UNIVERSITY OF SOUTHAMPTON

FACULTY OF ENGINEERING AND THE ENVIRONMENT

AERONAUTICS, ASTRONAUTICS AND COMPUTATIONAL ENGINEERING

Development of a Decision Support Framework for Systems Architecting in Aerospace Applications

by

Amrith Surendra

Supervisor: Prof James Scanlan
Second Supervisor: Dr Hakki Eres

Thesis for the degree of Doctor of Philosophy

August, 2015

UNIVERSITY OF SOUTHAMPTON

ABSTRACT

FACULTY OF ENGINEERING AND THE ENVIRONMENT

AERONAUTICS, ASTRONAUTICS AND COMPUTATIONAL ENGINEERING

Doctor of Philosophy

DEVELOPMENT OF A DECISION SUPPORT FRAMEWORK FOR SYSTEMS ARCHITECTING IN AEROSPACE APPLICATIONS

by Amrith Surendra

The exploration of the architectural solution space tends to be iterative, where multiple system architectures are explored over several cycles before a final solution is selected. However the time and cost required to conduct this activity can be significant in the presence of multiple system architectures. This thesis presents a decision support framework that assists the system architect in generating, analysing, and identifying optimal system architectures. The framework achieves this by using a formal modelling approach that represents the architectural decision-making process as a Constraint Optimisation Problem (COP), resulting in a graph representation of interconnected architectural decision variables. The graph-based approach provides computational tools that enables the system architect to automatically synthesise viable architectures based on the constraints defined, and calculate high-impact decision variables within the network. This capability is enabled by synthesising concepts from decision theory, multi-objective optimisation, and centrality measures from network analysis to provide a visual representation of high-impact decision variables. In applying this framework to the design of a low-cost Unmanned Aircraft System (UAS), we identify the choice of design alternatives relating to the implementation of the payload sensor system to have a high-impact on system properties and network connectivity. Suggesting that the exploration of the solution space should be focused towards payload implementation.

Table of Contents

List of Figures

List of Tables

DECLARATION OF AUTHORSHIP

I, Amrith Surendra declare that the thesis entailed 'Development of a Decision Support Framework for Systems Architecting in Aerospace Applications' and the work presented in it are my own and has been generated by me as the result of my own original research. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University;
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated;
- Where I have consulted the published work of others, this is always clearly attributed;
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work;
- I have acknowledged all main sources of help;
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself;
- Parts of this work have been published as:

.1. Surendra, A, Ferraro, M, Schumann, B, Van Schaik, J, Daniel, J, Gorissen, D, Scanlan, J.P. and Keane, A.J. (2013) The challenges of using value-driven design for practical design of UAVs. Journal of Aerospace Operations, 1, (4), 377-386. (doi:10.3233/AOP-120025).

.2. Schumann, B, Ferraro, M and Surendra, A, Gorissen, D, Scanlan, J.P. and Keane, A.J. (2013). Better design decisions through operational modelling during the early design phases. AIAA Journal of Aerospace Information Systems Vol. 11 (4), pp195-210 (doi: 10.2514/1.I010149).

.3. Surendra, A, Scanlan, J.P, Eres, H and Harman, A (2015). Development of a decision support framework for systems architecting in aerospace applications. Journal of Aerospace Information Systems (ISBN: 2327-3097) (Submitted).

Signed:.....

Date:.....

Acknowledgements

I would like to thank both my supervisors, Professor Jim Scanlan and Dr. Hakki Eres for their guidance and support over the last four years. I am thankful for their patience when things didn't work; their guidance, when things did work; and the general pleasantness of their conversations.

I also would very much like to thank Adam Harman, Martin Johnson, John Warbutton, and very one else within Advanced projects in Rolls-Royce for allowing me to spend two years of my research life in industry. The experience gained from my time over there has been invaluable and a great learning experience.

And of course, I would like to thank my parents, who have always stood by my side. My close friends for making my life a lot more fun and interesting, especially when things got hard and stressful at work.

Thanks is also due to University of Southampton, for providing me the opportunity to conduct research in the university's premises, the then School of Engineering Sciences (SES) now Faculty of Engineering & the Environment (FEE) for providing me a studentship.

Thank you all!

Amrith Surendra

List of Abbreviations

Abbreviations	Descriptions
ADG	Architectural Decision Graph
AHP	Analytical Hierarchy Process
AoS	Algebra of System
ASK	Amplitude-shift keying
BER	Bit error rate
BLOS	Beyond line-of-sight
BVLOS	Beyond visual line of sight
CBA	Cost Benefit Analysis
CER	Cost Estimation Relationships
CODA	Concept Design Analysis
CSP	Constraint-Satisfaction Problems
DARPA	Defense Advanced Research Projects Agency
DBPSK	Differential binary phase shift keying
DOE	Design of Experiments
DSS	Decision Support Systems
EA	Evolutionary Algorithm
ECS	Environmental Control System
EO	Electro-Optic
ESF	Epoch Syncopation Framework
FMA	Functional-Means Analysis
DVS	Decision variable Sensitivity
FOV	Field of View
FPA	Focal Plan Arrays
FQFD	Fuzzy Quality Function Deployment
FSK	Frequency-Shift Keying
GA	Genetic Algorithm
GP	Gaussian Process
GSD	Ground Sampling Distance
GTOM	Gross Take-Off Mass
HALE	High Altitude Long Endurance
IIA	Independence of Irrelevant Alternative
IRMA	Interactive Reconfigurable Matrix of Alternatives
LCC	Life-Cycle Cost
LLT	Lifting-Line Theory

LOS	Line-of-Sight
MADM	Multi-Attribute Decision Making
MAU	Multi-Attribute-Utility
MAUT	Multi-Attribute-Utility Theory
MCDM	Multi-Criteria-Decision-Making
MDP	Markov Decision Process
MDT	Maintenance Down Time
MIT	Massachusetts Institute of Technology
MODM	Multi-Objective Decision Making
MTBF	Mean Time Between Failure
MTBM	Mean Time Between Maintenance
NASA	National Aeronautics and Space Administration
NBI	Normal Boundary Intersection
NC	Normal Constraint
NPV	Net-Present Value
NSGA	Non-Dominated Sorting Genetic Algorithm
ODM	Overall Design Merit
OPN	Object-Process Network
OSD	Office of the Secretary of Defense
PSK	Phase-Shift Keying
QFD	Quality Function Deployment
QP	Quadratic Programming
QPSK	Quadrature Phase Shift Keying
RDT&E	Research, Development, Test and Evaluation
RF	Radio-Frequency
ROA	Real Options Analysis
SATCOM	Satellite Communication
SCVR	Standardised Cross Validation Residual
SEAD	Suppression of Enemy Air Defence
SFC	Specific Fuel Consumption
SNR	Signal-to-Noise Ratio
SPEA	Strength Pareto Evolutionary Algorithm
SQP	Sequential Quadratic Programming
TDN	Time-Expanded Decision Network
TIES	Technology Identification, Evaluation and Selection
TOP	Take-Off Parameter
TOPSIS	Technique for Ordered Preference by Similarity to the Ideal Solution

TOS	Time On Station
UAS	Unmanned Aircraft System
VCDM	Value-Centric Decision Making
VDD	Value-Driven-Design
VSP	Vehicle Sketch Pad

List of Symbols

Symbols	Descriptions
$\mathbf{J}$	Objective vector
$\mathbf{f}(\mathbf{x})$, $\mathbf{g}(\mathbf{x})$, $\mathbf{h}(\mathbf{x})$	Functions with input design variable vector $\mathbf{x}$
A_{ij}	Adjacency Matrix
U	Overall utility value
$U(X_i)$	Utility value of system attribute X_i
k_i	Scaling factor of the i^{th} system attribute
K	Normalising parameter
K_f	Relaxation factor
d_i	Euclidian distance between filtered i^{th} data point and the nearest member of the Pareto front
$N_{filtered}$	Total number of filtered data points
$\delta_{i\,,\text{shift}}$	Utopia point shift for a given system architecture C_i
μ_j^i	Utopia point of objective j of the i^{th} system architecture
μ_j^o	Global utopia point of objective j of the global Pareto front
σ_Z^2	Variance of the Gaussian Process
θ_i	Hyperparameter in Gaussian process correlation function
$\boldsymbol{\theta}$	Hyperparameter vector for θ_i
Γ	Covariance function
$\mathbf{\Gamma}$	Covariance matrix
s^k	Search direction of the k^{th} iteration
x_k	k^{th} decision variable
$\widetilde{D}_k$	Set of value assignments for x_k that exists as feasible value assignments
$d_{k,i}$	The i^{th} alternative in the decision variable x_k, $d_{k,i} \in \widetilde{D}_k$
$E(\delta_{shift})$	Mean of the utopia point shift over all the system architectures
$E(\delta_{shift}\|x_k = d_{k,i})$	Mean of utopia point shift over all feasible system architectures with the decision variable assignment $x_k = d_{k,i}$
$\|\widetilde{D}_k\|$	Number of feasible alternatives of the set $\widetilde{D}_k$
$x_{c,k}$	Centrality measure of decision variable x_k

α	Positive constant, typically chosen to be less than 1
x_j	Neighbouring nodes of decision variable x_k
k_j^{out}	Out-degree of the neighbouring nodes
$D_{Antenna}$	Antenna Drag, (N)
ρ	Air density, (kgm^{-3})
V	Velocity, (ms^{-1})
d_{ant}	Antenna diameter (m)
C_D	Drag coefficient
t/c	Thickness-to-chord ratio
GSD_H, GSD_V, $GSD_{geometric\ mean}$	Ground sampling distance in the horizontal plane, vertical plane, and the geometric mean
R	Slant range from target to the sensor, (m)
θ_{Look}	Look angle (deg) between the sensor and the target
H_{pix}, V_{pix}	Number of detector elements in the horizontal and vertical plane

Operators

Symbols	Descriptions
∇	Del operator
$\nabla(.)$	Gradient of $(.)$
$\nabla^2(.)$	Divergence of the gradient $(.)$
$[.]^{\mathrm{T}}$	Matrix transpose of $[.]$
$[.]^{-1}$	Matrix inverse of $[.]$
$E[.]$	Expectation operator
$\lvert\cdot\rvert$	Absolute value and vector norm
$\lVert\cdot\rVert$	Matrix norm

Chapter 1: Introduction

The design of engineering systems can be described as an interdisciplinary approach that involves not only technical effort, but also non-technical management activities and processes throughout the systems life-cycle to ensure that the product satisfies customer needs [1]. The design process, in simplified terms, follows a logical sequence of activities and decisions that transform the operational needs of a system into a description of a preferred system configuration [2]. The choice of a solution tends to depend on the prior experience of the design team, and by the analysis carried out to explore multiple system architectures. From a systems engineering perspective, a system architecture relates to the earliest decisions made in the design of a complex system. A system in general is represented by a decomposition of the system-elements into its associated sub-elements, and the interrelationships that exists between those sub-elements. This view is represented by Crawley et al [3] as "*an abstract description of the entities of a system and the relationships between those entities*".

System architecting on the other hand is defined as a process of decision-making. Hazelrigg [4] defines three key elements in the decision-making process as, identification of options or choices, development of expectations on the outcomes of each choice, and the formulation of a system value function for ranking different system architectures. The process of system architecting, as defined by Rechtin [5] is one that is driven by a customer's purpose or purposes, it involves working jointly with the customer to identify value judgements of the client and a solution definition. The solution definition should not only cover the physical representation, but other aspects such as cost, performance, human organisation, and other elements that characterise the system definition. Rechtin, distinguishes system architecting from systems engineering in its greater use of heuristic reasoning, lesser use of analytics, closer ties to the customer, and a particular concern with the certification of readiness for use. This view of architecting is also encompassed as the embodiment stage in classical engineering design, in which functions are translated into physical entities, which are then arranged in space such that they can achieve the desired function [5], [6].

A common approach to system architecting follows the process of decomposition, where the functional intent of a system architecture is decomposed into its associated sub-functions. System architectures are then defined by allocating several design alternatives to each function, at a given level of system abstraction. This mapping between functions to elements of form are represented in two arrangements. One is a representation of the elements of an architecture in a matrix form, and the other is a representation of the elements as a graph network. Figure 1.1 provides an illustrative example of the graph and matrix representation,

in which a set of system functions and their associated design alternatives are represented in a matrix. This is then translated into a graph representation by transforming the adjacency matrix A_{ij} into a series of nodes and edges. In this instance the nodes of the graph represent functions and the connecting edges represent relationships between those functions. In this example the relationships are assumed to be bi-directional.

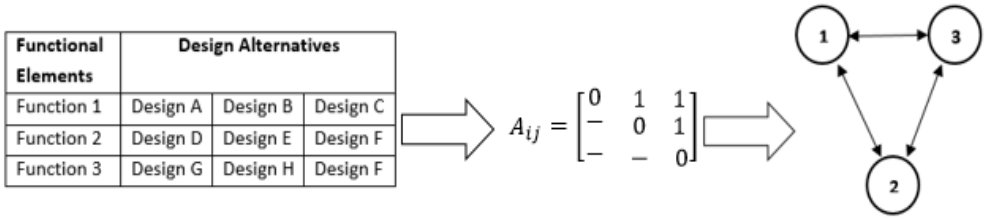


Figure 1.1 Matrix and network representation of the relationship between functions and elements of form.

This can be mathematically represented by defining system functions as decision variables $\mathbf{X} = [x_1, x_2, \dots, x_N]$, each decision variable being assigned a domain of values $D_i = [v_1, v_2, \dots, v_n]$, where $[v_1, v_2, \dots, v_n]$ represents n design alternatives. The adjacency matrix A_{ij} is used to quantify the existence of relationships between decision variables x_i and x_j, where $i \neq j$.

The allocation of design alternatives to each decision variable is generally considered to be the divergent phase of design, as multiple design alternatives are considered for each decision variable with no predefined solution in mind. The convergent phase of design starts with ruling out infeasible or sub-optimal design alternatives, based on some pre-defined criteria, from the architectural solution space. This gives a set of feasible design alternatives that are carried forward for further analysis. This exploration process is illustrated in Figure 1.2.

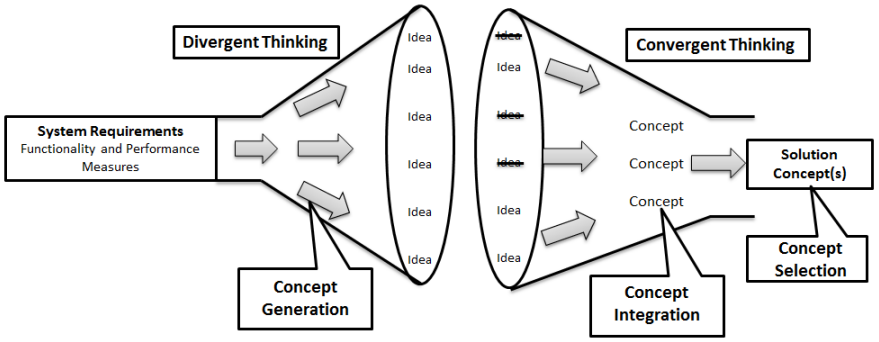


Figure 1.2: Illustration of the concept exploration process [7].

This process generally tends to be iterative as the architectural solution space is explored multiple times before a final system architecture is selected. The drawback in applying this exploration strategy is that the down selection process is predominantly based on prior experience alone, which at the start of the program is difficult to judge due to the uncertainties in requirements and the operating environment. However, if a pure exploration

strategy is adopted, one in which all design alternatives are taken forward for further analysis without the initial down selection step, the number of architectural trades will grow rapidly. Resulting in the work load required to assess multiple system architectures to increase significantly [8]. Thus, a balance needs to be struck between engineering judgment and exploration, where exploration of the architectural solution space is carried out only when the impact of changing design alternatives can have a significant effect on improving system properties.

This thesis tackles the problem of architectural exploration by encoding the architectural decision-making process as a Constrain Optimisation Problem (COP). The COP lends itself to representing the system architecting problem as a graph, with decision variables represented as nodes, and the connecting edges representing compatibility constraints between different decision variables. The graph-based model provides a foundation to incorporate domain independent tools, such as Constraint Satisfaction Problem (CSP) algorithms, and network centrality measures to aid the architectural exploration process. The performance and cost benefits of viable system architectures are then quantified by applying Multi-Attribute Utility Theory (MAUT), which is then fed into a multi-objective optimisation framework to calculate Pareto optimal solutions of utility and cost. The novelty of the framework presented in this thesis is in its ability to calculate and visualise high-impact decision variables. High-impact decision variables are quantified as those that are sensitive to changes in system properties, due to changes in design alternatives within its domain, and strongly connected to other decision variables within the network.

Standard techniques from global sensitivity analysis, such as Sobol/Saltelli sensitivity indices are typically used to explore the parameter space to provide robust sensitivity measures in the presence of nonlinearity and interactions amongst model parameters. The Sobol'/Saltelli method provides variance-based sensitivity indices that quantify the relative contribution of each parameter to the uncertainty in outputs. This requires the parameter set to be continuous, however the domain of design alternatives within each decision variable are discrete categorical parameters, and are therefore not appropriate for global sensitivity analysis. The ADG (Architectural Decision Graph) framework, developed by Simmons [9], overcomes this by applying a modified version of the main effects analysis to calculate decision variable sensitivity. However, the ranking of important decision variables within the ADG framework is qualitative, and can result in rank reversal based on the bias of the decision maker. This thesis builds on the ADG framework to provide a quantitative measure of high-impact decision variables, by accounting for both decision variable sensitivity and network connectivity, by applying PageRank centrality [10]. In doing so, the focus of exploration is aimed towards decision variables that have a significant impact on the Pareto optimal solution set that is generated.

1.1 Design Process

Design in engineering systems has attracted a vast amount of research across numerous aspects of the design process and its applications. According to the source: *The Design*

Society [11], the number of books and proceedings published in the field of engineering design is over 150, and the number of papers published is over 6300. These numbers are taken from a single source, and is used to quantify the term 'vast'. However, the true number of publications are unknown, but for illustrative purposes the former numbers stated will suffice. Due to the large number of publications within engineering design methods, multiple definitions of *design* have emerged over the years, of which the most prominent description of *design* being defined as a decision-making process [18-21]. Howe [16] states the role of decision-making in engineering design as:

"Engineering design is a non-unique iterative process, the aim of which is to reach the best compromise of a number of conflicting requirements. Whether the need is for a totally new item or for a development of an existing one, the design procedure commences with an interpretation of the requirements into a first concept. This is essentially a synthesis process which involves decision-making. Once the first concept has been derived it can be analysed in the context of the requirements. The concept is refined by an iterative synthesis/analysis/decision-making sequence until an acceptable solution is achieved."

This statement identifies that decision-making is inherently embedded within the design process, and is core to all design activities. The development and synthesis of a system architecture generally involves a series of decisions taken by a group of decision makers. The outcome of these decisions typically define the requirements, targets, and architectural solutions, based on the knowledge and experience of the decision makers to create an optimal and robust system solution. However, most of the high level decisions are made early on in the design process where the information available is limited, and requirements are highly uncertain. To support the decision-making process multiple methods have been developed, which all fall under the category of Multi-Criteria Decision Making (MCDM). Sen and Yang [15] give an overview of MCDM methods that have been most prominently used in engineering design.

The overall design process can be categorised into a series of phases, starting with conceptual design, than preliminary design, and detailed design. This simplified overview is represented in many research papers and design books [5, 23-24]. Conceptual design starts with the identification of end-user needs, problem definition, collection of information in regards to the operational use of the system, and the definition of a system architecture [18]. Preliminary design involves the refinement of the selected architecture, which is studied in enough detail such that the decision maker is confident that the system will meet the specified requirements. Resources can then be committed to manufacturing and production of the system. Detailed design produces a complete engineering description of the system ready for production, where the designed parts are fabricated, and assembled [5, 23]. This iterative process is typically represented as a design spiral, as seen in Figure 1.3.

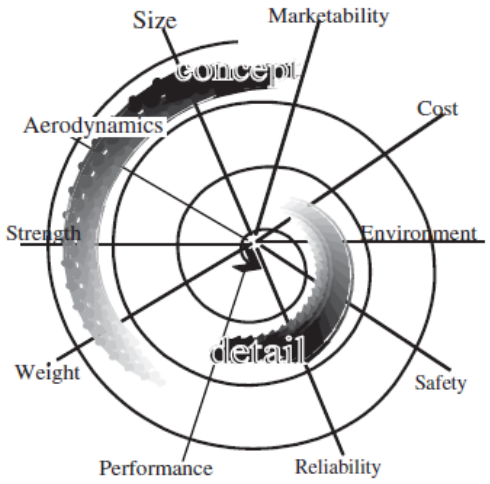


Figure 1.3: The design spiral [13].

This thesis primarily focuses on the conceptual design stage. It is in conceptual design that several high level questions are answered and decisions made, such as configuration arrangement, size, mass, and aircraft performance. It is also the stage in design where there is the greatest flexibility to influence system solution by exploring multiple design alternatives and performing several trade studies. Figure 1.4 provides an illustrative description of the traditional lifecycle cost curve seen in engineering design. The greatest opportunity to influence the cost of a system lies in the conceptual design phase, this is backed up by the fact that if most of the changes are made in the latter stages of design. The effort required for re-design and the cost of making changes will dramatically increase, due to design freedom being highly limited. To prevent costly re-designs much knowledge as possible should be ascertained and be made available at the early stages of design, allowing changes to be made before the cost is locked in. Thus, supporting the decision-making process at the conceptual design stage has much wider implications down the design process in developing an optimal and robust system solution.

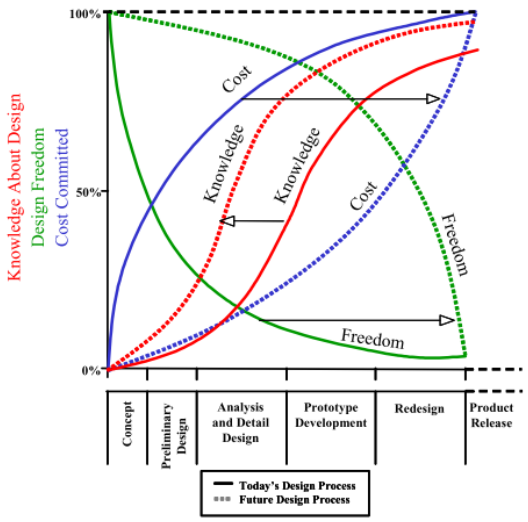


Figure 1.4: Cost-Knowledge-Freedom Relations [20].

1.1.1 Design from a Systems Engineering Perspective

There are several definitions of systems engineering. The one that is most noticeable and most prominent in the world of aerospace engineering is that defined by the National Aeronautics and Space Administration (NASA):

"*Systems engineering is a methodical, disciplined approach for the design, realization, technical management, operations, and retirement of a system. A "system" is a construct or collection of different elements that together produce results not obtainable by the elements alone. The elements, or parts, can include people, hardware, software, facilities, policies, and documents; that is, all things required to produce system-level results. The results include system-level qualities, properties, characteristics, functions, behaviour, and performance"* [1].

The key concept of systems engineering is to take consideration of several aspects that encompasses a collection of activities, processes, and techniques that are beyond the scope of the technical design challenge. Its purpose is to enable the generation of a system that provides a design solution that meets the needs and requirements of the system stakeholders. This is achieved by providing a methodical approach that enables the design and management of a complex system, where decisions are made based on higher level system criteria, rather than individual performance attributes [18][1]. Therefore, systems engineering is seen as a means to achieving a design that is capable of meeting requirements that are often conflicting in nature. The systems engineering process is typically iterative and is applied more than once during each stage of the development cycle. A brief description of the process, taken from the *Systems Engineering Fundamentals* handbook, published by the Department of Defense (DoD) [2] is depicted in Figure 1.5.

The first step in the process is to analyse inputs, which tend to be requirements and constraints defined by the system stakeholders. Based on the analysis of inputs, a set of functional and performance requirements are characterised [18]. These higher level functional requirements are further decomposed into lower-level functions, and its associated performance requirements are also decomposed and allocated to lower level functions. The end result of this analysis is a detailed description of the system, which explains what the system does and how well it must do it at each level of decomposition. In performing these steps the designer will gain a deeper understanding of the requirements, and will often find issues within the original requirements set. This results in the designer to revisit the requirements analysis stage of the process to redefine some of the requirements that were deemed unobtainable. This iterative exercise is referred to as the requirements loop.

[1] Attributes are defined as parameters of the system under consideration that are used to describe the system. For example this could be performance or dimensional parameters such as thrust, mass, etc.

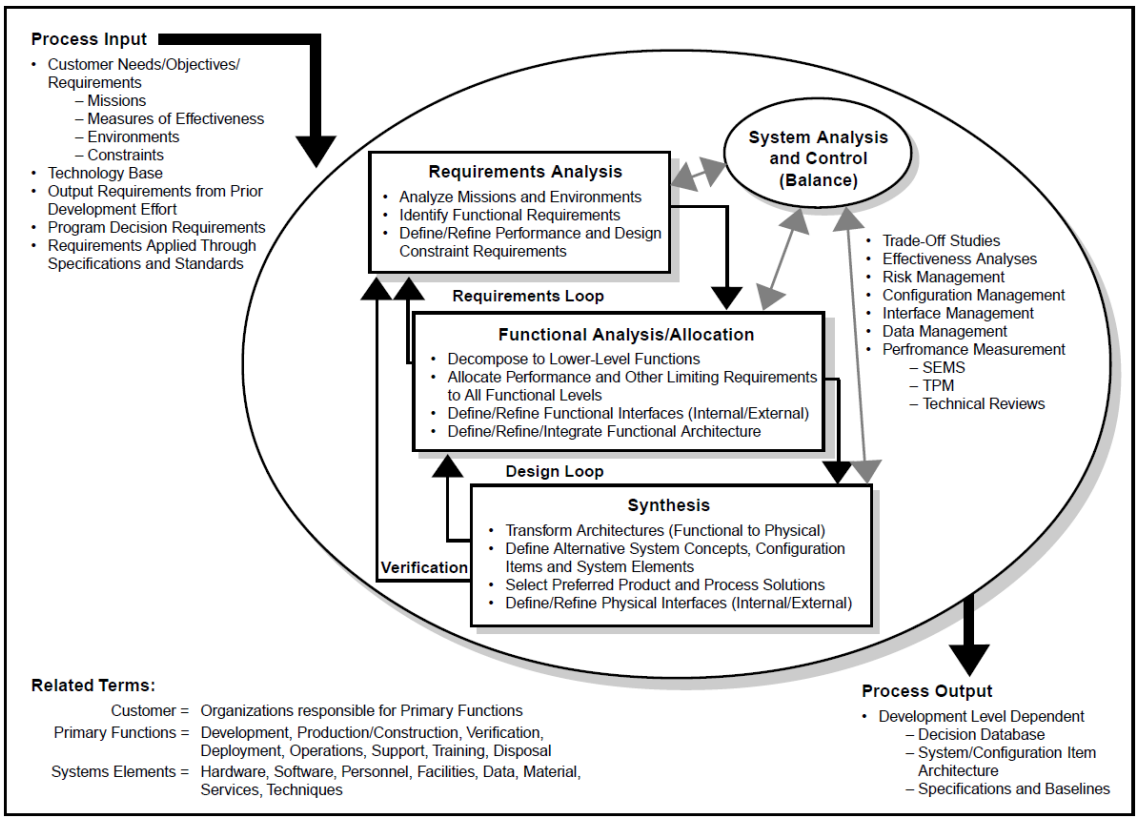


Figure 1.5: Transformation of system requirements to system synthesis [2].

The next step in the process is known as design synthesis, in this stage the physical architecture of the system is generated. The physical architecture aims to meet all the system functions, and its associated performance parameters defined in the previous step. The detail to which the physical architecture is defined is dependent on the phase of the design process. For example in the early phase of design, the architecture is defined only at a conceptual level i.e. the system definition is abstract and does not define the detailed implementation of sub-systems and components. The process of design synthesis is also an iterative process that allows the designer to revisit the functional and performance requirements in order to ensure that the physical system can meet the functional and performance criteria. This process also requires the designers to carry out "*technical management*" activities such as trade-off studies and effectiveness analysis [18]. This ensures that alternative solutions are considered, and the relationships between multiple aspects of the systems are defined. Technical management activities also consist of risk management and progress assessment, these activities must be monitored throughout the process cycle, which may involve conducting technical reviews on a regular basis.

The output of the process is a wealth of data that describes the system configuration, and the design processes that are required to develop a system. However, the framework does not explicitly address the complexities in the decision-making process at each stage of system development. This has resulted in a recent increase in research interest in the field of Decision Support Systems (DSS), which are capable of coping with complex decision-

making problems in engineering design. The aim of DSS is to aid the selection of optimal system architectures that meet the defined criteria.

1.2 The Decision-Making Process

Decision-making in engineering design is a complex process that involves many organisations, and generates of vast amounts of information during the design process. Decision support can therefore be described as assisting the decision maker in making decisions by providing a rational procedure for arriving at a solution. This is achieved by identifying an appropriate course of action, among alternative courses, that aims to attain the organisational goals, or maximise the satisfaction of stakeholder needs. Simon [21] describes the decision-making process to be categorised into four major phases, intelligence, design, choice, and implementation. A conceptual picture defining this process is represented in Figure 1.6. In this process there is a continuous flow of activity from intelligence to design to choice, with each phase being revisited as more information is gathered and knowledge ascertained.

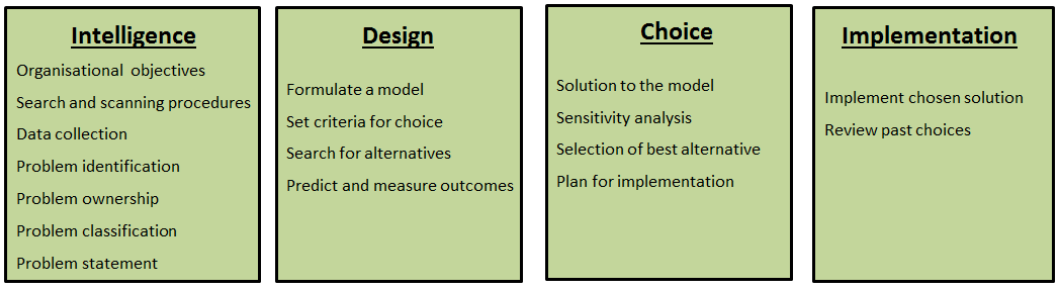


Figure 1.6: Simon's four stages of decision-making [21].

The decision-making process starts at the intelligence phase, which aims to identify problems and opportunities from different data sources. This begins with the identification of customer/end-user goals, system objectives, and a means of satisfying them. Problems arise as a result of differences between the defined goals, and the feasibility of the system in satisfying them. At this phase attempts are made to find the problem symptoms, the magnitude of its effect, and defined measures of addressing them.

The design phase involves developing models to analyse a set of feasible design alternatives. Modelling of a system and its outputs (which for example include cost, performance, reliability, and other system attributes) requires simplifications to be made of the operational environment and the design itself. However, a fine line should be drawn in the assumptions made, such that they do not over simplify the scenario and deviate away from reality. The outputs of these models are used to quantify system performance and cost attributes, which in-turn enables the decision maker to define courses of action in the implementation of the system.

The choice phase is where decisions are made, and where an action plan is drawn to follow a certain course of system implementation. The boundary between choice and design phase tend to overlap at times, as they are closely interlinked. Since models and its output parameters are continuously changing as more information is acquired, the choice activities frequently revisit the design activities. In this respect, the choice phase is considered to encompass the search, evaluation, and recommendation of an appropriate solution to the modelling of the design phase. Finally, in the implementation phase, decisions (but not necessarily the system) are implemented. Successful implementation leads to problems being solved, and failure results in returning back to an earlier phase of the process.

Simon [21] suggests that the most central and time-consuming part of the four phases are the intelligence and design phase, due to their wider impact of influencing the final solution. The effort required in designing and analysing a feasible set of design alternatives in systems architecting is time consuming, as it often introduces a search space that is divergent in the number of possibilities. Due to their importance in facilitating the decision-making process, this thesis mainly focuses on developing tools and methods to address the intelligence and design phase.

1.3 Decision Support Systems (DSS)

To handle the four phases of decision-making in engineering design many tools have been developed, which fall under the category of Decision Support Systems (DSS). DSS can be described as tools that an organisation uses to support and enhance the decision-making activity [27-28]. DSS provides a simplified representation of a situation that is understandable to the decision maker, thus allowing the decision maker to run sensitivity analyses of inputs, and conduct what-if scenarios to better understand the behaviour a system. A detailed literature review on the development and use of DSS, and its applications in industry has been given by Power and Sharda [24]. In the decision support literature five categories of DSS are recognised, these include: model-driven, communication-driven, data-driven, document-driven, and knowledge-driven [24].

Model-driven DSS uses computerised systems that account for financial models, representational models, and/or optimisation models to assist the decision-making process. The development of quantitative models is a dominant component in the DSS architecture, it makes use of datasets and parameters provided by the decision-makers to help analyse situations and aid the decision-making process [24]. Communication-driven DSS provides its functionality via information technology systems to support shared decision-making. Data-driven DSS includes management reporting systems, data storage and data analysis systems, and information systems. The functionality of these systems is provided by gaining access to and manipulating large datasets to identify trends, or important system parameters that would influence the view of the decision maker. Document-driven DSS makes use of document retrieval and analysis capabilities to support the decision-making process. Finally, knowledge-driven DSS suggests courses of actions, based upon information/data that has already been stored. This system makes use of sophisticated tools such as artificial

intelligence, and statistical analysis tools such as case-based reasoning, and Bayesian networks [25].

In reality a decision problem is addressed by many decision support tools that span across all these five categories. For example, information systems may include model-driven DSS and knowledge-driven DSS modules for pre- and post-processing. This thesis focuses on developing methods in model-driven DSS, though the application may have other decision support subsystems incorporated within it. Model-driven DSS offers a wide variety of tools to assess multiple decision problems that span across a wide variety of applications. This is seen to be especially useful early on in the decision-making process, where data and information that is available is limited. Model-Driven DSS broadly spans three sub-categories, decision analysis, optimisation, and simulation. Decision analysis refers to methods that provide quantified evaluations of possible courses of action. These evaluations often include the assessment of the decision maker's value structure to identify the decision maker's needs and objectives.

The application of multiple assessment methods in decision-making is described by Multi-Criteria Decision Making (MCDM). The main focus of this thesis is in identifying an appropriate assessment method that captures the value of the system stakeholders, and in-turn aid the architectural exploration process. The following sub-sections provide a brief introduction to MCDM and systems architecting.

1.4 Systems Architecture and Systems Architecting

There have been numerous definitions over the years in describing system architectures. This thesis takes the approach presented by Dickerson and Mavris [18], and focuses on the definition given by the electrical engineering and software engineering community. From: IEEE Std 610.12-1990:

"[System] Architecture is the organization of the system components, their relations to each other and to the environment, and the principles guiding its design evolution."[26].

From Miller and Mukerji:

"The architecture of a system is the specification of the parts and connectors of the system and the rules for the interactions of the parts using the connectors."[27].

System architectures generally tend to evolve over long periods through the program life, as technology matures to a point where the system architect feels that the behaviour and interactions of the system/sub-systems are understood. Crawley [3] identifies the importance of a systems architecture, in that it provides a means of understanding the complexity of the system, a means to design systems, and a means to manage systems. The importance of system architectures is also identified by its influence on the system "ilities"-flexibility, scalability, maintainability, reliability, and survivability, its influence on functional

behaviour, which can be intended or unintended (emergent behaviour), and its influence on the complexity of system operations. This is illustrated by Figure 1.7.

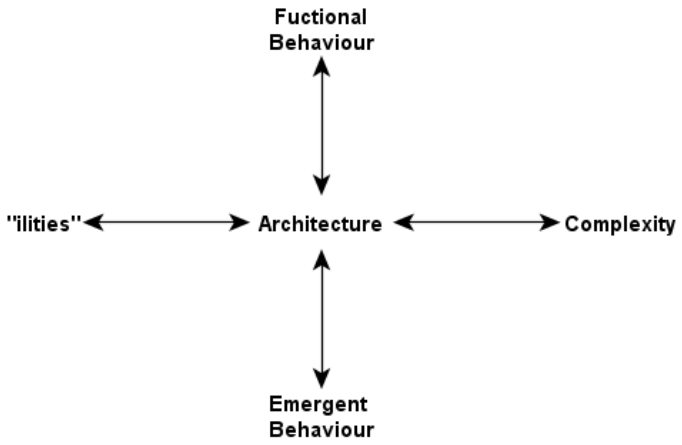


Figure 1.7: The central role of system architecture on complexity, "ilities", functional behaviour, and emergent behaviour [3]**.**

Systems architecting is referred to the process of creating a system architecture [9], which in general involves identifying system functionality i.e. what the system is supposed to do and how it will do it. The process starts with an initial set of requirements from the customer/end-user, describing the system outcomes and constraints imposed by the operational environment. This provides an initial boundary of the possible design alternatives that are viable in the implementation of the system. At this stage several system architectures are evaluated and information relating to system behaviour is obtained. Based on this information, and from previous experience of the decision maker, decisions are made regarding the implementation of the system. This in general will result in a reduction of the design space by removing infeasible system solutions, which in turn will impact the feasible of other system functions. This process is represented in Figure 1.8. The decision-making process thus results in the architectural solution space to become constricted, by reducing the set of design alternatives that are acceptable. The process of decision-making can therefore be defined through a network representation, where each node defines a system function and the arrows represent the constraints imposed by the decision maker that results in the reduction of the architectural solution space. The network representation depicted in Figure 1.8 shows each point in space as a feasible system architecture, which to some extent satisfies all of the needs and goals specified. In this space decisions are made to define a set viable system architectures that are carried forward for further analysis. This can be thought of as a partition and selection operation in the architectural solution space.

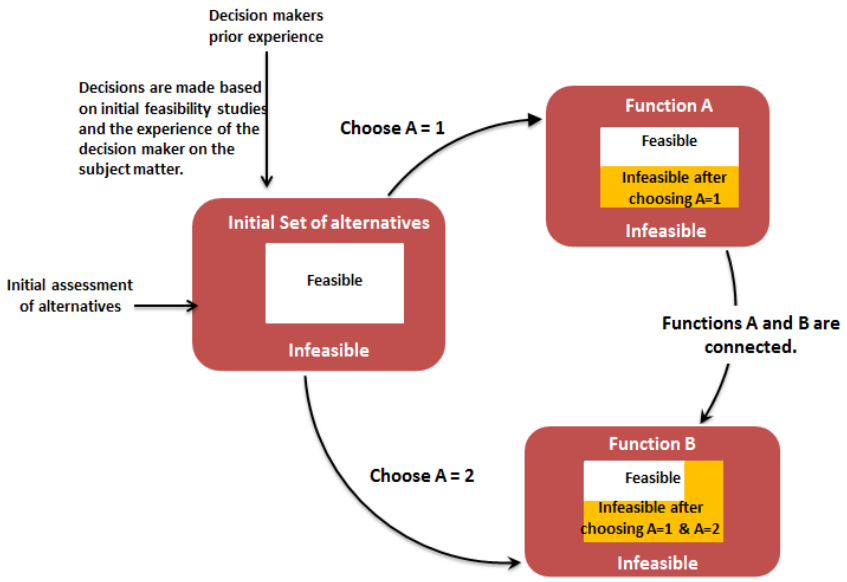


Figure 1.8: Impact of the system architecting process on the architectural solution space [9].

1.5 Multi-Criteria-Decision-Making (MCDM)

To develop a successful design various stakeholder needs have to be met, which can include cost, performance, environmental impact, operational availability, and other metrics of interest. In order to find the best compromise design solution, the decision maker is required to balance multiple, and potentially conflicting, objectives and transform these objectives/needs into a solution. The balancing of objectives/needs requires the use of MCDM methods. MCDM methods provide a systematic approach that employs decision rules and algorithms to formulate the design problem and provide support to the decision maker in reaching a final system solution [19]. In order to solve a MCDM problem some necessary information needs to be determined prior to applying any method. This includes a well-defined measurable criteria, preference information on the criteria, alternative system architectures, and a repeatable and transparent decision-making method.

The gathering of this information beforehand can be achieved by applying qualitative and quantitative methods, which are broadly grouped into two classes, Multi-Attribute Decision Making (MADM) and Multi-Objective Decision Making (MODM) [15], [28]. MADM techniques focuses on ranking and selecting the best solution from a set of design alternatives, based on a prioritised set of attributes of those alternatives [19], [28]. An example of this is an airline selecting an aircraft from a competing set of aircraft concepts. MODM methods on the other hand are more appropriate for design applications, as they focus on multiple objectives within the continuous design space, and are subject to active constraints. MODM methods relate to identifying the best design, within the design space, that satisfies a given set of constraints. This relates to an optimisation problem, where system objectives are maximised or minimised in order to identify a solution with the best values of those objectives. The combination of methods used in MODM and MADM form the overall solution for MCDM, as

depicted in Figure 1.9 . Since MCDM focuses on selecting the best solution from a set of conflicting objectives, it requires trading objectives to identify a set of designs from the feasible design space that are considered optimal. Identifying a set of design alternatives that are within the optimal solution set and are worthy of further consideration are formally referred to as the set of non-inferior or non-dominated solutions. The concept of non-inferior or non-dominated solutions can best be described via the concept of Pareto optimality.

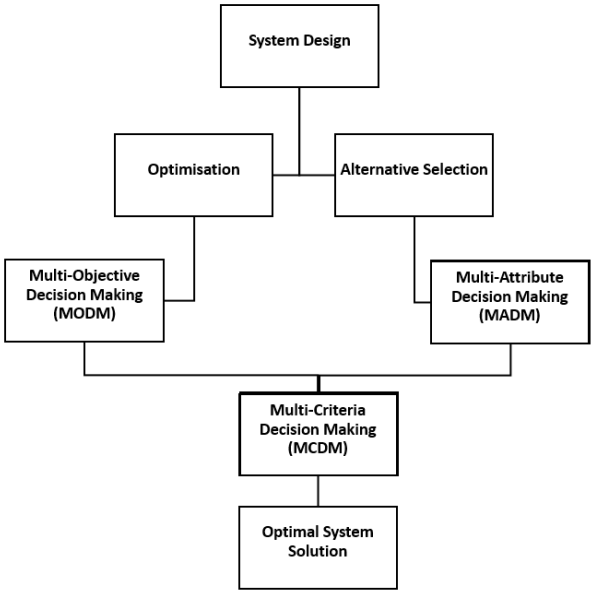


Figure 1.9: Multi-Criteria Decision Making (MCDM) Framework [19].

1.5.1 Pareto Optimality

Pareto optimality can be described by first defining the multi-objective optimisation problem. The multi-objective problem can be defined mathematically as a set of objectives $\mathbf{J} = \mathbf{f}(\mathbf{x}) = [(f_1(\mathbf{x}), \ldots\ldots, f_k(\mathbf{x})]$, where each objective is a different function of the inputs, $\mathbf{x}$, such that, $J_k = f_k(\mathbf{x})$. The optimisation problem can also be subject to a set of equality constraints $h_i(\mathbf{x})$, inequality constraints $g_i(\mathbf{x})$, and the design variables $\mathbf{x}$ upper and lower bound values. The aim of the optimisation problem presented in this thesis is to minimise the set of objective functions that depend on the vector of design variables $\mathbf{x}$, and on a set of constraint functions $[h_i(\mathbf{x}), g_i(\mathbf{x})]$: This is represented mathematically as:

$$\begin{aligned} &\text{Minimise} && \mathbf{J} = \mathbf{f}(\mathbf{x}) = [(f_1(\mathbf{x}), \ldots\ldots, f_k(\mathbf{x})] \\ &\text{Subject to.} && h_i(\mathbf{x}) = 0, \qquad i = 1,2, \ldots, p \\ & && g_j(\mathbf{x}) \leq 0, \qquad j = 1,2, \ldots\ldots, m \\ & && x_q^L \leq x_q \leq x_q^U, \quad q = 1,2, \ldots\ldots, n \end{aligned} \tag{1.1}$$

The solutions obtained from the optimisation problem are said to be Pareto-optimal, in that they are non-dominated in the objective space. Non-dominated solutions can be defined as a solution $\mathbf{J}^*$ for a given set of inputs $\mathbf{x}^*$ such that no other feasible solution exists for the same $\mathbf{x}$ that is better on all objectives J_k i.e. there are no other feasible solutions in the design space that has the same or better performance than the Pareto optimal solutions, considering all the objectives [29]. Mathematically dominance can be defined as being strong or weak dominance. For example, consider output solutions $\mathbf{J^1}$ and $\mathbf{J^2}$, in the case that $\mathbf{J^1}$ dominates $\mathbf{J^2}$ weakly if:

$$J_i^1 \leq J_i^2 \; \forall i \in \{1,2,\ldots.,k\} \text{ and } J_i^1 > J_i^2 \text{ for at least one } i \tag{1.2}$$

$\mathbf{J^1}$ dominates $\mathbf{J^2}$ strongly if and only if:

$$J_i^1 < J_i^2 \; \forall i \in \{1,2,\ldots.,k\} \tag{1.3}$$

Figure 1.10 shows a hypothetical two-dimensional Pareto front, which contains Pareto optimal solutions for a set of system architectures. A system architecture in this instance is represented as a combination of design alternatives, which is defined through a feasible value assignment from the domain of each decision variable i.e. a system architecture $\mathbf{X_j}$ is defined as: $\mathbf{X_j} = [x_1 = v_a, x_2 = v_a, x_3 = v_a, \ldots x_N = v_a]$, where v_a is some feasible value assignment within the domain of each decision variable. System designs on the other hand are variations of a given system architecture, where each system design is defined by a unique set of values, for the design variables $\mathbf{x}$. These values are represented by the interior points in the hulls of $[\mathbf{X_1}, \mathbf{X_2}, \mathbf{X_3}, \mathbf{X_4}]$, as shown in Figure 1.10. The global Pareto front of system architectures is obtained by concatenating the Pareto frontiers of all non-dominated system designs, from each system architecture. This method was originally presented by Messac and Mattson [26] and is defined as the set based Pareto frontier (S-Pareto front), which defines the notion of dominance for concepts i.e. sets of solutions, as opposed to individual solutions.

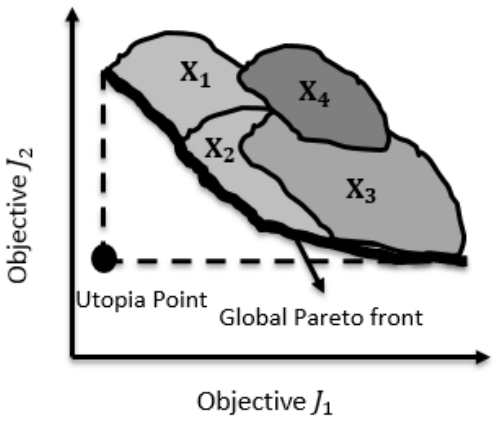


Figure 1.10: Two-dimensional Pareto frontier and feasible design space [30].

The selection of a solution from the Pareto frontier still requires a judgment to be made in regards to the relative importance of each objective. These judgments are generally specified

by the decision maker, which in turn is backed up by market valuations and customer demands. However, as the number of objectives are increased, and thus the complexity of the decision-making process increases, the number of non-dominated solutions will also increase dramatically [36-37]. The allocation of weights to each objective becomes harder to quantify by the judgement of the decision maker alone. This requires the use of MCDM methods to aid the identification of optimal system solutions in a complex multi-objective problem.

MCDM methods have been developed to handle more complex decision problems with multiple objectives, most of which provide a set process for the decision maker to follow through to get to the final solution. This generally involves converting a large multi-objective problem into a smaller multi-objective problem with one or two objectives. This has been the focus of MADM. On the other hand, MODM focuses on methods to identify the Pareto frontier. This is typical achieved via the weighted function approach, where a set of weights are introduced for each objective and the multi-objective problem is transformed into a single objective problem. The single objective problem is in-turn optimised to find the optimal solution set for the given weighting function [33]. Repetition of this process with different weighting functions allows most of the non-dominated solutions to be determined. To avoid weighting functions designers have turned to population based search schemes to construct the Pareto front. In such schemes designs are compared with each other to identify high quality designs. The spacing between competing designs are also compared such that the search can be tailored to progress towards the Pareto front that is evenly distributed [34]. There are multiple variations to the two methods just described, which aim to provide more efficient search strategies by speeding up the convergence process.

In recent years, much research has emerged in addressing architectural optimisation and design space exploration in aerospace systems. Buonanno et al. [9,10] have developed a multi-criteria interactive genetic algorithm (GA) in application to the design of a supersonic transport aircraft. In this a GA is used as a coarse search method in exploring the architectural solution space [11]. Similar approaches have also been taken by Brown and Thomas [35], and Singh and Dagli [36], in the exploration of Pareto optimal system architectures by making incremental changes to a starting baseline architecture. Ross [37] presents a review of several tradespace exploration methods that expand upon the Pareto front analysis to account for performance and cost, system flexibility, and robustness. However, these approaches require the architectural tradespace to be fully enumerated prior to search. Thus, the search for Pareto optimal solutions is dependent on the initial set of system architectures that were chosen to be enumerated.

In addition to searching the solution space for optimal standalone system architectures, several decision support tools have been developed to study the evolution of an architecture through multiple design alternatives. Chapter 3 presents a brief overview of some of the methods that have emerged over recent years in addressing optimisation problems in

engineering design. For a more in-depth review of the advancements in different optimisation methods and its applications the reader is referred to Keane and Nair [13].

1.6 Research Aims and Objectives

The following section outlines the scope of this research, and its objectives to achieve the proposed research aim. The research aim is as follows:

To develop a decision support framework that is able to aid the system architecting process to identify optimal system architectures, and identify high-impact decision variables.

The objective of this research is focused towards developing a decision support framework that captures the knowledge of the decision maker, in combination with quantitative data from concept analysis. This in-turn is used to identify high-impact decision variables that have a significant impact on system properties and strongly influence the feasibility of design alternatives of other decision variables. The framework should be able to capture the knowledge of the decision maker by first enabling the decision maker to generate a set of design alternatives that satisfies system functionality, and second identify the compatibility of different design alternative. The captured knowledge should in-turn facilitate the understanding of the decision rational, and provide a justification to the exploration of different system architectures.

The second part of this framework is focused towards knowledge representation, where optimal system architectures, and high-impact decision variables are readily presented to the decision maker. This requires the definition of optimality i.e. the choice of an appropriate objective function, or a set of objective functions that captures the needs of the system stakeholder. The research objective of this thesis is therefore focused towards defining an appropriate MADM method that captures the needs of the system stakeholders. This requires capturing stakeholder needs, which in-turn is used to quantify Pareto optimal architectures and high-impact decision variables.

1.6.1 Research Question and Hypothesis

Based on the literature review presented in this chapter it has been established that effective decision-making in engineering design is integral to the successful implementation of the system within its operational environment. It has also been established that decisions made based on pure judgement of the decision maker alone does not quantify for rigorous exploration of the solution space. However, a strategy based on pure exploration, where every architectural combination is analysed is not feasible. This exploration process is limited due to the availability of time, and the allocated budget for architectural exploration. Therefore, the following question is posed:

Is there a middle ground where the exploration of the architectural solution space combines both the experience and knowledge of the design team, with the quantitative analysis of multiple system architectures to identify an architectural exploration strategy?

In order to answer this question, the following hypothesis is made, which is the subject of this research study:

The needs for the exploration of the architectural solution space can be met by incorporating methods from systems engineering, decision theory, design of Experiments (DoE), multi-objective optimisation, and graph theory. The integration of these methods into an overarching framework will enable a more focused architectural exploration strategy, with the aim of maximising system utility and minimising LCC.

The hypothesis presented above is motivated through several key concepts, which can be best introduced through a series of questions. The questions are as follows:

Question 1. How to capture and represent the knowledge of the decision maker?

Question 2. How to encode the captured knowledge of the decision maker to represent decision variables with a set of design alternatives, a set of constraints between decision variables, and a means of removing infeasible system architectures?

Question 3. How to capture the needs of system stakeholders, and translate them into a set of objective functions to allows for system optimisation?

Question 4. How to represent the captured information/data to the decision maker, such that optimal system solutions, and high-impact decision variables can be readily identified?

Through the course of this thesis the above question will be answered, which in-turn should validate the overall research hypothesis. The concluding answers to these questions will be presented in the final chapter of this thesis.

1.7 Thesis Overview

This section provides a brief overview of the thesis outline. The overview is divided into the contents of each chapter.

Chapter 2 focuses on identifying an appropriate MADM, or Value-Centric Decision Making (VCDM) method that captures the value stream of the system stakeholders. This chapter provides an overview of different MADM and VCDM methods that are already available in literature. In comparing all the available methods it is found that Multi-Attribute Utility Theory (MAUT) best meets the needs of value-focused decision. MAUT achieves this by providing a means of aggregating multiple system attributes to form a single utility value.

Chapter 3 gives an overview of MODM methods, where the focus is split into prior and posterior articulation of preference information. This thesis makes use of a combination of methods in prior and posterior articulation. Multi-Attribute Utility Theory (MAUT) is used to capture the preference information of the system stakeholders prior to optimisation.

Posterior methods, such as NSGA-II and surrogate based optimisation are applied to identify Pareto optimal solutions, with the aim of maximising the utility function and minimising the life-cycle cost (LCC) function. The chapter concludes by describing the optimisation framework that will be applied in this thesis to identify Pareto optimal solutions.

Chapter 4 addresses the methods available in literature to construct and aid the architectural decision-making process. Following a similar structure to Chapters 2 and 3, it provides an overview of methods that are available in literature to construct the architectural decision problem. Here, three common metric are introduced to evaluate the validity of different methods: *Construction*, *Search,* and *Representation*. In applying these three metrics it was found that a combination of methods were required to address the architectural decision problem.

Chapter 6 introduces the reader to the decision support framework developed in this thesis, which combines methods and tools defined in previous Chapters. The framework can be broken down into three categories: *Construction, Search,* and *Representation*. The *Construction* and *Search* phases have already been addressed in Chapter 4. The *Representation* phase breaks presentation of data into two formats. One representing Pareto optimal solutions by plotting data on a utility versus Life-Cycle Cost (LCC) axis, which can be used to identify the Pareto front. The other format focuses on representing high-impact decision variables to the decision maker, in order to aid the architectural exploration process. The combination of all three phases encapsulates the architectural decision problem, which in turn should aid the decision maker to effectively search the architectural solution space.

Chapter 7 introduces the Unmanned Aircraft System (UAS) case study that is used in validating the developed decision support framework. This case study is loosely based around the 2Seas research project conducted by the University of Southampton. The 2Seas research project is focused towards the development of an UAS for maritime surveillance. The aim of this chapter is to introduce the reader to the integrated systems approach in designing an UAS. This not only includes aerodynamics and structure, but includes other elements such as communications system, payload sensors, and the delivery of electrical power to on-board avionics and payload systems. This thesis present a modelling environment that captures all the major system elements within an UAS. However, for simplicity and to ensure that the simulation environment is within the scope of this thesis, the methods encoded are low fidelity models that are intended for conceptual analysis. The simulated results are compared against the real 2Seas UAS system values to validate the accuracy of the modelling environment. Finally the modelling environment is integrated into the optimisation framework to identify Pareto optimal solutions.

Chapter 8 applies the decision support framework to the design of a small low-cost Unmanned Aircraft System (UAS). In applying the decision support framework, multiple design alternatives pertaining to the propulsion system type, communications systems, electrical power generation systems, and payload sensor systems where identified. To ensure

that only viable system architectures are analysed, physical/preferential constraints are captured using logical constraint statements. This information is then passed onto the *Search* phase to generate a set of viable system architectures. The generated system architectures are modelled using simple semi-empirical and analytical models to identify Pareto optimal solutions. The concatenation of Pareto fronts from all system architectures results in the calculation of the global Pareto front. The Pareto front data is also used to capture the sensitivity of each decision variable to changes in system properties, which is then fed into the network centrality measure to identify high-impact decision variables.

Finally, Chapter 9 concludes with a list of contributions made to address the questions defined in Chapter 1. Recommendations for future work in the areas of system architecting are made to address issues that have not been tackled in this thesis.

Chapter 2: Value-Centric Decision Making

Design of an engineering system involves the satisfaction of multiple objectives. It is often the case that improving one objective will inevitably result in at least another objective being worst off. Thus, the design of a system generally tends to be a compromise between multiple objectives. To find the best compromise design solution decision makers are required to take in account of a variety of system attributes of interest. In the example of commercial aircraft design, the decision makers will have to contend with cost, emissions, aircraft performance, and other design factors. Thus, design may inevitably be formulated as a Multi-Criteria Decision Making (MCDM) problem. To solve a MCDM problem some necessary factors have to be known beforehand, which include a well-defined measurable criteria, the preference information of the criteria, design alternatives, and a repeatable and transparent decision-making method. The chosen criteria needs to be well defined, such that system objectives can be fully represented. This typically includes a set of system attributes, or a single attribute that captures the preference structure of the stakeholders. To aid the decision-making process several decision support tools have been developed to provide a systematic process to formulate the decision problem, and provide guidance to the decision maker in reaching a final solution.

This chapter focuses on identifying an appropriate value centric decision-making (VCDM) or multi-attribute decision-making (MADM) method that identifies and captures the preference structure of system stakeholders. This will enable optimal system architectures to be identified that meet the needs and objectives set out at the start of the program. This chapter starts by defining the term value with respect to the system under consideration, and follows on to describe different VCDM and MADM methods that are available in literature. Finally, the chapter concludes by choosing an appropriate method that, 1) meets the stakeholder needs and objectives, and 2) represents the reality of decision-making in systems architecting.

2.1 What is Value?

The definition of value is dependent on an individuals or organisations perception of how to quantify value. Stakeholder's perception of value is driven by policies, politics, user needs, the operational environment, and other external factors. The perception of value also changes over time due to changes in the organisation structure, leadership style, and other exogenous influences. Hence, the definition of value is by no means definitive. However to gauge the idea of value, with respect to engineering systems, this thesis defines value from

both an economic and decision theoretic perspective. Ross et al. [38] identify various definitions of value by referring to Merriam-Webster online dictionary [39], which defines value as:

- A fair return or equivalent in goods, services, or money for something exchanged.
- The monetary worth of something: Market price.
- Relative worth, utility, or importance, a good value at that price.

The first two definitions are more in-line with the traditional economic definition of value. From an economic perspective the numerical worth of value is defined as the net worth of a product or a system, where net worth[2] is represented in monetary terms. This interoperation of value has been applied mainly in profit driven ventures, where shareholder satisfaction is derived from the creation of monetary value. The assessment of net worth requires defining the creation of resources to be assessed in terms of monetary value, and the value of alternative options to be assessed in terms of the monetary value added by these options [38]. In the aerospace industry this view of value has mainly been adopted by the civil airline sector, who are driven to maximise the net worth of their company [40].

From a decision theoretic perspective value is not only derived in monetary terms, but from multiple other sources. The source of value is dependent on the context that the system is operating under. For example, in military aircraft systems this could be the value in the aircrafts ability to capture information in reconnaissance operations. The third definition of value from Merriam-Webster online dictionary is more in-line with the decision theoretic view. Utility is defined as a non-dimensional measure of the satisfaction of an individual or entity in-exchange of a service, or the acquisition of goods. Since utility is non-dimensional it can be applied to any attribute within the system, hence the creation of value from a decision theoretic point of view is achieved by maximising the utility of the system [29].

2.2 Why use Value-Centric Decision-Making?

The decision-making element of design, by means of maximising system value, is defined in literature as Value-Centric Design (VCD) [43-44]. VCD combines the methodologies developed in Value-Driven Design (VDD), with systems engineering to provide a means of assessing the value of different system architectures. The assessment of value in essence is achieved by combining analytical/physics based models with cost models. Collopy et al [43] describes the VDD framework as one where no requirements are applied to extensive attributes such as reliability, maintainability, performance attributes, and all aspects of cost. In the VDD framework engineering teams rather follow an objective function, which is a combination of all system attributes aggregated together to provides a scalar value that indicates the 'goodness' of the overall system design. Design teams can therefore use this

[2] Net worth in economic terms is defined as the amount by which assets exceeds liability i.e. the total assets that an individual or business has minus the total liabilities of that individual or business.

scalar value to identify the best solution, which is achieved by designing a system that provides the highest value score.

In comparing the classical systems engineering approach to VCD, the systems engineering process is highly driven by requirements where the satisfaction of requirements guides the functionality of the product. Consider the case where requirements are provided for the design of an aircraft, normally this includes the range of an aircraft, payload mass, take-off distance, and other performance/design implementation requirements. These requirements are used to fix certain aspects of the aircraft, such as wing-loading (W/S) and thrust-to-weight ratio (T/W). The decision process now mainly focuses on the selection of aircraft sub-systems that best meets the requirements. These decisions are made based on prior experience and knowledge of the design team. The decisions made on these high level parameters causes the value of the system to be almost fixed, and restricted in improving it any further. By the time detailed aerodynamic and structural analysis are carried out the constraints are already fixed and there is little movement in the design space [44]. The resulting effect of this is that the program could require extensive redesign if certain requirements are not met, or if the requirements itself have changed. All of which can result in cost overruns and scheduling problems.

"The integration of VCDM methods into the conceptual design stage allows the design team to capture the complex interactions between each sub-system as design changes are made". To get a better understanding of this statement let's consider what the value metric defines. The aim of a value metric is to provide a score that indicates the 'goodness' of a given design (i.e. the higher the score the better the design). This is achieved by combining different disciplines of the design process i.e. economics, optimisation, and systems engineering [45]. So each design team bases its decision on maximising the score provided by the value model (objective function), and are not driven by satisfying specific requirements in isolation of other system attributes.

2.3 Value-Centric and Multi-Attribute Decision Making

The fundamental ideas behind VCDM and MADM are inherently the same, as both methods strive to find the best solution, from a set of alternatives, based on the evaluation of system attributes and their preference structure. The inherent difference between the two methods is evident when comparing the choice of the objective function. The VCDM method promotes the use of a single scalar objective function to capture the stakeholder preference structure, which can be monetary or have non-dimensional units. In comparison to MADM, which requires system attributes to be aggregated into a non-dimensional objective function. But no implicit method is defined that captures the stakeholder's preference structure, this is rather left to the system modeller. The following sub-sections provide a review of some of the widely used VCDM and MADM methods applied in engineering design.

MADM methods focus on defining a rank order for a set of design alternatives. The data required to rank designs can be defined as qualitative or quantitative. However, in reality

MADM problems may involve the use of both data types[15]. Qualitative evaluations can be defined as a MADM problem where each alternative is not numerically measured over a set of attributes, but is relatively evaluated by judgments of the decision maker. Quantitative evaluations use analytical or physics based models to calculate attribute values of design alternatives, the attribute values are then used to rank different design alternatives. Sen and Yang [15] have classified different MADM methods based on the required input evaluation data, and how designer's preferences are captured. This is shown in Figure 2.1.

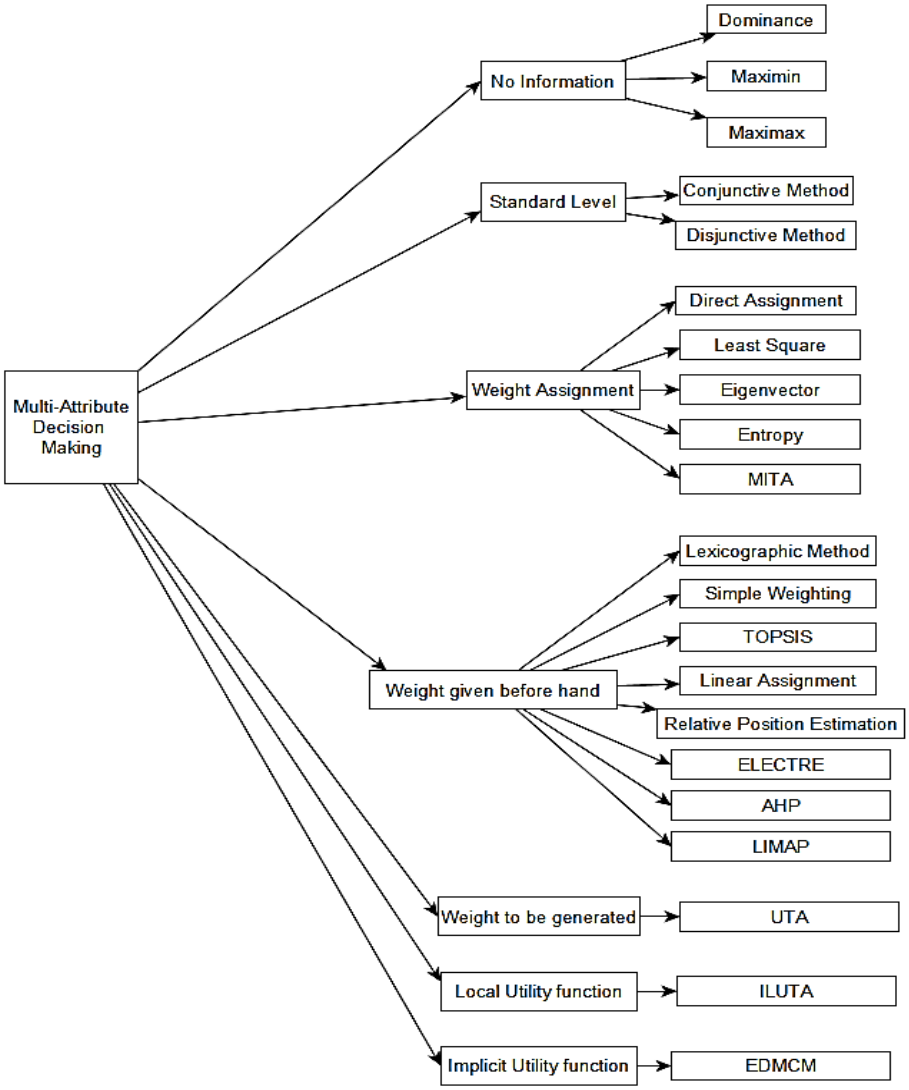


Figure 2.1: Classification of Multi-Attribute Decision Making methods [15].

Before presenting different MADM and VCDM methods, it is worth to first define a set of desirable properties that are required for a design alternative selection method. This in turn can be used as an evaluation criteria to assess different methods. The thesis examines different methods based on Arrow's general possibility theorem, and from an economic perspective. The core of Arrow's general possibility theorem is formed by three properties [46].

1. *Unrestricted domain:* States that the values that can be taken up by the criterion and the decision function should be unrestricted, implying that alternatives should not be excluded based upon irrelevant constraints [46].

2. *Pareto Optimality:* Pareto optimality defines a state in which it is impossible to make one attribute better without making at least one other attributes worse off.

3. *Independence of Irrelevant Alternative (IIA):* States that the rank order of a given alternative should not depend on the alternative set i.e. if A is ranked before B from the alternative set $\{A, B\}$ then A should still be ranked before B when a new alternative is introduced to the set $\{A, B, C\}$. Dependence of rank order on the alternative set implies that different outcomes are possible by varying the alternative set.

The design of an engineering system should also be considered from an economic perspective. Typically decisions made in regards to the acquisition of a system, allocation of resources, and manufacturing of components. These factors are strongly influenced by the economic worth of a system.

2.3.1 Analytical Hierarchy Process (AHP)

AHP was originally developed by Saaty [47] as a means of assisting the decision maker in aggregating multiple system objectives. AHP uses the judgment of the decision maker to decompose the problem into hierarchies of desirable system objectives that are assigned a set of weights by pairwise comparisons. These comparisons can be taken to reflect the relative strength of the preferences of the decision maker, which in turn are used as a means of ranking different system architectures. The modelling of a decision problem using AHP can be decomposed into four steps, as follows:

Step 1. Define the decision problem and state the goals or objectives.

Step 2. Define the criteria or factors that influence the objectives.

Step 3. Construct a pairwise comparison matrix, where each element in the upper level is compared against the elements in the level immediately below. The calculation of the weight scales are based on simple matrix algebra. The relative weight scale for each criterion derived from the pairwise comparison matrix is calculated by solving:

$$Aw = \begin{matrix} A_1 \\ \vdots \\ A_n \end{matrix} \begin{bmatrix} \frac{w_1}{w_1} & \cdots & \frac{w_1}{w_n} \\ \vdots & \ddots & \vdots \\ \frac{w_n}{w_1} & \cdots & \frac{w_n}{w_n} \end{bmatrix} \begin{bmatrix} w_1 \\ \vdots \\ w_n \end{bmatrix} = n \begin{bmatrix} w_1 \\ \vdots \\ w_n \end{bmatrix} \tag{2.1}$$

where A is the comparison matrix, w is a vector of pairwise comparisons, and n is the dimension of the matrix i.e. the total number of criteria being assessed.

This is a formulation of an eigenvector problem. The calculated weights are exact for a consistent matrix, and since the principal eigenvector is normalised (also called priority vector) the sum of weights exactly equals 1.

$$\sum w_i = 1 \tag{2.2}$$

Step 4. Rank the design alternative set based the relative weights vector obtained from the previous step. The design alternative with the highest value is the most favourable.

AHP has attracted much interest in industry and academia, mainly due to its simple mathematical properties and the fact that relatively few inputs are required to construct the model. As a decision support tool AHP offers a multi-level hierarchical structure of objectives, criteria, sub-criteria, and alternatives, resembling that of a decision tree. This allows the formulation of a decision-making problem at multiple system levels [48].

However, the application of AHP as a value model is limited due to the drawbacks present in the formulation and construction of the AHP model. One of the most crucial steps in AHP is pairwise comparisons, which can be described as a one-to-one mapping of the decision criteria. The mapping technique assumes that the decision criteria are independent of each other, and hence does not capture the decision makers view of the interactions between multiple decision criteria i.e. many-to-many mapping. The method also suffers from rank reversal of alternatives, depending on the number of alternatives being assessed. This implies that the rank order is dependent on the alternative set, violating Arrow's IIA property.

The output of the AHP process is a set of weights that identifies the decision maker preference to each decision criteria. Since preferences of the decision maker were captured using binary preference scales (the Saaty scale), the resulting output is also binary. Suggesting that the preferences of the decision maker to the decision criteria are linear. However, literature in the field of decision theory has identified that preferences of an individual or a group is generally non-linear [27, 39, 50]. For a more in-depth description on the limitations of using AHP in engineering applications the reader is referred to Traintaphyllou and Mann [48].

2.3.2 Technique for Ordered Preference by Similarity to the Ideal Solution (TOPSIS)

TOPSIS is a useful technique for ranking a set of alternatives and selecting the best solution by using distance measures. Hwang and Yoon [51] proposed that the ranking of alternatives is based on the shortest Euclidean distance from the positive ideal solution, and the farthest from the negative ideal solution. The chosen solution should thus be close to the positive ideal solution as possible, and as far from the negative ideal solution as possible. The procedure for TOPSIS can be described as follows.

Step 1. Calculate the normalised decision matrix.

A decision matrix of m alternatives and n criteria is formulated first, and then the normalised value r_{ij} is calculated.

$$r_{ij} = \frac{f_{ij}}{\sqrt{\sum_{j=1}^{m} f_{ij}^2}} \tag{2.3}$$

Where f_{ij} is the value of the i^{th} criterion function for the alternative A_j $(j = 1, \ldots\ldots, m), (i = 1, \ldots\ldots, n)$.

Step 2. Calculate the weighted normalised decision matrix.

The weighted normalised value v_{ij} is calculated as:

$$v_{ij} = w_i r_{ij} \tag{2.4}$$

w_i are the weights of the i^{th} criterion and should sum to unity.

Step 3. Determine the positive ideal and negative ideal solutions.

The positive ideal A^* and the negative ideal solutions A^- will be:

$$\begin{aligned} A^* &= \{v_1^*, \ldots\ldots, v_n^*\} = \left\{\left(\max_j v_{ij} \middle| i \in I'\right), \left(\min_j v_{ij} \middle| i \in I''\right)\right\} \\ A^- &= \{v_1^-, \ldots\ldots, v_n^-\} = \left\{\left(\min_j v_{ij} \middle| i \in I'\right), \left(\max_j v_{ij} \middle| i \in I''\right)\right\} \end{aligned} \tag{2.5}$$

Where I' refers to the 'benefit' criteria and I'' refers to the 'cost' criteria.

Step 4. Calculate the separation distance.

The separation of each alternative from the ideal solution, using the Euclidian distance, is given as:

$$\begin{aligned} D_j^* &= \sqrt{\sum_{i=1}^{n} (v_{ij} - v_i^*)^2} \\ D_j^- &= \sqrt{\sum_{i=1}^{n} (v_{ij} - v_i^-)^2} \end{aligned} \tag{2.6}$$

The D_j^* and D_j^- terms represent the distance from the positive ideal and negative ideal solutions.

Step 5. The relative closeness to the ideal solution is calculated.

$$C_j^* = \frac{D_j^-}{(D_j^* + D_j^-)} \tag{2.7}$$

This metric is used to rank the alternative set and the highest ranking alternative is selected.

TOPSIS offers a simple process that can be easily implemented for consistent decision-making, and has been applied in various engineering design problems [52]. In fact, TOPSIS is a utility based method, in that it provides an aggregate measure of benefits derived from a set of outputs. However, unlike MAUT, TOPSIS assumes mutual additive (preferential) independence, which limits its capability in capturing human decision-making [38]. TOPSIS, like AHP, applies a simple linear weight assignment to each attribute and does not capture the non-linearity of the decision makers preference structure. Furthermore, TOPSIS does not offer a weight allocation method, nor a consistency checking process for judgments like AHP does. But unlike AHP, TOPSIS alleviates the requirement of pairwise comparisons, allowing it to be applied to cases where a large number of attributes are used, and cases where quantitative data is given [52]. However, both methods suffer from rank reversal based on the alternative set, hence violating Arrow's IIA property. In addition TOPSIS assumes a direct and unlimited compensation among all attributes in its distance measure [15]. Such compensations ignore important features of design with respect to their attributes, resulting in a solution that is dominated by another design with better average features with regards to all attributes [15]. This violates Arrow's Pareto optimality property.

2.3.3 Quality Function Deployment (QFD)

Although not mentioned in Figure 2.1, QFD is a well-established approach in systems engineering for requirements prioritisation and relating stakeholder needs to technical requirements. The process involves transforming the customer requirements ("What's") into product attributes ("How's"). This transformation can be further extended and applied at multiple levels of system decomposition [18]. The use and application of QFD has been explored and reviewed extensively and has gained much popularity in its application in industry over the years [53–55]. The elements involved in the enumeration of information in the QFD house of quality matrix are depicted in Figure 2.2.

The process starts with the identification of customer requirements at a given level of system decomposition, which are placed in the left-hand corner of the house of quality matrix. Next, similar to AHP, weighting are assigned to each customer requirement based on their relative importance (*Importance Ranking*) with respect to each other. Top-level system attributes are listed on the top of the matrix. The direction of improvement of each attribute is quantified by using an upward arrow, downward arrow, and zero, which denotes maximise, minimise, and a target value to be met. Another correlation concerning the interaction between different system attributes is captured at the top of the house of quality matrix. This correlation indicates weather an increase in the value of a system attribute will cause an increase, decrease, or no effect on another system attribute. A similar correlation matrix is also built for system requirements.

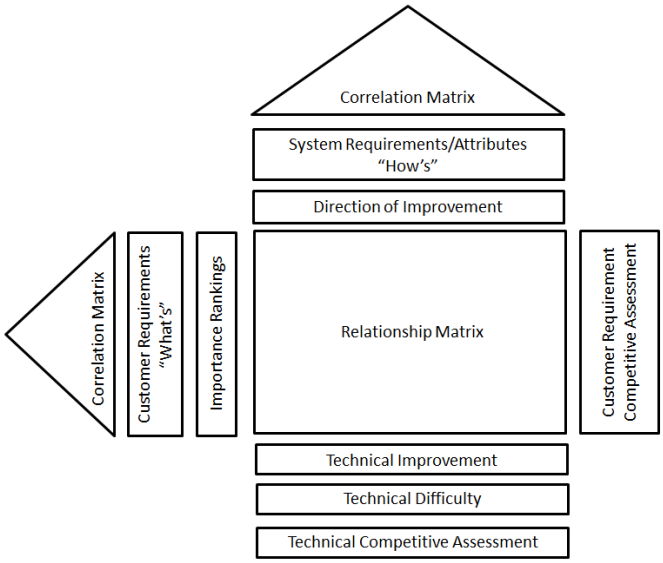


Figure 2.2: Main elements of the QFD house of quality matrix.

The central box in the house of quality matrix indicates the importance of each attribute in relation to meeting different system requirements. This is captured using a discrete scale that indicates the strength of the relationship. The values inputted in the central relationship matrix are used to calculate the technical importance of each attribute, which are estimated by summing all the customer requirement relationship values for that attribute. These are presented at the bottom of the house of quality matrix. The technical difficulty matrix for each system attribute is assessed using a binary scale and also documented at the bottom of the matrix. The combination of technical difficulty and technical improvement provides a measure of technical risk, which in-turn identifies the key design drivers that may require close attention [18].

The process just described requires substantial subject matter judgments to be made by the decision maker, making the enumeration of data tedious if a large number of customer requirements or system attributes are to be incorporated [53]. Also, the process of assigning weightings and correlation strengths between attributes and requirements in general cannot be assessed by precise values, as the information available to make precise judgments at the conceptual design stage is limited. This introduces vagueness in the weightings and correlations assigned. The presence of vagueness has been addressed by fuzzy set theory [53], and second-order polynomial models to quantify the functional relationship between the customer requirements and system attributes [56]. Vanegas and Labib [57] presented a fuzzy multi-criteria decision-making procedure to find optimal target values for system attributes, whilst accounting for system constraints. However, the fuzzy logic method provides a limited gain in accuracy when used to evaluate design concepts [58]. Methods such as neural networks have been applied to tune the fuzzy relationship function in order to provide a more realistic attribute-requirement relationship. But, similar to the fuzzy set approach, the neural network requires large data samples for it to be applied effectively, which is not always available at the conceptual design stage [59].

The requirement for the enumeration of large qualitative data from the decision maker and the uncertainty present in these judgments makes the QFD approach not suitable for engineering optimisation and trade-off studies [55]. Assessing QFD based on Arrow's properties indicates that QFD suffers a similar problem to that of AHP and TOPSIS, in that the rank order is dependent on the alternative set [46]. This again violates Arrow's IIA property.

2.3.4 Concept Design Analysis (CODA) Method

The CODA method offers an enhancement to the standard QFD approach by providing a more sophisticated mapping between system requirements and design attributes [60]. The CODA method ranks different design alternatives using a dimensionless merit factor, which is generated from translating a set of design attributes into a set of non-dimensional value functions. The value functions are assessed over a range of system attribute values, and the shape of the value function is dependent on the decision maker's preference relationship in regards to maximising value. The preference relationships can be categorised into three types, maximisation, minimisation, and optimisation. This is based on Taguchi's quality loss function [61], nominal-the-best, smaller-the-better, and larger-the-better [60]. These representative value functions are presented in Figure 2.3.

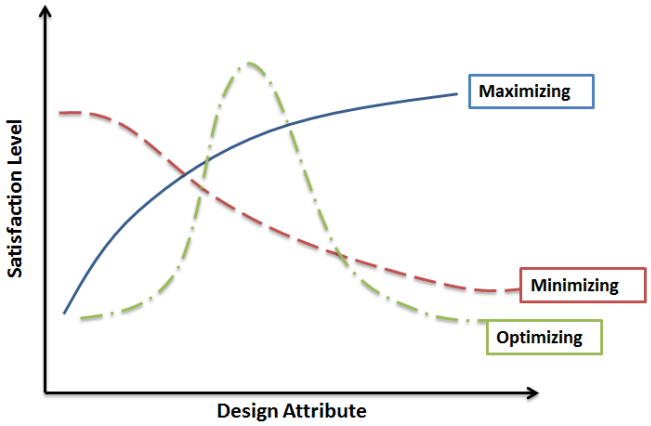


Figure 2.3: : Representative value functions for various design attributes [55]**.**

The shape of the maximisation and minimisation functions are defined by identifying a neutral point of the design parameter, which is defined as the point at which the decision maker is 50% satisfied (a value of 0.5) [55]. The optimisation function on the other hand requires two inputs, the optimum value and a tolerance value from the optimum. To account for the relative importance of different customer requirements/needs, weights are applied by using a binary weighting model. Similar to QFD, the CODA method also defines the strength of the relationship between design attributes and customer needs. These correlations can be described as being strong, medium, or weak. The overall design merit can be calculated using the following equations:

$$CS_i = \frac{N_i}{SCF_i} \sum_{j=1}^{N} MV_{ij} . CF_{ij}$$

$$ODM = \sum_{i=1}^{M} CS_i \tag{2.8}$$

where $MV_{ij} = F_{ij}(\rho_j)$ is the value function of each design attribute, CF_{ij} is the correlation matrix between design attributes and customer needs, SCF_i is sum of correlation factors for each customer need, CS_i is the satisfaction level of each customer need, and ODM is the overall design merit.

The CODA methodology has been applied in a variety of applications, ranging from the design of medical devices (Woolley et al [60]) to civil aerospace applications (Eres et al [55]). It has provided an improvement to the standard QFD method, by reducing the number of judgments required by the decision maker and the representation of non-linear preference relationships. However, the CODA method still suffers from some of the problems present in QFD. The allocation of importance weights to customer needs is based on pairwise comparisons. This suggests that customer needs are assumed to have mutual additive (preferential) independence, which limits its capability in capturing human interactions between customer needs. Also, since the CODA method uses value functions to quantify customer preferences, it does not capture the attitude of the decision maker towards risk and uncertainty during the decision-making process.

Value functions capture the elicitation of customer preferences to the outcome of a situation, which is known with certainty. However, in engineering design attribute performance levels are often uncertain due to various sources, such as incomplete information, modelling errors, and engineering assumptions. Thus, the use of value functions in engineering design does not characterise behavioural decision-making under uncertainty [38]. This implies that the outcome of a situation, that depends on a level of uncertainty having a certain probability distribution, cannot be characterised by a value function, or by any of the previous methods described. Even though there aren't any distinct violations of Arrow's general possibility theorem by the CODA method, its limitation to account for uncertainty puts it at a disadvantage.

2.3.5 Multi-Attribute Utility Theory (MAUT)

In economic theory, utility is defined as a numerical measure of a preference relationship. Relating this to engineering systems, utility measures a benefit that a system attribute provides to a preference relationship of the stakeholder. An example of a system attribute is the mass of a system, and the preference relationship of the stakeholder could be to minimise mass. In Particular, utility theory quantifies decisions made under uncertainty, comparing this to value-functions, where decisions are made with certainty. Hence, utility functions represent the preference relationship of the stakeholder based on the uncertainty of outcomes.

To quantify uncertainty utility functions are mapped by presenting the stakeholder with a set of probabilistic lottery scenarios based on variations in system attributes. A lottery is characterised by a set of possible outcomes of a system attribute a_i, which will occur with probability P_i. This is written in a more compact notation $(a_1, P_1\,; a_2, P_2; , , ,)$, which represents a sequence of pairs of outcomes and associated probabilities. The order in which these pairs are presented to the decision maker has no particular significance. The output of this lottery scenario is a probability value, which is defined as the value at which the stakeholder feels indifferent to the outcome of the lottery scenario, known as the indifference probability value.

This indifference probability value is used to define a utility curve that denotes the preferences of the stakeholder based on the uncertainty of an outcome. The scale on the utility curve can be established by fixing a set of outcomes. Typically the scales are fixed based on the best possible outcome as $U(S) = u_{\top} = 1$, and the worst possible outcome as $U(S) = u_{\perp} = 0$ (this is referred to as the normalised utility scale). Therefore, maximising utility signifies maximising system value.

The derivation of the MAUT equation depends on the condition of independence assumed during its derivation. These assumptions refer to the way a person values the interaction between attributes. If the assumptions are acceptable during its derivation, then there exists a means of obtaining a multi-attribute utility function, whilst still accounting for the interaction between attributes. There are four main independence conditions relating to the issue of multiple objectives, these are preferential, weak-difference, utility, and additive independence. In this thesis we address two of the four assumptions, utility and additive independence, as these two conditions are important in relation to addressing uncertainty.

Utility independence defines the indifference between a lottery scenario and a certainty equivalent, and doesn't depend on the level of other attributes. This makes it possible to measure the changes in utility over one attribute independently of all other attributes. Mathematically this is defined as:

$$\begin{aligned} &\text{If,} && (a_i|a_j) \sim (a_i|a, \mathrm{P}; a_i''|a_j) \\ &\text{Implies,} && (a_i|a_j') \sim (a_i'|a_j', P; a_i''|a_j') \end{aligned} \tag{2.9}$$

Where $a_i|a_j$ indicates conditionality (which is to be read as "a_i exists when condition a_j prevails"), and a_i and a_j represent two different attributes, where $i \neq j$.

Additive Independence for a set of attributes $\{a_1, \ldots\ldots, a_N\}$ can be assumed if the preference order of lotteries do not depend on the joint probability distribution of these lotteries, but only on their marginal probability distribution [62]. This is represented mathematically as follows:

$$(a_i|a_j, P = 0.5; a_i'|a_j') \sim (a_i|a_j', P = 0.5; a_i'|a_j) \tag{2.10}$$

Note that both lotteries have an equal chance ($P = 0.5$) at either a_i or a_i' occurring, and also have an equal chance of a_j and a_j' occurring. If the above condition holds then the marginal probability distribution on each attribute a_i and a_j are the same in both lotteries, thus adhering to the additive independence condition.

Based on the conformance of the independence condition, an additive or multiplicative form of the utility function may be derived. A multiplicative form of the utility function is derived if the set of attributes conform to the utility independence condition. The multiplicative form of the utility function is defined as:

$$K \cdot U + 1 = \prod\prod (K \cdot k_i \cdot U(a_i) + 1) \text{ where } K = -1 + \prod_{i=1}^{n} (K \cdot k_i + 1) \tag{2.11}$$

where, U represents the overall utility of satisfying multiple objectives, $U(a_i)$ represents the utility of system attribute a_i, k_i is the Individual scaling factor or weighting factor for the i^{th} system attribute, and K is the normalising parameter that insures consistency between the definitions of U and $U(a_i)$.

If the additive condition holds between the attribute set, then $\sum k_i = 1$ and utility function is reduced to an additive form.

$$U = \sum_{i=1}^{N} k_i U(a_i) \tag{2.12}$$

The main advantage offered by MAUT is its capability to capture uncertainty and risk during the decision-making process. More recently MAUT has been applied in an optimisation framework in the design of space systems [63]. Ross and Rhodes [64] have adopted the MAUT method to develop a dynamic tradespace exploration approach that can be used to help quantify utility/value robustness. The approach explores the changes in expectation of value over the system life cycle, which is further decomposed in to a series of contexts with fixed value expectations known as Epochs [66-67].

However, the MAUT approach presents its own limitations. One limitation that has been previously identified, and widely debated, is the validity of the independence axiom [4], [43]. The independence axiom does not accurately conform to the observed behaviour of human decision-making under uncertainty, where the axioms of rational behaviour are not necessarily obeyed in a 'real' decision-making scenario. The independence axiom leads to the assertion that a person's preference should vary linearly with the probability of its occurrence. The controversy about this axiom arises because repeated experiments have demonstrated that people often act as if their preferences are nonlinear in probability [67]. A famous example of this behaviour, illustrating nonlinearity, is the Allais "paradox". This classic example is explained by presenting a situation where a person is asked to make a choice between a definite (or certain) fortune or gamble for a greater amount, but with a small probability of getting nothing. Kahneman and Tversky [68] showed that in this example

most people violated the independence axiom, and exhibited nonlinear probability valuations, especially towards the certainty option.

The second notable limitation of the MAUT method is associated with the operationalisation of the model, which requires judgments to be made by the decision maker on the assignment of the scaling factors, k_i. The argument relating to the allocation of values to the scaling factors is that the alternatives provided in the lottery assessment are hypothetical. Experimental evidence suggests that individuals first find it hard to comprehend lottery scenarios, and second a behaviour mismatch is noted between real and hypothetical situations [46]. Another limitation of MAUT, is its inability to aggregate the preferences of a group of individuals to an average representative individual. This is best described in Arrow's General Possibility theory, which defines the nonexistence of a social welfare function [69].

2.3.6 Net-Present Value (NPV)

The methods that have been described thus far have mainly focused on aggregating multiple objectives and attributes into a single non-dimensional cardinal scale that can be used to rank alternatives. However, these aggregation methods pose some common problems, one being that of weighting attributes and system objectives based on their relative importance. The weighting process is biased by the operating environment, the organisational structure at the time of weight allocation, and personal and political interests [46]. The weights defined are therefore biased towards a particular group and may not be representative of the true value of the system. Another problem that is identified is associated with presenting system value in a non-dimensional scale. The non-dimensional scale does not provide any descriptive insight into the benefits offered by the system architectures i.e. value is not represented as an absolute number, rather is represented in relativistic terms, which can only be used as a comparative tool. A more intuitive scale such as monetary value can provide a much clearer representation of value added to the system, as the benefits offered by the system are clearly quantifiable. The use of single criteria alternative selection methods, such as NPV (Net-Present Value) and CBA (Cost-Benefit Analysis) removes the need for weight allocation and aggregation of attributes, and presents value in a monetary scale that is more intuitive to the decision maker.

NPV is a measure of the profitability of an investment in monetary terms. NPV is employed as a measure to assess the value added to the firm (or the stakeholders), resulting from an investment in an asset (or an engineering system). NPV achieves this by accounting for the present value of all future cash flows generated by an asset over time, more specifically it represents the present value of all cash inflows minus all cash outflows [70], [71]. The present value of future cash flows is determined by defining a discount rate, which is defined by how much future cash flows are discounted in order to make it correspond to an equivalent amount today. Hence, the choice of a discount rate is a key factor in assessing the performance of an asset over time. Mathematically NPV can be defined as follows:

$$NPV = D_o + \int_{t_j}^{t_k} \frac{D(t)}{(1+r(t))^t} dt \sim D_o + \sum_{t_j}^{t_k} \frac{D(t)}{(1+r(t))^t} \sim D_o + \sum_{t_j}^{t_k} \frac{D(t)}{(1+r)^t} \quad (2.13)$$

where D_o represents the initial investment before time t_j , D_t is the net cash flow, which is revenue minus cost, at time t, and $r(t)$ is the discount rate (i.e. the rate of return on future investments) at time t.

The most inherent assumption in the use of NPV is that stakeholders perceive value only in monetary terms. For some engineering systems, especially commercial aerospace systems, this assumption holds true. Castagne et al [72] and Chung et al [45] have applied NPV as an objective function in the design of commercial aerospace systems. These studies have clearly identified that stakeholders of a commercial aerospace system perceive value in monetary terms, and have demonstrated the use of NPV in improving overall system design. However, in the presence of non-commercial aerospace systems, the perception of value is not so clear-cut. Consider the following case study by DARPA (Defense Advanced Research Projects Agency). The DARPA F6 program is a study of the feasibility of fractionated spacecraft's that are wirelessly linked [41]. The study invited industry led teams represented by Boeing, Lockheed Martin, Northrop Grumman, and Orbital Sciences, to create value models to identify the best system design from a set of system architectures. The value models defined by each team however differed, as each team perceived stakeholder value differently. Boeing defined value as being a ratio of performance over cost. Lockheed Martin defined value more in an economic context, where a price was assigned to operating time, then subtracted with cost to yield profit. Northrop defined value by capturing customer preferences by applying utility theory, and compared the gain in customer preferences (or utility) to the cost of the system. Orbital sciences developed a sophisticated model for pricing of data, based on market dynamics, to calculate the NPV of the system. The outcome of this study suggests that non-monetary value of a system cannot be captured by NPV or other monetary measures alone and requires other methods to define non-monetary value [41].

Other limitations of in the use of NPV is also found when considering the derivation of the NPV metric. The NPV calculation assumes that cash-flow, discount rate, and inflation rates can be appropriately and accurately predicted for a give system [38]. However, in reality there are high levels of uncertainty in predicting cash flows, as they are prone to fluctuations due to market uncertainties. Ross et al [38] identify that the calculation of a discount rate and cash flow require detailed market predictions to be made in a potentially volatile business environment, making it highly uncertain.

However, uncertainties in the market are to some extent addressed by applying Real Options Analysis (ROA). ROA accounts for changes in system value due to uncertainty over the life of the system by using stochastic processes to calculate the value of options. Options are used as a means to consider only the outcomes that are favourable to system value (i.e. options that maximise the NPV). In other words, ROA recognises that decision makers react to

changes in the business environment or market uncertainties, and direct the system (i.e. make changes to the system – options) to maximise system value. Justin et al [73] make use of ROA (more specifically, the study makes use of the Black-Scholes equation to price options[3]) to evaluate and optimise commercial aircraft development strategies over time. In this study several market uncertainties such as fuel prices and emission costs are introduced into the simulation. However, the use of ROA still does not address uncertainty in the calculation of cash flow over time, which has a detrimental effect in the evaluation of system value.

2.3.7 Cost-Benefit Analysis (CBA)

Cost-Benefit Analysis (CBA) is performed to determine the advantage of one alternative over another in terms of the net benefits offered (benefits minus cost) by the system. CBA has been widely used in government decision-making as a tool to evaluate the net aggregate benefits (or value) achieved from the output of the project, in comparison to the net aggregate cost of the project. The output of CBA is a monetary value that characterises the net aggregate benefits offered by the project or system under consideration. This is achieved by representing benefits[4] in monetary terms, and adjusting it for the time value of money. The assessment of benefits and costs in monetary terms draws on a verity of economic theories – welfare economics, public finance, resource economics, etc. It tries to combine these elements to form a coherent economic model. As CBA make use of the discounted cash flow methods, it suffers from similar limitations as NPV. However, CBA differs from NPV as it monetises attributes of the system regardless of their contribution to the revenue stream. For a more in-depth review of CBA the reader is referred to *Cost-Benefit Analysis: Concepts and Practice* [74].

The derivation of the CBA formula follows a similar procedure to that of NPV. The net benefits of a system are described as the sum of net benefits minus cost, where the sum is discounted at a specified discount rate (similar to the NPV calculation). The difference lies in the transfer functions that transform the benefits into a monetary value. The transfer function depends on the system under consideration and other external factors. The derivation of the transfer function is specific to a given scenario and is therefore not addressed in the current discussion. The general form of the CBA formula is defined as follows:

$$NB = \left(\sum B_o - \sum C_o\right) + \left(\sum_{t_j}^{t_k} \frac{\sum B(t)}{(1+r)^t} - \sum_{t_j}^{t_k} \frac{\sum C(t)}{(1+r)^t}\right) \tag{2.14}$$

[3] Black-Scholes model is a mathematical model that gives a theoretical estimate of the price of options. For further detail the reader is referred to *Black-Scholes and Beyond: Option Pricing Models* [147].

[4] The definition of costs in CBA not only includes actual monetary value (i.e. manufacturing costs, operating cost, etc.), but also the costs due to negative social impacts of the project (i.e. cost of emissions, cost of deteriorating human/animal health, etc.).

where B_o and C_o are the initial benefits and cost of the system, r is the discount rate (i.e. the rate of return on future investments), B_t and C_t are the benefits and cost of the system estimated at discrete points in time, and t_j and t_k represent the time intervals over which the cost-benefits are quantified.

CBA suffers from the same limitations that are present in NPV, which are associated with the choice of the discount rate, and in the inaccuracies present in calculating cash-flows over a long period of time. However, unlike NPV, CBA allows the aggregation of multiple system attributes in monetary terms. CBA can therefore be thought of as filling in the gaps between NPV and MAUT, in that CBA uses an economic approach to evaluate system value by aggregating non-monetary benefits into a monetary value. However, the major short-coming of this evaluation method lies in the transfer function that transforms non-monetary benefits into a monetary value. For example, Ferraro et al [75] make use of the CBA method to assess the overall system value in the design of a small low cost search and rescue Unmanned Aircraft System (UAS). The assessment captures system benefits by identifying the number of lives saved due to the actions of the UAS, and monetises this benefit by prescribing a monetary value to human life - defined as the statistical value of life. This value is used by government agencies to assess health and safety influences of a project. In conclusion, the monetisation of benefits requires several assumptions to be made that are not always clear and might be considered controversial [76].

2.4 Choosing an Appropriate Value Model

To identify an appropriate value model it is important to define the environment that the system is operating under, and the assumptions made on the perception of value. Once these are represented, the next step in the process is to define the assumptions and limitations of the value models that are being considered for system valuation. The underlying assumptions of the value models can be correlated against the system operating environment and the stakeholder's perception of value. This allows the decision maker to more readily identify the applicability of different value models. This methodology was adopted from Ross et al [38], where the selection of the best value model is based on minimising the number of assumption that do not correlate to the system under consideration. The table presented in Appendix A assumes that both NPV and CBA are calculated deterministically. If non-deterministic dynamic models are used, certain limitations regarding *Calculating Value*, *Stakeholder Perceptions*, and *Market Predications* can be ignored to some extent.

This thesis adheres to the assumptions made by Ross et al [38] when identifying an appropriate value model i.e. all value models are derived deterministically, where uncertainty in market predications and cash flow are not accounted for, and the perception of value does not change over time.

2.4.1 Perception of Value

The perception of value in this thesis is seen through the eyes of the engineering firm. Here the perception of value is defined as the capability to deliver a system that meets all the expectations and needs of the end-user, and the value of maximising the net worth of the engineering firm through the sale and operational use of the system. The above statement identifies two different value streams:

1. End-User value – where value is perceived as the ability of the system to meet key objectives/needs outlined by the end-user. These are commonly a combination of non-monetary and monetary benefits.
2. Engineering firm value – where value is perceived as the capability of the system to generate profit through the sale and operational use of the system.

Due to the complexities in predicating the revenue stream of a large engineering firm, this thesis limits its study to capture the value of the end-user alone. Typically, end-user value is quantified by mission success rate, where sophisticated operational simulation models are used to simulate the performance of the system in an operational environment. The measure of mission success rate is identified by the successful completion of mission objectives. However, this requires detailed information regarding the operational environment and the overall system performance, which are highly uncertain at the early conceptual design stage. For example, assessment of military systems typically requires the analysis of sortie rate, survivability, effectiveness of weapon systems, and other operational factors. Such detailed understanding of the mission environment is highly uncertain, and the time required to construct a representative modelling environment would be a significant effort.

2.4.2 Choosing an Appropriate MADM Method

Most of the multi-attribute ranking methods, such as AHP, TOPSIS, and QFD violate Arrow's IIA property, and failed to address uncertainty during the decision-making process. MAUT offers a solution to this problem by capturing the decision makers risk attitude towards uncertain outcomes, which in-turn can be used to evaluate uncertainty. However, the assumptions of independence in the derivation of the MAU model means that uncertainty in human decision-making is not accurately captured. Also, the allocation of values to scaling constants k_i makes the operationalisation of the method time consuming.

To avoid aggregating multiple attributes and allocating weighting factors or scaling constants, single criteria valuation methods are used to capture stakeholder value. These include methods such as NPV and CBA. However, the fundamental drawback with using single criteria valuation methods, such as CBA and NPV, is that value is presumed to be perceived only in monetary terms. The evaluation of different value methods are based on the validity of assumptions used to derive the value model, in comparison to the assumptions of stakeholder perceived value. The choice of the most appropriate MADM method should be the one that least violates the perceived value assumptions of the stakeholders. The

assumptions made can have a direct impact on the accuracy of value assessments, and its conformance to stakeholder-perceived value [38].

This thesis focuses on the design of a non-commercial aerospace system, where the use of the system by the customer does not result in the generation of revenue, but rather provides a service that enables the customer to benefit in other means. For example, the military acquisition of an aerospace system is representative of a non-commercial aerospace system. Since it has been established that value is not perceived in monetary terms, both NPV and CBA can be neglected. The next criterion focuses on the decision-making behaviour of the stakeholders in the design of a system. It is assumed that the design of an engineering system is made in an uncertain environment at the early conceptual design stage. Thus, capturing uncertainty of system attributes and the operating environment is vital in designing a robust system solution. Accounting for both these criteria leads to the selection of MAUT, as it least violates stakeholder assumptions.

Chapter 3: Multi-Objective Decision Making and Optimisation

Multi-Objective Decision Making (MODM) encapsulate methods that have been developed to handle Multi-Criteria Decision Making (MCDM) problems, where the "best" design is selected from a large set of alternatives within the design space. The means of selecting a "best" design follows the process of optimisation, where the aim is to maximise and/or minimise a set of desired criteria. Thus, MODM techniques are more appropriate for design applications within a continuous solution space, where the design envelope is subject to active constraints. In aerospace design the continuous solution space generally relates to geometric design variables (such as wing area, aspect ratio, engine bypass ratio, etc.), and the desired criteria generally relates to system performance and cost attributes (such as life-cycle cost (LCC), gross take-off mass (GTOM), and specific fuel consumption (SFC)). In general, these desired criteria, or objectives, tend to be conflicting in nature and the selection of an optimal design solution requires a compromise that best satisfies all the objectives. The definition of an optimal solution in this case is not intuitive and requires specifying some form of weighting or preference function that ranks competing designs.

Sen and Yang [15] define a classification for different MODM methods based on the type of preference information, and the stage at which preference information is elicited. This is represented in Figure 3.1. The focus of this thesis is mainly on prior and posterior articulation of preference information.

Prior articulation is applied to capture the preference structure of system stakeholders, such that an aggregate function consisting of multiple system objectives can be created to define an overall 'goodness' value. This 'goodness' value represents a scalar value that can be optimised to find the "best" solution to the multi-objective problem.

Posterior articulation methods require no preference or goal information from the system stakeholders prior to optimisation. Instead, posterior methods seek to identify solutions that are candidates of the Pareto optimal set.

Cardinal data refers to data associated with quantitative information, but not necessarily a preferential order. Data retrieved from engineering analysis generally relates to cardinal data, in that they are quantitative and represent physical aspects of the system. However, some information relating to the system may be ordinal in nature where alternatives are ranked

based on subjective views or opinions on the 'goodness 'of the system. For example, 'ease of use' might be qualitatively measured as being low, medium, or high – with 'high' representing very easy to use. The use of ordinal data is far less common in engineering analysis, and for this reason this chapter only elaborates upon some of the cardinal methods represented in Figure 3.1. For a more complete description of these methods the reader is referred to Sen and Yang [15].

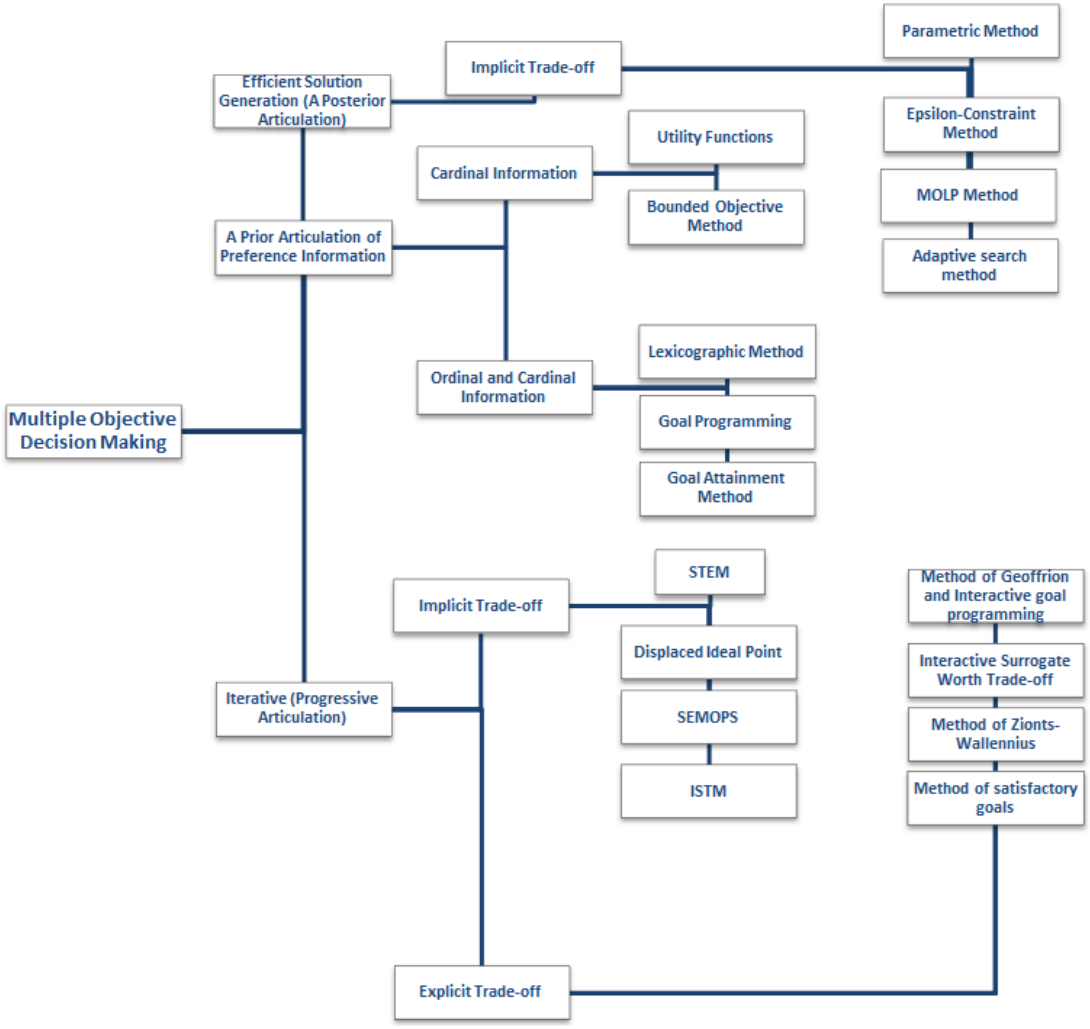


Figure 3.1: Classification of MODM methods [15].

This chapter also introduces the reader to different optimisation algorithms that have been widely used in engineering design to search for optimal solutions that are possible candidates for the Pareto set. The chapter specifically focuses on response surface based, or surrogate based optimisation methodologies that aim to identify solutions within to the Pareto optimal set, with minimal computational cost requirements.

3.1 Prior Articulation

3.1.1 Weighted-Sum Approach

This method employs a weighted sum of the objectives to derive a single objective function; where the resulting single objective problem can then be solved to obtain Pareto optimal solutions. The objective function for a problem with N objectives is given by:

$$Min \quad \mathbf{f}(\mathbf{x}) = \sum_{n=1}^{N} w_i f_i(\mathbf{x})$$

$$s.t. \quad X \in \Omega \quad W = [w_1, w_2, \dots, w_N] \tag{3.1}$$

where W is the weight vector representing the relative importance of objectives, and $f_i(\mathbf{x})$ is the i^{th} objective function. In practice each i^{th} objective function is often normalised such that the weights w_i are more directly related to the decision-makers preferences.

Though the weighted sum approach offers a simple and easy method to implement and understand the multi-objective problem, the method is not well suited for rigorous decision-making. The drawbacks of this method are as follows, an even distribution of the weights among objective functions does not always result in an even distribution of solutions on the Pareto front. The method also fails to identify solutions on the non-convex regions of the Pareto front, and will instead only choose the extremes [77]. Figure 3.2 represents the applicability of the method for a convex and non-convex objective space. Optimal solutions on the Pareto front are represented by points that lie on the hyperplane $L = \{f(X) | \sum_{n=1}^{N} w_i f_i(\mathbf{x}) = c\}$, which is tangential to the feasible space Λ, and the slope of $L = -w_1/w_2$. It can be noted that for a non-convex region, the Pareto optimal solutions between A and B are not identified by this method, regardless of the weighting vector values.

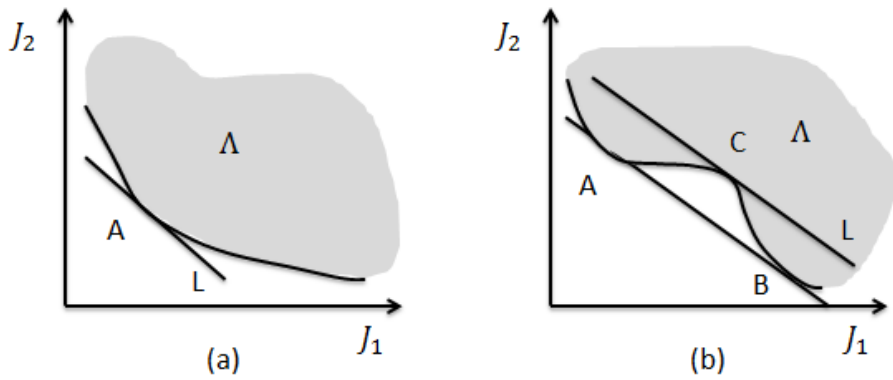


Figure 3.2: Additive weighted sum method for (a) convex region, (b) non-convex region [78].

3.1.2 Goal Programming

In certain circumstances it is preferred to search for a specific improvement, by specifying the required objective function value, rather than searching for an unknown optimum. Goal

programming is a problem formulation method that seeks to find an optimal compromise solution by minimising the deviation from the defined goals.

In most cases of goal programming the defined objectives are not "hard" constraints that the search algorithm has to meet; rather the defined goals may have some ambiguity. In this case goals are treated as "soft" constraints, which allow the search algorithm to violate the condition of meeting a specified objective value. Rather, the "best" solution can be defined to have the minimum weighted deviation from the ideal solution, where all the goals meet the originally specified value. This method also allows the decision maker to add bias to the search algorithm, in-terms of the importance of meeting certain specified objective values in comparison to others. In this case all deviations are multiplied by weights, which reflect their relative importance, and summed together to form a single objective function. The method can be formulated as follows:

$$\text{Min} \quad \left\{ \left[\sum_{j=1}^{k} \left(w_j^+ d_j^+ + w_j^- d_j^- \right)^p \right]^{\frac{1}{p}} \right\}$$

$$\text{s.t.} \quad X_b \in \Omega_b$$

$$\Omega_b = \left\{ \begin{array}{c} f_j(\mathbf{x}) - d_j^+ + d_j^- = \hat{f}_j \quad j = 1, \dots, k \\ d_j^+ \cdot d_j^- = 0 \quad \mathbf{x}_b = [\mathbf{x}^T \, d_1^+ \, d_1^-, \dots, d_k^+ \, d_k^-] \\ d_j^+, d_j^- > 0 \quad \mathbf{x} \in \Omega \end{array} \right\} \tag{3.2}$$

where d_j^- and d_j^+ are the deviation variables form the specified goal value. The variables w_j^- and w_j^+ are the relative weights for the corresponding deviation variable, and $\hat{f}_j$ is the goal of the j-th objective.

A very common form of this problem formulation follows the lexicographic goal programming approach [79]. The distinguishing feature of this approach is the existence of the number of priority levels to be specified. This leads to the following formulation:

$$\text{Min} \quad a_l = p_l h_l(D^+, D^-)$$

$$\text{s.t.} \quad \mathbf{x}_b \in \Omega_b, \;\; l = 1,2, \dots, L \tag{3.3}$$

where $D^+ = [d_1^+, d_2^+, \dots, d_k^+]^T$, $D^- = [d_1^-, d_2^-, \dots, d_k^-]^T$, and p_l represents the priority weights, and L is the number of priority levels.

The algorithm solves the above equation by first minimising a_1 to obtain a_1^*, then a_2 is minimised such that $a_2 \leq a_1^*$. This process is repeated until a_L is minimised such that the compromised design of the MODM problem X^* can be obtain. If the criteria are linear functions of $\mathbf{x}$, a modified Simplex algorithm can be used to solve this problem. Any single objective nonlinear optimisation technique can be utilised iteratively to solve the problem with nonlinear objectives.

However, the application of goal programming does present disadvantages. It is often difficult for the decision maker to set goals for all the objectives when information available is limited, and there is large ambiguity in the choice of the objective value. Other disadvantages relate to the derivation of the method itself, where it has been found that goal programming is not efficient in finding designs for non-convex problems [15]. For a more in-depth discussion on goal programming the reader is referred to Jones and Tamiz [79].

3.1.3 Utility Theory and Physical Programming

As discussed in the previous chapter, utility theory offers a powerful method to address the formulation of MODM problems. The method seeks to create a set of non-linear utility functions that completely expresses the decision makers preference structure. In addition, utility theory offers the capability to capture uncertainties in the decision-making process, which offers a probabilistic assessment of the impact of uncertainties on the value of designs. However, the formulation of non-linear utility functions is not trivial and can be error prone, as the assessment of preference information is subjective and requires a significant body of knowledge of the system operating environment to make accurate judgements.

Physical programming is closely related to goal programming; in that it can be used to create utility functions through the use of a set of class functions to represent the decision makers physical preferences. The objectives are classified in an intuitive manner from 'highly desirable' to 'unacceptable'. This allows the decision maker to quickly create single attribute utility functions, which are then additively aggregated to form a scalar objective function. The method is formulated by the following equation:

$$\begin{aligned} \text{Min} \quad & \log_{10}\left[\frac{1}{n_{sc}}\sum_{i=1}^{n_{sc}} \overline{f}_i[f_i(\mathbf{x})]\right] \\ \text{s.t.} \quad & \Omega \end{aligned} \tag{3.4}$$

where Ω represents the design space, and n_{sc} is the number of soft constraints.

However, physical programming also represents drawbacks in its application. In the problem formulation phase the decision maker is still required to specify weightings to each of the class functions. Secondly, physical programming is a deterministic design method that does not capture the uncertainties in the decision-making process.

3.2 Posterior Articulation

The methods defined in posterior articulation aim to determine non-dominated solutions, of which the most desirable is selected based on some defined criteria that reflects the judgements of the decision maker. This means that trade-off information is used after the non-dominated solutions have been found. Hence, these techniques do not require any assumptions or information regarding the preference structure prior to optimisation, which can be advantageous. A typical method of identifying the Pareto optimal set is by applying

successive optimisation runs using one of the prior aggregation methods, and different weight vectors. This technique works by varying the values of the weight vector, resulting in each optimisation run to converge to a different point on the Pareto front. However, the problem posed with this approach results in the need for multiple optimisation runs, which can cause the computational expense of the problem to grow out of control. Another issue to this approach is related to the distribution of solutions along the Pareto front, where it can be mathematically proven that a well distributed set of weight vectors does not necessarily result in a good distribution of point along the Pareto front.

3.2.1 Normal Boundary Intersection (NBI) and Normal Constraint (NC) Method

Das and Dennis [80] proposed the NBI method to find a uniform spread of solutions on the Pareto front, resulting in an even distributed set of points on the Pareto set, given an evenly distributed set of weights. The idea behind this approach is defined by the following observations, the intersection point between the normal emanating from any point on the convex hull of individual minima[5] and the boundary of the objective space is probably a Pareto optimal point i.e. the point of intersection closest to the utopia point is a Pareto minimal point, while the one furthest is a Pareto maximal point. This approach is formulated as follows:

$$\begin{aligned} &\text{Min} \quad \lambda \ \mathbf{x} \in \mathbf{X}, \lambda \\ &\text{s.t.} \quad \mathbf{\Phi w} + \lambda \mathbf{n} = \mathbf{F(x)} - \mathbf{F}^o \end{aligned} \tag{3.5}$$

where $\mathbf{\Phi}$ is a $k \times k$ pay-off matrix in which the ith column is made up of the vector $\mathbf{F}(\mathbf{x}_i^*) - \mathbf{F}^o$, in which $\mathbf{F}(\mathbf{x}_i^*)$ is the vector of the objective functions evaluated at the minimum of the ith objective function and $\mathbf{F}^o$ represents the utopia point. The diagonal elements of $\mathbf{\Phi}$ are zeros, $\mathbf{w}$ is a vector of scalars such that $\sum_{i=1}^{k} w_i = 1$. $\mathbf{n} = -\mathbf{\Phi e}$, where $\mathbf{e} \in R^k$ is a column vector of ones in the objective space, and $\mathbf{n}$ is called the quasi-normal vector. Given a convex weighting $\mathbf{w}$, $\mathbf{\Phi w}$ represents a point in the convex hull of individual minima, and $\mathbf{\Phi w} + \lambda \mathbf{n}$ represents the set of points on that normal. The points of intersection between the normal and the objective boundary closest to the utopia point is represented by the solutions of equation (3.5). As $\mathbf{w}$ is modified systematically, the solutions obtained yields an even distribution of Pareto optimal points. However, the method may also yield non-Pareto optimal points, but Das and Dennis describe this as not necessarily being a disadvantage, as it results in a smoother approximation of the Pareto boundary [80].

[5] Convex hull of individual minima: Let x^* be the representative of the optimum variable for the global minimum of $f_i(x), i = 1, \dots, k$. Let $F_i^* = F_i^*(x_i^*), i = 1, \dots, k$. Then the set of points in R^k that are convex combinations of F_i^* is referred to as the convex hull of individual minima.

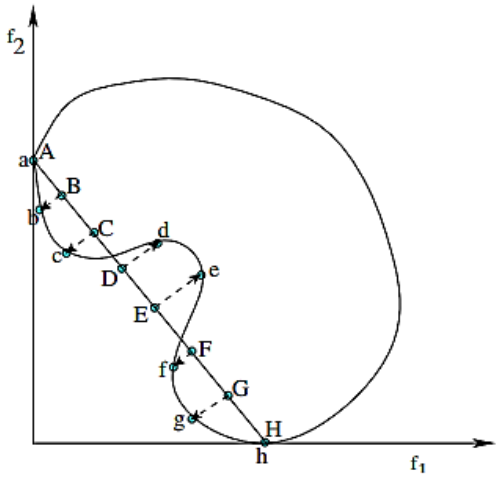


Figure 3.3: Representation of solution obtained using the NBI method.

The normal constraint (NC) method provides an alternative to the NBI method with some improvements [81]. In this method the utopia line/plane is first found and its components are used to normalise the objectives. A sample of evenly distributed points are generated on the utopia plane by projecting each sample point onto the Pareto optimal surface by solving a constrained optimisation problem for each of these points. However, similar to NBI, the method described thus far may generate dominated solution points. To mitigate this, a Pareto filter is applied to search for and delete any dominated solutions, which is done by comparing each solution point with other solution points and all dominated points are discarded [81].

3.2.2 Multi-Objective Genetic Algorithms

The methods presented thus far have involved unique formulations that are solved using standard single-objective optimisation methods. Approaches such as genetic algorithms (GA) can be tailored to solve a multi-objective problem directly. GA is a type of evolutionary algorithm (EA) that mimics the evolutionary process in nature for optimisation. The algorithm can be defined by using the following operators, *representation*, *fitness assignment*, *selection*, *crossover,* and *mutation*. A GA achieves this by first generating a set of randomly generated states called the population, where each state is represented using a binary string of zeros and ones. The generated states of the population are assigned a fitness value based on the objective function value at that point. The fitness values of the states are used by the selection operator to select candidates for mating in order to generate the next generation of states. The goal of the selection operator is to create more copies of designs with relatively high fitness, while diminishing the number of weak candidates. New candidate solutions are generated by applying the crossover operator. The goal of the crossover operator is to create new candidate solutions (child solutions) from members of the mating pool. The crossover operator picks two candidate states at random from the mating pool, and a crossover point is chosen randomly (which is based on the crossover probability) from the positions in the binary string i.e. the binary string of each pair to be mated are exchanged to generate a new

state or child. Finally, each location of the binary string in the child state is subject to mutation, which is dependent on the mutation probability. For a binary coded GA the mutation operator flips a bit in the string at a random position i.e. the mutation operator, with a given mutation probability, changes an arbitrary bit in the genetic sequence from its original state. The mutation operator, similar to the crossover operator but not to the same extent, aims to maintain genetic diversity from one generation of a population to the next.

By applying the three operators, *selection*, *crossover*, and *mutation*, at each generation of the optimisation process the GA increases the number of optimal solutions i.e. solutions with higher fitness values. The *crossover* and *mutation* operators combine features of the high fitness solutions of the previous generation in the hope of creating better solutions in the next generation. The application of this process over multiple generations will in theory lead to better solutions and converge towards the Pareto optimal set.

This evolutionary optimisation approach can be extended to a multi-objective optimisation problem. Of the many evolutionary algorithms proposed for multi-objective optimisation, the most popular and widely used is the modified non-dominated sorting genetic algorithm (NSGA-II) [32]. The following sub-section provides a brief over of NSGA-II. For a more complete survey of other multi-objective evolutionary algorithms, the reader is referred to the text by Deb [32].

3.2.2.1 Non-dominated sorting genetic algorithm

The application and development of EAs for multi-objective optimisation problems have been vast, of the many proposed EAs the most popular are the non-dominated sorting genetic algorithm (NSGA) and the improved strength Pareto evolutionary algorithm (SPEA). Goldberg [82] describes the development of a new method of multi-objective fitness assignment known as non-dominated sorting. The procedure of fitness assignment, as illustrated in Figure 3.4, calculates the rank of each individual according to the number of other fronts that dominate the front that it is a member of.

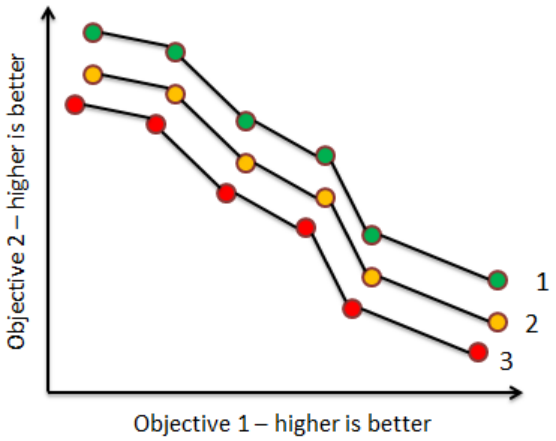


Figure 3.4: Principle of non-dominated sorting. Solutions belonging to dominated fronts are assigned successively inferior fitness values.

For a more comprehensive review of the applicability of these methods the reader is referred to Steuer [32]. This thesis mainly focuses on the NSGA procedure, which will be described here after.

The NSGA algorithm is similar to the standard GA apart from the way the selection operator works. The non-dominated individuals are first identified from the current population and assigned a large dummy fitness value. In order to maintain diversity in the population, these dummy fitness values are then degraded through the use of sharing [83], where the fitness value is degraded by dividing the dummy value by a quantity σ_{share} proportional to the number of individuals around it. The non-dominated solutions are temporarily ignored and the procedure is repeated for the remaining individuals, using a smaller assigned dummy fitness value at each step. This process is continued until the entire population is classified into several fronts [83]. The algorithm proceeds in the same fashion as the GA, with the selection operator applied, where selection preference is given to individuals with large dummy fitness values.

However, several criticisms of the NSGA algorithm emerged, the most prominent of which were its computational complexity, lack of elitism, and for choosing the optimal parameter value for the sharing parameter σ_{share}. This led to the development of a modified version of the algorithm, NSGA-II proposed by Deb et al. [84], which is an improvement of the earlier proposed NSGA algorithm. NSGA-II incorporates elitism, better sorting algorithm, and requires no sharing parameter to be chosen.

NSGA- II works by first classifying the population into non-dominated sets. The first set contains all the non-dominated solutions of the population, and the second front being dominated by the individuals of the first front and so on. In this way the entire population (which includes the parent and child population) is classified into different Pareto sets by assigning a non-dominated rank (fitness), where a rank of 1 is assigned to all members of the first non-dominated set, rank of 2 for the second, and so on. In addition to fitness values, a new parameter called crowding distance is calculated for each individual, which measures how close an individual is to its neighbours. A larger crowding distance will result in a more diverse population. The algorithm continues to select parents from the population by using binary tournament selection, based on the rank and crowding distance of individuals. Here, individuals are selected if the rank is lesser than the other, or if the crowding distance is greater than the other (note, crowding distance is only compared if the rank for both individuals are the same).

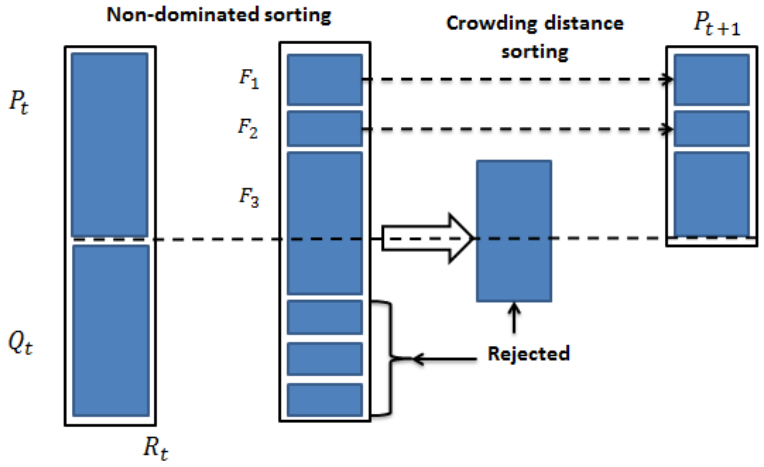


Figure 3.5: Schematic of the NSGA-II procedure [32]**.**

The outline of the NSGA- II algorithm is represented in following steps:

Step 1. Create a parent and offspring population and combine both to generate $R_t = P_t \cup Q_t$. Then, perform a non-dominated sorting of R_t and identify different fronts F_i $i = 1,2, \ldots,$ etc.

Step 2. Perform archive update to generate $P_{t+1}$of size N i.e. until $|P_{t+1}| + |F_1| < N$, perform $P_{t+1} = P_{t+1} \cup F_i$ and $i = i + 1$.

Step 3. Perform crowding-sort procedure to include only the most widely spread solutions using the crowding distance values in the sorted F_i to P_{t+1}.

Step 4. Perform selection on P_{t+1}, binary tournament selection with replacement to fill the mating pool by using the crowding distance.

Step 5. Apply crossover and mutation operators to the mating pool from the previous step to generate Q_{t+1}.

3.2.3 Surrogate Assisted Optimisation

The idea behind surrogate assisted optimisation is to use approximation techniques to speed up the design optimisation procedure, by accelerating the search procedure of a variety of optimisation methods, including gradient-based and EAs. This was first studied by Schmit, Farshi, and Miura [85], [86], which focused on structural optimisation using mathematical programming techniques. Surrogate assisted methods can be described as a technique that allows complex and computationally expensive optimisation problems to be transformed into simpler, less time intensive, functions that can be adapted into an optimisation framework. The surrogated assisted methods can in essence be described as a curve fitting technique that allow the 'real' objective or constraint function to be replaced with an approximation of that function. The central theme behind such techniques is to identify an appropriate approximation function $\hat{y} = \hat{f}(\mathbf{x}, \alpha)$, where α is a vector of unknown parameters that are determined either by a black-box based approach or a physics based approach [13]. The physics-based approach aims to exploit the governing equations (either the continuous or

discrete form of the governing equations) in order to determine the functional form $\hat{y} = \hat{f}(\mathbf{x}, \alpha)$. In contrast, the black-box approach involves running the analysis code at a number of preselected input points $\mathbf{x} = \left[\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots, \mathbf{x}^{(n-1)}\right]^{\mathrm{T}}$, where the sample plan is generated by applying techniques developed in design of experiments (DoE). The analysis code is then run at these points to generate a set of outputs $\mathbf{y} = \left[y^{(1)}, y^{(2)}, \ldots, y^{(n-1)}\right]^{\mathrm{T}}$, resulting in the generation of a set of input-output data. This data set is in-turn used to tune the model parameters to fit a surrogate model $\hat{f}$ to create a mapping of the objective function.

This thesis primarily focuses with the black-box approach, which offers the practical advantage of not modifying the analysis code to determine the α parameters. The overall back-box approach follows an iterative procedure involving the following steps: (1) data generation, (2) model-structure selection, (3) parameter estimation, and (4) model validation. The application of this approach results in the construction of a surrogate model for each objective function. The constructed surrogates are in-turn used in a search scheme, such as a genetic algorithm (GA), to construct the Pareto set. This thesis adopts the framework developed by Voutchachkov et al. [87], where the surrogate based approach is combined with NSGA-II, and an update strategy. The update strategy allows the identification of infill points in regions of interest, such that the model parameters of the surrogate model can be re-tuned in order to effectively identify the Pareto set.

The following subsection provides an overview of the Gaussian Process or Krig predictor surrogate model, as it forms the basis for the optimisation and update point selection method applied in this thesis. However, the topic of surrogate model construction and parameter estimation is vast. For a more complete review of the methods available, the reader is referred to MacKay [88], Box and Draper [89], and Keane and Nair [13].

3.2.3.1 Gaussian Process Surrogate Model

Gaussian Process (GP) modelling originates from the work by Krig [90], who developed the method to predict mineral concentrations, in the area of geostatistics. The use of GP models allows the user to control the amount of regression, and also provides a statistical framework for assessing the accuracy of the model predictor. This in-turn can be used in assessing where to place any future update points to the surrogate model.

In the GP model the function outputs $y(\mathbf{x})$ is treated as a random variable, in the sense that the model output in unknown until the model is run at that particular point. In essence the model outputs are assumed to be realisations of the Gaussian random field with mean β and covariance Γ. The model structure can thus be represented as:

$$Y(\mathbf{x}) = \beta + Z(\mathbf{x}) \tag{3.6}$$

where $Z(\mathbf{x})$ is a Gaussian stochastic process with zero mean and covariance given by:

$$\text{Cov}\,(Z(\mathbf{x},\mathbf{x}') = \Gamma(\mathbf{x},\mathbf{x}') = \sigma_z^2 R(\mathbf{x},\mathbf{x}') \tag{3.7}$$

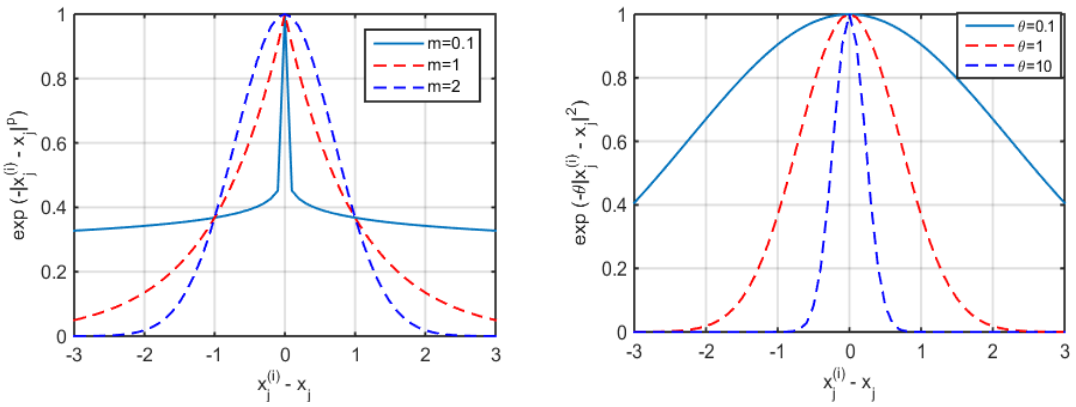
Here σ_z^2 represents the process variance and $R(\mathbf{x},\mathbf{x}')$ is a parameterised correlation function that can be tuned to the training data set. A common choice of the correlation function is:

$$R(\mathbf{x},\mathbf{x}') = \exp\left|-\sum_{j=1}^{n}\theta_j\left|x_j - x_j'\right|^{m_j}\right| \tag{3.8}$$

where $\theta_j \geq 0$ and $0 < m_j \leq 2$ are the undermined hyperparameters. From this an $n \times n$ correlation matrix of the observed data set can represented as:

$$\boldsymbol{R} = \begin{bmatrix} R(\mathrm{x}^{(1)},\mathrm{x}^{(1)}) & R(\mathrm{x}^{(1)},\mathrm{x}^{(2)}) & \cdots & R(\mathrm{x}^{(1)},\mathrm{x}^{(n)}) \\ R(\mathrm{x}^{(2)},\mathrm{x}^{(1)}) & R(\mathrm{x}^{(2)},\mathrm{x}^{(2)}) & \ldots & R(\mathrm{x}^{(2)},\mathrm{x}^{(n)}) \\ \vdots & \vdots & \ddots & \vdots \\ R(\mathrm{x}^{(n)},\mathrm{x}^{(1)}) & R(\mathrm{x}^{(n)},\mathrm{x}^{(2)}) & \ldots & R(\mathrm{x}^{(n)},\mathrm{x}^{(n)}) \end{bmatrix} \tag{3.9}$$

Figure 3.6 represents different correlations due to variations in m and θ. Here m represents the 'smoothness' parameter. Setting $m_j = 2$ results in the correlation to be 'smooth' and continuous, and infinitely differentiable. Reducing the value of m_j increases the rate at which correlation drops as $\left|x_j^{(i)} - x_j\right|$ increases. The hyperparameter θ_j controls the influence of the sample point as $\left|x_j^{(i)} - x_j\right|$ increases. Forrester et al. [34] describe θ_j as being a measure of how 'active' the function we are approximating is due to variations in x_j i.e. a high value of θ_j indicates that all points have a high correlation, while a low value of θ_j indicates a significant difference between the $Y(x_j)$'s θ_j.



(a) (b)

Figure 3.6: Correlations with varying (a) m and (b) θ.

Since the observed outputs $\mathbf{y} = \left[y^{(1)}, y^{(2)}, \ldots, y^{(n)}\right]^{\mathrm{T}}$ are assumed to be reflections of the Gaussian random field, the joint distribution of the set of data is also a joint normal distribution. Consequently the likelihood of the observed data being generated by the parameterised Gaussian random field (represented as a log of the likelihood function) is:

$$\ln\left(L(\boldsymbol{\theta}, \beta, \sigma_z^2)\right) = \frac{1}{2}\left|n\ln(2\pi) + n\ln\sigma_z^2 + \ln|\boldsymbol{R}| + \frac{1}{\sigma_z^2}(\mathbf{y} - \mathbf{1}\beta)^T\boldsymbol{R}^{-1}(\mathbf{y} - \mathbf{1}\beta)\right| \tag{3.10}$$

where $\boldsymbol{\theta}$ is a vector of the $\theta_i, i = 1, \dots n$, hyperparameters, $\boldsymbol{R} \in \mathbb{R}^{n\times n}$ is the correlation matrix, and $\mathbf{1}$ represents a matrix of ones – $[1,1,\dots.,1]^{\mathrm{T}} \in \mathbb{R}^n$. In order to estimate the hyperparameters $\boldsymbol{\theta}, \beta$, and σ_z^2 the likelihood function is maximised, which is equivalent to minimising the negative log of the likelihood function. By taking derivatives of the above equation and setting them to zero, the hyperparameters β and σ_z^2 can be estimated as follows:

$$\beta = \frac{\mathbf{1}^T\boldsymbol{R}^{-1}\boldsymbol{y}}{\mathbf{1}^T\boldsymbol{R}^{-1}\mathbf{1}} \tag{3.11}$$

$$\sigma_z^2 = \frac{(\boldsymbol{y} - \mathbf{1}\beta)^T\boldsymbol{R}^{-1}(\boldsymbol{y} - \mathbf{1}\beta)}{n} \tag{3.12}$$

These estimates of the hyperparameters can be substituted back into equation (3.10) and the constant terms removed to give the *concentrated ln-likelihood function:*

$$\ln(L) \approx -\frac{n}{2}\ln(\sigma_z^2) - \frac{1}{2}\ln|\boldsymbol{R}| \tag{3.13}$$

The value of this function is dependent on the hyperparmeter. However, unlike β and σ_z^2, differentiating equation (3.10) with respect to $\boldsymbol{\theta}$ does not yield an analytical solution for the estimate of $\boldsymbol{\theta}$**.** Instead, some form of numerical optimisation method is employed to calculate the values of $\boldsymbol{\theta}$**.** Typically a GA or simulated annealing is employed as an optimisation algorithm to generate the best result [34].

Having determined the hyperparameters that maximise the likelihood function, new function values can be predicted at unobserved data points $\hat{\mathbf{x}}$. A new prediction $\hat{y}$ is augmented with the observed data to form a joint normal distribution, which can be written as follows:

$$\begin{vmatrix}\boldsymbol{y}\\ \hat{y}\end{vmatrix} \sim \boldsymbol{N}\left(|\mathbf{1}_{n+1}\beta|, \begin{vmatrix}\boldsymbol{\Gamma} & \gamma(\hat{\mathbf{x}})\\ \gamma(\hat{\mathbf{x}})^T & \boldsymbol{\Gamma}(\hat{\mathbf{x}}, \hat{\mathbf{x}})\end{vmatrix}\right) \tag{3.14}$$

where $\gamma(\hat{\mathbf{x}}) = [\sigma_z^2 R(\hat{\mathbf{x}}, \mathbf{x}^{(1)}),\ \sigma_z^2 R(\hat{\mathbf{x}}, \mathbf{x}^{(2)}), \dots, \sigma_z^2 R(\hat{\mathbf{x}}, \mathbf{x}^{(n)})]^T$. The prediction of $\hat{y}$ can now be determined by taking the conditional distribution of $\hat{y}$ given the data set $\boldsymbol{y}$ i.e. the posterior probability distribution, which is given as:

$$\hat{y}|\boldsymbol{y} \sim \boldsymbol{N}\left(\beta + \gamma(\hat{\mathbf{x}})^T\boldsymbol{\Gamma}^{-1}(\boldsymbol{y} - \mathbf{1}\beta), \gamma(\hat{\mathbf{x}})^T\boldsymbol{\Gamma}^{-1}\gamma(\hat{\mathbf{x}})\right) \tag{3.15}$$

Given the above, the posterior mean and posterior covariance can be written as:

$$\hat{y}(\mathbf{x}) = \beta + \boldsymbol{\tau}(\mathbf{x})^{\boldsymbol{T}}\mathbf{R}^{-1}(\boldsymbol{y} - \mathbf{1}\beta) \tag{3.16}$$

where $\boldsymbol{\tau}(\mathbf{x}) = \left[R\left(\mathrm{x}, \mathrm{x}^{(1)}\right), R(\mathrm{x}, \mathrm{x}^{(2)}), \dots, R(\mathrm{x}, \mathrm{x}^{(n)})\right]$, which represents the correlation of new points. The mean of the posterior distribution is seen as the prediction of the output at the new

unobserved point $\hat{\mathbf{x}}$, and the variance of the output at this point is seen as the uncertainty in the prediction of the output.

$$\sigma^2(\mathrm{x})=\sigma_z^2\,(1-\boldsymbol{\tau}(\mathbf{x})^T\mathbf{R}^{-1}\boldsymbol{\tau}(\mathbf{x})) \tag{3.17}$$

3.2.3.2 Multi-Objective Expected Improvement

Having constructed an initial surrogate model using the sampled data points, $\mathbf{x}=\left[\mathbf{x}^{(1)},\mathbf{x}^{(2)},\ldots,\mathbf{x}^{(n-1)}\right]^{\mathrm{T}}$, it is worth considering adding further infill points to enhance the accuracy of the model in predicting the true objective function f. The GP model allows the calculation of mean-square error $\hat{s}^2(\mathbf{x})$ of the model prediction, which in-turn allows it to estimate the uncertainty of the prediction and thus define an infill strategy. The infill points are located where the mean-square error is maximum. However, a pure error based infill strategy leads to filling the gaps in the design space and not exploring to find any minima's within the objective space. A balanced exploration-exploitation aims to minimise $\hat{y}(\mathbf{x})$ and at the same time minimise mean-square error $\hat{s}^2(\mathbf{x})$ of the model prediction. This can be achieved via the incorporation of statistical improvement criteria, which attempts to balance exploration and exploitation. This section provides details of the probability of improvement and expected improvement criteria. For a more in-depth review of other alternatives the reader is referred to Keane and Nair [13]. The probability of any infill point representing an improvement over the current best point $f_e^{\min}(\mathbf{x})=\min\left(f_e^{(1)}(\mathbf{x}^{(1)}),f_e^{(2)}(\mathbf{x}^{(2)}),\cdots,f_e^{(N_0)}(\mathbf{x}^{(N_0)})\right)$ is given as $P[I]=P\left[y\left(\mathbf{x}^{(N_0+1)}\leq f_e^{\min}(\mathbf{x})\right)\right]$. In a multi-objective sense this can be considered as the probability that the new design point will dominate a single member of the existing Pareto set, this concept is illustrated in Figure 3.7. Here, the solid line represents the Pareto front, the shaded region represents the region in which the new point will extend the Pareto set, and the hatched region represents the region in which at least one set member [34].

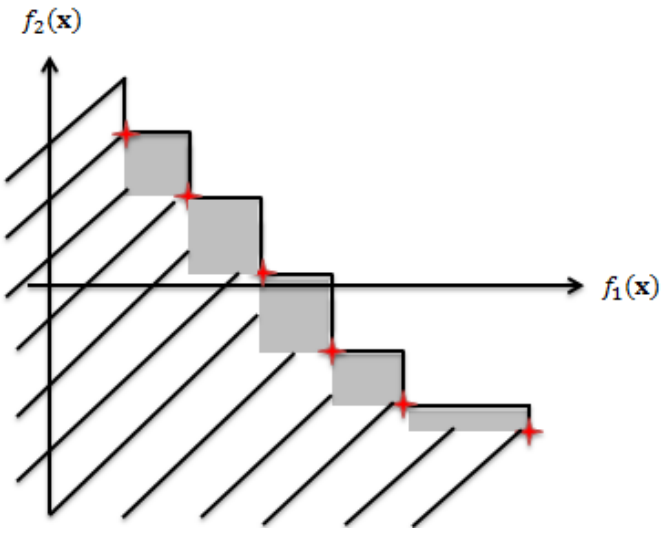


Figure 3.7: A Pareto set illustration for a problem with two objectives [34].

For a two-dimensional problem with two objectives $f_1(x)$ and $f_2(x)$, and x consisting of only one design variable. The initial Pareto set can be constructed from the DoE runs, which are represented as:

$$f_{1,2}^{*} = \left\{\left[f_1^{(1)*}(x^{(1)}), f_2^{(1)*}(x^{(1)})\right], \left[f_1^{(2)*}(x^{(2)}), f_2^{(2)*}(x^{(2)}), \ldots, \left[f_1^{(M_0)*}(x^{(M_0)}), f_2^{(M_0)*}(x^{(M_0)})\right]\right]\right\} \quad (3.18)$$

The superscript * indicates that the designs are non-dominated. Here it is assumed the GP models built for the two objective are independent, where the mean $\hat{y}_1(x)$ and $\hat{y}_2(x)$ and variance $\hat{s}_1(x)$ and $\hat{s}_2(x)$ are used to construct a joint Gaussian probability density function. The probability that a new design point at x will dominate a single member of the existing Pareto set is given to be $P\left[Y_1(x) < y_1^{*(i)} \cap Y_2(x) < y_2^{*(i)}\right]$, which is given by integrating over the hatched area in Figure 3.7.

$$\begin{aligned} P\left[Y_1(x) < y_1^{*(i)} \cap Y_2(x) < y_2^{*(i)}\right] & \\ &= \int_{-\infty}^{y_1^{*(1)}} \int_{-\infty}^{\infty} Y_1 \phi(Y_1, Y_2) dY_2 dY_1 \\ &+ \sum_{i=1}^{m-1} \int_{y_1^{*(i)}}^{y_1^{*(i+1)}} \int_{-\infty}^{y_2^{*(i)}} Y_1 \phi(Y_1, Y_2) dY_2 dY_1 \\ &+ \int_{y_1^{*(m)}}^{\infty} \int_{-\infty}^{y_2^{*(m)}} Y_1 \phi(Y_1, Y_2) dY_2 dY_1 \end{aligned} \quad (3.19)$$

where $\phi(.)$ represents the joint Gaussian probability density function. However, the defined criteria does not explicitly define the degree of improvement being achieved and thus does not necessarily encourage exploration when used as an infill criterion. This is addressed by the expected improvement criterion. The derivation of the expected improvement criterion is taken from Forrester et al. [34].

The expected improvement $E[I(\mathbf{x})]$ is the first moment of the integral over the area below and to the left of the front, and taken about the Pareto front. $P[I(\mathbf{x})]$ represents the integral over the probability density function, with limits below and to the left of the Pareto front where the improvement can occur. The calculation of the centroidal distance of the expected improvement integral from the Pareto front $(\bar{Y}_1\ \bar{Y}_2)$ is determined by $E[I(\mathbf{x})]$ (which is the integration with respect to its origin) divided by $P[I(\mathbf{x})]$. The expected improvement criterion, based on any location on the front, can now be calculated by measuring the Euclidian distance from the centroidal location $(\bar{Y}_1\ \bar{Y}_2)$ to the closest set member of the Pareto front from the centroid $(y_1^*(x), y_2^*(x))$ and multiplying it by $P[I(\mathbf{x})]$.

$$E[I(\mathbf{x})] = P[I(\mathbf{x})]\sqrt{(\bar{Y}_1(x) - y_1^*(x))^2 + (\bar{Y}_2(x) - y_2^*(x))^2}, \quad (3.20)$$

where

$$\bar{Y}_1(x) = \left\{ \int_{-\infty}^{y_1^{*(1)}} \int_{-\infty}^{\infty} Y_1 \phi(Y_1, Y_2) dY_2 dY_1 + \sum_{i=1}^{m-1} \int_{y_1^{*(i)}}^{y_1^{*(i+1)}} \int_{-\infty}^{y_2^{*(i)}} Y_1 \phi(Y_1, Y_2) dY_2 dY_1 + \int_{y_1^{*(m)}}^{\infty} \int_{-\infty}^{y_2^{*(m)}} Y_1 \phi(Y_1, Y_2) dY_2 dY_1 \right\} / P[I(\mathbf{x})] \quad (3.21)$$

and $\bar{Y}_2(x)$ is defined similarly. The expected improvement criterion encourages a more evenly spread Pareto front distribution, as the centroidal positions lying near a large gap between the points forming the Pareto front will get a proportionally higher value of $E[I(\mathbf{x})]$ since the Euclidian distance to the nearest point will be greater. In addition, the $E[I(\mathbf{x})]$ criterion also encourages a more wide spread exploration if the data points used to construct the GP model are widely spaced. This results in the error terms to be large, which tends to further increases exploration.

3.3 Optimisation Framework

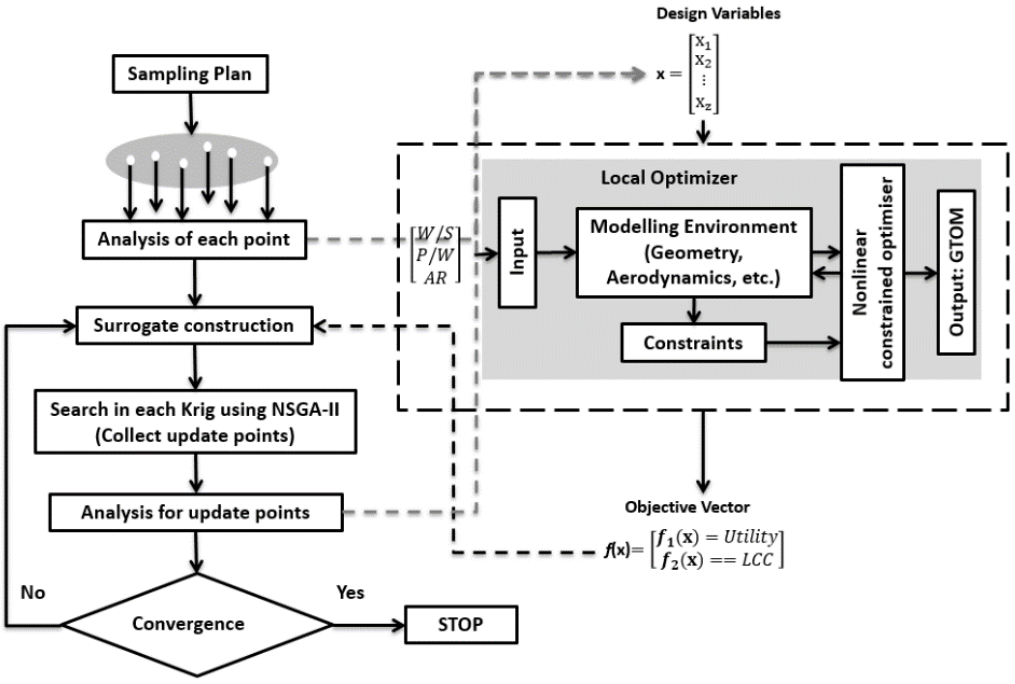


Figure 3.8: Optimisation Framework.

This thesis adopts the framework developed by Voutchachkov et al. [87], where the surrogate based approach is combined with the NSGA-II multi-objective optimisation algorithm, and an update strategy to identify infill points in regions of interest. This allows the model parameters of the surrogate model to be re-tuned to effectively identify the Pareto optimal set. This study uses a Gaussian Process (or Kriging) response surface model to fit the sample data set. The expected improvement criterion is used to define the update strategy, as it

provides a balanced exploitation and exploration strategy of the objective space [34] i.e. it provides a balance between optimality and design space uncertainty. The optimisation framework also applies a local nonlinear constraint optimiser to effectively act as a constraint satisfier. This is used to scale the wing area, engine power-to-weight ratio, and wing aspect ratio to meet a defined set of constraints, and with the aim of minimising GTOM (Gross Take-off Mass). The overall optimisation framework is depicted in Figure 3.8.

3.3.1 Constrained Optimisation

Before defining the optimisation procedure it is worth describing the optimisation problem being addressed. Here, the optimisation problem concerns two objectives, one being the maximisation of the utility function, and the other being the minimisation of the Life-Cycle Cost (LCC) function. The optimisation problem is also subject to inequality constraints $g_i(\mathbf{x}), i = 1, \cdots, n$ (but not equality constraints). The details describing the constraint functions and design variables $\mathbf{x}$ used in this optimisation problem will be described in later chapters. However, in a general form the optimisation problem can be defined as follows:

$$\begin{aligned} &\text{Minimise} && \mathrm{f}(\mathbf{x}) = [-U(\mathbf{x}), LCC(\mathbf{x})] && \\ &\text{Subject to} && g_i(\mathbf{x}) \leq 0. && i = 1, \cdots, n \\ & && \mathrm{x}_q^L \leq \mathrm{x}_q \leq \mathrm{x}_q^U && q = 1, \cdots, z \end{aligned} \tag{3.22}$$

In the above formulation a negative sign is added to the utility function in order to convert it to a minimisation problem, which is equivalent to maximising the positive utility function. The inequality constraint represents the constraint boundaries that must be satisfied. In addition each design variable $\mathrm{x}_q \in \mathbf{x}$ can also be subject to upper and lower bound constraints, defined as $\mathrm{x}_q^L \leq \mathrm{x}_q \leq \mathrm{x}_q^U$. The satisfaction of these constraint functions are addressed by a local nonlinear constraint optimiser, which aims to solve the following sub-problem:

$$\begin{aligned} &\text{Minimise} && f(\mathbf{x}) && \\ &\text{Subject to} && g_i(\mathbf{x}) \leq 0. && i = 1, \cdots, n \\ & && \mathrm{x}_q^L \leq \mathrm{x}_q \leq \mathrm{x}_q^U && q = 1, \cdots, z \end{aligned} \tag{3.23}$$

In the above formulation a single objective function $f(\mathbf{x})$ is defined, which is not the same as the multi-objective formulation in equation (3.22). This single objective function can be representative of the systems performance i.e. mass, total efficiency, platform signature, etc.

The above single objective constraint problem can be written as an unconstrained minimisation problem by defining the Lagrangian function ($\mathcal{L}$):

$$\mathcal{L}(\mathbf{x}, \lambda) = f(\mathbf{x}) + \sum_{j=1}^{n} \lambda_j g_j(\mathbf{x}) \tag{3.24}$$

Or

$$\mathcal{L}(\mathbf{x}, \boldsymbol{\lambda}) = f(\mathbf{x}) + \boldsymbol{\lambda}^T \mathbf{g}(\mathbf{x}) \tag{3.25}$$

where $L(\mathbf{x},\lambda)$ is the Lagrangian function, and λ_j is the j^{th} Lagrange multiplier, and the vector $\boldsymbol{\lambda} = [\lambda_1, \lambda_2, \dots, \lambda_n]^T$ are vectors of the Lagrange multipliers for the inequality constraints. The addition of Lagrange multipliers combines the constraints to yield an unconstrained optimisation problem. The above formulation also requires the optimisation algorithm to differentiate between active and inactive constraints, where a constraint $g_i(\mathbf{x})$ is said to be active at a point $\mathbf{x}^*$ if the point falls on the constraint surface i.e. $g_i(\mathbf{x}^*) = 0$, else the constraint is defined to be inactive. Based this information, the first-order optimality condition, known as the Karush-Kuhn-Tucker (KKT) conditions, for a point $\mathbf{x}^*$ to be a local minimum of equation (3.24) can be defined as follows:

If $\mathbf{x}^*$ is a local solution to equation (3.23), the function $f(\mathbf{x})$ and $g_i(\mathbf{x})$ are continuously differentiable, and the set of active constraints at the point $\mathbf{x}^*$ are linearly independent, then there exists Lagrange multiplier vector $\boldsymbol{\lambda}$ such that the following conditions are satisfied:

$$\begin{aligned}
&\nabla_{\mathbf{x}} \mathcal{L}(\mathbf{x}^*, \boldsymbol{\lambda}) = 0 \\
&g_j(\mathbf{x}) \leq 0. && j = 1, \cdots, n \\
&\lambda_j \ g_j(\mathbf{x}) = 0 && j = 1, \cdots, n \\
&\lambda_j \geq 0 && j = 1, \cdots, n
\end{aligned} \tag{3.26}$$

Where $\nabla_{\mathbf{x}} \mathcal{L}(\mathbf{x}^*, \boldsymbol{\lambda}) = \nabla f(\mathbf{x}) + \sum \lambda_j \nabla g_j(\mathbf{x})$. The above conditions implies that either λ_j is equal to zero, or the constraint $g_j(\mathbf{x})$ is active – defined by $\lambda_j \ g_j(\mathbf{x}) = 0$. Several non-linear constrained optimisation algorithms are available to solve for the Lagrangian function and satisfy the first order optimality conditions i.e. the KKT conditions. These algorithms can generally be grouped into: quadratic programming, penalty and augmented Lagrangian method, linearized search techniques, cutting plane method, feasible direction method, and sequential quadratic programming (SQP) method. The SQP methods are one of the most effective methods in solving the non-linear constrained optimisation problem, and will be discussed in further detail in this section. For an overview of the other methods the reader is referred to Bonnas et al. [91].

The Lagrangian function is solved by applying the sequential quadratic programming (SQP) method, which solves a quadratic sub-problem in each iteration. A quadratic sub-problem of equation (3.23), at the current point $\mathbf{x}_k$ has the following form:

$$\begin{aligned}
&\text{Minimise} && \tfrac{1}{2} \mathbf{d}_\mathbf{k}^\mathbf{T} \mathbf{B}_k \mathbf{d}_\mathbf{k} + \nabla f(\mathbf{x}_k)^T \mathbf{d}_\mathbf{k} \\
&\text{Subject to} && \nabla \mathbf{g}(\mathbf{x}_k)^T \mathbf{d}_\mathbf{k} + \mathbf{g}(\mathbf{x}_k) \leq 0
\end{aligned} \tag{3.27}$$

where $\mathbf{d_k} = \mathbf{x} - \mathbf{x}_k$ and $\mathbf{B} = \mathbf{I}$ (identity matrix) initially, which is then updated to approximate the Hessian matrix $\mathbf{H}$. This sub-problem can now be solved using a quadratic programming (QP) algorithm. The methods to solve QP problems can be divided into three groups: active-set, gradient projection, and interior-point methods. In the presence of inequality constraints the SQP method solves a sequence of quadratic sub-problems iteratively, where the objective function is quadratic in form and the constraints are linear. This however is not appropriate when the constraint functions are non-linear. To account for non-linearity of the constraints, while using the linearized form in the quadratic sub-problem, the SQP method chooses a quadratic form of the Lagrangian as the objective function, hence $\nabla f(\mathbf{x}_k) = \nabla_{\mathbf{x}} \mathcal{L}(\mathbf{x}_k, \boldsymbol{\lambda}_k)$. Active set methods work by maintaining the estimate of the inequality constraints that are active in each iteration, where the active inequality constraints can be posed as equality constraints, resulting in quadratic sub-problem. However, in the active set method the active constraints change in each iteration, requiring a large number of iterations to reach convergence. The gradient projection method on the other hand allows the active constraint set, otherwise known as the active set, to change rapidly from iteration to iteration. Each iteration of the gradient projection algorithm consists of two stages. In the first stage the algorithm searches along the steepest descent direction. If a constraint boundary is encountered the search direction is 'bent' to remain in the feasible region. Thus, the search is made along the piecewise linear path to locate the first local minimiser, called the Cauchy Point. In the second stage the active set is updated to the inequality constraints that are active at the Cauchy point. The QP sub-problem is solved with this new working set at the Cauchy point to obtain the next iterate [91].

Interior point method aims to solve a sequence of approximate minimisation problems. This method converts the inequality conditions to equality conditions by introducing slack variables. In addition, logarithmic terms called barrier functions are added to the objective function, which results in the following problem definition:

$$\begin{aligned} \text{Minimise} \quad & \tfrac{1}{2}\mathbf{x}^T\mathbf{G}\mathbf{x} + \mathbf{x}^{\mathbf{T}}\mathbf{p} - \mu \textstyle\sum_{k=1}^{m} \log s_k \\ \text{Subject to} \quad & \mathbf{a}_j^T\mathbf{x} = b_j \qquad\qquad j = 1,2,\dots,l. \\ & \mathbf{c}_k^T\mathbf{x} - d_k - s_k = 0 \qquad k = 1,2,\dots,m. \end{aligned} \tag{3.28}$$

where s_k are the slack variables corresponding to the inequality constraints, and μ is a barrier parameter. As μ approaches zero, the solution of the above problem approaches the solution of the QP problem, as specified in equation (3.27).

The solution to the quadratic sub-problem, from any one of the methods defined thus far, is used to form the following new iteration:

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{d}_k \tag{3.29}$$

where α_k is the step length parameter, which is determined in order to produce a sufficient decrease in a merit function $\phi(\mathbf{x})$. In this method the objective function and constraint functions are combined together in each iteration of the algorithm to form an unconstrained optimisation problem. The merit function used by Han [92] and Powell [93] are of the following form, which is widely used in many SQP implementations:

$$\phi(\mathbf{x}) = f(\mathbf{x}) + \sum_{i=1}^{m} r_i \cdot g_i(\mathbf{x}) + \sum_{i=m+1}^{l} r_i \cdot \max[0, g_i(\mathbf{x})] \tag{3.30}$$

Powell recommends setting the penalty parameter r_i as:

$$r_i = (r_{k+1})_i = \max\left\{\lambda_i \frac{(r_k)_i + \lambda_i}{2}\right\}, \quad i = 1, \ldots, m. \tag{3.31}$$

This allows positive contributions from constraints that are inactive in the QP solution, but were recently active.

The iterative scheme also requires updating the Hessian matrix, or an approximation of the Hessian matrix, after each iteration. The Hessian matrix is typically updated by applying the BFGS method [91]. The BFGS updating method, named after Broyden, Fletcher, Goldfarb, and Shanno, constructs the Hessian at each major iteration as follows:

$$\mathbf{H}_{k+1} = \mathbf{H}_k + \frac{q_k q_k^T}{q_k^T s_k} - \frac{\mathbf{H}_k s_k s_k^T \mathbf{H}_k^T}{s_k^T \mathbf{H}_k s_k} \tag{3.32}$$

where

$$s_k = \mathbf{x}_{k+1} - \mathbf{x}_k$$

$$q_k = \nabla_{\mathbf{x}+1} \mathcal{L}(\mathbf{x}_{k+1}, \boldsymbol{\lambda}) \tag{3.33}$$

Based on the information provided thus far, the overall SQP method can be defined as a series of steps:

Step 1. Start with an approximation of $(\mathbf{x_0}, \boldsymbol{\lambda_0})$ and $\mathbf{B_0}$. Set $k = 0$ and define the merit function $\phi(\mathbf{x})$.

Step 2. Formulate the quadratic programming sub-problem, as defined in equation (3.27), at the current iteration $\mathbf{x}_k$. Solve the quadratic programming sub-problem to obtain $\mathbf{d_k}$ from the solution.

Step 3. Choose the step length α_k by solving the unconstrained optimisation problem of minimising the merit function along $\mathbf{d_k}$ i.e. $\phi(\mathbf{x}_k + \mathbf{d_x})$.

Step 4. Calculate the new iterate by using equation (3.29).

Step 5. If the solution is converged, stop.

Step 6. Update $\mathbf{B}_k$using the BFGS algorithm to obtain $\mathbf{B}_{k+1}$.

Step 7. Increment $k = k + 1$ and go to **Step 2**.

The SQP algorithm in this thesis is applied as a constraint satisfier, and a local single objective optimiser for minimising system mass in Chapter 7.

3.3.2 Multi-Objective Optimisation Procedure

Having presented an overview of both surrogate modelling and nonlinear constrained optimisation algorithms in this chapter, the overall optimisation framework, as defined in Figure 3.8, is outlined. The overall optimisation process can be broken down in the following steps:

Step 1. Define a sampling plan to generate a set of points, $\mathbf{X} = \{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \mathbf{x}^{(3)}, \dots, \mathbf{x}^{(n)}\}$, within the design space where simulation program will be run.

Step 2. Run the simulation at each specified point from step 1 to calculate the output response $\mathbf{y} = \{y^{(1)}, y^{(2)}, \dots, Y^{(n)}\}^T$. This generates an input-output relationship, which is stored in a dataset: $\mathcal{D}^0 \equiv \{\mathbf{x}^{(i)}, y^{(i)}\},\ i = 1, \dots, n$. Set the update counter $\kappa = 0$.

Step 3. Construct the GP surrogate model using the dataset $\mathcal{D}^0$ and tune the hyperparameters.

Step 4. Search the GP surrogate model of the utility function $\widehat{U}(\mathbf{x})$ and LCC function $\widehat{LCC}(\mathbf{x})$, using the expected improvement criterion in order to identify update points $\mathbf{x}^{\dagger}$.

Step 5. Select designs that are best in terms of ranking and space filling properties. See Section 3.2.2.1.

Step 6. Evaluate the selected designs and add to the data set $\mathcal{D}^{\kappa} \equiv \{[\mathbf{x}^{(i)},\ \mathbf{x}^{\dagger}],\ [\,y^{(i)}, y^{\kappa}]\}$.

Step 7. Produce Pareto front and compare with previous. If two or three consecutive Pareto fronts are identical then stop, otherwise continue.

Step 8. Repeat step 3 and re-tune the hyperparameters with the updated data set $\mathcal{D}^{\kappa}$.

Chapter 4: Architectural Exploration

This chapter builds on the concept of systems architecting, as described in Chapter 1, and identifies the need for architectural decision-making in engineering design. Hazelrigg [4] defines three key elements in the decision-making process as identification of options or choices, development of expectations on the outcomes of each choice, and formulation of system values for ranking different system architectures. This is best illustrated through Figure 4.1, which shows the mapping of stakeholder objectives to functional intent of the system, describing what the system has to do in order to satisfy the objectives. This is then mapped to the physical architecture of the system, defined by system elements. The mapping between system elements can be representative of many forms, such as information flow, physical interface (mechanical or electrical), compatibility relationships, etc. This thesis represents the mapping between system elements as compatibility constraints, which is representative of the viability of system architectures.

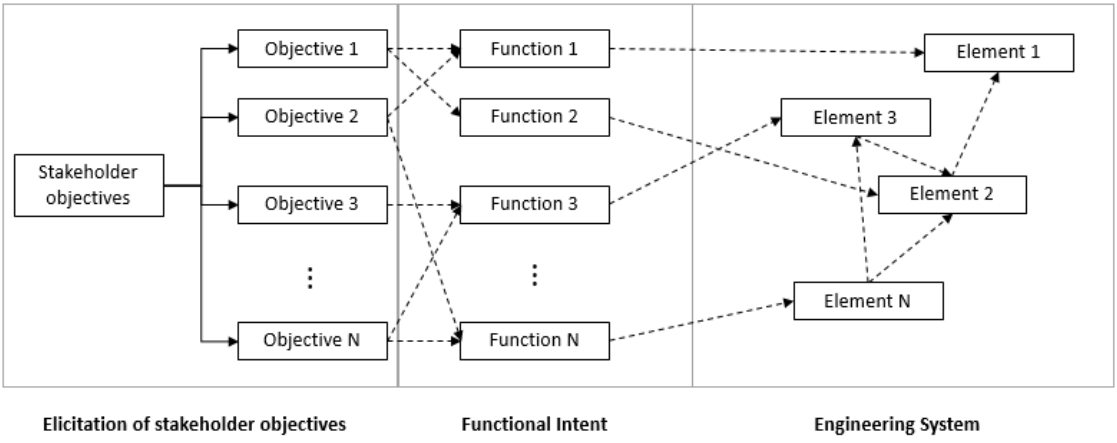


Figure 4.1: Functional view of the system architecting process.

The assessment of stakeholder objectives has already been addressed in previous chapters, where value assessment was discussed in great detail. This chapter addresses the translation of functional intent into a physical system architecture, which encompasses the decision-making process of defining a set of viable design alternatives that are candidate solutions for the system under consideration. The decision-making process in systems architecting can be defined as identifying a set of viable design alternatives, based on knowledge and experience of the design team, and down-select a design to generate a final system architecture. This chapter provides a literature review of methods that are available to assist the architectural exploration and down-selection process. Based on the literature review a

set of architectural exploration methods are selected that best meet the needs of architectural decision-making.

4.1 Architectural Decision-Making

System architecting can be viewed as a decision-making process, pertaining to Simon's four stages of decision-making: *Intelligence*, *Design*, *Choice*, and *Implementation*. Thus, to address the decision problem, the decision maker needs effective methods for each of the four stages. According to Simon [16, 26] the most central and time-consuming part in the architecting process is the *Design* phase. Even if *Intelligence*, *Choice*, and *Implementation* activities have perfect solutions, the *Design* activity is still potentially the most difficult and time-consuming phase of the decision-making process. As the number of system architectures increase the design effort required to explore the architectural tradespace space increases significantly. Therefore, the thesis focuses on addressing the needs of the *Design* phase, as it is considered to be the most resource intensive of all the four phases.

In order to evaluate the use of different methods and tools within the *Design* phase, three common metrics are used as a means of evaluation – *construction, search*, and *representation*. These aspects were identified through a survey of decision support literature [14, 27].

1. *Construction* – Allows the decision maker and system modeller to construct the design problem that is human understandable, and can also be encoded into a programmable environment.

2. *Search* – Allows the capability for reasoning about the structure of the decision problem itself. This includes the capability to effectively search the architectural tradespace space, based on the constraints imposed by the decision maker.

3. *Representation* – Provides a means of representing the outputs of the model such that the behaviour and benefits of the system are clearly highlighted to the decision maker. This can be thought of as knowledge representation.

The following section outlines some of the methods used to aid the system architecting process in the design phase.

4.2 Table and Matrix based Methods

4.2.1 Morphological Matrix

Morphological Analysis was originally developed by an American astrophysicists Zwicky [23]. It is used to facilitate the rigorous investigation of large multidimensional spaces by enumerating the possible alternatives for each decision variable, which is represented in a matrix format known as the morphological matrix. An extension of the morphological matrix is Functional Means Analysis (FMA) [5], which flows the logic of function to element of form

in order to create a system architecture. Burge [5] describes FMA as a being a highly structured approach in generating, selecting, and documenting system design concepts. For example, in the design of a civil airliner some of the system functions are defined as, Deliver electrical power, Generate lift for steady flight, Provide necessary thrust, and, Maintain stability and control. Each function is then allocated a set of design alternatives within its domain.

However, the inclusion of many alternatives into the matrix can result in a large number of system configurations, which can become unmanageable for the system modeller to evaluate, given the limited availability in time to explore the solution space, and the budgetary constraints imposed on the exploration process at the conceptual design stage. The morphological matrix, as described by Mavris [18], is an inherently a static document and does not lend itself very well in its basic formulation for electronic design reviews, where decisions are dynamically challenged and rectified on site [18]. Engler et al [94] have made use of the morphological matrix and addressed its shortfalls to develop a process known as Interactive Reconfigurable Matrix of Alternatives (IRMA). IRMA is used to instantiate architectural solutions in an interactive form by allowing it to be dynamically reconfigured during the concept selection process [18].

4.2.2 Decision Support Matrix

The Design Structure Matrix (DSM) provides a means of mapping one element of a system to another. The mapping of system elements can be physical connections, information flow (input–output) connections, or constraint connections between two or more elements. The DSM is represented by an $n \times n$ matrix, where the matrix representation generally tends to be symmetric. DSM has been widely used in systems engineering to identify interfaces between different sub-systems or components. In this way DSM can support the top-down identification of interfaces at each system level.

Function 1 (F1)	$F1 \rightarrow F2$	$F1 \rightarrow F3$	$F1 \rightarrow F4$	$F1 \rightarrow F5$
$F2 \rightarrow F1$	**Function 2 (F2)**	$F2 \rightarrow F3$	$F2 \rightarrow F4$	$F2 \rightarrow F5$
$F3 \rightarrow F1$	$F3 \rightarrow F2$	**Function 3 (F3)**	$F3 \rightarrow F4$	$F3 \rightarrow F5$
$F4 \rightarrow F1$	$F4 \rightarrow F2$	$F4 \rightarrow F3$	**Function 4 (F4)**	$F4 \rightarrow F5$
$F5 \rightarrow F1$	$F5 \rightarrow F2$	$F5 \rightarrow F3$	$F5 \rightarrow F4$	**Function 5 (F5)**

Figure 4.2: A representative example of a DSM matrix.

In systems architecting DSM is used to capture the interrelationship between system elements. These interrelationships are used to identify the compatibility of solutions between one decision variable to another, by defining the compatibility or constraint rules between

decision variables. This results in a reduction in the number of possible combinations of system architectures. The design effort required to explore the architectural tradespace space is now more manageable [18]. However, DSM matrix on its own fails to meet the *construction* and *search* criteria, as it does not allow the decision maker to construct the design problem and identify different design alternatives that are available. DSM is therefore only considered as a *representation* tool to identify the interconnectivity between decision variables. Using structural reasoning algorithms, the matrix representation can be sorted to show sets of elements that are tightly interconnected. This in-turn can also be used to identify clusters of closely interconnected elements, and separate them from other clusters in order to create sub-problems that can be solved independently of each other.

4.3 Taxonomies and Classification Hierarchies

4.3.1 Ontologies

A taxonomy defines a hierarchical classification or categorisation of systems, in reference to a means of organising concepts of knowledge. In a broader sense this comes under the field of knowledge organisation, which represents all schemes for organising information content and knowledge management. The term knowledge organisation encompasses classification schemes that organise material at a general level, headings that provide more detailed access to sublevels, and control variant versions of key information content. This includes schemes, such as semantic networks and ontologies. An ontology is defined as a general description of each concept (or class), where the properties associated with each concept describes various features and attributes of that concept (properties that belong to a class), and the restrictions associated on those properties. An ontology, together with a set of individual instances of classes, can be regarded as a knowledge base [95]. For example, an ontology of the system architecting process can be created by defining decision variables as classes. These classes are then arranged in a hierarchical taxonomy (subclass – superclass). The design alternatives belonging to each decision variable (or classes) are defined through properties. The relationships between different decision variables (or different instances of classes) are defined by describing allowed values for properties of each class. In recent years new tools and languages have emerged for the purpose of providing the capability in supporting the development of ontologies, and provide technical support for knowledge-driven systems. Ontology toolkits such as Protégé [95] have allowed the exploration of different ontology languages, such as XML, RDF and OWL. For a more in-depth review of these different ontology languages the reader is referred to Allemang and Hendler [96].

The complexity of designing and manufacturing engineering systems often requires the understanding of technological advancements, design constraints, best practice knowledge, customer requirements, and cost management [97]. This requires knowledge from multiple domains to be brought together to support the decision-making process. Ahmed et al. [98] have developed an methodology to support the creation of ontologies for the purpose of searching, indexing, and retrieving engineering design knowledge. The developed

methodology has allowed researchers and industry practitioners to create ontologies to aid engineering decision-making [99, 100]. Van Ruijven [99] has also presented an ontology for systems engineering by describing a set of information models. An illustration of this is represented in Figure 4.3. This developed ontology represents an interpretation of the systems engineering process, which in-turn can be used build or configure information systems to support the systems engineering process [99].

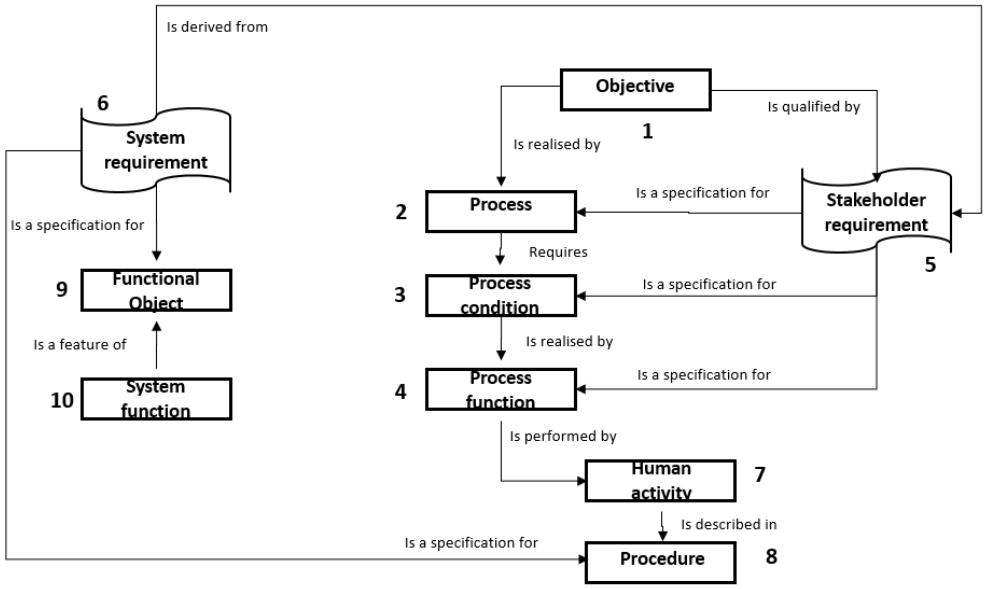


Figure 4.3: Illustration of an information model of the breakdown of the systems design process [99].

However, the development of ontologies in engineering systems thus far have been domain specific. But to address the system architecting process it is required that ontologies be modular, such that concepts can be generalised into separate ontologies [97]. This should allow for better flexibility, modularity, and maintainability of the knowledge content [97]. The adoption of ontologies in engineering design is also limited due to limited availability of approaches or standard protocols to follow for ontology development, and no practical methods for supporting engineering decision-making [100], [101]. Even in the case where a generic and modular ontology framework is available, the effort required by system modellers to enumerate and maintain the ontology is significant, especially for system architectures consisting of large hierarchical structures [97]. For this reason the adoption of an ontology framework to facilitate the engineering design process has been limited.

4.4 Directed Graph-Based Decision Support

4.4.1 Markov Decision Process (MDP)

A Markov Decision Process (MDP) is a sequential decision problem, in which a systems utility depends on the sequence of decisions taken over a period of time. A MDP can thus be defined as a sequential decision-making problem for a fully observable, stochastic environment with

a Markovian transition model, and an additive reward function. A MDP consists of a set of states s, a set of actions in each state $A(s)$, a transition model $P(S_{t+1} = s'|S_t = s, a)$, and a reward function $R(s)$. A Markovian transition model can be defined as process whereby the future state S_{t+1} depends only on a finite fixed number of pervious states $S_{0:t}$. The simplest form of the Markovian transition model is the first-order Markov process, in which the future state depends only the current state, and not on the entire state history. This is mathematically defined as follows:

$$P(S_{t+1} = s'|S_{0:t} = s) = P(S_{t+1} = s'|S_t = s) \tag{4.1}$$

A MDP can thus be formally written as a 4-tuple, $\langle S, A, P_a, R_a \rangle$, where S is a state space, A_x is a set of actions possible at state s, $P_a(s, s')$ is the probability of transitioning from state s to state s', and $R_a(s)$ is the reward at sate s when action a is taken. Figure 4.4 gives a representative illustration of a hidden Markov model, where S represents states, y represents observations, a represents actions, and b represents output probabilities [9]. MDPs are generally solved using dynamic programming techniques, such as value iteration and policy iteration. Policy is defined as what the agent should do for any state that the agent might reach.

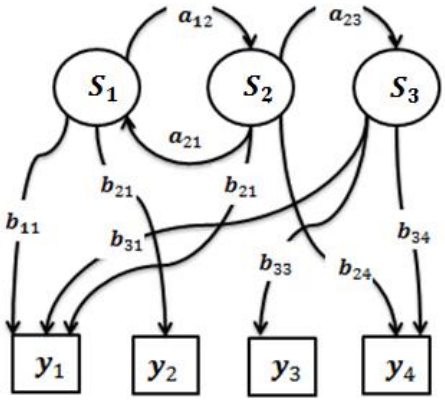


Figure 4.4: Representation of a Markov Decision Process (MDP).

The value iteration algorithm solves a MDP by iteratively stepping through the Bellman equation, which is defined as:

$$U(s) = R(s) + \gamma \max_{a \in A(s)} \sum_{s'} P(s'|s, a) U(s') \tag{4.2}$$

where $U(s)$ is the utility function for the current state, and γ is a discount factor which describes the preference of an agent for current rewards over future rewards. In the value iteration algorithm the Bellman equation is solved iteratively by defining initial values for the utilities, and then updating the utilities of each state from the utilities of its neighbours. This iteration step is defined as follows:

$$U_{i+1}(s) \leftarrow R(s) + \gamma \max_{a \in A(s)} \sum_{s'} P(s'|s,a) U_i(s') \tag{4.3}$$

where $U_i(s)$ is the utility value for state s at the ith iteration.

Policy iteration offers a simpler algorithm for solving MDPs, in comparison to value iteration. Policy iteration begins with some initial policy π_0 to calculate the utility of each state, if π_i were to be executed i.e. calculate $U_i = U^{\pi i}$. At the ith iteration, policy π_i states the action to be carried out in state s. The algorithm calculates a new maximum expected utility policy π_{i+1} using one-step look-ahead based on U_i. The algorithm terminates when the policy improvement step yields no improvement in utility. Because the action in each state is predefined by the policy π_i, and there are only a finite number of policies for a finite state space, policy iteration must terminate with finding the optimal policy within in a finite time frame. The implementation of the policy improvement step is easy, in comparison to the standard Bellman equation [102], as the action in each state is fixed. This results in a simplified version of the Bellman equation relating to the utility of s, under π_i, to the utilities of its neighbours.

$$U_i(s) = R(s) + \gamma \sum_{s'} P(s'|s, \pi_i(s)) U_i(s') \tag{4.4}$$

It can be noted that the 'max' operator has been removed, making the equations linear. For n states we have n linear equations with n unknowns, which can be solved exactly using standard linear algebra methods [102]. For a more in-depth review of value iteration and policy iteration the reader is referred to Russell and Norvig [102].

Fulcoly et al. [103] make use of MDP, in combination with the epoch-era framework, to analysis potential system change mechanisms that alter a systems behaviour over time in response to epoch shifts. Here an epoch is defined as a static snapshot of contexts with accompanying stakeholder needs over a fixed period of time [103], which can change over the lifecycle of the system. An 'era' is defined as the time-ordered sequence of epochs [66]. The developed framework, known as epoch syncopation framework (ESF), was applied to the design of an orbital transfer vehicle that can be used for a variety of on-orbit servicing missions. In this study several design variables, in combination with three change strategies were considered, such as the manipulator size, propulsion system type, and fuel mass. A change strategy determines when change mechanisms will be executed, which change mechanism will be executed, and what design will be transitioned to form a system architecture.

Frameworks similar to the MDP approach have been applied in many engineering decision-making problems. They are best suited for structured decision-making problems where the possible set of states, actions, and transition probabilities can be derived before the computation can proceed. However, for an unstructured decision problem, where prior information in regards to scenarios, transition probabilities, and actions are unknown, an MDP model fails to explicitly represent the decision-making problem. In addition the Markov

transition model fails to account for previous state history, which can prove to be detrimental in systems architecting, where future decisions are dependent on multiple past decisions. Thus, the Markov assumption does not hold.

4.4.2 Time-Expanded Decision Network (TDN)

The Time-Expanded Decision Network (TDN) framework, developed by Silver and De Weck [104], avoids the limitations of the Markov assumption by including the state history. This allows the TDN framework to run quantitative scenario planning analysis for system architectures, in which system responses are programmed through the time-expanded network [104]. This is achieved by capturing the cost of operating and switching system configurations through time, whilst seeking the least-cost path through the network in a changing operating environment. To frame the evolution of a system through its life-cycle as a dynamic flow problem, the TDN framework first represents a set of system configurations, and the costs associated with developing, operating, and switching between system configurations as a static network. This is illustrated in Figure 4.5, where S is the source and Z is the sink. In this example three system configurations are considered, A, B, and C, which are represented as nodes. The operating cost of each system configuration are split as nonrecurring engineering and facilities costs C_D, fixed recurring costs C_F, and variable recurring costs C_V. The switching costs between system configurations are represented as C_{SW}.

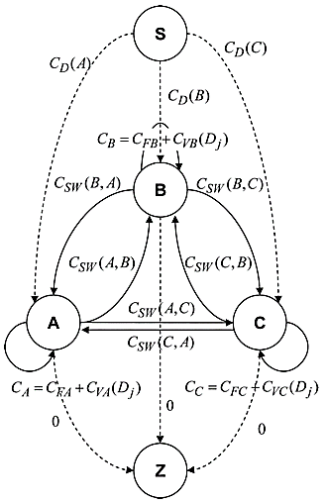


Figure 4.5: An example of a static network, representing switching costs, staying costs, and development costs [104].

The TDN can now be created with the addition of a time element to each link, resulting in the reformulation of a dynamic network into a time-expanded static network. The problem now is in finding the minimum cost flow from source S to a sink Z in a directed network, where each arc in the network is associated with a traversal time t, and a traversal cost c, and each node in the network is duplicated at each time period. In addition, the network decouples operating costs and switching costs by splitting each time period with chance nodes (circles)

and decision nodes (squares). This avoids the problem of the min-cost flow algorithm to selectively avoid 'operating arcs' and only choose 'switching arcs' in time periods where the demand is high. In reality existing systems will continue to operate to meet existing demands whilst a new system configuration is being developed [104].

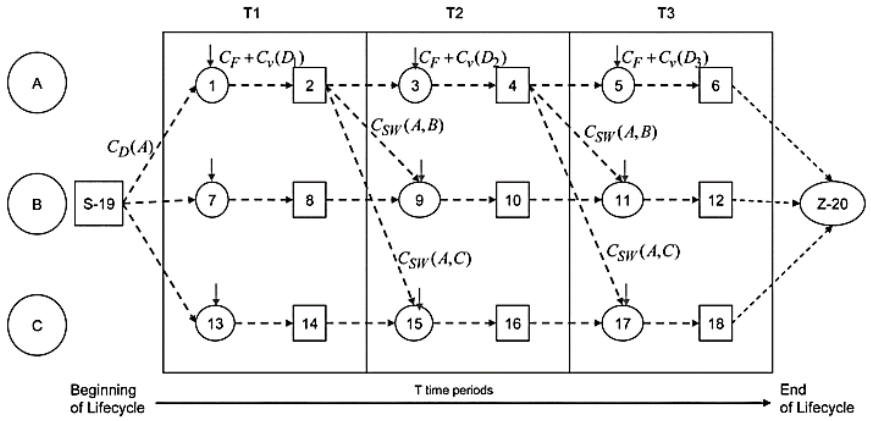


Figure 4.6: An example of Time-Expanded decision Network (TDN). Operating and switching costs over time are modelled for three concepts [104]**.**

The representation of a decision problem using the Time-Expanded Network allows the introduction of operational demand scenarios over time. This lead to the ultimate goal of finding sets of design configurations that minimise LCC, which is achieved by finding the least-cost path through the network. Since the defined network is acyclic, the nodes can be arranged topologically in terms of their distance from the source, allowing the use of a simple reaching algorithm to find the shortest path. The output of the TDN framework can thus be used as a representational tool to generate optimal strategies for switching between system architectures, depending on the demand scenario. The framework also provides a means to quantify the value of real options in terms of the effect in reducing switching costs [9].

Although the TDN framework mitigates the Markov assumption problem, it still faces the same issues as a MDP. The TDN framework is unable to represent an unstructured decision problem, where prior information about the initial set of system architectures, switching costs, and operating demand scenarios are unknown. In addition, this thesis assumes that the initial set of system configurations are not available at the outset, and focuses on the design-activity stage of decision-making to identify an optimal set of initial system configurations that should be considered by the decision maker.

4.5 Constraint Graph Based Methods

4.5.1 Constraint-Satisfaction Problems

A Constraint Satisfaction Problem (CSP) is a mathematical representation of a constraint problem, which is characterised by a set of decision variables, each with a finite domain of possible assignments [102].These decision variable are interconnected by constraints that

represent the compatibility of possible assignments for each decision variable. CSPs thus offer a natural representation for a wide variety of problems, and can be used to search the solutions space with multiple constraints imposed on the feasibility of solutions. To identify feasible solutions, the CSP search algorithm makes use general-purpose heuristics to eliminate large portions of the search space all at once by identifying variable-value combinations that violate the constraints [102]. Thus, the CSP search algorithm goes hand-in-hand with the specified *search* property of the evaluation criteria. The constraints imposed by the decision maker on the compatibility of different design alternatives can be propagated through the architectural solution space to remove infeasible system architectures.

A CSP can be expressed as triple $(\mathbf{X},\ \mathbf{D_X}, C_\mathbf{X})$, where $\mathbf{X}$ is a set of decision variables $\mathrm{x}_i \in \mathbf{X}$ that ranges over a finite domain $D_{x_i} \in \mathbf{D_X}$, $C_\mathbf{X}$ defines a set of constraints relating to the set of decision variables $\mathbf{X}$. Solutions to a CSP is found when all constraint functions are satisfied i.e. when $C_\mathbf{X} = \mathbf{true}$, where the vector $\mathbf{true} = [true_\mathbf{1}, true_2, true_3, \dots, true_{N_c}]$, and N_c represents the total number of constraints.

4.5.1.1 CSP Formalism

A CSP can be formulated based on the types of constraint being imposed on the decision space. The simplest type is a binary constraint, which relates to the restriction placed on the value of two variables e.g. $v_1 \neq v_2$. This can be extended to describe higher-order constraints that relate to more than two decision variables. Other common constraints are linear constraints on integer values, in which each variable appears in linear from e.g. $v_1 + t_1 \leq v_2$, where (v_1, v_2) are values associated to domains (D_1, D_2).

A CSP can also be formalised based on the domain size D of a decision variable. The simplest kind is that of a discrete finite domain, where variables can be thought of as discrete values that have limited ranges. However, CSPs in real world applications generally tend to have continuous domains, where the variable v in domain D can have any real value in the range of a and b - $v = (a, b) = \{v \in \mathbf{R} \mid a < v < b\}$. This thesis however only focuses on discrete, finite domain variables, as design alternatives generally pertain to configurationally changes that are discrete and finite.

The constraints that have been described thus far have been absolute constraints, where the violation of a constraint will result in the ruling out that particular solution. However, constraints can also be setup based on preferences – preference constraints indicate solutions that are preferred by the decision maker i.e. soft constraints. These constraints are generally encoded as cost functions on an individual variable assignment. A Preference constraint set $C_p = \{F_1, \dots \dots, F_m\}$ contains a set of m constraints, which are made up of cost functions F_j. The means of identifying the most preferred solution is by minimising the cost function, which is generally solved with an optimisation algorithm.

An extension of CSPs are Constrain Optimisation Problems (COPs), which can be defined as $(\mathrm{CSP},\mathbf{g})$ or $(\mathbf{X},\ \mathbf{D_X}, C_\mathbf{X}, \mathbf{g})$, where $\mathbf{g}$ is a set of objective functions. Several studies have

represented the system architecting process as a multi-objective COP [105]. Ehrogott and Gandibleux [106] provide a detailed survey on the development of COP methods in solving real world decision problems. Rayside and Estler [107] describe the development of a user interface that enables the system architect to define a multi-objective COP, and provides a means to solve the COP. The developed interface is however aimed towards solving COPs with discrete variable domains with many constraints, and is not intended for deign variables that are continuous. Work by Lin [105] also represents the system architecting process as a multi-objective COP, where a generalised Conflict-directed A^* search is used to find Pareto optimal solutions. This developed framework was applied to Apollo mission mode case study to find the optimal mission mode [105].

4.5.1.2 Methods of Solving CSPs

To illustrate the methods used in solving a CSP the following map colouring example is used. The map colouring problem is a classic example that has been used in many literatures regarding CSPs [102]. Here the task is to colour each region of the map in such a way that no neighbouring regions have the same colour.

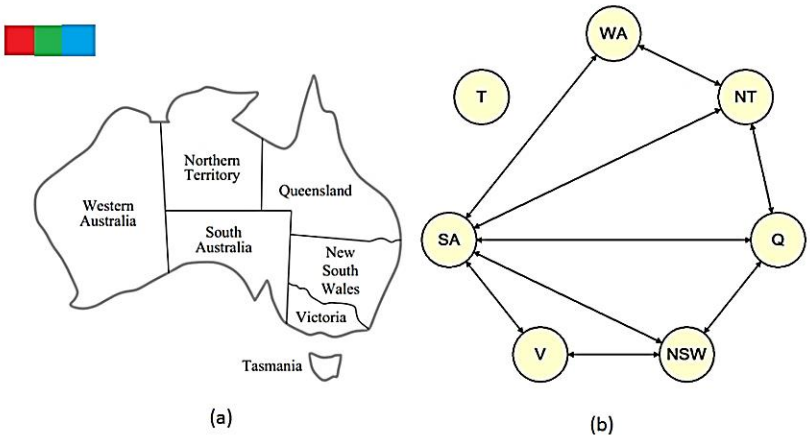


Figure 4.7: An example of a map colouring constraint satisfaction problem. (a) The states and territories of Australia that need to coloured using red, green and blue. (b) The constraint map of the map-colouring problem [102].

Using the map-colouring example in Figure 4.7 the decision variables can be defined as $\mathbf{X} = \{WA, NT, Q, NSW, V, SA, T\}$, and the domain for each decision variable is given as $\mathbf{D_X} = \{red, green, blue\}$. The constraints imposed on the decision variables can be described as binary constraints (i.e. no neighbouring regions can have the same colour), which can be mathematically represented as follows:

$$C_{\mathbf{X}} = \{SA \neq WA, SA \neq NT, SA \neq Q, SA \neq NSW, WA \neq NT, NT \neq Q, Q \neq NSW, NSW \neq V\} \quad (4.5)$$

The aim here is to present a series of methods that can be used to obtain a solution to an arbitrary constraint problem. Hence, the focus is not in finding the most efficient method for

solving the map-colouring problem, rather the following sections outline some general purpose methods in solving CSPs, and illustrate its use by using the map-colouring problem as an example.

4.5.1.3 Constraint Propagation

Before the search begins in finding a feasible set of solutions it would be useful to remove as many infeasible values from the solution space. This will reduce the search time, as the search space would consist mostly of viable solutions. The removal of infeasible solutions prior to search is achieved by the application of a domain specific of inference function, called constraint propagation. The idea in constraint propagation is to achieve local consistency. Local consistency is achieved by treating each variable as a node in a graph and each constraint as an arc. Consistency is enforced across each part of the graph until all inconsistent values have been removed. There are different degrees to which local consistency may be imposed, some of which are, node-consistency, arc-consistency, and k-consistency. Node-consistency is enforced by ensuring all constraints are satisfied in the domain of a single decision variable. Arc consistency extends this idea and ensures that all constraints are satisfied across a constraint arc connecting two decision variables. k-consistency is a general form approach to constraint propagation and is the strongest form of constraint propagation, where the k represents all the nodes/decision variables in a constraint graph. k- consistency is achieved by ensuring that constraints of the decision variables domains are satisfied across all arcs and nodes in the graph. A CSP is said to be strongly k- consistent if constraints are satisfied across all $k, k-1, k-2, \ldots\ldots.,$ all the way down to a single decision variable. However the time and memory required to achieve k-consistency is exponential in k. Therefore in practice arc-consistency or less commonly path-consistency (or 3-consistency) is adopted. However, it should be noted that full consistency is only achieved if k-consistency is applied. Applying the more commonly used arc-consistency does not remove all the inconsistent solutions from the domain of each decision variable. For a more in-depth definition of constraint propagation the reader is referred to Russell and Norvig [102].

4.5.1.4 Search

It has been established that inference alone does not find all viable solutions, unless k-consistency is applied. Therefore, there comes a time when it is required to search the decision space in order to find viable solutions. Seen as we are interested in finding all the viable solutions that are available, a depth-first search strategy is adopted to achieve this. Depth-first search is a graph-based search strategy where the graph is first represented as an acyclic directed network, or a tree network where the root node is at the top of the network and a branching structure traverses down. The search expands the deepest node of the network where the node has no successors, when the deepest node of a branch has been explored the search 'backs-up' the tree to the next node that has unexplored successors [102]. An example of a depth-first search is given in Figure 4.8.

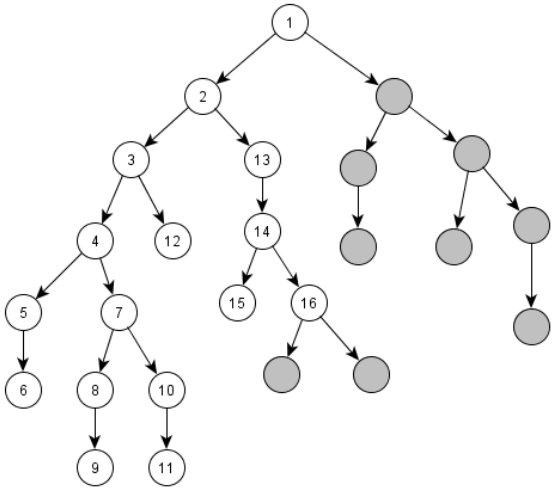


Figure 4.8: Depth-first search strategy applied to a tree network.

The search strategy described thus far has not addressed removing infeasible solutions from the decision space. To address this issue a variant of depth-first search called backtracking search is adopted. Backtracking search chooses values from the domain of each decision variable and assess if there is a viable solution at each successor node. If none exists it backtracks and tries different value assignments until a viable solution has been found i.e. one that does not violate any constraints. To illustrate the process the map-colouring problem is used as an example, here we assign values to decision variables in the order ,(WA, NT, Q).

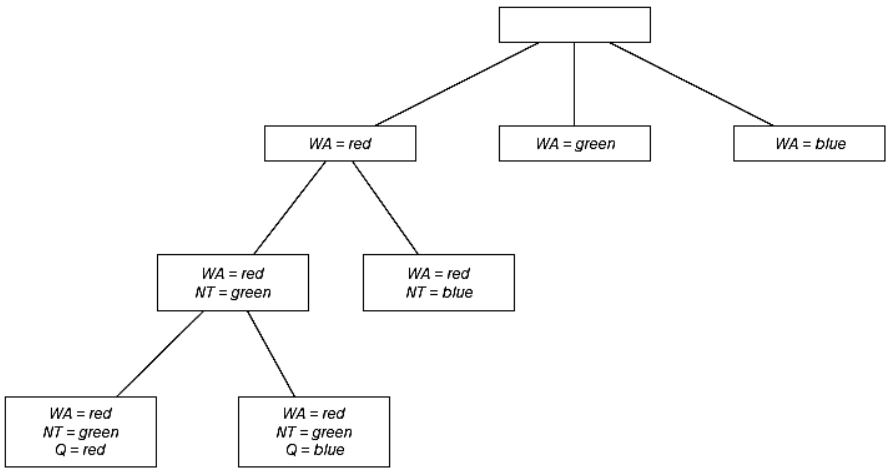


Figure 4.9: Backtracking search of part of the map-colouring example [102]**.**

The disadvantage with using backtracking search is that only one successor is generated at a time rather than all of the successors. This can result in a large computational time, especially when the network is densely branched and complex constraint functions are imposed on the decision space. To addresses the issue of performance there have been several methods that have been developed over the years that improve the backtracking search strategy. Although the details of these methods are not presented here, as they are

beyond the scope of this thesis, the functions of the search problem that they address are presented as follows:

- In what order should the tree network be organised and in what order should the values in the domain of the decision variable/node be tried.
- Can inferences be integrated into each step of the search (such as constraint propagation) in order to remove infeasible values of successor decision nodes/variables.
- Can the search process avoid repeating failures where there are constraint violations.

As stated there are numerous methods available to address these functions, and the choice of the method to be used is entirely dependent on the search problem at hand. For more information, in regards to improving the performance of the search, the reader is referred to Russell and Norvig [102].

4.6 Meta-Language Based Methods

4.6.1 Object-Process Network (OPN)

The OPN framework provides a methodology to formulate a complex problem by expressing it in a computational meta-language[6] that is customised for the decision problem being considered. The OPN framework was developed by Koo [108] and was originally applied to create a moon-mars exploration architecture generator. The use of a meta-language is useful in system architecting, as it provides a decision maker with a means to describe all parts of the system and their interactions. OPN makes use of a bipartite graph representation to define a network of objects and processes to represent the design alternative space. An *Object* in OPN stores the intermediary states of the executing OPN model, and a *process* in OPN stores the transformation rules that change the state of an executable model. The interconnecting edges in the network represent the relationships that depict pre- and post-conditions for the execution of a model. Figure 4.10 represents an annotated diagram showing the syntax of OPN. The processes are represented as ellipses, and objects are represented as rectangles. The *token* is an abstract data structure that records the execution path and reflects the sequence of events required to generate that *token*. The *token* in OPN is also used to carry data through the network as the model is executed.

[6] A meta-language is a language or system of symbols used to discuss another language or system. The prefix "meta" means "origin" or "one level of description higher", and "language" is used as a system of signs, symbols, or rules that are used for communication[108].

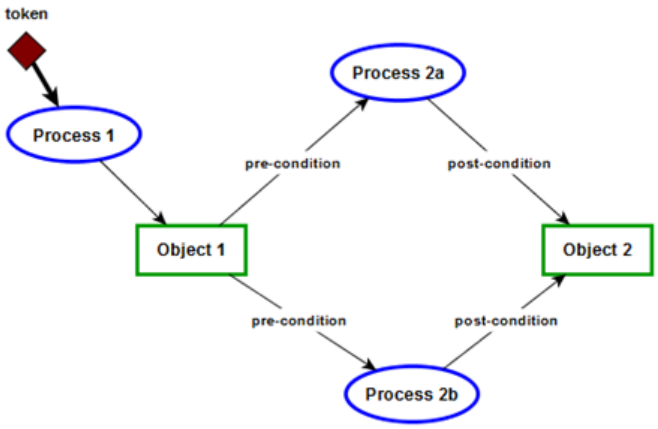


Figure 4.10: A simple OPN representation in a graphical format [108].

From a system architecting sense, the different paths through the network represent different possible architectures. The pre-condition for a process will prevent a *token* from proceeding down a specified path, resulting in the infeasible system architecture to be pruned from the network. This allows the OPN model to be executed through each path of the diagram to generate all the viable system architectures. Each of these processes can contain a programmable code that is capable of constructing equations or calculating metrics of interest [108]. The OPN framework has been validated by applying it to several case studies, such as the human moon-mars exploration program. In addition the OPN framework was also applied to model the mission-mode decision problem [109], the cargo launch vehicle configuration problem[110], and NASA's stakeholder policy problem [111].

The OPN framework meets the needs of architectural decision-making by enabling the construction of individual instances of architectural models, and automating the computational tasks of design space exploration. However, according to Simmons [9], the application of the OPN framework in practice was found to be conceptually difficult for system modellers. Also, the OPN model does not enable algorithms for structural reasoning, and thus does not manipulate the structure of the decision problem to increase computational efficiency, or provide information about the connectivity or the sensitivity of decision variables subject to architectural changes. Besides these limitations, the OPN framework is still powerful tools that provides numerical and symbolic solvers, in addition to providing viewing tools that generate plots of feasible system architectures.

4.6.2 Algebra of Systems (AoS)

The AoS framework is an extension of the OPN framework. The AoS methodology represents system models as algebraic entities, and formulates model transformation activities as algebraic operators to describe and evaluate the design space for a wide range of engineering systems. In AoS the knowledge of the design space is encoded in a triple $\langle P, B, C \rangle$. Each domain is made up sub-algebraic domains, which correspond to the domains operands and operators. Table 4.1 gives the domains, and their respective operands and operators that are used in AoS.

Table 4.1: AoS algebraic domains[112].

Domain	Operands	Operators
Algebra of Systems Domain, AoS.	$\langle P, B, C \rangle$	*{encd,enum,eval}*
Properties Domain, P	$\{\langle key, value \rangle^*\}$	*{merge, substitute, interp, delete}*
Boolean Domain, B	$\{TRUE, FALSE\}$	*{and, or, negate, interp}*
Composition Domain, C	$\langle \{Obj\}, \{Proc\}, \{pre\}, \{post\} \rangle$	*{union, subtract}*

The properties domain P denotes the quantitative and qualitative properties of a system by representing it in a formal data structure $\{\langle key, value \rangle^*\}$.The property domain also provides a set of operators to manipulate the data structure $\{merge, substitute, interp, delete\}$, which allows new information content to be created. The Boolean domain B denotes the Boolean value status that defines whether or not an AoS model satisfies certain constraints. These constraints are defined using the operators provided within the Boolean domain $\{and, or, negate, interp\}$, which are applied to the information content in the properties domain. The composition domain C denotes the compositional structure of a system by using the OPN framework. The system is encoded as a collection objects, processes, and connecting relationships. The connecting arcs are associated with a Boolean expression that defines the constraint between the object and process. The operators $\{union, subtract\}$ are used to combine and divide the information content of the graphs.

Finally, the AoS can be defined as: $AoS = \langle \{P, B, C\}, \{encd, enum, eval\} \rangle$, the three operators $\{encd, enum, eval\}$ – {*encode, enumerate, evaluate*} computationally derive elements of knowledge about the design solution space. This is similar to source-code compilation process, where knowledge about the system composition is processed in several stages. This process can be thought of as a knowledge compilation process, where the process converts *Available Knowledge* elements and *Enumerable Model* elements into *Generated Sub-models* and back into *Available Knowledge* [112]. This process is depicted in Figure 4.11. The design cycle ends when the Available Knowledge domain has sufficient information to implement a design.

Similar to OPN, the AOS framework meets the needs of architectural decision-making, and its principles are applied in various applications. For example, the ADG framework developed by Simmons makes use of the OPN and AoS principles to develop a decision support framework to study the configuration of human lunar outpost architectures [9]. The AoS framework satisfies the *construction* and *search* criteria, but does not address the representation of information, as defined by the *representation* criteria. Though the framework provides the capability to extract information and represent it in any format possible, it does not define a particular viewing method that best represents the information to the decision maker.

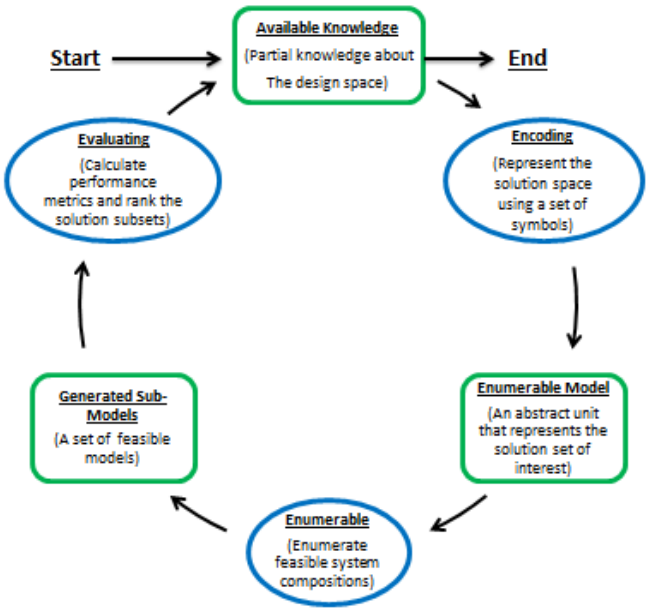


Figure 4.11: Algebra of System (AoS) knowledge compilation process [112].

4.7 Summary

This chapter presented a series of approaches that are used as a means of enabling the decision maker to create and search for viable system architectures. In order to evaluate the applicability of these different modelling approaches three evaluation criteria were established, *construction*, *search*, and *representation*. These criteria are aimed at capturing the needs of the decision maker in creating a system architecture.

Table 4.2: Comparison of different system architecting methods.

Approaches	Construction	Search	Representation
Morphological Matrix	✓		
DSM	✓		
Markov Decision Process (MDP)	✓		
Time-Expanded Decision Network (TDN)	✓		
Constraint-Satisfaction Problem	✓	✓	
Object Process Network	~✓	~✓	~✓
Algebra of Systems	✓	✓	~✓

Based on the comparisons made in Table 4.2, three methods are found to be of particular interest, CSP, OPN, and AoS. The CSP method provides the ability to encode a decision problem with alternatives, and logical constraints in a graph representation. In addition, the CSP method provides structural reasoning features to manipulate the decision problem and effectively search the solution space for viable system architectures. However, the CSP method does not provide any capability to represent data to the decision maker in a meaningful manor.

The OPN framework uses a meta-language to represent an architectural decision problem. The meta-language provides sufficient syntax and semantics to create a customised model of an architectural solution space. The OPN framework has been effectively applied in several architecting problems. However, applying OPN has also shown that it is conceptually difficult for a system modeller to define a decision problem using this approach. The OPN framework also lacks structural reasoning capabilities to effectively search the alternative space. In terms of representation the OPN approach does provide some capability provide a graphical view of the decision network made up of objects, processes, and connecting relationships. However, the OPN framework does not explicitly identify a set of representational approaches that would best translate the information content to the decision maker, such that optimal architectures are identified.

The AoS approach is an extension of the OPN framework. In AoS the knowledge of the design space is encoded in a triple $\langle P, B, C \rangle$, where P denotes the quantitative and qualitative properties, B denotes the Boolean value status, and C denotes the compositional structure of a system [112]. The AoS framework goes further by offering the capability to encode, enumerate, and evaluate system solutions by generating sub-models to capture new available knowledge. However, similar to OPN, the drawback in applying the AoS framework is that it can be conceptually difficult for system architects to represent decision problem using a formal meta-language [9]. Also, the AoS approach does not offer any explicit knowledge representational views that best translates valuable information to the decision maker. This is rather left to the system modeller to encode such capability into the framework.

Both OPN and AoS offer powerful simulation features which can generate a set of viable system architectures, and calculate numerical properties via its processing capabilities. Also, both approaches have been developed to encode engineering design problems into a programming environment by the use of a formal meta-language. Although, these approaches have presented themselves to be ideal candidates for addressing the architectural decision-making problem, this study focuses on the CSP approach to address the *construction* and *search* phase. This is because, in order to create a formal modelling approach that encodes all the capabilities required to address architectural decision-making into one simulation environment would be time-consuming, and is beyond the scope of this thesis. This is not to say that the CSP approach offers any less of a capability in terms of modelling an architectural decision-making problem. The CSP approach is a more generic tool that addresses many constraint problems and is not specifically restricted to architectural decision-making. Thus, the development of computational tools to model CSP has been vast, and is widely available for use by many open-source programming packages.

Chapter 5: Centrality Measure in Networks

An integral aspect of this research is in the identification of high-impact decision variables that aid the architectural exploration of design alternatives. To define the importance of each decision variable within a network, this thesis adopts methods from graph theory and from the network analysis literature. A network can be defined as a pattern of connections in a system, which can be represented by a series of nodes and edges. Each element of a system is represented by nodes within a network, and the edges between nodes represent relationships between system elements. The structure of such networks, in particular the pattern of interactions, can have a big impact on system behaviour [113].

Mathematically networks can be represented in a number of different ways. Take an undirected network with n nodes, where each node is labelled with integer values ranging from $1, \ldots, n$, and the edges between the nodes i and j can be denoted as (i, j). This allows the complete network to be defined by giving a value of the node n, and a list of all edges for each node. For example, take the network represented in Figure 5.1, here if we take $n = 4$ the corresponding edge list would be $[(1,2), (1,3)]$, as it is connected to nodes 2 and 3. Another way to represent networks, and arguably a more efficient representation, is in a matrix form, known as the adjacency matrix. The adjacency matrix of a network A_{ij} is defined as follows:

$$A_{ij} = \begin{cases} 1 & \text{If there is an edge between vertices } i \text{ and } j. \\ 0 & \text{Otherwise.} \end{cases} \quad (5.1)$$

This chapter provides a review of the fundamentals of network analysis and centrality measures. The chapter starts with an overview of different representations of networks that are available, and the implications of choosing a network type to represent the system architecting process, as described in Chapter 1. The chapter follows onto define different measures of centrality, more specifically this chapter focuses on degree, eigenvector, Katz, and PageRank centrality. As they are closely related to measuring the influence of a node within a network. Though, it should be pointed out that other centrality measures, such as, closeness centrality, betweenness centrality, hubs and authorities' centrality, and other centrality measures are also available for use. However, these centrality measures are not necessarily focused towards capturing the influence of a node within a network. For example, closeness centrality measures the mean geodesic distance from node i to j, averaged over all nodes j in the network. Nodes with low closeness centrality measures show that they are

separated from others by only a short geodesic distance on average. Conceptually such nodes might be thought of as having better access to information from other nodes, or more direct influence on other nodes [113]. For a more in-depth review of centrality measures and network analysis the reader is referred to Newman [113].

5.1 Types of Networks

The existence of different types of networks is defined by the connection of edges between different nodes. This thesis mainly focuses on networks which are undirected with no self-edges and no multi-edges. To elaborate on this point the following sub-sections provide a brief overview of different network representations.

5.1.1 Multi-Edge and Self-Edge Networks

These networks allow more than one edge between a pair of nodes- *multi-edges*, or have edges that connect nodes to themselves- *self-edge*, which is illustrated in Figure 5.1. Multi-edges are represented by setting the corresponding matrix elements in the adjacency matrix to the multiplicity of the edge. For example, a double edge between nodes i and j is represented by $A_{ij} = A_{ji} = 2$. Self-edges from node i to itself is represented by setting the corresponding diagonal element $A_{ii} = 2$. In this instance a value of two is assigned as self-edges have two ends, both of which are connected to node i.

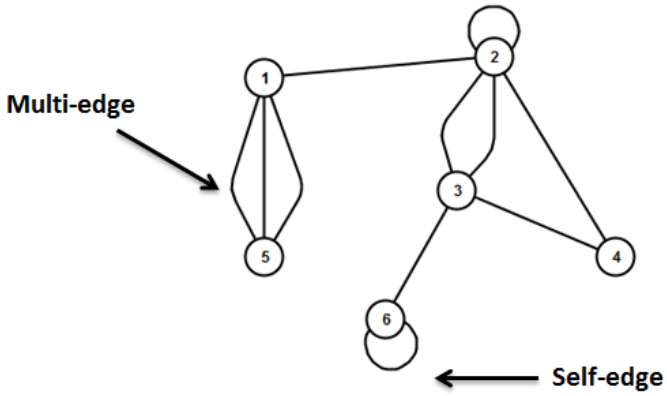


Figure 5.1: Illustration of a multi-edge & self-edge network.

Self-edges and multi-edges are very rare in systems architecting or decision networks, as it goes against the assumptions made about the architecting process in this thesis. The representation of compatibility constraints within the network are assumed to exist between a set of nodes, rather than to a node itself i.e. no self-relationships exist. But, it is useful to recognise that such networks exist and can be modelled using a simple matrix representation [113].

5.1.2 Acyclic Directed Network

An Acyclic directed network has no closed loops of edges and does not contain any self-edges or multi-edges. A closed loop in a network is defined as the arrows on each of the

edges pointing in the same direction around the loop. Thus, a network with no cycles is called an acyclic network. An interesting feature to note in an acyclic directed network is that if the nodes are ordered sequentially, as in Figure 5.2, then there can only be an edge from node j to node i if $j > i$. Translating this feature to the construction of the adjacency matrix will results in all non-zero elements to be above the matrix diagonal i.e. all non-zero elements are present in the upper triangular of the matrix. The adjacency matrix for an acyclic network also presents the property that all of its eigenvalues are zero, which acts as an indicator to prove that a given network in acyclic.

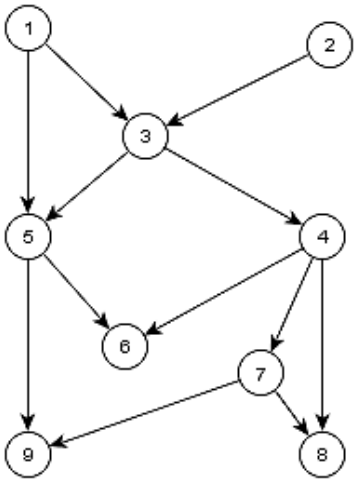


Figure 5.2: Illustration of an acyclic directed network.

Acyclic directed networks can therefore be used to represent sequential decision-making problems, where all edges in the directed network point forward in time. This is represented graphically in Figure 5.2. However, as described in Chapter 1, the system architecting process is assumed to be represented by an undirected network, as the compatibility relationships do not signify a direction between the connecting nodes. Therefore, an acyclic directed network would not be an appropriate representation of the system architecting process.

5.1.3 Bipartite Network

In a bipartite network there are two different node representations, one representing the original node and the other representing the group to which it belongs to [113]. For example, from a system architecting perspective the group nodes can be represented as decision variables, and the design alternatives can be represented as nodes connected to the group node that they belong to. Thus, the bipartite network can be thought of as a network representation of the morphological matrix. An example of a bipartite network is shown in Figure 5.3.

The representation of a bipartite network in a matrix form differs slightly from the traditional adjacency matrix, where the matrix represented is no longer a square $n \times n$ matrix. If there

are n original nodes and g group nodes the matrix can represented by a $g \times n$ matrix, having B_{ij} elements such that:

$$B_{ij} = \begin{cases} 1 & \text{If vertex } j \text{ belongs to group } i\,. \\ 0 & \text{Otherwise.} \end{cases} \tag{5.2}$$

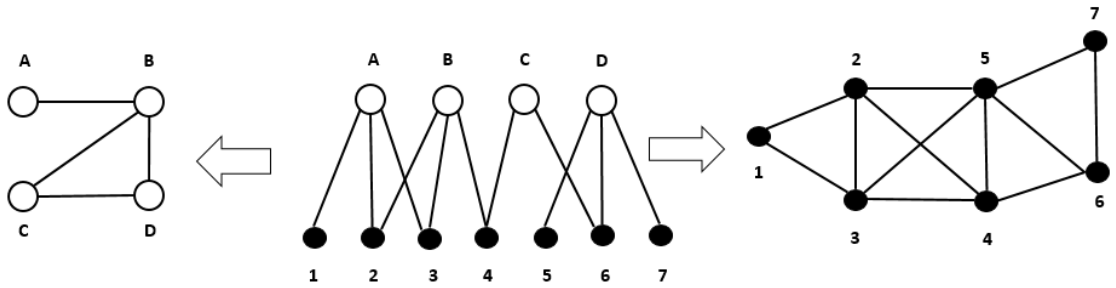


Figure 5.3: A two-mode projection of a bipartite network, where the open circles represent four groups, with edges connected to each node to represent the group it belongs to [113]**.**

Although the bipartite network may give a complete representation of a network, it is more convenient to represent the constraint relationships between decision variables using direct connections between nodes. This is more efficieny rather than forming a bipartite network consisting of decision variables and a set of design alternatives that belong to those decision variables. For example, taking the bipartite network illustrated in Figure 5.3, links between node sets (A, B, C, D) and sets $(1,2,3,4,5,6,7)$ can be infered by creating a one-mode projection from a two-mode bipartite network, as shown in the left and right hand portions of Figure 5.3. However, the one-mode projection does not accuratly represent the system architecting process, as there is no differentiation between edges that represent constraint relationships between decision variables and edges that represent design alternatives that belong to a given decision variable. Thus, representing the system architecting process as a bipartite network will result in a loss of information.

5.2 Centrality Measure

The representation of networks is only part of the solution. In order to extract information about the network structure, several analysis methods have been developed over the years, and it has been an area of active research in the fields of computer science and applied mathematics. This thesis presents some of the basic network analysis methods that can be used to extract information about the system architecting network, and in turn aid the decision-making process. Centrality is defined as a measure of a nodes importance in a network i.e. the most central node in the network. Depending on the network structure a variety of useful quantities or measures can be calculated that captures particular features of the network topology. There are multiple methods of measuring centrality that are available in literature [113]. The following sub-sections provide a review of some of these methods in measuring the importance of decision variables in a system architecting network. To aid the understanding of each method a notional network is created, which will hereafter be used to

illustrate the concepts behind different centrality measures. This notional network is represented in the figure below.

0	1	0	0	0	0	0	1	0	0	1
–	0	1	0	0	1	0	0	0	0	0
–	–	0	1	0	0	1	1	0	0	1
–	–	–	0	0	0	0	0	1	0	1
–	–	–	–	0	0	0	0	0	0	0
–	–	–	–	–	0	0	0	0	0	1
–	–	–	–	–	–	0	0	1	0	0
–	–	–	–	–	–	–	0	1	0	0
–	–	–	–	–	–	–	–	0	1	0
–	–	–	–	–	–	–	–	–	0	0
–	–	–	–	–	–	–	–	–	–	0

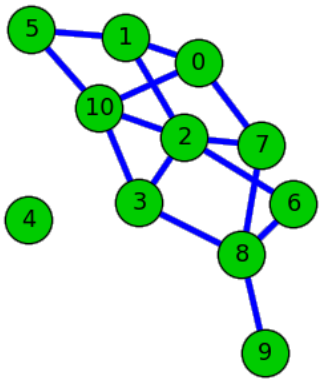


Figure 5.4: The adjacency matrix and a pictorial representation of the notional network used to illustrate the concepts behind different centrality measures.

The above network is an undirected network with no self-edges and multi-edges, and is illustrative of a typical system architecting network. The interrelationships between the nodes are assumed to be undirected and relationships only exist between a set of nodes with a minimum of at least two nodes i.e. no self-edges. In this network each node is also given a relative importance value that is based on a non-network factor, which is representative of the decision variables impact on system properties such as value and cost. These relative importance values are defined in the table below, where the range in the importance scales symbolise a value of 0 being not important and a value of 1 being very important.

Table 5.1: Relative importance of nodes based on non-network factors

Nodes	0	1	2	3	4	5	6	7	8	9	10
Importance	0.3	0.6	0.1	0.4	0.8	0.32	0.65	0.11	0.05	0.55	0.6

5.2.1 Degree Centrality

Degree centrality is the simplest measure of centrality in a network. The degree of a node in a network is defined by the number of edges connected to that node. For an undirected network the degree x_i of node i can be calculated in terms of the adjacency matrix as:

$$x_i = \sum_{j=1}^{n} A_{ij} \tag{5.3}$$

For a directed graph however there exists two degree metrics, the in-degree and out-degree. In-degree x_i^{in} represents the number of incoming edges to node i and the out-degree x_j^{out} represents the number of outgoing edges from node j.

$$x_i^{in} = \sum_{j=1}^{n} A_{ij} \quad \text{and} \quad x_j^{out} = \sum_{i=1}^{n} A_{ij} \tag{5.4}$$

The measure of a nodes degree in a system architecting network can be of importance, as it represents the degree to which a decision variable is influenced by other nodes (x_{in}) and the degree to which a given decision variable is influencing other nodes (x_{out}). Using the network defined in Section 5.2, degree centrality identifies the most connected node and therefore the most central node in the network as being node 2, the least important being node 4. This is illustrated in Figure 5.5, where the size of the node in the network represents the nodes importance in the network i.e. the bigger the node the more important it is.

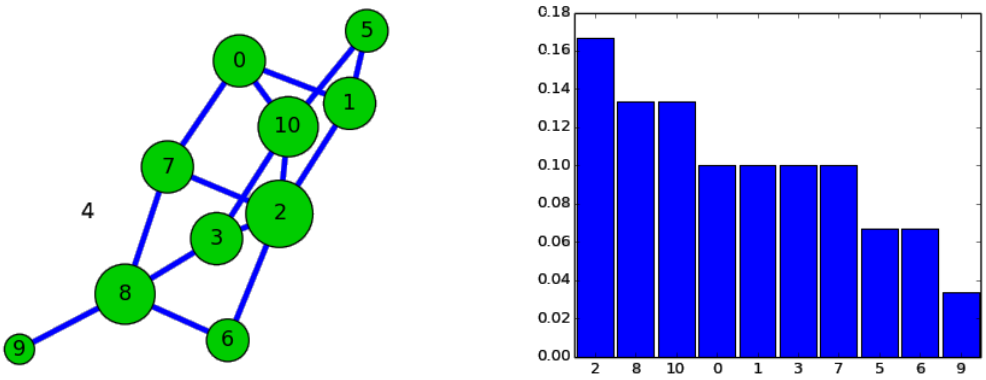


Figure 5.5: Degree centrality of the network.

Although degree centrality is a useful measure, it does not account for the importance of neighbouring nodes themselves, nor does it account for non-network factors that influence centrality. In many circumstances a nodes importance in a network is increased by having connections to other nodes that are themselves important.

5.2.2 Eigenvector Centrality

Eigenvector centrality addresses the limitations of degree centrality by scoring nodes proportional to the sum of the scores of its neighbours. That is, eigenvector centrality defines a nodes importance in a network to be increased by having connections to other nodes that are themselves important. This is defined mathematically as:

$$x_i = k_1^{-1} \sum_j A_{ij} x_j \tag{5.5}$$

Where, x_j is the neighbouring node of x_i, and k_i are the eigenvalues of the adjacency matrix A_{ij} and k_1 is the largest of them. Eigenvector centrality can be applied to both directed and un-directed networks. However, problems arise when measuring centrality of directed networks. If a node in a network has only outgoing edges or no connecting edges, then the centrality of that node will be zero. This conceptually might not seem to be a problem as a node that no one points to could be considered to have centrality zero. However, consider the case of a node that may be pointed by many others that they themselves are pointed to by many more, and so through the generations, but if the progression ends up at a node that

has zero in-degree then the final value of centrality will still be zero. This property is defined in mathematical terms as: only nodes that are in a strongly connected component of two or more nodes, or the out-component of such a component, can have non-zero eigenvector centrality [113].

The eigenvector centrality measure of the notional network is calculated and represented in Figure 5.6. It can be noted that the rank order of the nodes differs from degree centrality. For example node 8 is no longer ranked second and is pushed further down the order. It should also be noted that the centrality of node 4 is still zero, even though the relative importance of this node is high, as defined in Table 5.1. This is the inherent limitation of both degree and eigenvector centrality, where the non-network impotence factor of a node is not accounted in the centrality calculation.

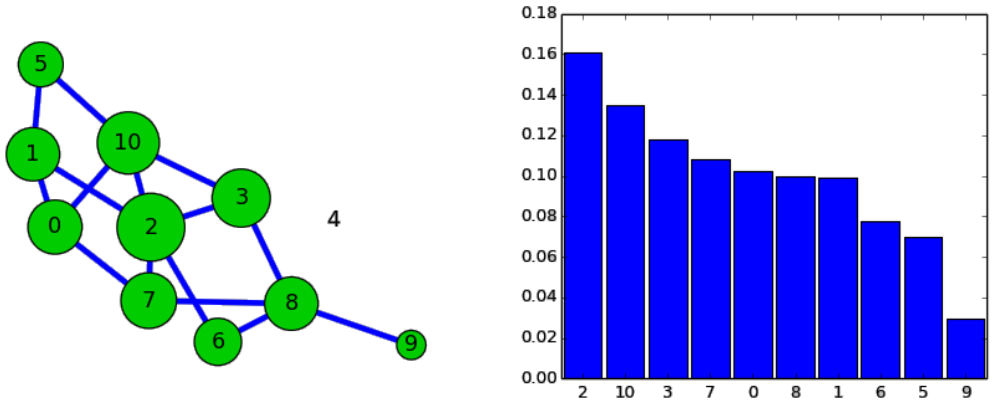


Figure 5.6: Eigenvector centrality of the network.

5.2.3 Katz Centrality

Katz centrality provides a solution to the problem of zero centrality measure by introducing a small amount of centrality for each node, regardless of its position in the network or the centrality of its neighbours. Katz centrality is defined mathematically as:

$$x_i = \alpha \sum_j A_{ij} x_j + \beta_i \tag{5.6}$$

where α is a positive constant. The second term in the equation allows nodes with zero in-degree to have a centrality of β. This allows non-network importance measures to be included into the centrality measure. Hence, any node that is pointed to by many other will have a high centrality, and those that are pointed to by others with high centrality themselves will still do better. In a matrix form this can be written as:

$$\mathbf{x} = (\mathbf{I} - \alpha \mathbf{A})^{-1} \boldsymbol{\beta} \tag{5.7}$$

The choice of the α value governs the balance between the eigenvector term and the constant term. Letting $\alpha \to 0$ would mean that only the constant term β would survive and centrality would only be measured by the β values. Increasing the α term from zero would increase the

centrality and eventually there comes a point at which it diverges. This happens at the point where $(\mathbf{I} - \alpha\mathbf{A})^{-1}$ diverges i.e. when the $\det(\mathbf{I} - \alpha\mathbf{A}) = 0$. As α increases, the determinant first crosses zero when $\alpha = 1/k_1$ i.e. the inverse of the largest eigenvalue of the adjacency matrix. Thus, the value of α should be less than this if the centrality measure is to converge.

The use of Katz centrality however still has its limitations. If a node with high Katz centrality has edges pointing to many other nodes then these nodes also inherit high centrality. This feature is not always desirable as the centrality gained by virtue of receiving an edge from a highly important node is diluted by being shared with so many other nodes [113]. The concept of Katz centrality can be illustrated by measuring the centrality of the notional network, which is depicted in Figure 5.7. From the centrality measure it can be instantly noted that node 4 no longer has a centrality measure of zero, instead the non-network importance values seem to dominate the centrality measure, thus placing node 4 on the top of the rank order list even though node 4 is not connected to any other nodes. Katz centrality also indicates nodes 0, 2, and 5 gaining centrality by being connected to node 1, which in itself is relatively important. However, node 2 is placed relatively low in the rank order even though it has a high degree of connectivity.

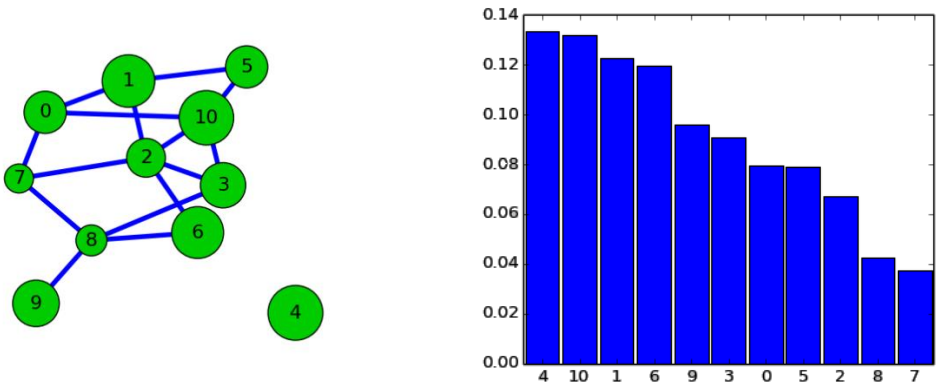


Figure 5.7: Katz centrality of the network.

5.2.4 PageRank Centrality

PageRank centrality was originally developed by Page et al [10] and is used as a central part of Google's web ranking technology. PageRank is an extension to Katz centrality, in that the nodes that point to many other nodes pass only a small amount of centrality on to each of the other nodes even if their own centrality is high. To get a better compromise between connectivity and non-network importance factors, PageRank divides the centrality of the neighbouring nodes by their out-degree. This is represented mathematically as:

$$x_i = \alpha \sum_j A_{ij} \frac{x_j}{k_j^{out}} + \beta_i \tag{5.8}$$

This however introduces a problem when the out-degree $k_j^{out} = 0$, which makes the solution indeterminate. The problem is fixed by artificially setting $k_j^{out} = 1$ for such nodes. In a matrix form this can be written as:

$$\mathbf{x} = \alpha \mathbf{A}\mathbf{D}^{-1}\mathbf{x} + \boldsymbol{\beta} \tag{5.9}$$

with $\mathbf{D}$ being the diagonal matrix with elements $D_{ii} = \max(k_i^{out}, 1)$, k_i^{out} being the out-degree of the neighbouring node x_j, and $\boldsymbol{\beta}$ being the personalised vector (similar to that defined in Katz centrality), which defines the personalised ranking value for each node. The above equation can be rearranged to give:

$$\mathbf{x} = \mathbf{D}(\mathbf{D} - \alpha \mathbf{A})^{-1}\boldsymbol{\beta} \tag{5.10}$$

Similar to Katz centrality, the value of the constant term α is determined by the inverse of the maximum eigenvalue of the matrix. In this case α should be less than the inverse of the largest eigenvalue of $\mathbf{A}\mathbf{D}^{-1}$. For an undirected network it can be shown that the largest eigenvalue is equal to 1 and the corresponding eigenvector is $(k_1, k_2, k_3, \ldots)$, where k_i is the degree of the ith node [113]. Thus, the value of α should be chosen to be less than 1. It should be noted that the calculation of the largest eigenvalue and its corresponding engine vector is done iteratively by using the power iteration method, which is described in Appendix B. The power iteration method is also applied to in the calculation of Katz centrality.

PageRank is very useful in cases where having edges connecting a node from other important nodes elsewhere in the network is a good indication that the connected node is also important. However, PageRank also ensures that a high centrality node pointing to a large number of other nodes does not pass on large centrality scores to those other nodes, which is ensured by dividing the neighbouring node by its out-degree. This is illustrated by using the notional network that was defined at the beginning of section 5.2, the results of which are defined in Figure 5.8. Node 4 is no longer at the top of the rank order, even though its non-network importance factor is very high. This is because node 4 is not connected to any of the other nodes in the network, and therefore its influence on the network is non-existent. For this reason its rank order in the network is reduced. It can also be noted that node 10 has the highest centrality in the network but it does not pass all of its centrality onto other connected nodes, such as nodes 0, 2, 3, and 5.

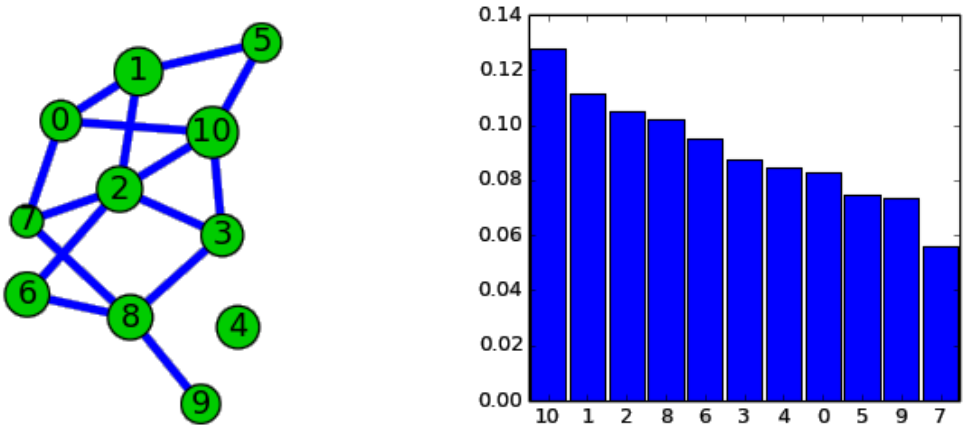


Figure 5.8: PageRank centrality of the network.

5.3 Summary

Representing the system architecting process as a network is a very effective means of visualising the relationships between different decision variables, and the imbedded knowledge of the design team during the decision-making process. The graph-based methods described in this chapter allow the decision maker to easily represent the architectural decision problem as a network. This is achieved by defining the adjacency matrix of the network, which in turn can be easily translated into a programming environment. In addition, the network analysis methods described in the previous section allow the decision maker to gain information about the network structure and understand the behaviour of the network, in terms of identifying important decision variables within the network that have a strong influence on system properties.

In representing the architectural decision-making process as a network several assumptions have been made. The network is assumed to be un-directed, where the connected nodes influence each other equally, it is also assumed that there are no self-edges i.e. a node cannot add a relationship to itself or multi-edges between nodes.

Once the representation of the system architectural network is satisfied, the next step requires identifying an appropriate means of measuring the relative importance of each node. The previous section made a comparison of several commonly used centrality measures, degree, eigenvector, Katz, and PageRank centrality. By comparison it was established that both degree and eigenvector centrality do not offer a suitable match to measure the relative importance of each node from a system architecting perspective. As they do not account for non-network related importance factors such as the influence of a decision variable on system properties like value and cost. This leaves us with Katz and PageRank centrality, both of which account for non-network factors. However, Katz centrality poses one fundamental flaw in measuring the relative importance of each node from an architectural decision-making perspective. In Katz centrality the non-network centrality measure of a node seems to dominate the overall centrality measure, which in-turn is passed onto other nodes to which it is connected to. This artificially boosts the centrality of those connected nodes, even if they are unimportant, for the reason of being connected to an important node. PageRank centrality over comes this by dividing the connected nodes centrality by their out-degree, such that nodes that point to many others only receive a small amount of centrality. This allows a much better balance between non-network centrality measure and network related centrality measure, such its degree of connectivity.

Chapter 6: Decision Support Framework

The aim of this chapter is to present the reader with the decision support framework. The developed framework first captures the overall system value, and second provides a means of ranking high-impact decision variables based on their influence on system properties, and their influence on other decision variables within the network. The process of constructing the framework makes use of methods and tools described in previous chapters. Chapter 2: identified two value streams, one capturing end-user value and the other capturing the value of the engineering-firm. This thesis will only address end-user value, as capturing the revenue stream of a complex engineering firm is beyond the scope of this thesis. End-user value is captured by applying Multi-Attribute-Utility-Theory (MAUT).

Chapter 4 defined the decision-making process in system architecting, where the needs of the decision maker were identified to fit the following three criteria, *construction*, *search*, and *representation*. These criteria were used as a means of comparing different modelling approaches to aid the architectural decision-making process. Out of these approaches three stood out, Constraint Satisfaction Problems (CSP), Object Process Network (OPN), and Algebra of Systems (AoS). This thesis makes use of methods derived from the CSP approach (for reasons described in Chapter 4:) to identify feasible system architectures. The framework also integrates other approaches, such as morphological matrix and DSM to aid the construction of the decision problem.

6.1 Assumptions Made

To simplify the construction of the decision support framework several assumptions are made, in regards to the systems architecting process. This chapter first presents the underlying assumptions made in the construction of the decision support framework, and then follows on to describe the details of the framework.

Assumption 1. The end-user of the system perceives value in terms of non-monetary benefits. The system is assumed to operate in a non-commercial environment where the benefits ascertained by the system to the end-user are not through the revenue generated by the system. For example, this could be representative of a military acquisition of a combat aircraft.

Assumption 2. The value model is constructed by adopting MAUT. The MAUT approach assumes that the end-user makes decisions in an uncertain environment, where the outcome of the system is not known with certainty. The MAUT approach captures the preference relationships of the customer to uncertainties in the outcomes of the system attributes.

Assumption 3. The decision-making process in systems architecting is simplified into three phases, *intelligence, design*, and *choice*, as defined by Simon [21]. Chapter 4: identified the *design* phase of the process to be the most central, and also the most time consuming part of the decision-making process. The information ascertained at the *design* phase is used to capture system performance attributes, which in-turn is used to aid the decision-making process. For this reason, this thesis mainly focuses on addressing the *design* activity of the decision-making process.

Assumption 4. To model the architectural down selection process it is assumed that decisions are made as a means of down-selecting designs from a set of design alternatives in order to create a feasible system solution. The decisions made can therefore be regarded as an architectural refinement process, where the architectural design space is reduced to a set of possible candidates [9]. This allows the architectural decision-making process to be modelled as a network, where the nodes of the network represent a set of decision variables $\{X_1, \ldots\ldots\ldots, X_N\}$, each with a finite domain of design alternatives $\{v_1, \ldots\ldots\ , v_n\}$. The connecting edges represent the constraints imposed by the decision maker, in regards to the feasibility of different design alternatives.

Assumption 5. To simplify the modelling of the architectural decision-making process, the decision support framework assumes that all design alternatives are discrete variables with a finite set of alternatives. This allows the propagation of constraints over a discrete domain space, and thereby narrowing the feasible solution space using simple constraint satisfaction algorithms. It should be noted that the aim of the decision support framework is not to provide one single optimal solution, but to present a narrowed down set of feasible Pareto optimal solutions to the decision maker. This allows an informed choice can be made in regards to the system architecture.

Assumption 6. The decision support framework assumes that the system architecting process is not a sequential decision-making process. This is because the compatibility relationships between a set of decision variables generally tend to be bi-directional. From a logical constraint standpoint it doesn't matter which design alternative is selected first, as the order will not change the feasible domain of alternatives. However, many decision support frameworks, such as MDP, decision trees, and influence diagrams, assume that decisions are made in a sequential order,

which requires the decision makers to specify the order of evaluating decision variables. Such problems generally tend to represent planning actions over time, in which case a sequential representation is more appropriate.

6.2 Decision Support Framework

Much of the literature presented thus far has focused on tradespace exploration, by starting with a baseline architecture and making incremental changes, from a set of available candidate design alternatives, to find the global optimum. Other architectural exploration methods have focused on analysing the problem of evolving or extending an architecture, from a set of candidate design alternatives, in the presence of exogenous changes to the system. This thesis assumes that a set of candidate design alternatives are not fully defined, rather the developed framework aims to generate, and further explore the set of candidate architectures that are Pareto optimal in performance and cost.

In the context of system architecting, the decision support framework presented in this thesis has the following characteristics:

1. The ability to computationally capture the relationships between decision variables, and assess potential cost/performance tradeoffs between different system architectures.
2. The uses of Multi-Attribute Utility Theory (MAUT) to aggregate system performance contributions.
3. A novel approach to quantitatively calculate high-impact decision variables in order to steer the focus of the system architect towards decision variable that have a strong impact on the Pareto optimal solution set.
4. Provide a visual representation of high-impact decision variables within the network.

Before the framework is presented, the mathematical formulation of the system architecting process is defined to allow the reader to get an insight into the rationale behind each step of the framework.

6.2.1 Mathematical Formulation

The framework adopts a constrain-graph-based methodology to define the system architecting process as a COP. The COP approach offers a set generic tool sets that addresses many constraint satisfaction and optimisation problems, and is not restricted to systems architecting. Thus, the development of computational tools to model and solve COPs have been vast and are widely available for use by many open-source programming packages. The COP approach for the system architecting process can be formulated, as shown in equation (6.1).

Objectives, minimise	$\mathbf{J} = \mathbf{f}(\mathbf{x}) = [(f_1(\mathbf{x}), f_2(\mathbf{x})]$		
Decision Variables	$\mathbf{X} = [\mathrm{x}_1, x_2, \ldots\ldots, x_N]$		
Domain of value assignments	$D_{x_i} = [v_1^{(i)}, v_2^{(i)}, \ldots\ldots, v_n^{(i)}]$	$i = 1,2, \ldots\ldots, N$	
Metric Constraints	$h_j(\mathbf{x}) = 0$	$j = 1,2, \ldots, p$	
	$g_k(\mathbf{x}) \leq 0$	$k = 1,2, \ldots\ldots, m$	
	$\mathrm{x}_q^L \leq \mathrm{x}_q \leq \mathrm{x}_q^U, .$	$q = 1,2, \ldots\ldots, z$	
Assignment Constraint	$C_l = \langle scope, rel \rangle_l$	$l = 1,2, \ldots\ldots, N_c$	(6.1)

The objectives $\mathbf{f}(\mathbf{x}) = [f_1(\mathbf{x}), f_2(\mathbf{x})]$ is a vector of the two objective functions, one represents the Multi-Attribute Utility (MAU) function, and the other represents the Life-Cycle Cost (LCC) function. Ross et al [38] provide a summary of the applicability of MAUT in aerospace systems optimisation. The derivation of the MAU function depends on the conformance of system attributes to the independence conditions. A multiplicative form of the utility function is derived if the utility independence condition is upheld, else an additive form of the MAU function is derived if the additive independence condition is upheld. Utility independence is assumed if the decision maker is indifferent between a lottery scenario and a certainty equivalent, and is not dependent on the level of other attributes [62]. This allows the multiplicative form of the MAU function to be defined as:

$$K \cdot U + 1 = \prod (K \cdot k_i \cdot U(a_i) + 1) \text{ where } K = -1 + \prod_{i=1}^{n} (K \cdot k_i + 1) \tag{6.2}$$

U represents the overall utility of satisfying multiple objectives, $U(a_i)$ represents the utility of system attribute $a_i \in \mathbf{A}$, k_i is the individual scaling factor or weighting factor for the i^{th} system attribute, and K is the normalising parameter that insures consistency between the definitions of U and $U(a_i)$. The additive condition is assumed if the preference order for lotteries does not depend on the joint probability distribution of these lotteries, but only on their marginal probability distribution [62]. The additive MAU function is defined as:

$$U = \sum_{i=1}^{N} k_i U(a_i) \text{ where } \sum k_i = 1 \tag{6.3}$$

The constraints associated to the system architecting problem are sub-categorised into metric constraints and assignment constraints, which follows the definitions presented by Rayside and Estler [107].

- Assignment constraints as defined *the assignment of a certain value, which influences the assignment of other decision variables.*

- Metric constraints are defined as *constraints that ensure that a solution respects certain metric properties. For example, overall performance of a solution must be greater than a value* Y.

Assignment constraints represent logical constraint statements that define the compatibility of design alternatives within the domains of different decision variables. Equation (6.1) defines these constraints as a pair $\langle scope, rel \rangle$, where $scope$ is a set of decision variables that participate in the constraint relation, and rel is the relation that defines the domain values that the decision variables can take on [102]. Assignment constraints can essentially be treated as a CSP $(\mathbf{X}, \mathbf{D_X}, C_\mathbf{X})$, where $\mathbf{X}$ is a set of decision variables $\mathrm{x}_i \in \mathbf{X}$ that ranges over a finite domain $D_{\mathrm{x}_i} \in \mathbf{D_X}$. $C_\mathbf{X}$ defines a set of constraint relationships, relating to the set of decision variables $\mathbf{X}$. Solutions to a CSP are found when all constraint functions are satisfied i.e. when $C_\mathbf{X} = \mathbf{true}$, where the vector $\mathbf{true} = [true_1, true_2, true_3, \ldots, true_{N_c}]$, and N_c represents the total number of constraints. The optimisation of the objective functions and the satisfaction of the metric constraints are accounted for in the optimisation loop. The optimisation loop uses multi-objective optimisation algorithms to parametrically vary the design variables $\mathrm{x}_\mathrm{q} \in \mathbf{x}$, which provide a parametric representation of a system architecture within the continuous domain. The use of optimisation algorithms also enables the satisfaction of equality constraints $h_j(\mathbf{x}) = 0$, inequality constraints $g_k(\mathbf{x}) \leq 0$, and the upper and lower bound constraints $\mathrm{x}_q^L \leq \mathrm{x}_q \leq \mathrm{x}_q^U$, to find Pareto optimal solutions for each system architecture. The concatenation of the Pareto fronts from all feasible system architectures into a global Pareto front will result in a set of Pareto optimal architectural solutions.

6.2.2 Framework Representation

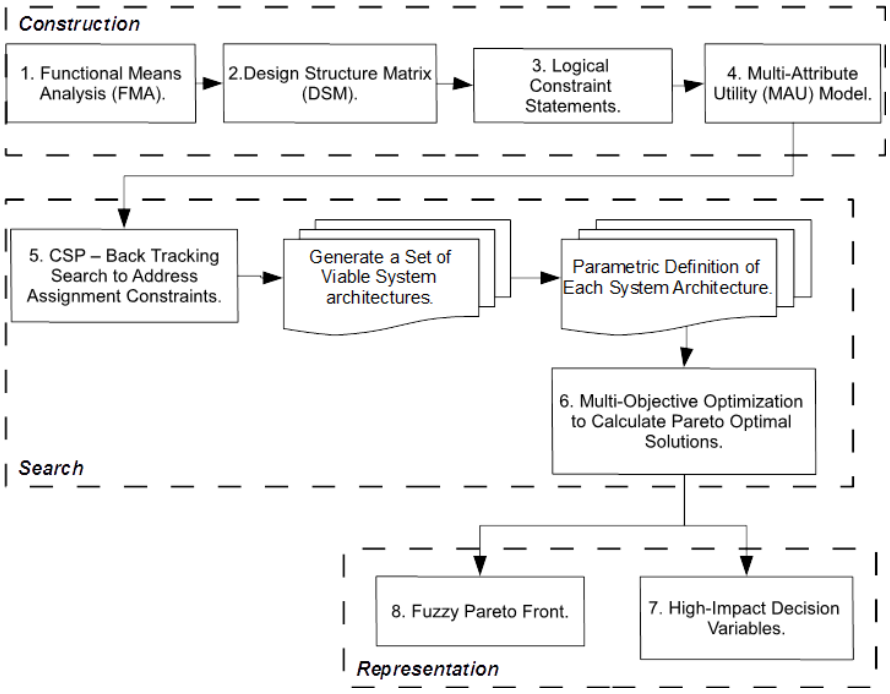


Figure 6.1: Decision support framework workflow.

Figure 6.1 represents a flow diagram of the developed framework, which can be categorised into three phases, *Construction*, *Search*, and *Representation*. The novelty of calculating high-impact decision-variables and visualising them is presented within the *Representation* phase. The *Construction* and *Search* phases are required to derive information that is needed to calculation high-impact decision variables. This is achieved by capturing the knowledge of the decision-maker, and calculating Pareto optimal solutions of different system architectures.

The *Construction* phase effectively acts as an enabler to defining the system architecting process as a COP. This is achieved by applying FMA and DSM to define a set of decision variables $\mathbf{X}$ of interest, domains of design alternatives D_{x_i} for each decision variable, a set of logical constraint functions between different decision variables $C_{\mathbf{X}}$, and a utility model to measure system performance. The *Search* phase uses logical constraint statements to define assignment constraints between decision variables, in order to find viable system architectures. This is achieved by employing the backtracking constraint-satisfaction algorithm to remove infeasible system architectures. Once a set of viable system architectures are identified, we continue to define a parametric representation of each system architecture. This in-turn enables the multi-objective optimisation algorithm to parametrically change each design variable $\mathbf{x}$ to calculate the Pareto fronts for each system architecture.

Once this is accomplished, data accumulated from multi-objective optimisation and DSM are used to represent information that is deemed vital to the decision maker in selecting a set of candidate design alternatives. Pareto optimal solutions are visualised using the Fuzzy Pareto front representation .[114]), and high-impact decision variables are visually represented using a network diagram of decision variables. The following sub-sections will provide details of the methods that are deemed vital in enabling the calculation of high-impact decision variables. This includes the back-tracking search algorithm that is used in the *Search* phase, the fuzzy Pareto front representation, and the calculation of PageRank centrality to measure high-impact decision variables.

The following sections provide a detailed description of the methods and tools used in each of the three phases, and a step-by-step process of constructing a decision support framework.

6.2.3 Construction

Before describing the *Construction* phase it is worth describing the use of the term 'system abstraction', in relation to physical and functional decomposition of a system. A system, in its most basic form, can be described as a collection of elements, where certain combinations of elements in a system can be viewed as a system in itself, which are regarded as subsystems of a larger system [18]. This suggests that there is a hierarchical relationship between a system and its elements, leading to the concept of a hierarchical structure. Within a

hierarchical structure a system exists at a certain level of abstraction, and its associated elements exist at a lower level of abstraction. This is referred to as system levels [18]. Thus, a system described in terms of its elements can be defined by the process of decomposition, where the system is decomposed into its lower level elements until the lowest element (or leaf element) of the hierarchy is reached. For example, consider an aircraft, which can be decomposed into a number of elements such as the wing, fuselage, and engine. These elements can in itself be considered as a system, and be decomposed into its own grouping of elements. Taking the jet engine for example, this can be further decomposed into inlet, compressor, combustor, turbine, and exhaust. A notional illustration of the hierarchical structure of an aircraft system is presented in Figure 6.2.

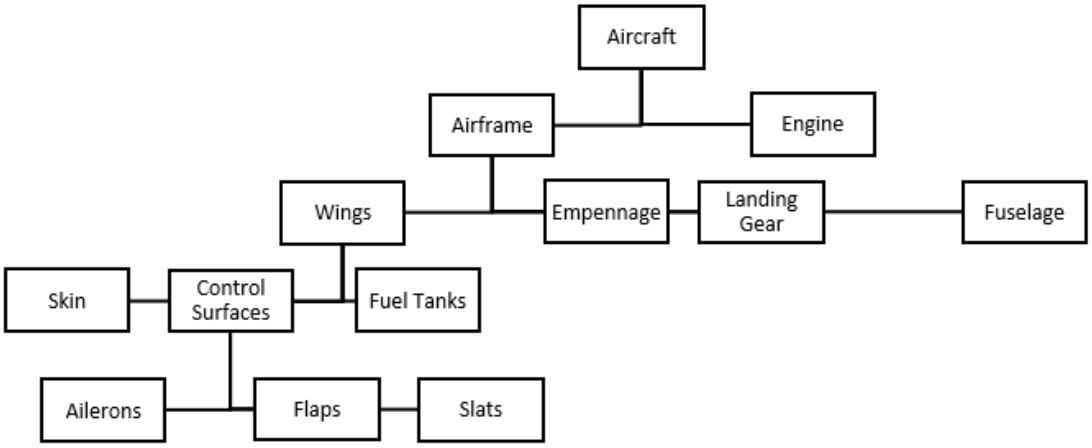


Figure 6.2: Notional physical decomposition of an aircraft [18]**.**

The discussion thus far has been focused towards the physical decomposition of a system. However, the same principals of hierarchical structure and decomposition can also be applied to functional decomposition. The functional decomposition of a system is achieved via requirements analysis, the process works by categorising the requirements based on their needs or wants to the stakeholder.

The categorisation in general results in two groups of requirements, functional and non-functional (or performance) requirements. A functional requirement is defined as what the system must do in order to meet its operational requirement. A non-functional requirement states to what extent the system needs to perform, or be implemented to meet the expectations of the stakeholders. Non-functional requirements tend to represent constraints placed on the design space in which the system must evolve, and are generally represented in terms of quality, quantity, scope, and availability [18].

Functional hierarchy establishes what the system has to do at each level of abstraction, and how well it must do it at each level. The root nodes of the functional hierarchy represent top-level functional requirements that describe the fundamental needs that the system must meet. Traversing down the hierarchy tree are lower level functional elements that describe the functional purpose of subsystems. Following on from this, non-functional requirements

associated with top level-functions are allocated to lower level functions. This process of decomposition is used to define the functional architecture at a greater level of detail through each level of the system. A notional example of a functional decomposition of an UAS, and its associated performance requirements have been represented in Figure 6.3.

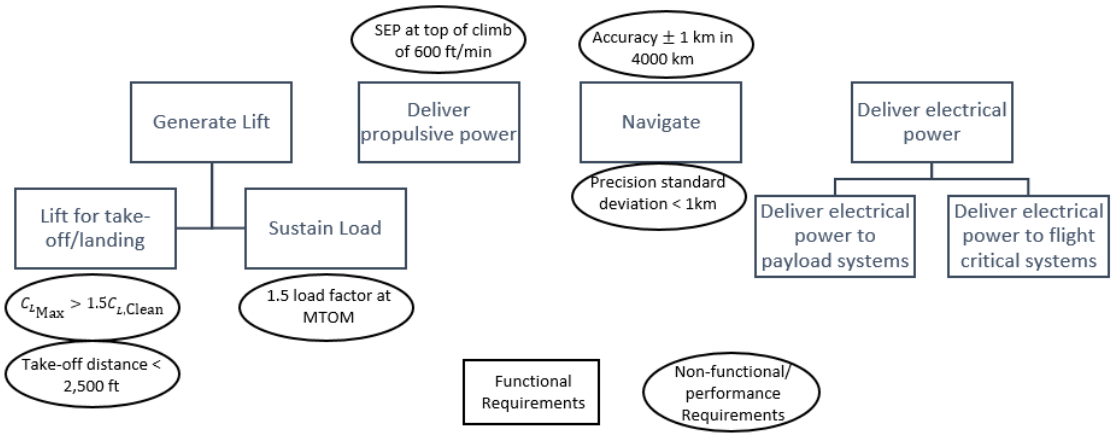


Figure 6.3: Notional functional decomposition of an UAS [18]**.**

Now that the concepts of functional and physical decomposition have been established, this section follows on to define design synthesis. The process of synthesis aims to generate a physical architecture of a system that performs all the functions established in the functional hierarchy. Design synthesis is achieved by linking the physical elements and the functional elements at the same level in the hierarchy structure. This thesis mainly focuses on the conceptual design stage, where the level of abstraction is limited to the top level of the hierarchy.

Though the design synthesis task is intrinsically functional, the decision maker may in reality carry through assumptions about the expected physical architecture and then perform functional decomposition with the notion of what subsystems will enable system level functionality [18].

The aim of the *Construction* phase is to apply the FMA approach to identify top-level system functions, and in-turn define a set of design alternatives that are linked to satisfying each function. The level of decomposition (at a given level of system abstraction) typically flows down to either one or two levels of the hierarchy until design alternatives can be defined. However, when performing functional decomposition the aim should be to reduce the number of levels in the hierarchy in order to minimise branching in the hierarchy tree. For example, consider the functional decomposition of the launch and recover systems, as shown in Figure 6.4. The launch and recover system is first decomposed into launch system and recovery system, and then further decomposed into convention and unconventional launch and recovery systems. This allows the physical architecture to be decoupled into different domains of design alternatives.

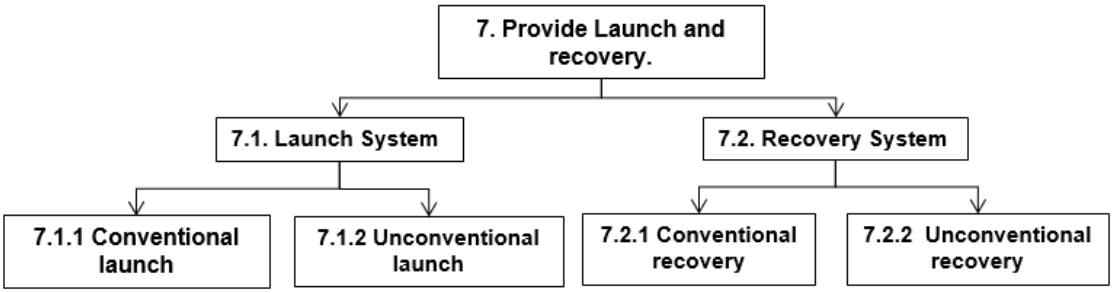


Figure 6.4: Functional decomposition hierarchy.

The decision support framework presented in this thesis formulates decision variables by taking only the leaf nodes of the functional decomposition hierarchy. Therefore, design alternatives may only be prescribed to the leaf nodes of the hierarchy, and only the leaf nodes of the hierarchy are carried forward into the COP representation of the system architecting process.

6.2.3.1 Decision Support Matrix

The reduction of the architectural solution space in the FMA table can exhibit interrelationships between different decision variables i.e. certain design alternatives might have been removed in one decision variable due to the decision made in selecting a particular design alternative in another decision variable – representative of Assignment constraints. The DSM is used as a first pass approach in capturing these interrelationships, which requires the decision maker to specify the existence of connections between decision variables when the solution space is reduced. To simplify the input of data, the DSM is assumed to be symmetrical. For example, if function A_1 has an alternative that is incompatible with an alternative in function A_2 then the alternative in function A_2 is also incompatible with the alternative in function A_1 i.e. the interrelationship is assumed to be undirected. This matrix representation can be directly translated into a graph representation of an undirected network.

6.2.3.2 Logical Constraints

Given a set of design alternatives, an overall system architecture can be generated by selecting one design alternative from each decision variable. However, certain combinations of design alternatives might be undesirable or technically infeasible, thus need to be removed from the solution space. These compatibility relationships can be represented as logical constraints, which are defined as a statement describing the relationship between a set of decision variables. Logical constraints provide a means of transforming a constraint statement defined by the decision maker into a programming environment, which can then be translated into the *Search* phase of the framework to remove infeasible system architectures. The logical constraints are constructed using simple logical operators and statements such as: *If/ else/ elseIf.*

Table 6.1: A list of logical operators used to construct logical constraints.

Symbol	Definition	Symbol	Definition
==	Equals to	>	Greater than
||	OR statement	<	Less than
&&	AND statement	>=	Greater than or equal to
~=	Not Equals to	<=	Less than or equal to

A set of viable system architectures can only be derived if none of the specified constraints are violated in the solution space. For example, if a compatibility constraint exists between decision variable A_1 and A_2 where if alternative 1 or 2 is selected in function A_1 then alternative 3 is not feasible in function A_2. This is represented in a mathematical form as, $(func\ A_1 == Alt1||Alt2)\ \ then\ \{func\ A_2 \sim = Alt3\}$.

6.2.3.3 Multi-Attribute Utility Model

The construction of the MAU model makes use of methods available in literature from systems engineering and decision theory, most of which stems from the work of Keeney and Raiffa [102]. Zhang et al.[115] provide a detailed overview in developing a MAU model at different levels of system decomposition, where three kinds of value models are prescribed, customer value model, system value model, and component value model. This thesis largely focuses on the development of a customer value model that identifies the impact of an overall system architecture rather than component and sub-system level designs. The decision support framework adopts the multiplicative form of the utility function as the utility independence assumption is a less restrictive independence assumption compared to additive independence. The attraction with using a multiplicative model as opposed to an additive one is that the additive model cannot express the value of any interactions between different attributes. Also, the multiplicative model allows the inclusion of proxy attributes[7] and many-to-one mappings between the objective and the attribute set.

6.2.3.3.1 Constructing Utility Functions

Utility function describes the decision makers preference relationship to a set of system objectives $\{O_1, \ldots\ldots, O_J\}$, due to changes in its associated system attributes $\{a_1, \ldots\ldots\ldots, a_N\}$. The shape of the utility curve is determined by the preference relationship of the attribute towards the objective i.e. maximise, minimise, or optimise, and the decision makers attitude towards risk. The assessment of risk attitude is based on the decision maker's preference to an uncertain outcome. Uncertainty is captured by defining a set of lottery scenarios to the decision maker, and determining the indifference probability value between the two lottery

[7] A proxy attribute is the level of an attribute is valued only for their perceived relationship to the satisfaction of a fundamental objective.

scenarios. The indifference probability is the probability value at which the decision maker is indifferent to the outcomes of the defined lottery scenario.

However if the attribute set is large, the process of determining the indifference probability values can become tedious, and not practically applicable in a decision-making context. To simplify the construction of utility relationships, general functional forms are assumed to represent different risk and preferential attitudes of the decision maker. These preference relationships are captured by three different functional forms, maximise, minimise, and optimise. In the case of maximise and minimise function, the risk attitude of the decision maker is represented by three different functional forms, risk averse (concave), risk neutral (linear), and risk prone (convex). This is illustrated in Figure 6.5. A^* and A_0 represent lower upper and lower bounds of an attribute. The indifference probability is determined based on the decision makers choice of risk attitude.

- Risk Prone – $P_e = 0.2$
- Risk Neutral – $P_e = 0.25$
- Risk Averse – $P_e = 0.36$

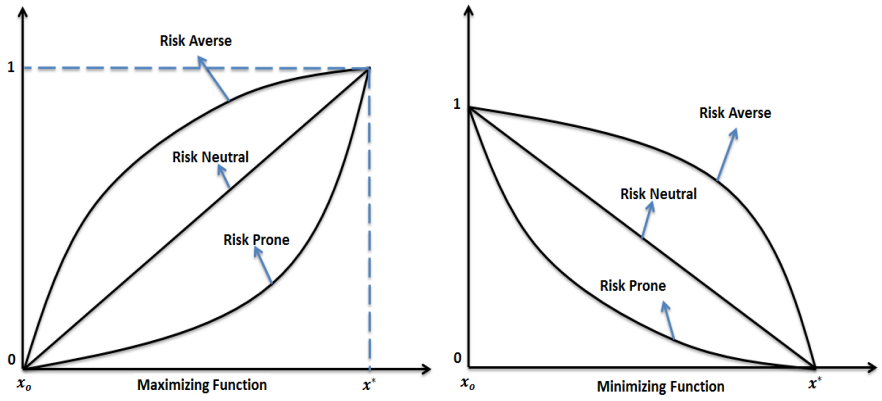


Figure 6.5: Representation of the maximising and minimising preferential utility functions with different risk attitudes.

The indifference probability value is translated to a utility value by multiplying the indifference probability by two. This utility value corresponds to the average of the upper and lower bound values of the attribute.

$$U = 2 \cdot P_e \rightarrow U\left(\frac{a + a^*}{2}\right) = 2 \cdot P_e \tag{6.4}$$

As a first order approximation these three points are used to define the utility function. Previous studies have applied exponential or linear functions to characterise the utility curve [116].However, the use of exponential functions results in a set of non-linear equations that requires an iterative procedure to find a solution. To reduce the complexity, whilst at the same time maintaining accuracy of the utility function, it was decided to interpolate between the three points.

The optimal utility function is represented by means of a normal distribution, with the mean and standard deviation known. The choice of a normal distribution as a utility function implies that the decision maker feels that some value of an attribute is optimal, which is represented by the mean of the normal distribution. The standard deviation of the normal distribution represents the tolerance of the decision maker to a deviation of the value from the optimal point. Since two thirds of the area of the normal curve lies within one standard deviation of the mean, the decision maker should specify a range about the optimal point such that the decision maker is willing to tolerate knowing that there is a two-thirds chance that the value will lie within this interval i.e. the decision maker should estimate the value of the standard deviation such that they would be willing to tolerate the attribute value falling within the interval $a_{opt} \pm \sigma_{tol}$ with odds of two out of three. The optimal utility function is represented as follows:

$$y = f(a|\mu,\sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{-(a-\mu)^2}{2\sigma^2}} \quad \rightarrow \quad U(A) = \frac{y}{\max(y)} \tag{6.5}$$

6.2.3.3.2 Scaling Constants

Referring back to the MAU equation, the scaling constants are defined as k_i. These are individual scaling factors for each dimension of the attribute set $\{a_1, \ldots\ldots\ldots, a_N\}$. They indicate the value trade-off between various pairs of attributes. In the assessment of scaling constants a set of equations are constructed with $k-$parameters as unknowns. These parameters are then calculated by solving a set of simultaneous equations. However, to solve these equations a pair of lottery scenarios are required, which define the indifference probability of the decision maker. Keeney [62] suggests that in practice it is best to fix the values of $N-2$ attributes and vary two attributes at a time.

This process is illustrated using the following example. Suppose we have three attributes a_1, a_2, and a_3, the aims is to calculate the three scaling constants k_1, k_2, and k_3. First, value trade-offs between attributes a_1 and a_2 are assessed by changing the values of these attributes and keeping a_3 fixed. This is achieved by defining the following indifference scenario, the decision maker is asked to identify the value of attribute a_1, call it a_1', such that (a_1', a_{2o}, a_{3o}) is indifferent to (a_{1o}, a_2^*, a_{3o}). This leads to the following equation.

$$k_1 U_1(a_1') = k_2 \tag{6.6}$$

Similarly, another indifference scenario is setup between attributes a_1 and a_3, where the decision maker is asked for a value of attribute a_1, call it a_1'', such that (a_1'', a_{2o}, a_{3o}) is indifferent to (a_{1o}, a_{2o}, a_3^*) . This leads to the following equation.

$$k_1 U_1(a_1'') = k_3 \tag{6.7}$$

If the MAU model is additive, then the sum of the scaling constants must equal one.

$$k_1 + k_2 + k_3 = 1 \tag{6.8}$$

Using the equations (6.6–6.8) the scaling constants can be calculated. However, if the MAU model is multiplicative the sum of the scaling constants is no longer equal to one, and an additional parameter needs to be assessed, which is the normalising parameter K. This results in four unknowns k_1, k_2, k_3, and K. In order to identify a closed-form solution to this problem another equation is required, which is defined as follows:

$$K + 1 = (Kk_1 + 1)(Kk_2 + 1)(Kk_3 + 1) \tag{6.9}$$

The solution to the above equation requires qualitative information from the decision maker, which is ascertained by presenting the decision maker with another lottery scenario. Using a probabilistic scale, the decision maker is asked to identify a probability value P_1 such that (a_1^*, a_{2o}, a_{3o}) is indifferent to $[(a_1^*, a_2^*, a_3^*)\ , P_1;\ (a_1^o, a_2^o, a_3^o)]$. The answer to which will result in the following equation:

$$k_1 = P_1 \tag{6.10}$$

However, the calculation of the normalising parameter K using equation (6.10) is not straight forward, as the equation is non-linear and requires an iterative scheme to obtain the solution. The convergence to a solution is facilitated by the fact that the normalising parameter is bounded by the sum of the scaling factors.

$$\sum k_i < 1.0 \text{ Implies } \quad K > 0$$

$$\sum k_i > 1.0 \text{ Implies } -1 < K < 0 \tag{6.11}$$

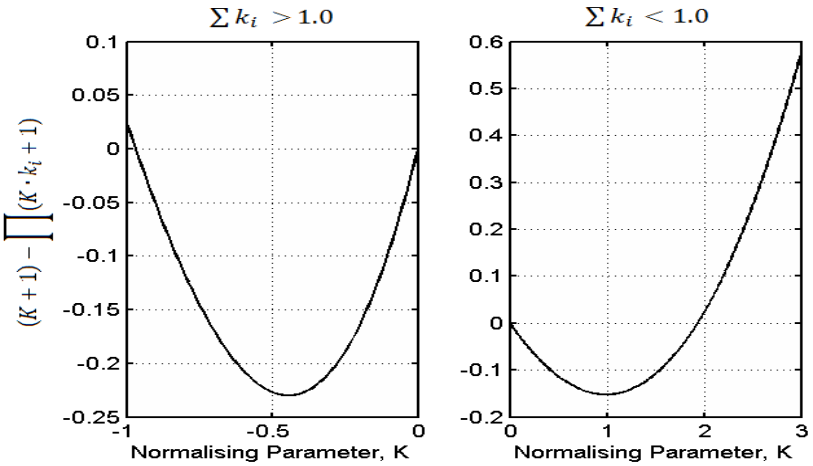


Figure 6.6: Illustration of the (n-1) dimensional polynomial equation of the normalising parameter – $n = 3$.

The method for solving the normalising parameter, K, by iterative means depends on the number of scaling factors, as the shape of the polynomial changes with the number of scaling factors (i.e. $(n-1)$- dimensional polynomial). This thesis adopts an iterative scheme based on Brent's method, which is a root finding algorithm that makes use of root bracketing, bisection, and inverse quadratic interpolation [117]. A brief overview of Brent's methods is

provided in Appendix C, for a detailed derivation of Brent's method the reader is referred to Brent [118].

The calculation of the scaling constants can be tedious, and is not always intuitive when presented to the decision maker. This is the major drawback in applying MAUT and care must be taken when assessing the scaling constants. To simplify the assessment process it is better to first identify the rank order of the scaling constants qualitatively based on discussions with the decision maker. Then, the process of calculating scaling constants can be carried out to verify that the rank order still holds.

6.2.4 Search

The *Search* phase aims to address Assignment constraints by representing the problem as a CSP ($\mathbf{X}$, $\mathbf{D_X}, C_\mathbf{X}$). The aim of the CSP methodology is not to optimise the system architecture, but to remove infeasible system architectures that do not satisfy the logical constraint relationship. To solve a CSP we need to define a state space and the notion of a solution. Each state in a CSP is defined by an assignment of values to some or all of the variables $\{x_i = v_i, x_j = v_j, \dots\}$ [102]. To ensure that all infeasible solutions are removed, a recursive backtracking algorithm is applied to search the solution space. Backtracking is a form of depth-first search that chooses one variable at a time and backtracks if no feasible solution is found [102]. A conceptual representation of the backtracking search algorithm is represented in Figure 6.7.

Algorithm: Backtracking algorithm for CSP.

```
function BACKTRACKING SEARCH (CSP) returns solution set, or failure
1       return BACKTRACK ( {}, CSP)
function BACKTRACK (assignment, CSP) returns a solution, or failure
2       if assignment is complete then return assignment
3       var ← HEURSTIC (CSP)
4       for each value in ORDER-DOMAIN-VALUES (var, assignment, CSP) do
5           if value is consistent with assignment then
6               add {var = value} to assignment
7               inference ← INFERENCE (CSP, var, value)
8               if inference ≠ failure then
9                   add inference to assignment
10                  result ← BACKTRACK(assignment, CSP)
11                  if result ≠ failure then
12                      return result
13          remove{var = value} and inference from assignment
14      return failure
```

Figure 6.7: Backtracking algorithm for CSP, with general purpose heuristics and inference functions [102]**.**

Backtracking search as a concept is simple and easy to implement, however problems arise when multiple constraint functions are added, and time and memory requirements are

limited. In order to improve the performance of backtracking search some simple heuristics are added to the search algorithm- $var \leftarrow \text{HEURSTIC}\ (CSP)$. The aim of the heuristic function is to answer the following question: *Which variable should be assigned next* $(var \leftarrow \text{HEURSTIC}\ (CSP))$*, and in which order should its values be tried* (ORDER-DOMAIN-VALUES) ? [102]. The simplest heuristic strategy to address decision variable assignment is the Select Unassigned Variable (SUV) heuristic, which chooses the next unassigned decision variable in an ordered list $\{x_1, x_2, \dots, x_N\}$. An extension of SUV is the Minimum Remaining Values (MRV) heuristic, which picks a decision variable that is most likely to cause a failure soon, and thereby pruning the search tree [102]. The algorithm applied in this study uses the MRV heuristic as a means for decision variable assignment.

Once the decision variable has been assigned, the algorithm must choose the order in which the domain of values will be examined. However, in our case the aim is to find all the solutions to the constraint problem, and not just the first one, thus the ordering of the domain of values does not matter. In this instance the ordering is assigned randomly within the domain of each decision variable.

To improve the efficiency of the search process an inference function $inference \leftarrow$ INFERENCE $(CSP, var, value)$ is added to the search procedure. This thesis adopts the forward-checking inference function to reduce the domain of values of neighbouring decision variables, such that backtracking search has less chance of failure. Forward-checking works by establishing arc-consistency[8] for the assigned decision variable x_i, which works by taking each of the unassigned decision variables x_j that are connected to x_i by a constraint, and deleting from x_j's domain any value that is inconsistent with the value chosen for x_i. For a more complete discussion on CSPs, and the methods used to solve them the reader is referred to Russell and Norvig [102].

In the above definition of a CSP, each decision variable should have one solution assigned to it after the constraints have been applied. If the application of constraints results in one or more of the decision variables to have no solution, then the backtracking algorithm fails and the CSP is unresolved. However, in some cases of the system architecting process, this is exactly what is required. For example, in some instances the decision maker might prefer to dynamically remove an entire domain of alternatives D_i belonging to a particular decision variable X_i. This can be achieved with the inclusion of a dummy value $null$ into the domain of alternatives D_i. The $null$ value states that no solution exists for the function node X_i, however as far as the backtracking algorithm is concerned the $null$ value is seen as a solution.

To illustrate this concept consider the following notional example. The design of an Unmanned Aircraft System (UAS) introduces flexibility as to how an UAS can be launched and

[8] A decision variable x_i is arc-consistent with respect to another decision variable x_j if for every value in the domain D_i there is some value in the domain D_j that satisfies the constraints on the arc (x_i, x_j) [102].

recovered. This can range from a conventional launch and recovery system to unconventional launch and recovery, or a combination of both conventional and unconventional launch and recovery mechanisms. The functional decomposition of the launch and recovery system is represented in Figure 6.8.

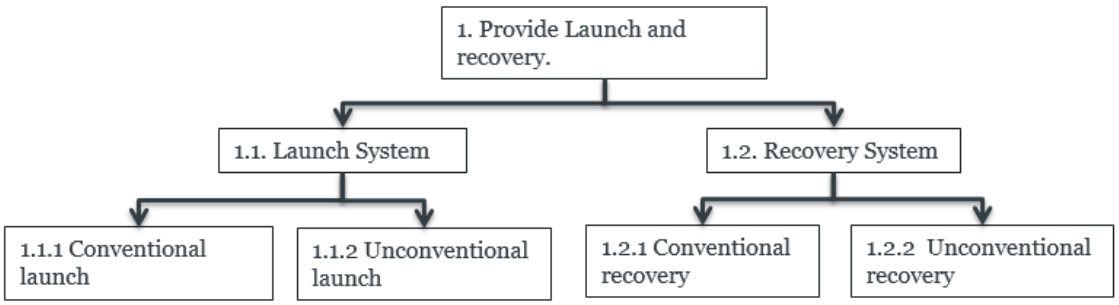


Figure 6.8: Notional functional decomposition for a launch and recovery system.

As previously defined, only the leaf nodes of the hierarchy are taken forward to the design synthesis phase, where design alternatives are allocated to each decision variable. This is illustrated with the following example, shown in Table 6.2.

Table 6.2: Notional functional-means analysis for a launch and recovery system.

1.1.1	**Conventional launch**	Quadricycle	Bicycle	Tricycle	*null*
1.1.2	**Unconventional launch**	Rail launch	Tensioned Line Launch	Ground-vehicle launch	*null*
1.2.1	**Conventional recovery**	Conventional landing	*null*		
1.2.2	**Unconventional recovery**	Skid and belly recovery	Net recovery	Cable-assisted recovery	*null*

The removal of an entire domain of values in a decision variable, based on the choice of solution in another decision variable is achieved by defining the constraint function, such that it points to the *null* value in the domain to be removed. This is illustrated by applying the following constraint functions for the launch and recovery system:

- If a solution from *Unconventional Launch* domain is selected then the solution set for *Conventional recovery* domain should be removed. This is achieved by defining the constraint function as:

$$\textbf{if } \{Conventional\ launch == null\} \textbf{ then } \{Unconventional\ recovery \sim = null\}$$

- The same logic as above is also applied if a solution from *Conventional recovery* domain is selected, resulting in the solution for the *Unconventional recovery* domain to have a *null* value.

$$\textbf{if } \{Conventional\ recovery \sim = null\} \textbf{ then } \{Unconventional\ recovery == null\}$$

- The difficulty comes in defining a means of combining solutions from *Conventional launch* and *Unconventional launch,* and a solution from *Unconventional recovery.* Let's assume that only the following combination is required: tricycle with tension line launch for the launch mechanism, and cable assisted recovery for the recovery mechanism. Let's also assume that the cable assisted recovery solution can only be selected if tricycle and tension line launch are selected. The implementation of these constraints is achieved by defining the following constraint functions:

  if $\{Conventional\ launch ==' Tricycle'\}$ **then** $\{Unconventional\ launch ==' Tensionline\ launch' || null\}$

  if $\{Conventional\ launch \sim =' Tricycle'\}$ **then** $\{Unconventional\ launch == null\}$

  if $\{Unconventional\ launch ==' Tension\ line\ launch'\}$ **then** $\{Unconventional\ recovery ==' Cable\ assisted\ recovery'\}$

  if $\{Unconvntional\ recovery ==' Cableassisted\ recovery'\}$ **then** $\{Unconventional\ launch \sim =' Rail\ launch' || 'Ground\ vel$

  if $\{Conventional\ launch ==' Quadricycle' || 'Bicycle'\}$ **then** $\{Conventional\ recovery ==' Conventionallanding'\}$

The last constraint in the above list is implemented to ensure that only the tricycle configuration is able to combine with a value from the *Unconventional recovery* domain. Applying the backtracking algorithm with the defined constraint functions, results in 8 feasible system solutions to be identified, these are represented in the table below:

Table 6.3: Architectural solutions for the notional launch and recovery system.

	Function 1.1.1	Function 1.1.2	Function 1.2.1	Function 1.2.2
1	*null*	Rail Launch	*null*	Net recovery
2	*null*	Rail launch	*null*	Skid and belly recovery
3	*null*	Ground vehicle launch	*null*	Net recovery
4	*null*	Ground vehicle launch	*null*	Skid and belly recovery
5	Quadricycle	*null*	Conventional landing	*null*
6	Bicycle	*null*	Conventional landing	*null*
7	Tricycle	*null*	Conventional landing	*null*
8	Tricycle	Tension line launch	*null*	Cable assisted recovery

6.2.5 Representation

The final phase of the decision support framework, the *Representation* phase, is the most demanding as it encapsulates both the simulation and the data analysis aspect of the decision support framework. The output of the *Search* phase is a set of feasible system architectures, which need to be evaluated in order to calculate the utility and LCC objective function values of the design, and ensure that the Metric constraints are satisfied. This requires the

development of analytical/ semi-empirical models which are used to calculate system level performance attributes and evaluate the objective functions. The utility and LCC objective functions of the design are in-turn fed into an optimisation framework to calculate Pareto-optimal solutions.

6.2.5.1 Pareto Front Representation

Non-dominated architectural solutions are represented by a Pareto plot of utility vs. Life-Cycle Cost (LCC). However, rather than eliminating all dominated solutions Smalling and De Weck [114] suggest to retain some of the solutions close to the Pareto front, as they might be more robust to implement or could provide other advantages that are not yet identified at the early conceptual design stage. This is represented by a fuzzy Pareto filter, where a relaxation factor K_f is introduced to retain solutions close to the global Pareto front. This thesis makes use of the fuzzy Pareto front representation to retain solutions close to the s-Pareto front that could indicate potential solutions for further analysis in order to identify their cost-utility benefits. The fuzzy Pareto front is represented as follows, $\mathbf{x^1}$ dominates $\mathbf{x^2}$ fuzzily if:

$$J_i(\mathbf{x^1}) + K_f\left(J_i^{\max} - J_i^{\min}\right) \leq J_i(\mathbf{x^2})\ \forall i \in \{1,2,\ldots.,k\},$$
$$\text{and } J_i(\mathbf{x^1}) + K_f\left(J_i^{\max} - J_i^{\min}\right) \leq J_i(\mathbf{x^2}) \text{ for at least one } i \text{ with } K_f \in [0,1]. \quad (6.12)$$

Setting the value of K_f to 0 would represent the weakly dominated Pareto front and setting the value of K_f to 1 would represent all the solutions in the objective space. If the value of K_f is in-between 0 and 1, solutions within the $K_f\left(J_i^{\max} - J_i^{\min}\right)$ hyper-rectangle offset from the s-Pareto front would be selected. This is graphically illustrated in Figure 6.9.

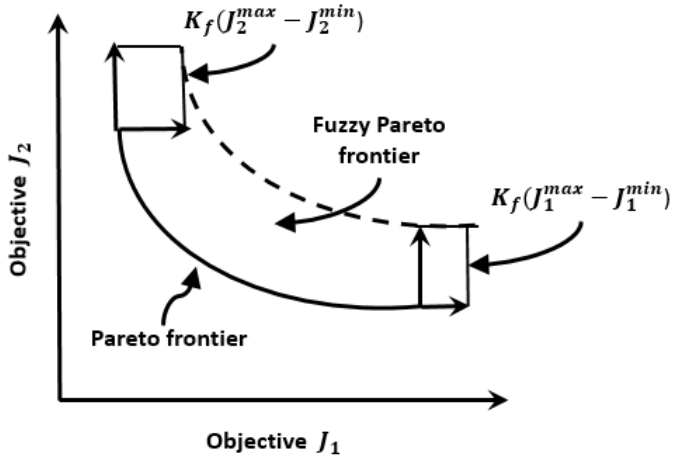


Figure 6.9: Illustration of the fuzzy Pareto front [119]**.**

Thus, the choice for the value for K_f has a direct impact on the solutions captured by the fuzzy Pareto filter. Guidelines for selecting a value for K_f is given by Smaling [119], where it is shown that increasing the value of K_f beyond the 0.4-0.6 range does not significantly improve design diversity. These values are chosen based on the design diversity metric

developed by Smaling [119], which consist of three parameters, design space envelop, number of designs contained in the design space envelope, and the dispersion of the designs within the design space envelope. However, the design diversity metric does not penalise solutions for being dominated solutions and far from the Pareto front, rather design diversity is the driving factor. To account for diversity and optimality the following set of metrics are proposed as a means of evaluating an appropriate choice of value for the relaxation factor K_f. These metrics are based on traditional hypervolume metrics and distance metrics [120]. The choice of an appropriate value for the relaxation factor K_f can be established by considering the following criteria:

1. Max dispersion of the filtered design space, MD_{DS}: This criterion is used as means of measuring the dispersion of solutions within the design space. It is represented as the average minimum distance between all points in the filtered data set, within in the design space. MD_{DS} is a summation over the Euclidian distance of each point i to its nearest neighboring point k. The nearest neighboring point k is found by taking the minimum of the Euclidian distance of all points $j = 1 \ldots . F - 1$ from point i. Where F is defined as the total number of points in the filtered data set, and $i \neq k$.

$$MD_{DS} = \frac{1}{D_P} \sum_{i=1}^{F-1} E_{\mathrm{x},i}.$$

$$\text{where } E_{\mathrm{x},i} = \min_{k=1,|F-1|} \sqrt{\sum_{j=1}^{M} \left(\frac{\mathbf{x}_j^{(i)} - \mathbf{x}_j^{(k)}}{\mathbf{x}_{j,max} - \mathbf{x}_{j,min}} \right)^2} \tag{6.13}$$

   Where E_{x} represents the Euclidian distance between points within the design space, and i and j can take values from 1 up to the size of the fuzzy Pareto optimal set F.

2. Max distance of the filtered data from the Pareto front, MD_{PF}: This criterion aims to measure the Euclidian distance of furthest data point of the filtered data set to the unfiltered Pareto front (i.e. when $K_f = 0$). This is achieved by first measuring the Euclidian distance between a point in the filtered data set $f(\mathbf{x})^i$ to all the points that lie on the Pareto front $f(\mathbf{x}_{PF})^k$**.** The nearest member to $f(\mathbf{x})^i$ is then determined by taking the minimum Euclidian distance value d_i between the two points. This is repeated for all the points in the filtered data set, resulting in a matrix of Euclidian distance values between the filtered data set and its nearest neighbour on the Pareto front $d_i \in \mathbf{d}$ where $i = 1, \ldots, F$.

$$MD = \max\ (\mathbf{d}), \text{ where } \mathbf{d}=[d_1, d_2, \ldots\ldots, d_{F,filtered}]$$

$$\text{and d } d_i = \min_{k=1,|PF|} \sqrt{\sum_{j=1}^{M} \left(f(\mathbf{x}_j)^{\boldsymbol{i}} - f(\mathbf{x}_{PF,j})^k \right)^2}. \tag{6.14}$$

   where $d_i \in \mathbf{d}$ and $i = 1, \ldots, F$, with F being the total number of points in the filtered data set. $k = 1, \ldots\ldots, K$, with K being the the total number of points that lie on the

Pareto front. M is the total number of objective functions, which in this case is equal to two.

6.2.5.2 High-Impact Decision variables

System architects at the conceptual design stage often face a challenge of selecting from a portfolio of potentially competing design alternatives, or technologies to create an overall system architecture. The choice of a system architecture has a significant impact on performance and cost attributes. This leads to the idea of measuring the impact of candidate architectures on the Pareto front.

To address the issue of architecture selection Mavris et al [121] proposed a method that is useful for sorting through a large set of candidate technologies to be infused in an aircraft system. The proposed methodology, known as Technology Identification, Evaluation and Selection (TIES), uses a GA to create a set of technology combinations. These combinations are compared with each other to identify superior and inferior solutions, the superior solutions are kept within the set while the inferior solutions are removed. The surviving solutions are used as parents in the next generation of system combinations. This is applied several times over many generations until the pollution converges to an optimal set of technology combinations. The TIES methodology is useful in identifying an optimal set of architectural combinations, but does not identify the impact of architectural changes on the objective values. De Weck and Chang [30] address the impact of architectural changes on the objective space by defining four metrics: (1) minimum utopia distance δ_{min}, (2) average utopia distance μ, (3) utopia point shift υ, and (4) the number of Pareto crossings χ. This is illustrated in Figure 6.10.

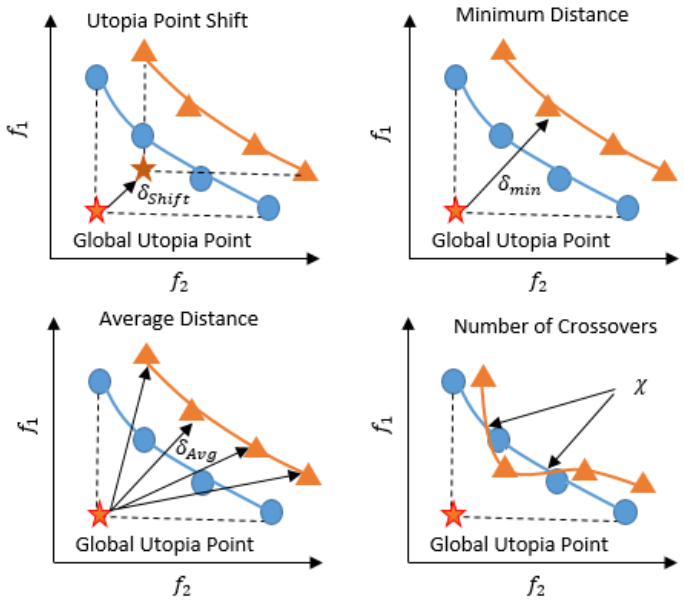


Figure 6.10: Pareto impact metrics [30].

The choice of an appropriate metric is dependent on the 'impact' characteristic that is of interest in capturing. From a systems architecting perspective important decision variables are defined as:

1. Decision variables that strongly influence system properties, such as utility value and LCC.

2. Decision variables that strongly influence the feasibility of other decision variables i.e. the connectivity of decision variables.

The 'impact' metric, from a systems architecting perspective, can be characterised by considering the three potential cases. First being the Pareto front of the system architecture X_i recedes from the global Pareto front. This suggests that the proposed architectural change has an adverse impact on the system under consideration. Second, if the entire Pareto front of concept X_i is moved towards the global utopia point, then the proposed architectural change shows promise. The third case is considered to be the most common [30], where there is crossover between Pareto fronts of multiple system architectures, resulting in the global Pareto set to be made up of solutions from multiple system architectures. Thus, in this case the proposed architectural change is optimal in certain regions of the objective space, and diminish in performance in other regions of the objective space.

Since we are interested in capturing the impact of architectural changes to the shift of the Pareto front, in relation to the global Pareto front, this thesis adopts one of the metrics presented by De Weck and Chang [30]. The influence of architectural changes on system properties, such as value and cost, is identified by measuring the shift in the utopia point of the Pareto front of architecture X_i from the global utopia point. This captures the ability of architectures to extend the solutions close to the global Pareto front. The shift in the utopia point is numerically measured by calculating the Euclidian distance of the utopia point from the global utopia point.

The Euclidian distance of the utopia point shift δ_{shift} for a given system architecture C_i is defined as:

$$\delta_{i\,,\text{shift}} = \sqrt{\sum_{j=1}^{k} (\mu_j^i - \mu_j^0)^2}$$

$$\text{with } j \in \{1, \ldots., k\} \text{ and } i \in \{1, \ldots., N\} \qquad (6.15)$$

Where $\delta_{i\,,\text{shift}}$ represents the utopia point shift of the Pareto front of architecture i, j represents the number of objectives, μ_j^i represents the utopia point value of objective j of the Pareto front of architecture i, and μ_j^0 represents the global utopia point value of objective j of the global Pareto front.

The impact of each function node on the utopia point shift can now be calculated by determining the sensitivity to the change in the utopia point shift due to architectural changes within the decision variables domain. This is measured by using a modified version of the main effects analysis, from the design of experiments (DOE) literature. This method was originally presented by Simmons [9] in the ADG framework. The main effect is a measure of the average change in a property caused by changing an independent variable at a specified level of another independent variable, or in this case by changing to a design alternative within the domain of the decision variable. However, the traditional main effects analysis is limited in its application from a system architecting perspective. The main effects analysis assumes that there is a baseline design and the decision variables are modified one at a time, and the modification of the decision variable is limited to two levels of assignment. To allow for more than two levels of assignment to each decision variable and not including a baseline design in the assessment, a modification to the traditional main-effects calculation is required. This modified version is represented by the Decision variable Sensitivity (DVS) equation, which is calculated over a feasible set of system architectures and their associated system properties by using the following equation:

$$DVS_{\delta_{\text{shift}},x_k} = \frac{\sum_{d_k \epsilon \widetilde{D_k}} |E(\delta_{\text{shift}}) - E(\delta_{\text{shift}}|x_k = d_{k,i})|}{|\widetilde{D}_k|} \tag{6.16}$$

where x_k is the decision variable k. $\widetilde{D}_k \subseteq D_k$ is the set of value assignments for x_k that exists as a feasible value assignment after the constraints have been applied to the decision variable x_k. $d_{k,i}$ is one of the alternatives in the decision variable x_k .The expression $d_{k,i} \in \widetilde{D}_k$ indicates that $d_{k,i}$ is a member of the set $\widetilde{D}_k$. $E(\delta_{\text{shift}})$ is the mean of system property δ_{shift} over all the feasible system combinations. $E(\delta_{\text{shift}}|x_k = d_{k,i})$ is the mean of all the feasible combinations with the decision variable assignment $x_k = d_{k,i}$. Finally, $|\widetilde{D}_k|$ is the number of elements of the set $\widetilde{D}_k$. The DVS is a measure of the average magnitude of change in the utopia point shift that occurs when changing the assignment of a decision variable. Here decision variables whom domain contains only one design alternative will have $DVS_{\delta_{\text{shift}},x_k} = 0$, as the assignment within its domain is fixed

An integral aspect of this framework is in its ability to quantify high-impact decision variables. To define the importance of each decision variable within a network, this thesis adopts centrality measures [113], which is defined as the measure of a node's importance within a network i.e. the most central node in the network. Depending on the network structure a variety of useful quantities or measures can be calculated that captures certain features of the network topology. This thesis proposes the application of PageRank centrality [10] to calculate high-impact decision variables. The method combines results from DVS and the adjacency matrix A_{ij} from the DSM to provide a quantitative measure of high-impact decision variables. The overall centrality measure is represented mathematically as follows:

$$x_k = \alpha \sum_j A_{kj} \frac{x_j}{k_j^{out}} + \beta DVS_{\delta_{\text{shift}}, x_k}, \quad (1)$$

where x_k is the centrality measure of node k. The terms α and β are positive constants, where $\alpha + \beta = 1$. The centrality measure is effectively a weighted sum of the non-network factor $DVS_{\delta_{\text{shift}}, x_k}$, and the network factor $A_{kj} \frac{x_j}{k_j^{out}}$. The term A_{ij} is the adjacency matrix of the network, where x_j are the neighboring nodes of node x_k, and k_j^{out} is the out degree of the neighboring nodes. It should be noted that both the non-network factor and the network factor are normalised, such that the values range from 0 to 1 before the summation takes place. This ensures that there are no scaling effects when the two terms are summed together. The centrality measure presented in this thesis is used as a relative measure of importance between decision variables in the network, and is used as a guide to identify the ordering of high-impact decision variables within the network.

6.3 Conclusion

The decision-making process in engineering design is considered to be challenging since decisions are generally non-routine, highly interconnected, and are significantly consequential on system properties. In identifying the shortfalls and gaps in the methods available in literature to aid the decision-making process this thesis has developed a coherent framework that supports the system architecting process. The framework brings together multi-objective optimisation methodologies, in combination with centrality measures from graph theory, and a modified version of the main effects analysis from the Design of Experiments (DoE) literature to identify an architectural exploration strategy with the aim of improving upon the solutions within the Pareto set.

The framework is split into three phases, *Construction*, *Search*, and *Representation*. The *Construction* phase facilitates the generation of system architectures and captures the relationships governing their choice by capturing the knowledge of the decision maker. One of the central aspects of this framework is in representing the architectural decision problem as a constraint optimisation problem. The problem is further sub-divided into a constraint satisfaction problem (CSP) to handle discrete logical constraints defining the feasibility of domain values of different decision variables. However, care should be taken when defining the constraint functions, as over constraining or under constraining the architectural solution space may result in too few or too many system architectures for the framework to be of any use in aiding the decision-making process. The framework also presents novel methods in visualising data generated from the analysis of system architectures. The representation of data is categorised into two parts, one representing the Pareto-optimal solutions on a cost-utility scale, and the other representing high-impact decision variables in the constraint network. High-impact decision variables are calculated by adopting a modified version of main effects analysis, where the sensitivity to the shift in the utopia point of the Pareto front from the global utopia point due to changes in design alternatives within its domain is

calculated. The decision variable sensitivity (DVS) data is then inputted into the network centrality measure, more specifically PageRank centrality, to classify high-impact decision variables.

Chapter 7: Case Study: Unmanned Aircraft Systems

The use of Unmanned Aircraft Systems (UASs) has many applications, covering a range of mission operations and system types [122]. This could represent military applications, such as the Predator, a medium-altitude long-endurance (MALE) UAS, to civil commercial applications such as the Zephyr, a lightweight solar-powered UAS. The attraction of using UASs mainly stems from the fact that there is no flight crew on-board the aircraft, which enables it to carry out missions that are otherwise seen as being "dull, dirty, and dangerous " [123].

Figure 7.1: Examples of different Unmanned Aircraft System (UAS) applications. Left image showing the Predator, and right image showing Zephyr.

Dull: These missions generally are long-duration with a low workload, and are considered to be repetitive in task that they perform. Such missions may include target coverage, communications relay, and air sampling [122].

Dirty: These missions would generally be considered dangerous for human crew, which might involve flying through contaminated air, high radiation environment, presence of biological agents, etc.

Dangerous: This mission types are generally focused towards military applications, where if flown with a manned aircraft it could potentially put a human life at risk. These missions include suppression of enemy air defence (SEAD), surveillance in enemy-controlled airspace, aerial targeting, etc.

The term 'system' in UAS suggests that there are multiple elements (which may include airborne and ground elements) that make-up an UAS. Typically, an UAS is made up of several key system elements, such as the aircraft itself, payload, communications systems, ground

control stations, launch and recovery systems, and other support equipment. Thus, the design and optimisation of an UAS depends on how these key system elements interact with each other, and the means by which they are integrated together.

7.1 Unmanned Combat Aircraft System Case Study

This case study focuses on the development of new propulsion system technologies for the future combat air system (FCAS) project. The development of future propulsion system architectures for the FCAS project was used as a case study to validate, and improve parts of the decision support framework presented in this thesis. The FCAS project is part of a research and concept development program undertaken by Rolls Royce plc, in conjunction with BAE systems, Dassault, Safran, Selex ES, Thales, and other industry partners. The FCAS project focuses on the development of an unmanned combat aircraft system (UCAS), in collaboration between the British and French governments. The development of future UCAS capabilities follows along the lines of Taranis and Neuron (see Figure 7.2), where the focus was on developing a stealth aircraft that is capable of supporting ISATR (Intelligence Surveillance Target Acquisition and Reconnaissance) capabilities [124]. The key attributes of this platform include, the ability to undertake long range missions, and to provide high levels of persistence and survivability in contested air space consisting of advanced air and ground threat systems [125].

BAE Systems - Taranis

Dassault - Neuron

Figure 7.2: Examples of different Unmanned Combat Air Systems (UCAS).

The propulsion system developed for this UCAS is required to not only provide propulsive power, but is also required to deliver electrical power to other flight systems on-board the aircraft, in addition to meeting the stealth requirements of the aircraft. Thus, the design problem posed for such a system is multi-objective in nature, requiring the design to satisfy several system objectives, such as maximise range, minimise infra-red signature, minimise radio frequency (RF) signature, minimise propulsion system volume, etc. The developed decision support framework in this thesis was used to define a utility function that combined all of the system objectives into a utility metric, which in-turn was compared against a cost metric the define the cost-utility benefits. In addition, a UCAS sizing and optimisation code was also developed that allowed several propulsion system architectures to be readily evaluated at a perform level. However, due to the sensitive nature of this work, the results obtained for this case study will not be presented or discussed in this thesis. For this reason,

another case study was chosen to illustrate the concepts of the developed decision support framework. The case study presented in this thesis is an illustrative example of the 2Seas project, in which the University of Southampton developed a small UAS for the purpose of maritime surveillance. The details of this case study are presented in the following sections of this chapter.

7.2 Overview

The case study presented in this chapter focuses on the design and development of a low-cost UAS to be used by civilian law enforcement authorities for maritime surveillance operations. Typically, many law enforcement authorities make use of manned helicopters to carry out search and rescue operations. However, the cost of operating a helicopter is significant, especially when law enforcement departments are constrained by budgets. UASs offer a low-cost, close-range surveillance capability that allows the system to operate without being detected by the target in comparison to manned helicopters. Thus, the use of UASs has gained much interest in recent years and have already been applied in various fields of law enforcement [122]. This case study is an illustrative example based around the European Union 2Seas research project (www.2Seas-UAS.com), which is carried out by the University of Southampton, Delft University, and other project partners from Holland, France, Belgium and the UK [126].

This chapter provides an overview of the modelling methods that are applied to capture the behaviour of some of the key system elements that describe an UAS. The UAS under consideration for this case study is known to be carrying an electro-optic (EO) camera system that provides high-definition motion imagery back to the ground control station. For the purpose of monitoring offshore activities in real-time and for long durations. The development of such a system requires a multidisciplinary approach that accounts for aerodynamics, structures, propulsion, avionics, payload integration, communication, etc. Thus, the development of an overall system architecture requires integration of several subsystems that are interlinked with each other to allow for effective system operations. The integration and development of subsystems also introduces multiple design alternatives to meet several system functionalities. Before defining the modelling methods applied, a brief overview of the UAS regulatory requirements are defined. The aim here is to present the reader with the operational constraints that regulate the application of UASs.

7.3 UAS Classification

Based on the Joint Doctrine Note (JDN) 3/10 [124], categorisation of UASs begins with the classification of mass, and the operational altitude of the UAS. Class I is defined to be less than 50 kg (which is further sub-divided based on operating altitude), Class II ranges in mass from 150 kg to 600 kg, and Class III is more than 600 kg (further sub-divided based on operating altitude). This classification guide is represented in Table 7.1.

Table 7.1: UAS classification guide [124][9].

Class	Category	Normal operating altitude	Civil category (UK CAA)
Class I < 150 kg	Micro < 2kg	Up to 200 ft AGL	Small Unmanned Aircraft (<20 kg)
	Mini 2-20 kg	Up to 3000 ft AGL	
	Small 20-150 kg	Up to 5000 ft AGL	Light Unmanned Aircraft (20-150 kg)
Class II 150-600 kg	Tactical	Up to 10,000 ft AGL	Weight category group 3 (> 150 kg)
Class III > 600 kg	Medium Altitude Long Endurance (MALE)	Up to 45,000 ft AGL	
	High Altitude Long Endurance (HALE)	Up to 65,000 ft AGL	

This thesis will focus on the Small category class of UAS (20 -150 kg), thus only Class I operational requirements will be discussed hereafter. As defined by the Unmanned Aircraft System Operations in UK Airspace – Guidance (CAP 722) report [127], UASs in the sub 20 kg and 20-150 kg class are restricted to operate within visual line of sight (VLOS). This is defined to be an altitude of 400 ft, and a flight radius of 500 m. Operations beyond 500 m or 400 ft is possible by complying with collision avoidance responsibilities. But collision avoidance is still required to be achieved through visual observations. Beyond visual line of sight (BVLOS) flight is defined as the distance where the remote pilot is no longer able to respond to or avoid other airspace users by visual means. UASs that intend to operate BVLOS require approved methods of aerial separation and collision avoidance. Advancements in the area collision avoidance for autonomous systems is an active area of research, and will not be addressed in this thesis (see Vahidi and Eskandarian [128] for more information).

Though there are restriction in altitude and range of flight, the design of a platform still needs to meet the operational requirements that will eventually enable the platform to be mature enough to overcome these restrictions. This may require the platform to be designed to fly at a higher altitude and longer range to meet the demands of the end user. For the current case study of maritime surveillance, the platform will be designed for altitudes and ranges greater than 400ft and 500m in order to meet the operational demands.

7.4 Mission Definition

This section defines the UAS mission profile, depicted in Figure 7.3. The cruise-out, and the cruise-back, phase are limited in distance by the maximum line-of-sight (LOS) communication range, as it is assumed that beyond line-of-sight (BLOS) capability via SATCOM (Satellite Communication) or by other relay nodes is not feasible. The cruise-out distance is set to be 90% of the maximum communication range to allow for some additional travel distance during loiter. During the loiter/search phase of the mission the optical camera

[9] AGL – Above Ground Level

system is turned on and data of the high definition digital motion imagery is transmitted back to the ground station. It is also assumed that data is transmitted to the ground station in real time, with no data storage on-board the UAS. Once the search phase of the mission is complete the UAS returns back to the ground control station, which is defined by the last two phases of the mission profile (6 and 7).

A detailed overview of the mission profile is provided in the table below. It can be noted that some of the mission variables are provided over a range. These act as design variables in the multi-objective optimisation framework, as the change in these mission variables can have a significant impact on the capabilities offered by the system, and the cost of implementing the system. It should also be noted that some of the fields in Table 7.2 are blank, as these values are an output of the model and are dependent on other design variable values, such as wing loading, engine thrust-to-weight ratio, and wing aspect ratio.

Table 7.2: Maritime surveillance mission profile definition.

	Segment	Time	Distance	Speed	Altitude	Thrust Setting
1	Take-off	–	100 m	–	Sea-level	Max continuous
2	Climb	5 min	–	–	150-400 m	Max continuous
3	Cruise-out	–	90% of max LOS range	30 m/s	150-400 m	–
4	Loiter	1-3 hrs	–	20-28m/s	150-400 m	–
5	Cruise-back	–	90% of max LOS range	35m/s	150-400 m	–
6	Descent	–	–	–	Sea-level	Idle
7	Land	–	100 m	–	Sea-level	Idle

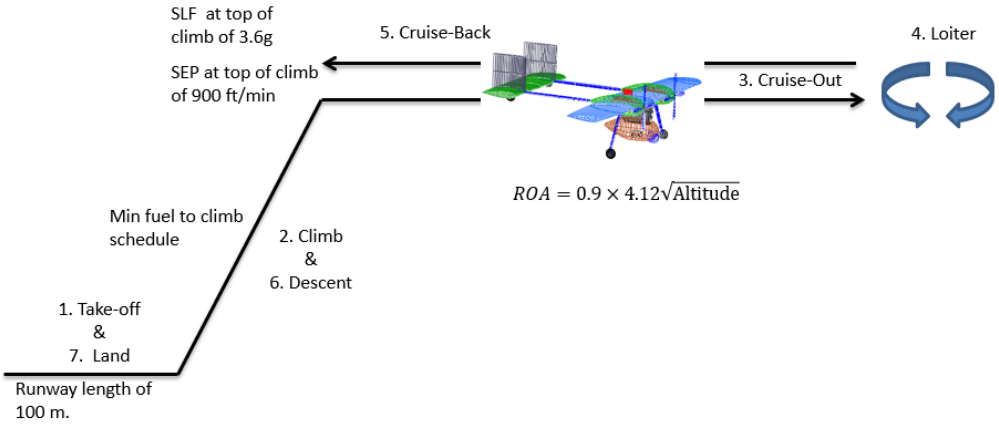


Figure 7.3: Maritime surveillance mission profile.

7.4.1 Communication Line-of-Sight Range

The line of sight (LOS) communication range is limited by the distance that the ground control station antenna can have a direct link with the on-board aircraft antenna. Beyond line-of-

sight communication (BLOS) occurs when the transmitter and receiver are no longer able to communicate with direct line-of-sight, and must use relay nodes to retransmit the signal. This case study assumes that the UAS operates under LOS communication, as BLOS requires the use of SATCOM or other relay mechanisms, which are deemed too expensive for data transmission [122].

For the theoretical calculation of LOS range, assuming a smooth Earth's surface with no terrain features, the range is assumed to be governed by the four-thirds Earth model. The presence of Earth's atmosphere causes the propagation of electromagnetic waves to refract. If the refractivity index is large enough it causes the electromagnetic wave to bend around the Earth's curvature, causing the range to the horizon to extend. A simple method to model this behaviour, for altitudes less than 3 km, is to replace the actual Earth with an imaginary Earth whose effective radius is calculated to be four-thirds of the Earth's radius (the value of four-thirds is valid under normal weather conditions). Using the four-thirds Earth model definition, the LOS range can be calculated from simple trigonometric relationships.

$$R = 4.12\left(\sqrt{H} + \sqrt{h}\right) \tag{7.1}$$

From the above equation it can be noted that the LOS range is defined as a function of cruise altitude [122], thus by increasing cruise altitude the LOS range is also increased. It should also be noted that the represented LOS range is a very simplified view intended for conceptual analysis, where several parameters have not been accounted for. In reality, the LOS range can depend on terrain features, obstacles such as buildings, and variation in atmospheric refraction (due to variation in atmospheric pressure) as a function of altitude. For a more in-depth review of wave-propagation effects the reader is referred to Blake [129]. Other affects such as frequency, atmospheric attenuation as a function of frequency will be discussed later on in this chapter.

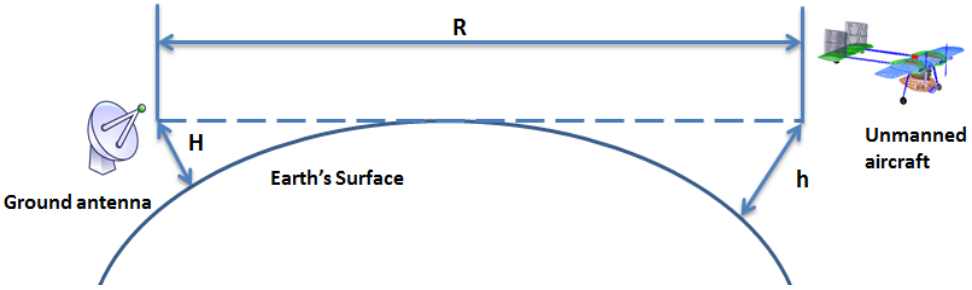


Figure 7.4: Line-of-sight geometry.

7.4.2 Power Requirements

In addition to the flight mission profile, the electrical power demand profile should also be taken into consideration. As meeting the power demand can have a significant impact on architecture selection and flight performance of the aircraft. Figure 7.5 shows an illustrative example of the average power demand profile (normalised power demand) during the course of the mission. Although the peak power demand would be greater, and is prone to

fluctuations during the mission, it is assumed that peak power demand is only required for short time periods. Thus, the power source is sized to meet the average power demand profile.

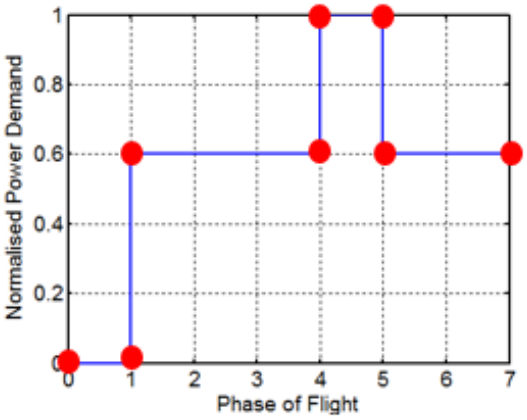


Figure 7.5: Illustrative electrical load profile of the UAS.

Phase 0: Start and Warmup of systems – This phase of the mission in the initial start-up of the aircraft, which requires main engine start-up, and system checks to be performed.

Phase 1, 2, and 3: Take-off, Climb, and Cruise-out – At this point the aircraft is operating autonomously. The aircraft demands power from control surface actuators, flight instrumentation, on-board sensors, and the antenna transmitter/receiver power.

Phase 4: Search – Also referred to as the loiter phase. This is the most demanding phase in terms of power demand. At this point the electro-optical camera system is turned on, and data generated from the high-definition camera is transmitted back to the ground station at a continuous rate.

Phase 5, 6, and 7: Cruise-back, Descent, and Land – Once the search phase is complete the payload system is turned-off and the aircraft returns back to base. During the return segment the UAS operates at a similar power level to phases 1, 2, and 3.

The power loads for the actuators, sensors, and flight instruments can be assumed to be approximately constant throughout the mission, and are thus grouped together into the avionics group as a constant power load. Changes in the power demand are mainly caused due the operational use of the payload sensor system during the search phase, and due to the transmission of data back to the ground control station. The following equation describe the overall power demand of an UAS at different phases of flight.

$$P_{overall} = P_{avionics} + P_{Comms\ flt\ crit\ data}$$

$$\text{With, } P_{avionics} = P_{actuators} + P_{sensors} + P_{flt\ instruments} \tag{7.2}$$

Equation (7.2) is applied to mission phases 1, 2, 3, 5, 6, and 7, when the payload system is turned off. Here, $P_{avionics}$ refers to the constant power demand from aircraft avionics[10], and $P_{Comms\ flt\ crit\ data}$ refers to the communications system power required to transmit flight critical data back to the ground control station. It is assumed that the rate of data transmitted back is constant and continuous throughout the mission.

During the search phase (phase 4), additional terms are added to account for the payload module power demand and the transmission of high definition motion imagery back to the ground station, as follows:

$$P_{overall} = P_{avionics} + P_{EO\ payload} + P_{Comms\ flt\ crit+HD\ data} \tag{7.3}$$

where $P_{EO\ payload}$ refers to the power demand of the electro-optical camera payload, and $P_{Comms\ flt\ crit+HD\ data}$ refers communications system power required to transmit flight critical data and high-definition imagery back to the ground control station. It is again assumed that the rate of data transmitted is constant and continuous during the search phase i.e. a continuous stream of high definition imagery and flight critical data is sent to the ground control station.

7.5 Modelling Approach

The analysis of several system architectures requires a modelling environment that integrates multiple models to capture the design attributes required for the MAU model. At the conceptual design stage the use of analytical and semi-empirical methods offers a set of simple parametric models that can be applied to gain visibility into the design space and better understand the characteristics of system solutions.

Typical legacy tools for conceptual design are generally developed to analyse a specific class or sub-class of aircraft configuration. These generally consist of modelling assumptions and correlations based on empirical data that relate to a particular class of aircraft. Thus, in order to model different system configurations, this study has developed its own modelling environment, which is focused towards a particular set of UAS classes and its sub-systems. The models developed in this research are simplified analysis methods that makes use of semi-empirical relationships to predict the system performance. It should be noted that the intent of this research is not to develop a novel aircraft modelling framework, but rather make use of standard textbook methods to predict system performance. For a more detailed description of the analytical/semi-empirical methods used the reader is referred to Raymer [17] and Gundlach [122].

The interaction between different analyses disciplines are represented in Figure 7.6. Here, the implementation of different analysis codes, and the communication between these

[10] Avionics refers to any electrical devices used on the UAS with the exception of the electrical power system, payloads, and communications system.

modules is executed in MATLAB, which offers an easy-to-implement programming environment.

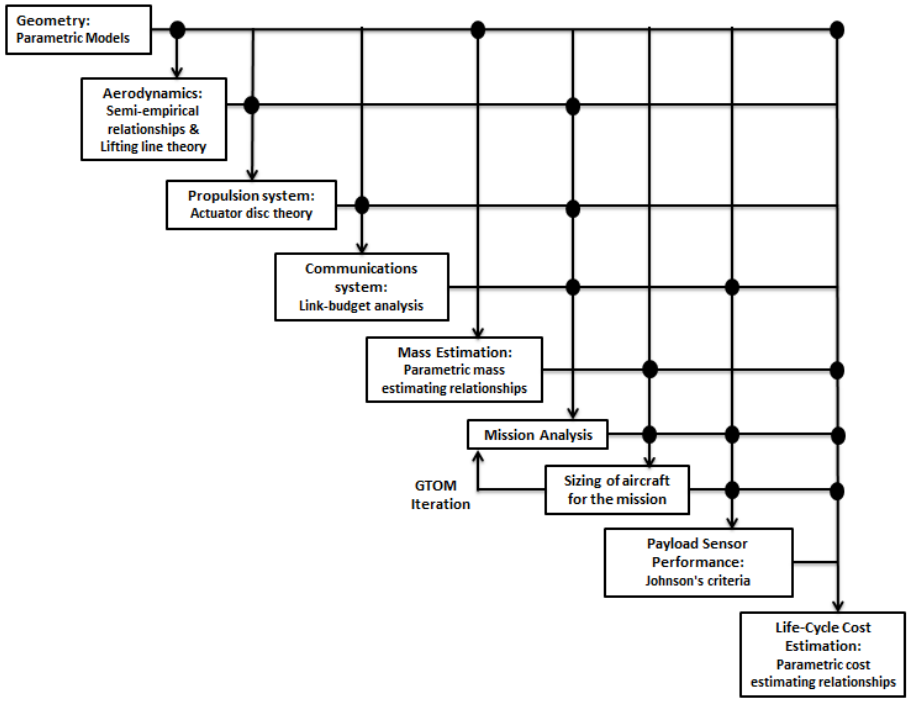


Figure 7.6: Schematic block diagram of the modelling environment.

7.5.1 Geometry

The representation of geometry is an important consideration in conceptual design, as there is always a trade-off between fidelity and the effort required to parametrically describe geometry. Commercial CAD packages offer great flexibility in generating a geometry with varying levels of fidelity. But the problem incurred in many CAD packages is the parameterisation of geometry can require substantial effort with a large number of parameters to describe the geometry [130].

At a conceptual design level much simpler representations of the geometry are required with a minimal number of parameters to parametrically represent the geometry under consideration. Vehicle Sketch Pad (VSP) offers this capability by providing a predefined aircraft geometry that is already parameterised. This allows the user to more readily represent the geometry with minimal effort in parameterisation [131].

(a) (b)

Figure 7.7: A geometric representation of the UAS using (a) CAD package and (b) Vehicle Sketch Pad (VSP).

Though VSP offers a user-friendly programmable environment to represent an aircraft geometry. For the current case study simple equations are used to represent the aircraft, as the geometry can be represented using simple shapes that can be modelled using simple equation, and in-turn calculate the wetted areas of components. VSP is only used for visualisation purposes and is not used in the optimisation framework to parametrically vary the geometry.

7.5.2 Aerodynamic Predictions

The aerodynamic drag characteristics of an aircraft can have a significant impact on fuel burn, which in-turn drives the take-off mass and the operating cost of the aircraft. Thus, much research has gone into developing computational methods to accurately predict the drag characteristics of an aircraft, and the means of reducing its impact. However, the requirement to evaluate several design configurations, and reduce the number of input arguments required for modelling, meant that sophisticated Euler or Navier-Stokes methods were not considered for aerodynamic evaluation. Instead, the aerodynamic calculations for this study rely upon semi-empirical and analytical modelling approaches.

The overall drag model can be represented as follows:

$$C_D = C_{DO} + C_{Di} \tag{7.4}$$

where C_{DO} is the zero-lift drag, which consists of friction drag and pressure drag. Friction drag is caused by the shear forces acting on the flow boundary layer, normal to the surface. Pressure drag is associated with wake formation caused to airflow interacting with the aircraft geometry. It is calculated by integrating the pressure differentials normal to the flight path. C_{Di} represents induced drag, which is caused due to vortex formation at the wing tip. The induced flow of the vortices decreases the effective angle of attack of the wing, which varies from wing tip to root. This generates a downstream-facing component of aerodynamic force of the wing, known as induced drag.

The zero lift-drag coefficient can be predicated by applying the following semi-empirical equation:

$$C_{DO} = \frac{1}{S_{ref}} \sum C_{f,seg} \cdot FF_{seg} \cdot Q_{seg} \cdot S_{wet} \tag{7.5}$$

where $C_{f,seg}$ is the flat plate skin-friction coefficient, FF_{seg} is the shape form factor, Q_{seg} is the interference factor, and S_{wet} is the wetted area of the body or the aerodynamic surface. The overall zero-lift drag value is the summation of all the individual components that makeup the UAS.

The calculation of the induced drag coefficient is achieved by incorporating a simple panel method. This study adopts the Prandtl lifting-line theory (LLT) method to predict the lift distribution along the wing span. Anderson [132] provides a more in-depth derivation of this method, and also provides a numerical lifting line theory that can address nonlinear lift curve

slopes for wings near or beyond the stall region. However, this study applies the matrix form of the Prandtl lifting-line theory, which is suitable for subsonic applications and linear regions of the life-curve slope (i.e. un-stalled regions). The matrix form of the LLT method is easy to implement, and offers a rapid calculation of the model. However, the LLT method is limited in modelling only one wing at a time, thus the downwash effects on the tail are not accounted, resulting in the tail effectiveness to be over-predicted, and the tail contribution to induced drag to be under-predicted. But, for the purpose of conceptual analysis, the LLT method will be implemented.

The LLT method predicts the lift distribution through a Fourier sine series. The method allows for variable chord geometry, camber, and twist distributions along the wing span. But the flowing limits are also placed on the wing geometry and flow conditions, such as quarter-chord sweep must be less than 10 degrees, the wing must have no dihedral, wing aspect ratio must greater than 5, and the flow must be incompressible. In meeting this criteria, the LLT method can be formulated as:

$$\frac{\pi \cdot c(\theta)}{2 \cdot b_w}[\alpha + \alpha_{twist}(\theta) - \alpha_{0L}(\theta)] \sin\theta = \sum_{n=1,odd}^{\infty} A_n \sin(n\theta)\left[\frac{\pi \cdot c(\theta) \cdot n}{2 \cdot b_w} + \sin\theta\right] \tag{7.6}$$

Where $A_n$is the influence coefficient, b_w is the wing span, c is the chord, α is the angle of attack, α_{twist} is the washout angle, and α_{0L} is the zero-lift angle of attack of the airfoil. The angle θ is a function of the semi-span ratio at a distance y from the wing root.

$$\theta = \cos^{-1}\left(\frac{-2 \cdot y}{b_w}\right) \tag{7.7}$$

The value y is divided up into N segments between 0 and $\pi/2$ radians, and n assumes odd integer values from 1 to $2N-1$. Here the chord at a given angle $c(\theta)$ is found through linear interpolation across the semi-span. Using the above parameterisation, the following matrix formula is solved for:

$$\mathbf{Ax} = \mathbf{b}$$

where $$\mathbf{A}(i,j) = \sin[n(j) \cdot \theta(i)] \cdot \left\{\frac{\pi \cdot c(i) \cdot n(j)}{2 \cdot b_w} + \sin[\theta(i)]\right\},\ i,j = 1,\dots,N$$

$$\mathbf{b}(i) = \frac{\pi \cdot c(i)}{2 \cdot b_w} \cdot [\alpha + \alpha_{twist}(i) - \alpha_{0L}(i)] \cdot \sin[\theta(i)],\ i = 1,..,N \tag{7.8}$$

where $n(j) = 2 \cdot j - 1$. The values of $\mathbf{x}$ are determined by applying the following operation $\mathbf{x} = \mathbf{A^{-1}b}$. The total lift, and thus the induced drag can now be calculated as follows:

$$C_L = AR \cdot \pi \cdot \mathbf{x}(1)$$

and $$C_{Di} = \frac{C_L^2}{\pi \cdot AR \cdot e}$$

with $$e_{theo} = 1/1 + \sum_{j=2}^{N} n(j) \cdot [x(j)/x(1)]^2 \quad (7.9)$$

where e_{theo} is the theoretical Oswald efficiency. In reality the Oswald efficiency factor will not be as big as the value predicted from the above equation. As viscous effects caused due to skin friction and flow separation can impact the Oswald efficiency factor. In addition, the presence of the fuselage, nacelles, and other aircraft components are not accounted for when calculating the lift distribution. To account for the impact of these components the following correction factors are added:

$$e = e_{theo} \cdot k_{e,F} \cdot k_{e,D_0} \quad (7.10)$$

where $k_{e,F}$ is the fuselage impact factor, and k_{e,D_0} is viscous impact factor. We assume that the impact of viscous effects on Oswald efficiency is negligible, and is therefore ignored in the calculation. The impact of the fuselage is calculated by apply the following semi-empirical formula :

$$K_{e,F} = 1 - 2\left(\frac{d_F}{b_w}\right)^2 \quad (7.11)$$

where, d_F is the max fuselage diameter. The application of the LLT method to an arbitrary wing, at different aspect ratios is presented in the figure below. The plot indicates that an increase in wing aspect ratio results in the reduction in induced drag coefficient, at a given lift coefficient, and the Oswald efficiency decreases with aspect ratio. These trends are in alignment with the trends predicted from theoretical and empirical data [17], [132].

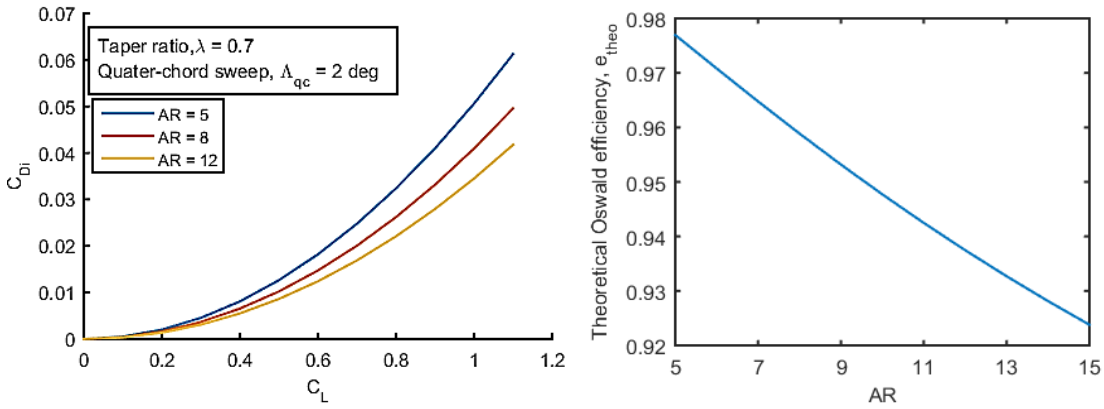


Figure 7.8: The application of the LLT method to an arbitrary wing.

7.5.3 Propulsion System

Majority of the UASs in Class I (<150 kg) are propeller driven [122]. It should be noted that propellers are only an element of the propulsion system. The shaft power required to drive the propellers must come from another device, such as a turboshaft engine, reciprocating engine, or electric motors. The choice of a power source to drive the propeller is dependent on various factors that influence the performance, cost, mass, and reliability of the aircraft,

and thus is an important architectural decision to be made. The choice of a power source in this case study is addressed in the next chapter of this thesis.

As the propeller is a common element within the propulsion system under consideration, a brief overview of the propeller model is described here. The propeller is modelled by applying the actuator disc theory to determine the shaft power required to generate thrust, at a given flight condition. The shaft power required to generate a given amount of thrust is defined as follows:

$$P_{Shaft,Prop} = \frac{T}{\eta_p} \cdot \left(V + \frac{1}{2} \cdot \Delta V\right)$$

$$\text{where} \quad \Delta V = \sqrt{V^2 + \frac{2T}{\rho \cdot A_p}} - V$$

$$\text{and} \quad \eta_{p,ideal} = \frac{2V}{V_e - V} = \frac{1}{\Delta V / 2V + 1} \tag{7.12}$$

Where T is the thrust required by the UAS, $\eta_p = \eta_{p,ideal} \cdot \eta_{p,nonideal}$ is the propeller efficiency (a nominal value for the non-ideal efficiency $\eta_{p,nonideal}$ is chosen to be 90%), V is the velocity of the aircraft, ΔV is the velocity difference between the exit velocity and inlet velocity, and A_p is the propeller disc area.

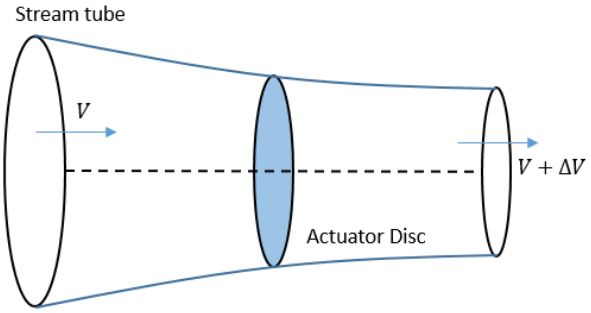


Figure 7.9: Propeller actuator disc model.

Once the required shaft power is obtained, it is compared against the available shaft power from the energy source, which could be a reciprocating engine, electric motor, turbofan, etc. It should be noted that in cases where the electrical power load of an aircraft is provided by an engine generator, the required shaft power will be the propeller shaft power plus generator load.

$$P_{shaft,Req} = P_{shaft,Prop} + \frac{P_{elec\ load}}{\eta} \tag{7.13}$$

where $P_{elec\ load}$ is the required electrical power by sub-systems on-board the UAS and η is an efficiency factor to account for losses in the delivery of the electrical power to the required sub-systems.

7.5.4 Communication Systems

The analysis of the communication link is an important aspect of the UAS's operational performance. It provides a means of calculating the reliability of a communications link in transmitting and receiving data to and from the UAS, and the ground control station. Communication systems are typically composed of many elements, each of which can be arranged into many configurations. To give a simplified overview of some of the main elements, a simplex one-way digital data link is shown in Figure 7.10. Here, the function of the modem is to modulate the input signal onto a carrier wave of higher frequency prior to transmission. Similarly, the modem at the receiver end, demodulates the signal to recover the original baseband signal. The modulation of a digital signal can in general be grouped into three categories, amplitude-shift keying (ASK), frequency-shift keying (FSK), and phase-shift keying (PSK). The choice of a modulation type is dependent on the application under consideration, and also on the required energy-per bit to noise-power-spectral-density ratio Eb/N_o. The transmitter takes the modulated signal from the modem and outputs a radio-frequency (RF) wave form. In consensus, the receiver takes the RF waveform from the antenna and outputs the modulated signal. Prior to transmitting the signal the amplifier increases the power of the RF waveform for long distance transmission. For a more in-depth review of the components involved in the communication link the reader is referred to Fortescue, et al [133].

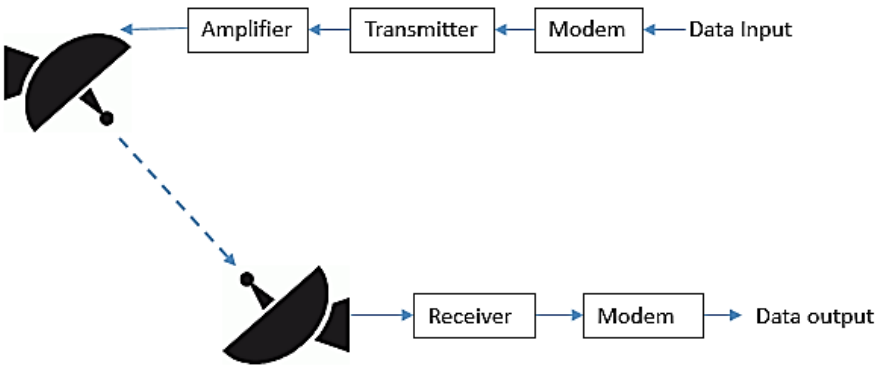


Figure 7.10: Simplified simplex one-way digital data link [122]**.**

At a conceptual design level, the link budget analysis methodology is applied as a preliminary tool to determine the transmission power required to transmit data from the UAS to the ground control station. For more advanced analysis methods, high fidelity finite element computational electromagnetic analysis codes are applied. These codes discretise the geometry of the aircraft, where the grid size is dependent on the frequency. Hence, high-frequency modelling of a large aircraft can be computationally intensive due to the fine grid size that is required. The ultimate aim of such tools is to aid the performance prediction and antenna integration trades [122]. However, the application of such tools is beyond the scope of this research. This study is focused at a conceptual level, hence in order to simplify the computation the link budget analysis is applied to predict the transmission power required.

In applying the link budget analysis, the performance of an antenna system is defined by the signal-to-noise ratio (SNR). This defines the strength of a signal at the receiver, in comparison to the noise generated by the receiver itself. A reliable communication link can only be achieved if the SNR is greater than some threshold value. This requires calculating the available SNR, which is typically represented in decibel form in order to make the calculations easier. The decibel form of the available SNR is defined as:

$$\begin{aligned} SNR(dB) &= P_T(dBm) + G_T(dBi) + L_T(dB) + G_R(dBi) + L_R(dB) \\ &+ 20\log_{10}\left(\frac{\lambda}{4\pi R}\right) - 10\log_{10}(1000k \cdot T)\left(\frac{dBm}{Hz}\right) \\ &- 10\log_{10}(BW)(dbHz) - NF(dB) \end{aligned} \tag{7.14}$$

where, P_T is the transmission power, G_T is the transmitter antenna gain. L_T is the signal loss through the transmitter antenna. L_P is the absorptive propagation loss. G_R is the receiver antenna gain. L_R is the receiver signal loss from receiver antenna through the amplifier. λ is the wavelength of the carrier signal, in this case study the carrier signal is transmitted at a frequency of 2 GHz – within the L band as it is deemed to be within an efficient bandwidth for digital video transmission [122]. R is the distance between the transmitter and receiver, k is the Boltzmann's constant, T is the ambient absolute temperature, BW is the effective noise bandwidth of the receiving process, and F is the noise factor.

The above equation provides a means of calculating the available SNR at the receiver end of the communications system. It can be noted that the calculation of the SNR is dependent on several variables, such as the distance the UAS is from the ground control station R, the propagation losses associated with the RF signal travelling through the earth's atmosphere L_P, signal losses associated with the transmitter and receiver L_T and L_P, and losses due to the presence of 'noise' in the electrical systems kTB – which defines the total noise power in bandwidth B. Table A.2 in Appendix D provides a list of values applied to each of these losses for the current case study. For a more in-depth description of the losses associated with the communications system, and the propagation of RF signals through the atmosphere, the reader is referred to Gundlach [122].

In calculating the available SNR, the link margin required to transmit data reliably to the receiver can be determined. The link margin is defined as the difference between the available SNR and the required SNR to transmit data to the receiver. A 10-dB link margin is recommended i.e. $\text{Link Margine}\ (10\ dB) = SNR_{Avail}\ (dB) - SNR_{Req}(dB)$, which is primarily to combat multipath effects and blockages at the extent of the communications range [122]. In knowing the required link margin, the transmission power P_T required by the communications system, on-board the UAS, to transmit data to the ground control station can be calculated. The required transmission power of the communications system is one of the crucial parameter influencing the choice of the power generation system on-board the UAS, and is secondary in its influence on the performance of the aircraft. The architectural choice of the on-board power generation system will be addressed in the next chapter of this thesis.

7.5.4.1 Required SNR

The required SNR is mainly driven by the type of modulation applied to the carrier waveform, and by the impact of system noise on error rates. In the case of digital signals, the mechanism by which system noise affects the output of the communications link is different in comparison to analogue signals. In this case, the demodulator at the receiver end contains threshold detectors, which allocates one of the permitted values to each received symbols[11] [133]. However, in the presence of noise i.e. if the noise voltage is large enough, the receiver output may lie on the wrong side of the threshold, resulting in an error in the interpreted signal. This can be defined by the bit error rate (BER – the probability of error in any one bit), which is a function of the ratio of energy per bit to noise power spectral density, E_b/N_o, and is dependent on the type of modulation applied on the carrier wave. Figure 7.11 shows the theoretical relationship between BER and E_b/N_o for three different modulation types, quadrature phase shift keying (QPSK), differential binary phase shift keying (DBPSK), and frequency shift keying (FSK). For a more in-depth description of these modulation techniques and the relationship between BER and E_b/N_o the reader is referred to Fortescue et al [133].

The required BER is a function of the type of data being transmitted, BERs of interest may range from 10^{-4} to 10^{-8}. For the current case study is assumed that the required BER for digital motion imaginary is in the order of 10^{-6}. The QPSK modulation is chosen as it provide a low E_b/N_o for a given BER in comparison to other modulation types, as illustrated in the figure below.

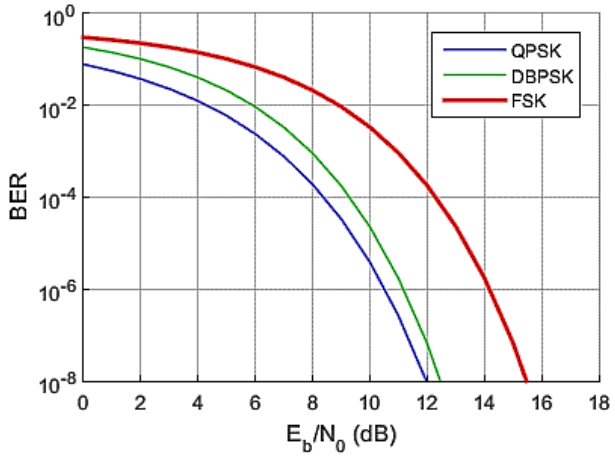


Figure 7.11: Bit Error rates for different modulation types.

The relationship between the required E_b/N_o and BER for QPSK modulated signal is represented in the equation below. The function $erfc()$ represents complementary error

[11] The symbols are used as a means of encoding the signal prior to transmission, such that the receiver may determine the bit information and signal levels in order to interpret the transmitted signal.

function. This relationship is applied to calculate the required energy-per bit to noise-power-spectral-density ratio.

$$BER = 1/2 \cdot erfc(\sqrt{Eb/No}) \tag{7.15}$$

The required SNR can now be determined by relating it to E_b/N_o, using the following linear relationship (in decibel form):

$$SNR_{Req} = Eb/No(\mathrm{dB}) + 10 \cdot \log_{10}\left(\frac{R_{Data}}{B}\right) \tag{7.16}$$

where, B is the bandwidth, and R_{Data} is the data rate. The ratio of R_{Data}/B is dependent on the modulation type. For QPSK modulation the ratio of R_{Data}/B is assumed to be approximately one [122].

7.5.5 Mass Estimation

The calculation of structural mass of various aircraft components is a very difficult task at the conceptual design stage, as information regarding the structural arrangement of the aircraft, or detailed CAD designs are not available. However, recent developments in modelling approaches make use of finite element methods, in combination with semi-empirical and analytical methods, to predict structural mass in conceptual design [134]. But the effort in constructing these models, and the number of input parameters that are required to run the model makes it infeasible to simulate multiple architectural configurations at a conceptual level. This leads us to adopt statistical-based approaches for structural mass estimation, usually relating to various parameters known to affect the structural mass i.e. GTOM, span, maximum load factor, etc. The application of such methods may provide unrealistic values where knowledge is not sufficiently available. This is especially true for small UASs, where the data available is relatively limited and configurations adopted have been unconventional in comparison to manned aircrafts. Thus, mass estimation based on statistical analysis may have serious disadvantages as extrapolation of the curve beyond its data range may provide misleading results.

Though the use of statistical approaches presents inherent disadvantages, this thesis adopts such methods to calculate the structural mass of the UAS [17], [122]. The reason being that statistical methods provide the benefit of carrying out mass predictions near instantaneously, with modifications (factoring of results) included to account of small UAS airframes, as presented by Raymer [17] and Gundlach [122]. But, it is recognised that a more detailed structural and mass analysis assessment will need be to be carried out prior to preliminary design.

7.5.6 Mission Analysis & Sizing

The mission analysis module take a given mission profile and calculates the fuel required to complete that mission. The predicted fuel burn for each phase of the mission segment is

calculated by using simple mission analysis calculations, such as the Breguet range equation. The output of the analysis is the required fuel mass and the GTOM of the UAS.

The calculated GTOM is however only a prediction, as the structural mass of the aircraft is calculated using statistical methods, which takes GTOM as an input. Thus, in order to solve for GTOM an iterative scheme is required, which starts with a predicted estimate of GTOM and iterates to a converged solution. The GTOM outputted from the mission analysis module is compared against the estimated GTOM at the previous iteration. If the error is substantial a new value of GTOM is predicted based on the size of the error. The new GTOM is fed back into the mission analysis module and the process is iterated until the error is within the tolerance limits. The GTOM iteration is defined using a simple equation, as follows:

$$GTOM_{new} = GTOM_{actual} - 0.7 \cdot GTOM_{estimated} \tag{7.17}$$

7.5.7 Aerial Remote Sensing

The main operational requirement of this UAS is to gather intelligence by capturing video imagery and relaying it back to the ground control station for further analysis. This is achieved by providing the optical sensor system an adequate field of view to be placed on the target, or scene of interest. Hence, the design of the UAS and its operational requirements are highly influenced by the payload sensor parameters. Therefore, it is imperative to accurately capture payload sensor performance, and its influence on UAS design, at the conceptual design stage in order to develop a well-balanced system.

Digital imaging cameras use a collection of individual detectors to form an image. These are arranged in rectangular arrays, known as the focal plan arrays (FPA). Each detector element generates an electrical signal based on the electromagnetic energy that reaches it [122]. The field of view (FOV) at the FPA can be described as an angular view of the focal plane. The field of regard (FOR) on the other hand is described as the maximum angular coverage that the sensor system is capable of achieving. In general, the FOV and FOR are identical for fixed optics with no zoom i.e. the FOR for fixed optics with no zoom capability is the widest FOV. For a more in-depth review of detector physics and design, the reader is referred to Leachtenauer and Driggers [135].

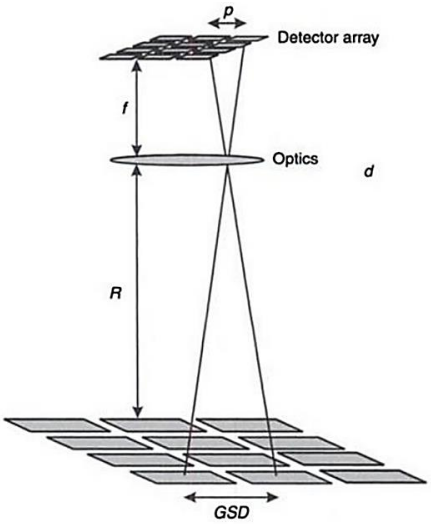


Figure 7.12: Ground sampling distance definition [135].

The measurement of image resolution, and thus the performance of the camera module is defined by the ground sampling distance (GSD). This parameter is used as a means of measuring the resolution limitations due to sampling, and thus the achievable image quality. The GSD is a function of the camera focal plane array, optics, and collection geometry. Although, other parameters are required to fully characterise the image resolution, GSD can be easily calculated based upon available engineering parameters. The GSD can be defined as the distance between the pixels projected on the ground at a slant range δR, using easily understood engineering parameters the GSD can be calculated as:

$$GSD_H = 2 \times \left(\frac{FOV_H}{2 \times Pix_H}\right) \times R \tag{7.18}$$

where GSD_H is the horizontal GSD, FOV_H is the horizontal FOV, Pix_H are the number of horizontal sensor pixels in the FPA, and R is the slant range. Similarly the vertical GSD is given as:

$$GSD_V = \frac{2 \times \tan(0.5 \times FOV_V \times Pix_H)}{\cos(\theta_{Look})} \times R \tag{7.19}$$

where θ_{look} is the look angle. It should be noted that parameters look angle and slant angle are both dependent of the UAS operational mission parameters– altitude and velocity.

The look angle θ_{Look} and the slant range δR, can be determined by considering the geometry of the UAS capturing imagery of an object. This is represented in Figure 7.13. The slant range between the aircraft and the object is defined as:

$$R = \sqrt{h^2 + GR^2} \tag{7.20}$$

where h is the aircrafts altitude and GR is the ground range from the UAS to the target.

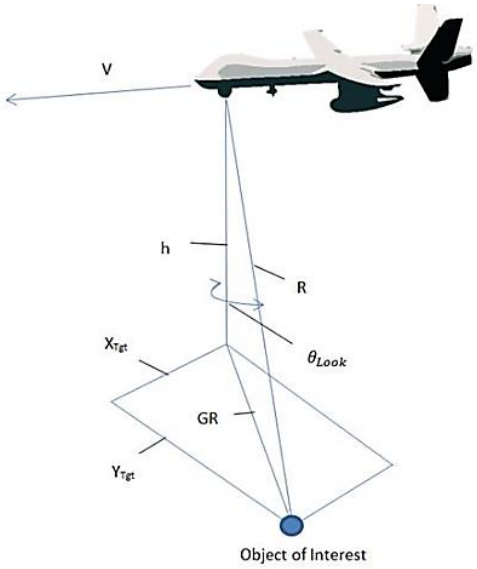


Figure 7.13: UAS remote-sensing geometry [122].

The focus of the discussion thus far has been on the use of GSD as a measure of resolution. However, GSD fails to explicitly state the quality of an image and its usefulness to the end user. Image quality prediction is arbitrary and is based on empirical approaches. The modelling approach defined in this thesis makes use of the Johnson criterion [135]. This method defines three levels of object discrimination for useful imagery as being detection, recognition, and identification. Detection can be defined as the probability that an imagery feature is recognised to be part of a general group (i.e. vehicle, ship, aircraft, etc.). Recognition is defined as the discrimination of a class of object (i.e. car, SUV, truck, etc.). Identification is the discrimination of the object type (i.e. if car then whether it is a BMW, Mercedes, Honda, etc.).

Determining the probability of detection, recognition, and identification for imagery is based on the sensors resolution. Here the object is replaced by black and white lines, each line constitutes a cycle. These groups of cycles are known as bar targets, which are commonly used for optical tests [122].

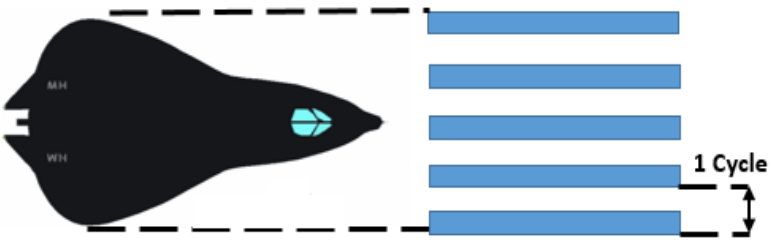


Figure 7.14: Four-cycle representation of an image.

The number of cycles across an object of interest is calculated by the target characteristic dimension divided by twice the averaged GSD. The characteristic dimension of an object is defined as:

$$d_c = \sqrt{W_{tgt} \cdot H_{tgt}} \tag{7.21}$$

where W_{tgt} and H_{tgt} are the object width and height (as viewed by the camera). The number of cycles across the object is given by:

$$N = \frac{d_c}{2 \times GSD_{Avg}} \tag{7.22}$$

Hence, the final probability of achieving the discrimination task for a given number of cycles is:

$$P(N) = \frac{(N/N_{50})^{(2.7+0.7\cdot(N/N_{50}))}}{1 + (N/N_{50})^{(2.7+0.7\cdot(N/N_{50}))}} \tag{7.23}$$

where N_{50} is the 50% probability of successfully performing the detection task. The N_{50} values for detection, recognition and identification are given as 0.75, 3.0, and 6.0 respectively.

7.5.8 Life-Cycle Cost

The estimation of life-cycle cost (LCC) is an important activity to be carried out, as it provides a means of evaluating the program viability, and its impact on budgeting and allocation of resources. The measurement of LCC provides an insight into near, mid, and long term technology, design, and operational investment needs. From an aerospace systems perspective, the LCC of a system consists of development cost, manufacturing and production cost, operating cost, and disposal cost. These contributions to the LCC are shown in Figure 7.15, where the relative magnitudes of the cost per phase shown in the figure are illustrative. For the current study disposal cost is ignored, as it only forms a small portion of the LCC, and the means of estimating disposal cost at a conceptual design stage is limited due to a lack of available knowledge.

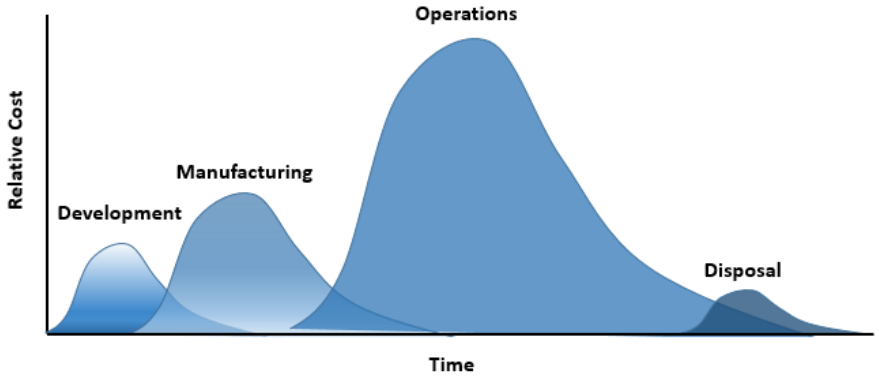


Figure 7.15: Life-cycle cost phases [136].

The estimation of cost at the conceptual design stage is largely statistical, where cost estimates are directly related to a set of system parameters known within the conceptual design framework, such as structural mass, engine thrust, or sortie rate, etc. These parametric approaches are largely based on trends derived from historic data that relate cost to some cost-driving variables in a mathematical form, via regression analysis. The implicit

assumption made in these models is that the same forces that affected cost in the past will affect cost in the future. The representation of such trends can be illustrated by NASA's estimate of software development cost as a function of software complexity, which is shown to have a linear trend.

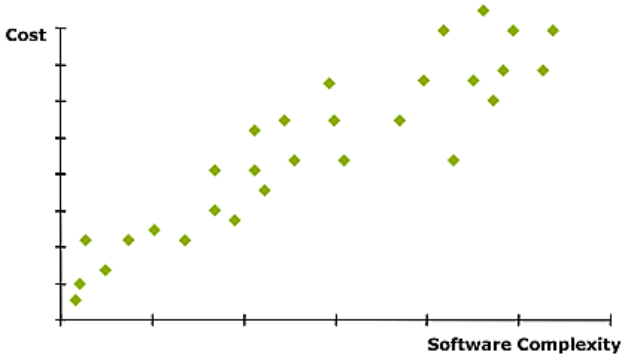


Figure 7.16: NASA data, relationship of cost versus software complexity [136].

However, applying statistical methods to estimate the cost of future system architectures may prove to be erroneous when the data used to derive the curve-fits are either from older programs, or systems that have had a drastically different development and operational program strategies, compared to the current system under consideration. This could result in different cost estimates for systems that belong to the same class. In addition, the prediction of cost for an UAS is further complication due to a lack of credible data this is available to develop a valid statistical model.

A more detailed approach to cost estimation takes a 'bottoms-up' approach, where individual estimates for each element is fed into the overall estimate. This methodology starts with a program definition, which requires the development of system requirements, concept of operations (CONOPs), a baseline system configuration, and a development time-line. Once the program is adequately defined the bottoms-up approach can proceed, which involves computing the cost at the lowest level of detail. Often the labour requirements are estimated separately from material requirements. Initial-bottoms-up cost estimates generally tend to be wrong, as these estimates are based on a best-case scenario with insufficient margins to account for uncertainties [122]. To compensate, a realism factor is applied, which accounts for all uncertainties within the program. This includes inflation, vendor price increase, requirement changes, budget cuts, etc. For a more in-depth discussion on the different phases involved in the bottoms-up cost estimation methodology the reader is referred to the NASA cost estimating handbook [136].

Though the parametric cost-estimation relationships (CERs) present several drawback in accurately predicting cost, this study adopts CERs as a means of estimating LCC. The reason being that CERs offer an estimate that can be predicated quickly, and can be easily replicated. Secondly, the information required to populate the CERs are minimal in comparison to the bottoms-up approach, where substantial detail regarding the system architecture and manufacturing methods, in combination with program operational details are required. This

information is however limited at the conceptual design stage, which makes the bottoms-up approach inapplicable at this stage of design. The CERs adopted for this study were derived by the RAND Corporation [137] and from Raymer and Gundlach [17], [122], which consists of data from both manned and unmanned aircrafts.

However, a problem still persists in that the data used in these CERs are mainly derived from large UAS and manned aircraft programmes that have a program life-cycles of decades. But, as previously stated, the focus of this study is on small UAS's, which generally have a program life of 5-10 years. To compensate for this Raymer and Gundlach [17], [122] suggest applying downward cost adjustment factors to the CERs in order to get the cost values approximately in line with small UAS programs. These are subjective adjustments based on expert opinion, which may compromise the validity of the cost estimates as there is no data, or a lack of it, to back-up the rational of the adjustment factors applied. Though the accuracy of the cost estimates may be compromised in applying these CERs to estimate the cost of a small UAS, this study is mainly interested in the general trends rather than the absolute values of cost, for the purpose of optimisation. For this reason, CERs are still deemed to be a useful methodology in calculating cost at the conceptual design stage.

The LCC for each system configuration can be estimated by decomposing LCC into, nonrecurring costs, fixed recurring costs, and variable recurring costs. Nonrecurring costs includes initial development, which consists of research, development, test and evaluation (RDT&E). Fixed recurring costs C_{Fi}, which are incurred at every time period regardless of demand or changes to the operating environment, including labour, facilities, and overhead. Variable recurring costs C_{Vi}, is dependent on the demand of system operation over a given time period, this includes the operating cost, material cost, variable labour cost, maintenance cost, fuel cost etc. In combining these three cost elements the LCC of a system configuration, for a number of time periods T, and a given demand in each time period D_j, can be predicted as follows:

$$C_{LCC,i}(D,T,r) = C_{RDTE,i} + \sum_{j=1}^{T} \frac{C_{Fi} + C_{Vi}(D_j)}{(1+r)^j} \tag{7.24}$$

In the above equation a discount factor r is included to represent the time value of money. The choice of a discount factor is dependent on the impact of future expenditures on the lifecycle cost relative to the initial expenditures. For the current study a discount rate of 10% is applied.

The estimation of the traditional cost elements such as RDT&E cost, manufacturing and production cost, labour cost, and material cost are predicted by applying CERs presented in several aerospace design text books[12]. For more details in regards to the development of

[12] The CERs applied in this study all contain a factor k, which scales the cost down to an appropriate level, such that it is in line with the small UAS class.

these CERs the reader is referred to Raymer and Gundlach [17], [122]. However, it should be noted that the cost associated with RDT&E of avionics, software, ground control station, payload sensors, and launch and recovery equipment, are not accounted for in this study. It is assumed that these subsystems are purchased off the shelf from external vendors, who will incur these cost elements. Only the cost of production of these sub-systems will be included in the final unit cost of the system, which are estimated by CERs developed by Technomics [138].

The following sub-section outline the assumptions made in defining the demand profile and maintenance schedule, such that the operating cost can be estimated.

7.5.8.1 Operating Cost

The cost of operating an UAS is dependent on the assumptions made as to how the UAS will be operated during the course of its program life. A major contribution to operating cost comes from fuel, labour, and maintenance. In reality other drivers, such as spares, storage, indirect labour, and support services can have a significant contribution to operating cost. But for the purpose of conceptual analysis these cost drivers are ignored due to a lack of available knowledge of the system operating environment. Before describing the cost elements involved, it is worth addressing the assumptions made in defining the demand profile during the program life of an UAS, and the assumptions made in the maintenance schedule of the UAS. The operational assumptions defined hereafter are partly notional and are partly based on operational data acquired from Schumann et al. [126].

For the current study the overall program life is assumed to be eight years, where the first two years are allocated for development and production of the UASs, and the next six years are allocated to the operational life of the UAS fleet. During the course of its operational life, it is assumed that the UAS's are utilised 5 months per year in operations, and it is also assumed that each UAS flies 10 training missions per year. This defines the demand profile of an UAS as a function of time.

The fleet size required to meet operational demands is dependent on the required surveillance time on the target area, the aircrafts time on station (TOS) or loiter time, and the operational availability of the UAS, A_O. The number of UAS's required to meet the operational demand can be defined as follows:

$$N_{AC} = \frac{T_{Surveillance}}{TOS \cdot A_O} \tag{7.25}$$

Here the required surveillance time $T_{Surveillance}$ at the target area is assumed to be 12 hours per day continuous. By determining the fleet size and the utilisation rate of the UAS per year, the number of missions flown per year can be calculated. Since the fuel mass per mission is known from the mission analysis module, the fuel cost per year can also be calculated.

The cost of maintenance is calculated by assuming a maintenance schedule for each of the major sub-systems of the UAS, which includes the airframe, power/propulsion, communications system, payload, and avionics. The scheduling assumptions are mainly driven by the reliability of system and sub-system components. The OSD (Office of the Secretary of Defense) reliability study [139] found that the major drivers in system failure were attributed to power and propulsion, followed by flight controls and ground control, and human interfaces. A breakdown of the finding of the report is presented in the figure below.

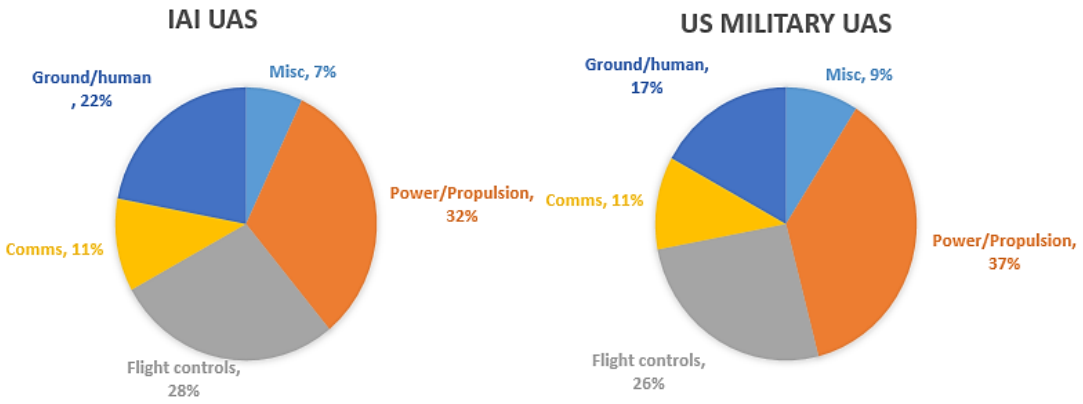


Figure 7.17: Average sources of system failures, data from 2002 [139].

The maintenance schedule is defined by the mean time between maintenance (MTBM) of the sub-systems, which in-turn is dictated by the mean time between failure (MTBF) values for each sub-system. At the conceptual design stage, it is difficult to predict the MTBF value of a component with no pre-existing data. Thus, an estimate of the MTBF value is made based on the MTBF values of similar components with pre-existing data. But, it is recognised that a more detailed assessment of failure modes will need to be carried out prior to preliminary design.

For the time being a set of baseline MTBF values are selected for different UAS sub-systems, which are representative of the sub-systems presented in Figure 7.17. These baseline values are factored to account for changes in the system configurations. The baseline MTBF values were taken from the data presented in the OSD report and by Gundlach [122], [139], which are defined in Table 7.3.

Table 7.3: Baseline MTBF values of critical UAS sub-systems.

Component	MTBF, hrs	Notes
Flight-critical avionics	5000	Includes autopilot, on-board processors, measurement sensors, etc.
Engine	1000	Non-aerospace engines
Communications System	5000	
Flight controls	300	Includes servos for control surfaces, linkage arms, wiring, etc.
Airframe	1000	Includes structural failure of the airframe.

The MTBM values for each sub-system is established by setting the maintenance to begin when the components reliability reaches a value of 0.55 i.e. the probability of failure is 0.45. This is defined as follows:

$$R = \exp\left[\frac{-t}{MTBF}\right] \tag{7.26}$$

As an example consider the flight controls sub-system, which has a MTBF value of 300 hrs. Using the equation above the MTBM value is calculated by setting the relaibility value, R, to 0.55 and calculating for time, t. For flight contols the MTBM is calculated to be 180 hrs, which is graphically llustrated in Figure 7.18.

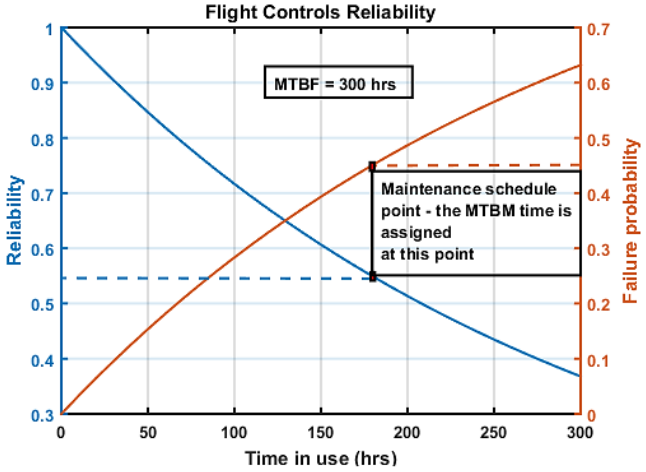


Figure 7.18: Flight controls reliability and system failure probability.

The reliability of a system can also be improved by adding redundancy into the system. Using the previous example, the flight control reliability can be improved by adding a duplex system with two strings, rather than using the simplex system. This in-turn will push the value of MTBM to be higher, and therefore the maintenance interval is less frequent. However, the addition of redundancy does have an adverse effect in terms of added complexity, mass, volume, power consumption, and cost. Thus, the choice of system architecture, in terms of redundancy, can impact the overall utility of the system. This architectural trade is addressed in more detail in the next chapter.

The maintenance cost can now be calculated by making assumptions in regards to the maintenance down time (MDT) for each major sub-system, the number of maintenance personal required for each sub-system, maintenance personal cost rate (£/hrs), and the consumable material cost during maintenance for each sub-system. In this simple model it is assumed that there are an infinite number of spares, and the spares availability is not considered in this modelling approach. The 'UAS checks' relates to system checks, replenishment of fuel, and reconfiguration of systems prior to launch. It should be noted that the data presented in Table 7.4 is illustrative, and is based on the authors' conversation with the 2Seas design team, and is not based on any previous program data (as the availability of data for maintenance is very limited for the class of UAS under consideration).

Table 7.4: Sub-system maintenance data.

Sub-System	MDT, hrs	Maintenance Personal	Man hour rate (£/hrs)	Consumable material rate (£/man-hour)
UAS checks	0.6	2	30	20
Engine	1.5	2	30	50
Airframe	3.0	2	30	20
Flight controls	0.8	1	30	30
Communications system	1.5	1	30	100
Flight-critical avionics	2.0	2	30	30

Now that the entire maintenance schedule has been established, the availability of a system at a given moment in time can now be determined. This is fully captured by operational availability, which is defined as the probability that a system operating under defined conditions will operate satisfactorily when called upon. Operational availability, A_O is defined mathematically as:

$$A_0 = \frac{MTBM}{MTBM + MDT} \tag{7.27}$$

Although reliability is typically associated with factors internal to the system, there are equally influential external factors that impact system reliability. These generally tend to be environmental influences. Though, some of these influences, such as precipitation, icing, and wind can be predicated to some extent, and be designed for. There are many unforeseen mishaps which lead to the loss of the UAS. The data from the OSD report [139] indicates that the UAS's with lower reliability are also those that are smaller, yet fly at higher profiles traditionally served by larger aircrafts. As a secondary effect to mishap rate, it was found that cruise Reynolds number is correlated to mishap rate of various manned and unmanned aircrafts.

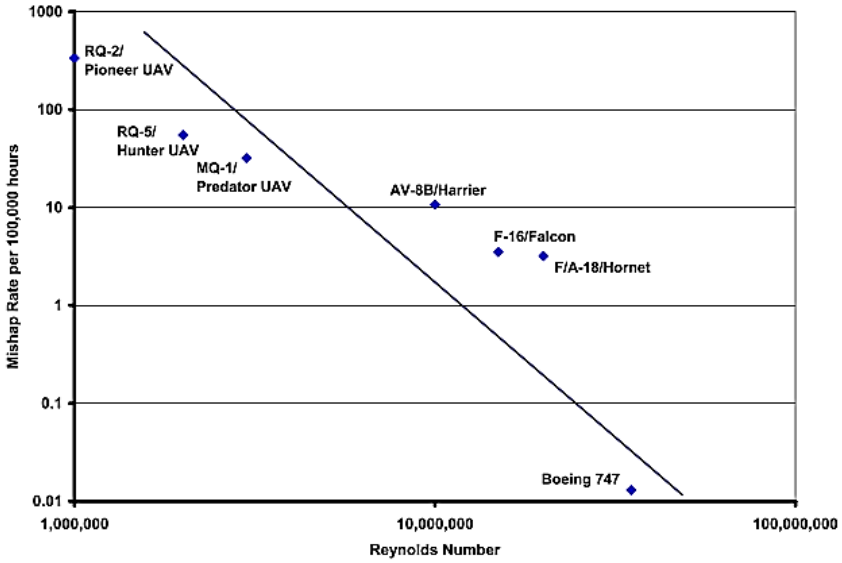


Figure 7.19: UAS mishap rate versus Reynolds number [139].

Base on this data, the accident rate per 100,000 flight hours $LR_{Accident}$ is calculated to be in the range of 5-10, depending on the system architecture under consideration. In the event of losing an UAS, the system is replaced, with the cost of replacement being defined as:

$$C_{Replace} = N_{AC,Loss} \cdot CP_{UAS,unit}$$

$$\text{Where, } N_{AC,Loss} = LR_{Accident} \cdot Flt_{Hours} \tag{7.28}$$

where $CP_{UAS,unit}$ is the unit cost of the system.

7.6 Model Validation

The accuracy of the simulation model is validated by comparing data from a UAS configuration, which is similar in configuration to that of the 2Seas UAS, developed by the University of Southampton (shown in Figure 7.20). The validation compares the structural mass and the operating mass of the system, at a fixed bassline mission profile.

Figure 7.20: The 2Seas UAS, developed by the University of Southampton.

The mass of a system is chosen as a metric for validation, as it is one of the most influential parameters on the vehicles performance and cost, and is therefore important to ensure that the mass estimates are approximately in-line with the real system.

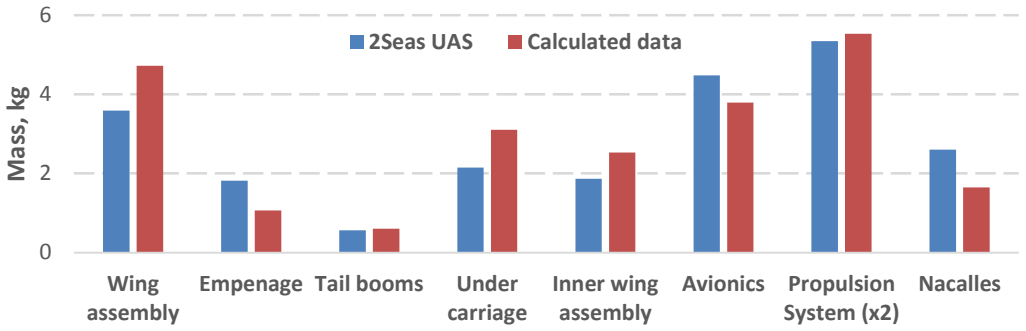
Before proceeding to model the UAS configuration, it is important to ensure that the UAS being modelled satisfies the constraints defined within the operating environment. The design of an UAS typically introduces numerous constraints, mainly derived from the limitations prescribed by the operating environment, and also by the system stakeholders. The constraints generated for the whole system can be vast, involving limitation on the voltage supply for different electrical systems, placement of electrical equipment, environmental protection systems, size limitations of payload systems, etc. Though all these constraints need to be taken into account to design a viable system architecture, this study limits the constraints relating to flight envelop performance, and geometric limitations of the UAS. This limits the scope of this study to a manageable size for conceptual analysis. The upper and lower values of the design and constraint variables used in this study are represented in Table 7.5.

Table 7.5: Design and constraint variable values.

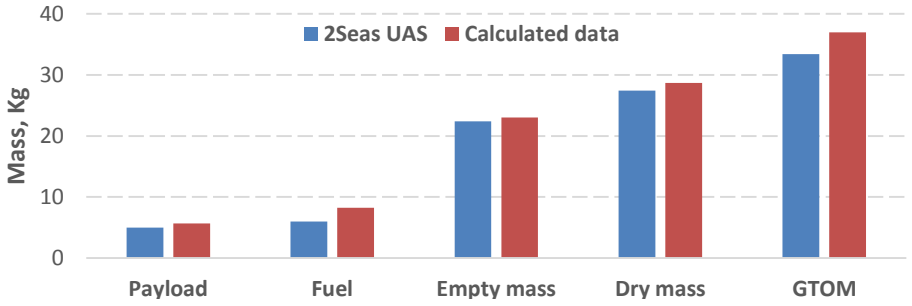
Design Variable	Upper bound	Lower bound
Wing Loading (kg/m^2)	40	20
Thrust-to-weight ratio	0.22	0.05
Wing aspect ratio	12	8
Constraint Variable	**Constraint Values**	
Take-off distance (m)	< 100	
Specific Excess Power (ft/min).	> 900	
Sustained Load Factor	> 3.5	
Wing Span (m)	< 3.2	
Wing quarter chord sweep (deg)	< 5	

The satisfaction of these constraints is achieved via a non-liner constraint optimiser, which effectively acts as a constraint satisfier. The optimiser is used to scale the wing loading W/S, the engine power-to-weight ratio P/W, and the wing aspect ratio AR in order for the UAS to meet the defined set of constraints. The study adopts the sequential quadratic programming (SQP) method to solve the nonlinear constrained optimisation problem. The SQP method works by creating a quadratic approximation to the Lagrangian, and creating linear approximations of the constraints to reduce the problem into a quadratic programming (QP) sub-problem. A more detailed description of the nonlinear constraint optimiser can be found in section 3.3.

The output of the simulation model is represented in Figure 7.21. In comparing the data, significant differences in the structural mass of system components can be noticed. This is especially true when comparing the masses of the wing assembly and under carriage, where the calculated data over predicts the mass by a significant amount. However, when comparing the masses of the empennage, nacelles, and avionics, the calculated mass is under predicted by an amount that is significant enough to offset the over predictions. This has resulted in the predicted empty mass to be relatively close to the actual empty mass of the 2Seas UAS, with a percentage difference of 2.6%. The overall predicted GTOM of the system however diverges by 10.6% from the actual, due to an overestimate of the required fuel mass. This is attributed to the differences in aerodynamic drag predictions. A more detailed mass breakdown of the system, with percentage differences can be found in Appendix E.



(a)



(b)

Figure 7.21: Comparison of (a) structural mass elements and (b) operating mass elements, of the calculated UAS data.

Though the predicted masses of the structural components are far from the actual, the overall empty mass and the GTOM of the system are within satisfactory bounds i.e. the masses are is in line with the class of UAS being modelled. For this reason it is believed that the modelling environment provides a good approximation to estimate the general trends in masses, due to changes in the design variables.

7.7 Multi-Objective Optimisation

Using the optimisation framework defined in section 3.3, the aim of this study is to find non-dominated Pareto optimal solutions for a given system architecture by varying a set of design variables. This optimisation problem can be defined as follows:

$$\text{Min} \quad \mathbf{J} = \mathbf{f}(\mathbf{x}) = [f_1(\mathbf{x}), f_2(\mathbf{x})]$$

$$\text{s.t.} \quad \mathbf{g}(\mathbf{x}) = [g_1(\mathbf{x}), g_2(\mathbf{x}), g_3(\mathbf{x}), g_4(\mathbf{x})] \leq 0$$

$$\mathbf{x}^L \leq \mathbf{x} \leq \mathbf{x}^U \tag{7.29}$$

Where $\mathbf{f}(\mathbf{x}) = [f_1(\mathbf{x}), f_2(\mathbf{x})]$ is a vector of the two objective functions – MAU function, and the LCC function. The definition of the utility function, in relation to the design of the UAS, will be addressed in greater detail in the next chapter of this thesis. The four inequality constraints $\mathbf{g}(\mathbf{x}) = [g_1(\mathbf{x}), g_2(\mathbf{x}), g_3(\mathbf{x}), g_4(\mathbf{x})] \leq 0$ are related to geometric and flight envelope constraints, which were defined in the previous section. The design variable vector $\mathbf{x}$ contains a set of four design variables, which are time on station, built in payload mass allowance,

cruise and loiter altitude, and loiter velocity. The choice of these design variables will again be addressed in the next chapter of the thesis. The upper and lower bound values for the design variables are presented in Table 7.6.

Table 7.6: Design variable upper and lower bound values.

Design Variable	Lower Bound	Upper Bound	Baseline Values
Time on station	1 hrs	3 hrs	2 hrs
Altitude	150 m	400 m	250 m
Payload mass allowance	0 kg	4 kg	0 kg
Loiter Velocity	24 m/s	30 m/s	26 m/s

This section discusses the results acquired from the optimisation framework. The optimisation framework follows a surrogate based approach, where a surrogate model is used to model the response of each of the two objectives. The process starts with a sampling plan, which in this case generates 40 infill sample points. The choice of 40 points is based on the general rule of sampling 10 times the number of design variables. The simulation model is run at each of these sample points, and the corresponding objectives of utility and LCC are extracted. Based on this input-output data set, $\mathbf{X} = \left[\mathbf{X}^{(1)}, \mathbf{X}^{(2)}, \ldots, \mathbf{X}^{(n-1)}\right]^{\mathrm{T}}$ and $\mathbf{y} = \left[y^{(1)}, y^{(2)}, \ldots, y^{(n-1)}\right]^{\mathrm{T}}$, a Gaussian Process (GP) model or Krig model is constructed for each one of the objectives.

7.7.1 Validation of Surrogate Model

To assess the validity of the surrogate models the leave-one-out cross-validation method is adopted. This method works by constructing the GP model by leaving one point out at a time, and then calculating the prediction value at the left-out point[13]. If a linear trend exists between the actual and predicted values, then this suggests that the GP model has captured the output response of the model accurately. Another method of assessing the validity of the GP model is via the evaluation of standardised cross validation residual (SCVR) values in the leave-one-out process, as proposed by Jones et al. [140]. The basic idea of SCVR is that it defines the number of standard errors by which the predicted and the actual values differ. Since the GP model is approximately 99.7% confident that the values lie within the mean prediction plus or minus three standard errors, an appropriate model validation test would be to see if the SCVR values lie in the interval [+3, -3]. If the values lie within this interval than the GP model is valid [140]. SCVR is defined as:

$$SCVR_i = \frac{y^{(i)} - \hat{y}_{-i}(\mathbf{x}^{(\mathbf{i})})}{s_{-i}(\mathbf{x}^{(\mathbf{i})})} \tag{7.30}$$

[13] When making cross-validation predictions, in theory one should re-estimate all the parameters of the GP model. However, dropping out a single observation in practice has a negligible effect on the maximum likelihood estimates. Therefore in practice the parameters are kept the same, but the correlation matrix $\mathbf{R}$ is recomputed using only $n - 1$ points [140].

where $y^{(i)}$ is the observed value at the i^{th} point that was left-out in creating the GP model, $\hat{y}_{-i}$ is the prediction at the left-out point, and s_{-i} is the cross-validated standard error of prediction.

Figure 7.22 shows the actual response plotted against the cross-validated prediction, and the corresponding SCVR values, for both the utility and normalised LCC outputs. These plots suggest that the GP model more accurately captures the response of the utility function, in comparison to the LCC function. This is indicated by the R^2 values on the plots, and also due to the fact that majority of the point for the utility function lie on the 45 degree line (as indicated by the red line). Figure 7.22 also plots the SCVR value for each point, it can noted that the majority of the points, with the exception of a few, are within the [+3, -3] interval. Hence, the surrogate models for both the objectives are deemed valid.

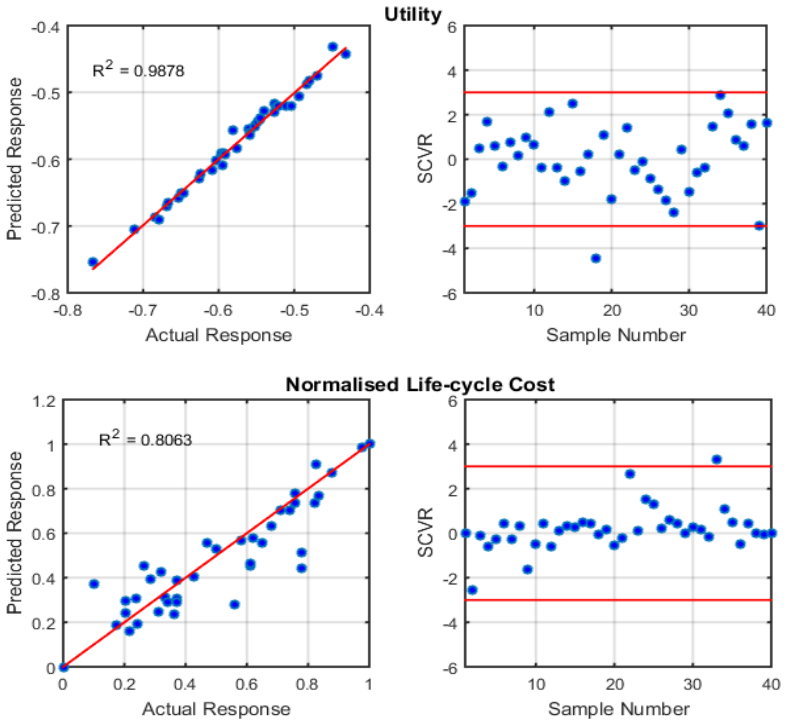


Figure 7.22: Surrogate model validation of the utility and LCC functions.

7.7.2 Update Points

The surrogate models constructed using 40 infill sample point are only approximates of the true functions that we wish to optimise. Thus, to improve the accuracy of the surrogate model in regions of interest, further infill points are required. Now, the question remains, where to choose the update point in order to improve the accuracy of the surrogate model. One may wish to select update points in the region of the predicted optimum by the surrogate to obtain an accurate optimal value quickly, which could be the local optimum- local exploitation. In other instances, one may wish to improve the accuracy of the surrogate more globally by placing update points where the mean squared error in the GP model is at its maximum - global exploration. This study adopts a balanced exploitation/exploration

approach by using the maximum expected improvement criterion $E[I(\mathbf{x})]$ (as defined in section 3.2.3.2). For a more complete description of multi-objective optimisation with surrogate models, and the mathematical formulation of the GP model and $E[I(\mathbf{x})]$, the reader is referred to Keane et al [13], and Forrester et al [34].

Figure 7.23 shows the progress of the search for five iterations. At each iteration contour plots of expected improvement $E[I(\mathbf{x})]$ for both the objective functions are plotted, in addition to the plot of the calculated Pareto front of utility and normalised LCC. The plots only show two out of the four design variables, time on station (TOS) and altitude. As these two variables have been shown to have a significant influence on the variation of $E[I(\mathbf{x})]$, with the other two variables having little to no influence. This is illustrated in Figure A.2 in Appendix F. From inspection it can also be noted that a majority of the update points are defined through the LCC function, with the utility function having minimal impact on $E[I(\mathbf{x})]$.

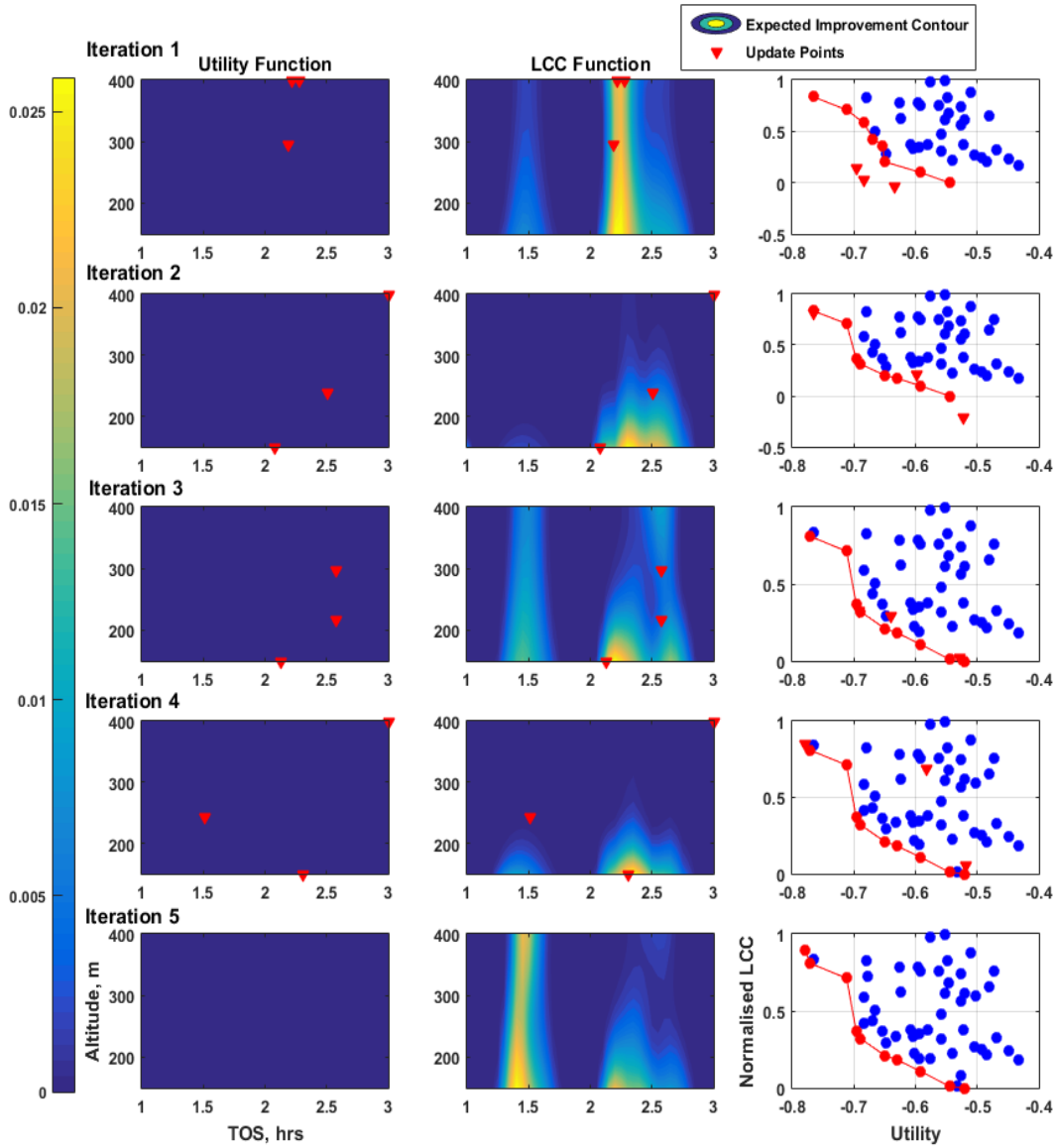


Figure 7.23: Progress of search of the multi-objective optimisation problem, using maximum $E[I(\mathbf{x})]$ strategy.

In this study three update points are selected at each iteration, where the location of the update point is based on maximising $E[I(\mathbf{x})]$, and in the diversification of the design variable space. This ensures that a well distributed Pareto front is obtained. The location of the update points in Figure 7.23 are represented by up-side down triangles. At the end of each iteration the GP model is re-tuned, with the inclusion of the update points, and the process of finding three new update points starts again. However, before progressing to a new iteration the solutions to the Pareto front are checked against a convergence criteria. This study defines the convergence criteria as being one where if the Pareto front is unchanged (unchanged is defined as less than two new points on the Pareto front) consecutively for two iterations, than the convergence is triggered. This is illustrated in iteration 3 and 4 in Figure 7.23, where only one new point is added to the front, thus terminating the optimisation at iteration 5.

7.7.3 Fuzzy Pareto Front Relaxation Factor Study

The fuzzy Pareto front representation requires the definition of a relaxation factor K_f value. This is achieved by applying the max dispersion of design space MD_{DS} metric, and max distance from the Pareto front MD_{PF} metric. Figure 7.24 represents the changes in MD_{DS} and MD_{PF}, due to variations in the relaxation factor. The aim is to find a value that maximizes the value of MD_{DS}, whilst at the same time minimises MD_{PF}.

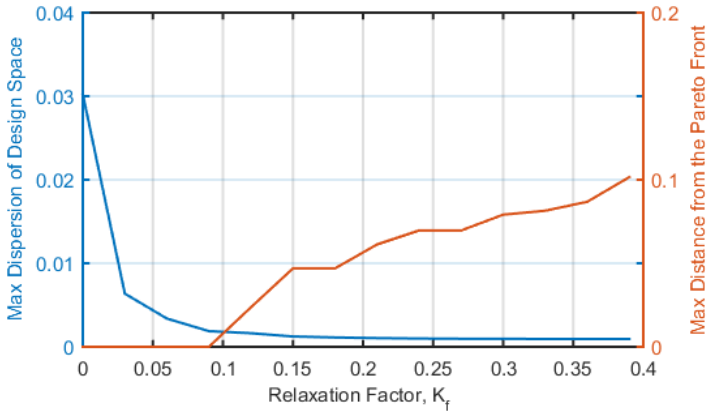


Figure 7.24: Impact of varying relaxation factor K_f.

The ideal choice for K_f, with $\max MD_{DS}$ and $\min MD_{PF}$, will lead to $K_f = 0$ i.e. the global Pareto front. However the aim here is to introduce points close to the global Pareto front, which requires a compromised solution that sacrifices both MD_{DS} and MD_{PF}. A value between 0.05 and 0.09 would provide a good compromise, where the value of MD_{PF} remains effectively constant, and MD_{DS} is in the decay region before reaching a steady state value. Thus, this study chooses a value for K_f of 0.05.

7.7.4 Optimisation Results

The final Pareto optimal solutions of the optimisation process is illustrated at the bottom right hand corner of Figure 7.23, where the solid red line indicates the Pareto front, and the

dotted blue points represent dominated solutions. The results of the Pareto optimal solution set are represented in Table 7.7. The data indicates that the maximum utility point is defined by the upper bounds of all four design variables. In comparison to the minimum LCC point, where the altitude and loiter velocity are at their lower bound values.

Table 7.7: Pareto-optimal solutions.

Utility	Normalised LCC	Time on Station (TOS), hrs	Alt, m	Payload allowance, kg	Loiter Velocity, m/s
0.78	1.00	3.00	400.00	4.00	28.00
0.77	0.90	3.00	400.00	3.00	28.00
0.71	0.80	2.95	398.33	1.84	24.12
0.70	0.41	2.28	400.00	3.24	28.00
0.69	0.36	2.22	400.00	2.69	28.00
0.65	0.24	2.03	390.04	1.88	27.78
0.63	0.20	2.19	294.92	1.92	28.00
0.59	0.12	2.00	272.28	1.92	24.38
0.55	0.01	2.06	162.64	1.88	27.67
0.52	0.00	2.07	150.00	1.90	21.00

To get a better understand of the impact of each design variable on the objective function, contour plots of the objective functions versus two of the four variables are plotted in Figure 7.25, with the remaining two variables held at their baseline value (the baseline values are defined in Table 7.6). First, in comparing the utility function plots, interactions between multiple design variables can be noticed. However in comparison, the normalised LCC function plots shown little interaction between the design variables. The variation in LCC is mainly dominated by TOS, and partly by altitude.

To be more numerate in measuring the impact of each design variable on the objective function, a simple main effects analysis was carried (see Box [141] for a more in-depth review of main-effects and interaction effects). The main effects, and corresponding interaction terms, of each design variable are calculated by running a two level full factorial design, generating a total of $2^4 = 16$ different sample runs. The range of variation of the design variables in the full factorial design is set to be $\pm 10\%$ around the baseline values. The evaluation of the output response at each of these points is carried out using the GP predictor model. In evaluating the output response, the main effects and the interaction terms are calculated, the results of which are represented in Table 7.8. To make the results more comparable between the two objective functions, the output responses were normalised such that they lie in the interval [0 1]. The results indicate that the main effects for the utility function are mainly spread across three variables, TOS, altitude, and loiter velocity, with altitude dominating the other two by a small amount. This is also true for the interaction effects, where the impact of the interacting variable on the utility function are mainly derived from TOS, altitude, and loiter velocity. The main effects of the LCC function on the other hand shows the TOS variable to have a dominate effect on the output response, relative to the

other variables. In terms of the interaction effects, only TOS and altitude show some interactive impact. There is little interaction effects with the other design variables, which is signified by the plots in Figure 7.25.

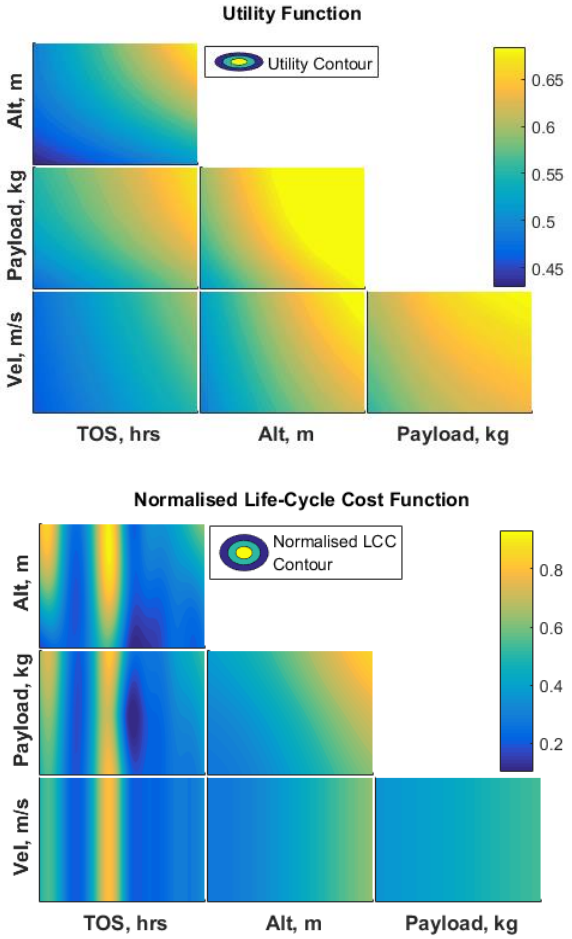


Figure 7.25: Contour plots of the LCC and utility of the Krig predictor function.

Table 7.8: Main-effects analysis and interaction effects.

		Utility	LCC
TOS	X1	0.152	-0.439
Alt	X2	0.162	0.054
Payload allowance	X3	0.043	0.006
Loiter Velocity	X4	0.142	0.000
Interaction Terms	X1:X2	0.027	-0.042
	X1:X3	0.000	-0.001
	X1:X4	0.030	0.000
	X2:X3	0.000	0.000
	X2:X4	0.030	0.000
	X3:X4	0.000	0.000

7.8 Conclusion

This chapter introduced the UAS design case study, providing a brief overview of the potential applications of UAS's, and its restrictions within the operating environment. In particular, this case study is an illustrative example based around the European Union 2Seas research project, in which University of Southampton have been involved in developing an UAS in application for maritime surveillance. As a means of modelling the system capabilities, in terms of its performance and cost elements, this chapter has provided a detailed breakdown of the modelling environment. The methods applied in modelling system performance, and its associated cost elements are predominantly based on low fidelity models, such that multiple system architectures can be evaluated rapidly. To validate the accuracy of the modelling environment, the results of the predicted masses were compared against the actual sub-system masses of the 2Seas UAS. The results indicated that the sub-system masses were off by a significant amount, which is not surprising as the predicted masses were calculated based on simple parametric relationships. However, the overall empty mass and the gross-take-off mass of the system were predicted with reasonable accuracy, thus validating the modelling environment to be reasonably accurate in predicting the overall system mass.

As a means of validating the optimisation framework, defined in section 3.3, the multi-objective problem of maximising utility, and minimising LCC was solved to obtain Pareto optimal solutions. This multi-objective problem was constrained to a single system configuration, where the system under consideration is the same as that used in the model validation section i.e. this system configuration is a replicate of the 2Seas UAS. The derivation of the utility function, and the choice of the design variables in the optimisation will be addressed in further detail in the next chapter of this thesis. The optimisation framework adopts a surrogate based approach to identify Pareto optimal solutions. Starting with 40-infill points, a Gaussian process (GP) model was constructed for both the objectives based on the output response of the modelling environment at those 40 points. The GP model was validated by calculating the errors in the actual and predicted values. To further explore the surrogate model in regions of interest, update points are identified based on maximising the expected improvement criteria $E[I(\mathrm{x})]$. Further update points were added to the surrogate model until the convergence criteria was met. This iterative process is depicted in Figure 7.23. The output of the optimisation framework is a set of Pareto optimal solution for the system under consideration, which has been the main focus of this chapter, and a precursor to the evaluation of high-impact decision variables in the next chapter.

Chapter 8: Implementation of the Decision Support Framework

This chapter expands upon the case study presented in the previous chapter to describe the study of architectural exploration of an Unmanned Aircraft System (UAS), for use in maritime surveillance. The exploration of the solution space considers multiple feasible system architectures that meet system functionality, whereby the process of exploration follows the decision support framework presented in Chapter 6: . The objective of this chapter is to generate and evaluate viable system architectures in order to identify the global Pareto front, and high-impact decision variables that have a strong impact on architectural decision-making. By identifying important decision variables the decision maker is able to identify the influence of each decision variable on system properties.

A notional set of system requirements are generated, which aim to be representative of the requirements that would be generated by the customer and other system stakeholders in a large aerospace design project. The availability of requirements allows the decision maker to begin the process of functionally decomposition, at a given level of system abstraction. Following the allocation of design alternatives to satisfy system functionality the decision support framework is applied: *construction*, *search*, and *representation*. The chapter concludes with quantifying the data generated by the decision support framework to make recommendations of decision variables that should be explored in future iterations of the exploration process in order to improve upon the Pareto optimal solution set that was generated from the first iteration.

8.1 Construction

8.1.1 Functional-Means Analysis

The task of functional decomposition may be performed by implicitly carrying through assumptions about the expected physical architecture in mind, with the notion of what subsystems are likely to enable system and subsystem level requirements. In this case study the decomposition of system functionality is performed at the whole system level, and not at a sub-system level. The functional hierarchy tree generated via the decomposition process is represented in Figure A.3 in Appendix G. Here, only the leaf nodes of the hierarchy are carried forward for the enumeration of design alternatives to each decision variable.

Table A.4 in Appendix H represents the alternatives generated from FMA. A whole system architecture is generated by selecting one alternative from each function. However, not all design alternatives are carried forward for further analysis, as they are deemed to be sub-optimal or not mature enough to be implemented into the system. This is representative of the decision maker's knowledge of the solution space. The design alternatives in the grey boxes in Table A.4 represent solutions that have been removed from the solution space. Inspecting Table A.4 it can be noted that solutions to certain decision variables have already been fixed. For example the decision variable *Deliver propulsive power*, the decision is fixed to implement twin 4-stroke piston engines. This is typical of many engineering design projects, where certain decisions are taken early on in the project, which in-turn act as an enabler to other system level decisions. Again, taking the example of *Deliver propulsive power*, the selection of a 4-stroke piston engine has enabled the selection of an engine generator for the decision variable *Deliver electrical power for flight critical systems*.

Once the design space has been reduced to a set of viable design alternatives, the interest lies in exploring multiple system architectures. The impact of these design alternatives on system objectives, such as value and cost, will influence the decision-making process. However, the inclusion of many design alternative will result in the total number of system architectures to grow to a number that might be unmanageable to evaluate in a limited time period and under budgetary constraints. To manage the size of the solution space additional constraints are added, which can represent physical or preferential constraints that define the viable combinations of design alternatives according to the compatibility rules specified.

8.1.1.1 Architectural Trades

This section defines the architectural trades that are carried forward from FMA. These trades relate to decision variables x_i, whose domain D_i contains more than one design alternative $\{v_1, \ldots\ldots, v_n\}$.

Deliver electrical power to payload:. The first alternative considers the generation of electrical power from the engine drive shaft via a generator. Thus, the total shaft power available to the propeller is defined as:

$$P_{shaft,Prop} = P_{shaft} - \frac{P_{Gen}}{\eta_{Gen}} \tag{8.1}$$

where P_{shaft} is the shaft power generated by the piston engine, P_{Gen} is generator load which is equal to the aircrafts power demand during operational use, and η_{Gen} is the generator efficiency, which for the current case study is assumed to be 70%.

However, as power demand is increased during the search phase of the mission i.e. when the optical camera system is turned on, the available propulsive power is reduced. This raises concern of drawing too much power from the engine, which can lead to performance degradations of the aircraft. To mitigate this issue, a separate battery source is considered which will provide electrical power to the payload system and the communications system

during the course of the search phase. However power to flight critical avionics will still be provided by the engine generator. The battery is sized to meet the power demand during the search phase, which is defined as follows:

$$M_{Batt} = \frac{Enery_{UAV}}{E_{spec}\,\eta_{Batt}\,f_{Usable}} \tag{8.2}$$

Where M_{Batt} is the mass of the battery which is dependent on energy demand of the UAS $Enery_{UAV}$(W-hrs). E_{spec} is the battery specific-energy (W-hrs/kg). η_{Batt} is the battery efficiency which is assumed to be 100%, even though not practically possible it gives a theoretical value for the battery mass. The f_{Usable} factor accounts for the battery pack depth of discharge, which is assumed to be 85%. Specific energy values of different battery types are represented in Table 8.1- this data is taken from Gundlach [122].

For the current case study a lithium-ion-polymer (Li-Po) battery is chosen, as lithium ion (Li-Ion) and lithium-ion-polymer batteries have been used widely in small UAS propulsion systems [122]. Though lithium sulphur (LiS) has the highest theoretical and practical specific energy values, the current cells have limited charge/discharge cycles. This limits its use to a small number of flights, in comparison to Li-Po and Li-Ion [122].

Table 8.1: Characteristics of battery chemistries [122]**.**

Battery Type	Theoretical Specific Energy, W-hr/kg	Practical Specific Energy, W-hr/kg	Specific Power, W/kg	Cell Voltage, V
Lead acid (Pb/acid)	170	30-50	180	1.2
Nickel cadmium (NiCd)	240	60	150	1.2
Nickel metal hydride (NiMH)	470	23-85	200-400	0.94-1.2
Lithium ion (Li-Ion)	700	100-135	250-340	3.6
Lithium polymer (Li-Po)	735	50.7-220	200-1900	3.7
Lithium sulphur	2550	350	600-700	2.5

Conventional and Unconventional launch and recovery systems: The demand for the UAS to take-off and land over a fixed runway length generally tends to dictate the engines thrust-to-weight ratio T/W and the aircrafts wing area. This in-turn can have a significant impact on the mass and size of the UAS. In order to minimise the size and mass, a tension-line launch and cable assisted recovery is considered to assist the UAS during take-off and landing. The idea here is to reducing the demand on the engines such that the mass and size of UAS can be reduced significantly. However, the use of external launch mechanisms results in the UAS to experiences greater inertial loads, caused due to faster rates of acceleration and deceleration from tension-line launch and recovery. This in-turn increases the structural mass of the UAS.

The take-off ground roll distance can be quantified by the Take-off Parameter (TOP) as follows:

$$S_{ground\ roll} = \alpha\, TOP = \alpha \frac{W/S}{\sigma C_{L_{TO}} BHP/W} \tag{8.3}$$

Where, α is a positive constant, W/S is the aircraft wing-loading, σ is the ratio of air density at take-off altitude to the air density at sea-level, $C_{L_{TO}}$ is the average lift coefficient at take-off, assumed to be 0.8, and BHP/W is the engine horse power-to-weight ratio.

The difference in ground roll distance between the two launch mechanisms considered are accounted for by simply changing the α parameter. For conventional launch α is set to 6.92, and for a tension-line launch α is set to 5.7.

Transmit and receive data: Communications systems are primarily used on-board the UAS to transmit and receive data to and from the ground control station. Thus, it is considered to be a vital sub-system for the effective operational use of an UAS. Figure 8.1 represents three different antenna systems considered in this architectural trade study.

Omni-directional, Vertical Polarisation.2dBi Gain

Bladed Omni-directional, vertical polarisation, 2dBi Gain

Omni-directional, circular polarisation, 3dBi Gain

Figure 8.1: UAS antenna systems.

Different antenna systems are conceptually assessed by considering the link budget analysis. The performance of an antenna system is defined by its signal-to-noise ratio (SNR), which defines the signal strength at the receiver in comparison to the receiver's noise. One of the main components in the loss of the receiver signal strength L_R is polarisation loss, which is caused due to polarisation mismatch between the transmitter and receiver antenna. When a vertically polarised antenna is used to transmit the signal, maximum signal power is achieved if the receiving antenna is also vertically polarised. However, polarisation mismatch may occur when the aircraft banks, causing a drop in the signal strength at the receiver. To counter this, a circular polarised antenna is considered. The circular polarised antenna allows the link budget to be maintained irrespective of the position of the aircraft. However circular polarised antennas have a large diameter, which in-turn can increase the aerodynamic drag of the antenna, and impact the integration of the communications system in terms of mass and size.

The impact of antenna integration on aerodynamic drag can be predicted by assuming that the antenna is a circular cylinder of diameter d and length l. It is also assumed that the operating Reynolds number of the flow across the antenna will be less than 3×10^5, which gives an approximate drag coefficient of 1.2. Using this information the antenna drag can be predicted as:

$$D_{Antenna} = 1.2 \cdot (1/2 \cdot \rho \cdot V^2) \cdot l \cdot d \tag{8.4}$$

In order minimise the drag of the antenna, a bladed antenna is considered. Here it is assumed that the antenna is contained within a symmetric aerofoil shape fairing. Given the thickness-to-chord ratio t/c of the aerofoil, the drag for a bladed antenna can be predicted as:

$$D_{Antenna} = C_D(t/c, Re) \cdot (1/2 \cdot \rho \cdot V^2) \cdot l \cdot d \tag{8.5}$$

Here $C_D(t/c, Re)$ is the drag coefficient of the aerofoil, which is a function of thickness-to-chord ratio and the flow Reynolds number.

Capture imagery over the target area: This function mainly focuses on the optical payload system. The intent of the optical payload system is to deliver high definition video imagery to the ground control station. The focus of this architectural trade is in the variation of the number of detector elements in the focal plane array, which has a direct impact on the field of view (FOV) of the optical system and the ground sampling distance. This governs the achievable image quality.

$$GSD_H = 2 \cdot \left(\frac{FOV_H}{2 \cdot H_{Pix}}\right) \cdot R$$

$$GSD_V = \frac{2 \cdot \tan(FOV_V/2 \cdot V_{Pix})}{\cos(\theta_{Look})} \cdot R$$

$$GSD_{geometric\ mean} = \sqrt{GSD_H \cdot GSD_V} \tag{8.6}$$

Where, GSD_V and GSD_H are the vertical and horizontal ground sampling distances and $GSD_{geometric\ mean}$ is the geometric mean value, R is the slant range from the target to the sensor, θ_{Look} is look angle between the sensor and the target, H_{Pix} and V_{Pix} are the number detector elements in the horizontal and vertical direction.

For a given slant range and look angle the image resolution can be increased by increasing the number of detector elements. However, the increase in the number of detector elements also results in a higher data rate that is required to be transmitted back to the ground control station. This in-turn results in a higher transmission power by the communications system to maintain the link margin. The raw data rate generated by the digital video stream is defined as:

$$Rate_{Data} = H_{pixels} \cdot V_{pixels} \frac{Bits}{Pixel} \cdot Rate_{Frame} \cdot F_{Compression} \tag{8.7}$$

Where, $Bits/Pixel$ is the pixel depth, which is dependent on the image standard. It is assumed that the pixel depth is 30 bits/pixel. $Rate_{Frame}$ is the frequency of image generation, for a typical video stream the image frequency is 24Hz. $F_{Compression}$ is the image compression factor, a 2:1compression with low loss is applied.

Protect system from overheating: The aim here is to keep system components within its defined temperature limits. This is achieved via the implementation of an environmental control system (ECS). Major sources of heat come from the control surface actuation mechanisms, and more dominantly from the piston engine. Providing no ECS control provisions has been considered as it simplifies the system architecture, however the trade-off comes with the cost incurred in terms of poor reliability. To improve reliability, specifically of the engine components, ram air cooling has been considered. Ram air cooling works by allowing air to enter through openings in the airframe and passes over the hot components before being exhausted. The cooling process of components is achieved through conduction and then convection of heat. However, the process of cooling results in an increase in drag through friction and separation. This cooling drag is modelled as a power loss to the engine. For conceptual design purposes Torenbeek [142] recommends the following cooling drag relationship for reciprocating engines:

$$\frac{D_{Cool}}{q} = C_{Cool} \cdot \frac{P_{Shaft} \cdot T_{amb}^2}{\sigma \cdot V} \tag{8.8}$$

Where the empirical constant $C_{Cool} = 4.9 \times 10^{-7}$ $\text{ft}^2/\text{lb-}^{\circ}\text{R}^2$.

Protect system from component failures: Avionics architectures typically have 1 to 3 strings – simplex, duplex, and triplex. Components that form a string are air data systems, autopilot, control surface actuator set, and the UA management system. The failure of flight critical components in a simplex string will result in the loss of the aircraft. Duplex and triplex strings improves reliability by offering redundancy. For a triplex system, in the event of one string failing command is handed over to another string by the use of a voting scheme, which requires implementing a voting algorithm. However, rather than applying a voting scheme a much simpler approach is considered, where two sets of control surfaces are made available for each function (i.e. two elevators, and two rudders). But this does come at a disadvantage, in terms of an increase in control surface actuation mass and a rise in power demand due to the doubling in the number of actuation systems.

The trade considered here focuses on reliability versus increase in mass, power, and cost. The trade focuses on the reliability of a low-cost duplex control surface actuation mechanism versus high-end single string actuation mechanism. For the high-end simplex actuation mechanism the $MTBF$ is assumed to be 700 hours, in comparison to the low cost actuation mechanism which is assumed to have a $MTBF$ of 300 hours. But the duplex system offers additional redundancy, which increases its reliability.

8.1.2 Design Structure Matrix

The reduction of the architectural solution space can exhibit interrelationships between different decision variables. The DSM is used as a first pass approach in capturing these interrelationships, which requires the decision-maker to specify the existence of constraint relationships between decision variables. The output of the DSM is an adjacency matrix A_{ij}, which describes the connectivity between elements i and j. Figure 8.2 shows an $n \times n$ DSM, which is representative of the adjacency matrix. The adjacency matrix A_{ij} from the DSM can be visualised as a network diagram, as shown in Figure 8.3.

Table 8.2: Leaf nodes of the functional decomposition used to represent decision variables.

System Functions	
1. Package and store payload and other system components.	7.1.2. Unconventional launch
2. Generate lift for steady level flight.	7.2.1 Conventional recovery
3. Deliver propulsive power.	7.2.2.Unconventional recovery
4. Integration of propulsion system.	8.1. Transmit and receive data
5.1. Deliver electrical power to flight critical systems.	8.2. Communication system integration. On-board the UAS.
5.2. Deliver electrical power to payload.	9.1. Capture imagery over the target area.
6.1. Provide longitudinal stability.	9.2. Integration of sensors.
6.2. Provide lateral stability.	10.1. Protect system from overheating.
7.1.1. Conventional launch	10.2 Protect system from component failures.

The nodes of the network are categorised into three sections. The nodes in green represent decision variables that contain more than one design alternative within its domain. The nodes in yellow represent decision variables that contain only one design alternative within its domain i.e. the solution is fixed, but are connected to other decision variables. Finally, nodes in red represent decision variables that contain only one design alternative within its domain, and are not connected to other decision variables i.e. they do not influence the feasibility of solutions of other decision variables.

The degree centrality measure represented in Figure 8.3 defines the degree of connectivity between each decision variable. The results indicate that decision variable 5.2 – *Delivery of electrical power to payload* is the most strongly connected within the network. This suggests that the decision variable *Delivery of electrical power to payload* has a significant impact on the feasibility of solutions of other decision variables within the network. The following sub-section provides a description of the connectivity of each node.

	1	2	3	4	5.1	5.2	6.1	6.2	7.1.1	7.1.2	7.2.1	7.2.2	8.1	8.2	9.1	9.2	10.1	10.2
1	0	0	0	0	0	0	**1**	**1**	0	0	0	0	0	0	0	0	0	0
2	-	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	-	-	0	**1**	**1**	**1**	0	0	0	0	0	0	0	0	0	0	0	0
4	-	-	-	0	0	0	0	1	0	0	0	0	0	0	0	0	**1**	0
5.1	-	-	-	-	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5.2	-	-	-	-	-	0	0	0	0	0	0	0	0	0	1	0	**1**	**1**
6.1	-	-	-	-	-	-	0	0	0	0	0	0	0	0	0	0	0	0
6.2	-	-	-	-	-	-	-	0	0	0	0	0	0	0	0	0	0	0
7.1.1	-	-	-	-	-	-	-	-	0	**1**	**1**	**1**	0	0	0	0	0	0
7.1.2	-	-	-	-	-	-	-	-	-	0	**1**	**1**	0	0	0	0	0	0
7.2.1	-	-	-	-	-	-	-	-	-	-	0	**1**	0	0	0	0	0	0
7.2.2	-	-	-	-	-	-	-	-	-	-	-	0	0	0	0	0	0	0
8.1	-	-	-	-	-	-	-	-	-	-	-	-	0	0	**1**	0	0	1
8.2	-	-	-	-	-	-	-	-	-	-	-	-	-	0	0	0	0	0
9.1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	0	0	0	0
9.2	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	0	0	0
10.1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	0	**1**
10.2	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	0

Figure 8.2: DSM matrix of the interrelationships between decision variables.

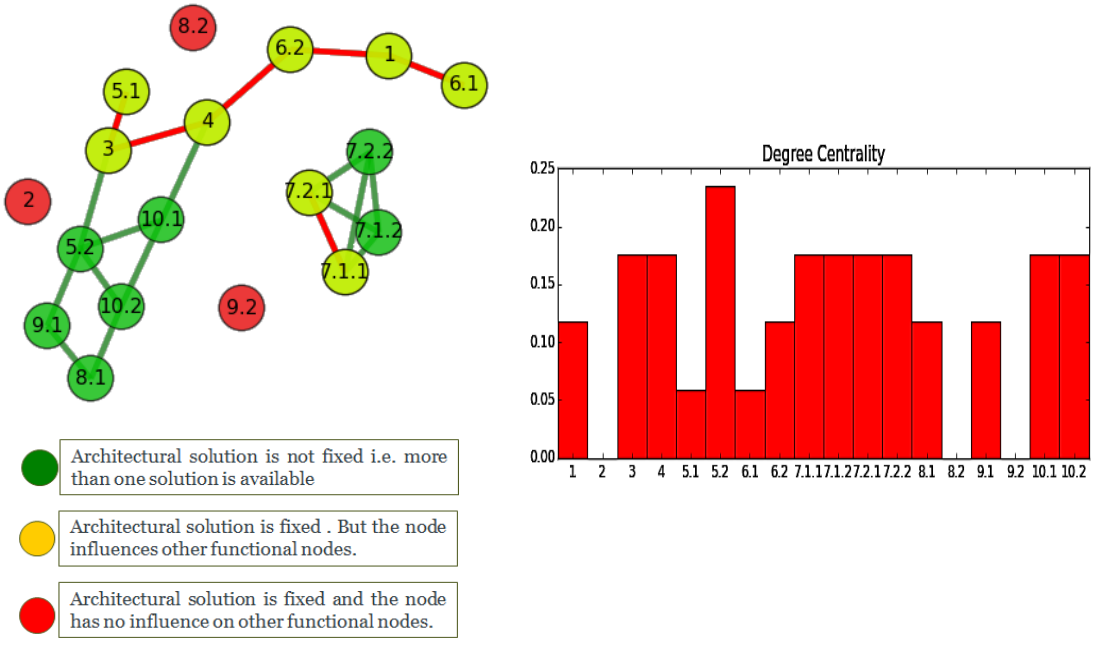


Figure 8.3: Network diagram of the interrelationships between decision variables.

8.1.2.1 Network Connectivity

Decision variable 1.0 to 6.1 and 6.2: The selection of a fuselage configuration to package payload and other system components has a direct impact on the design of the empennage. In this case, the choice of a detachable pod to package the payload and other system components provides modularity for different payload systems to be integrated into the UAS. However, the moment-arm required to maintain aircraft stability would be too small for the empennage to be directly attached to the detachable pod structure. Thus, to extend the moment arm, whilst at the same time aiming to minimise the structural mass of the aircraft, a boom configuration is considered.

Decision variable 3 to 4: The selection of a propulsion system to be used has a direct impact on the integration of the propulsion system onto the airframe. For example, the selection of solar cells can dictate the design of the wing to accommodate for the solar panels to be placed on the wing surface. The type of propulsion system, and its location can also impact the transmission of power to the propellers. For the current design a twin four stroke piston is selected. The integration of two engines on the airframe narrows down the available options for propulsion system integration. In this case a wing tractor configuration is chosen.

Decision variable 3 to 5.1 and 5.2: The delivery of electrical power to avionics, payload system, communications system, and other system components is a major driver in UAS design. For example, the use of solar power and fuel cells as a propulsion system would require a battery pack to store energy, which is than dissipated to other system components as required. However, the use of a reciprocating engine allows power to be taken directly from the engine drive shaft via a generator, which can be more efficient. But the performance of an aircraft can be affected if the power demand is too high. For the current case study it was decided that power to flight critical system components will be delivered by the engine generator, and power to the payload and communications system is left open to either a separate battery source, or via the engine generator.

Decision variable 4 to 6.2: The selection of a twin engine configuration can impact the design of the vertical stabilisers. In the case of a single engine failure the vertical stabiliser should be able to provide enough yaw authority to keep the aircraft in level flight. This requires the control surfaces to be sized to meet the single engine failure condition, and have a large enough moment-arm to provide the required yaw moment.

Decision variable 4 to 10.1 and 10.1 to 10.2: The reciprocating engine and on-board avionics are major sources of heat generation. Heat load generated by these components needs to be dissipated in order to improve component reliability and prevent system failure. The integration of the engine onto the airframe can have a major impact on the requirements to cool and dissipate engine heat. For example, installing the engine in an enclosed nacelle will require extra provisions to dissipate heat from the enclosed area to the surroundings, this is also true for enclosed avionics components. Thus, the integration of the propulsion system and avionics can impact the choice of design for the thermal management systems.

Decision variable 5.2 to 8.1and 9.1: The use of high resolution optical sensors can generate vast amounts of data, which can either be transmitted directly back to the ground control station, or stored and processed on-board the UAS. For the current design problem it was assumed that data would be directly transmitted back to the ground control station with no on-board data storage. The transmission of high resolution video imagery over long distances can require a relatively large signal transmission power from the communications system, which is delivered by an on-board power generation system. Combining this power demand with the power demand from other avionics components can produce a large electrical power load. This can impact the aircrafts performance if power is taken directly from the engine. To counter this effect a separate power source in the form of a battery is considered to deliver electrical power directly to the payload and communications system.

Decision variable 7.1.1, 7.1.2, 7.2.1, 7.2.2: The choice in implementing a launch and recovery system is highly interlinked. In most cases a conventional landing gear design will provide both the launch and recovery functionality. However, as the runway length/space is constrained to smaller distances, more unconventional approaches are required to launch and recover the system safely. The use of unconventional launch systems, such rail launch can significantly reduce the take-off distance. But with no landing gear other means of recovery need to be considered. This could range from a simple skid belly recovery to net and parachute recovery.

Decision variable 8.1 to 9.1: The selection of the communications system design can influence, or be influenced by the rate of data to be transmitted to and from the ground control station. For low resolution payload sensors the transmitted data rate can be relatively small, in which case a low gain antenna that can be implemented. These antennas have a small integration footprint on the airframe. However, as high resolution payload sensors are used the required transmission data rates increase significantly, which require high gain antennas to be implemented in order to maintain the link margin. The use of high gain antennas with low losses, such as the high gain circular polarised antenna, can have a relatively larger footprint on airframe integration.

8.1.3 Logical Constraints

The majority of the compatibility relationships applied in this case study were described in the previous section. The problem now relates to translating these compatibility relationships into a set of logical constraint statements. This enable the constraint logic to be encoded into a programming environment, and enable the search algorithm to remove infeasible system architectures.

Table 8.3 defines a set of compatibility constraints imposed on the architectural solution space. These constrains represents a combination of physical and preferential constraints imposed by the decision maker on the solution space.

Table 8.3: Compatibility constraints imposed on the architectural design space.

Constraint Description	*scope*	*rel*
Delivery of Electrical Power Constraint	(4.2, 8.1, 9.2)	**if**(8.1 == "*1280 x 720, FOV = 2, FOR =20*" **&&** 9.2 == "*Duplex with two sets of control surface*") **then** {4.2 == "*Separate battery source*"} **else** {4.2 == "*Engine Generator*"}
Launch and Recovery Constraint	(7.1.2, 7.2.2)	**if** (7.1.2 == "*Tensioned Line Launch*") **then** {7.2.2 == "*Cable-assisted recovery*"} **else If** {7.1.2 == "*null*"} **then** {7.2.2 == "*null*"}
Environmental Control System Constraint	(4.2, 9.2, 10.2)	**if** (4.2 == "*Separate battery source*" **&&** 9.2 == "*Duplex with two sets of control surfaces*") **then** {9.1 == "*Ram air cooling*"} **else** {9.1=="*No ECS*"}
Communications Antenna Constraint	(8.1, 9.1, 10.2)	**If** (8.1 == "*1280 x 720, FOV = 2, FOR =20*" **&&** 9.2 == "*Duplex with two sets of control surface*") **then** {7.1 == "*Omni- circular polarisation, Gain 3dBI*"} **else** {7.1==" *Omni-Dipole, vertically polarisation, Gain 2dBI*" || 7.1== "*Bladed Antenna, Omni-dipole, vertical polarisation, Gain 2dBI*"}

Deliver electrical power constraint: This constraint focuses on the worst case power demand scenario. To minimise the impact of power demand on the engine generator, the electrical load is shared between the engine generator and a separate battery source. The battery pack is used to power the payload and communications system, whilst the engine generator will provide power to flight critical avionics.

Launch and recovery constraint: This represents a physical constraint on the launch and recovery systems. The choice of a conventional landing gear system provides both launch and recovery functionality. For an unconventional launch and recovery system, *tension-line launch* and *cable-assisted recovery* are integrated with the *Tricycle* landing gear configuration to minimise the launch and recovery distance. However, in the case where unconventional launch and recovery functionality is not required, the *null* variable is chosen to artificially remove the decision variable from the solution space.

Environmental control system constraint: This describes a preferential constraint, which focuses on the worst case scenario in terms of heat generation. The heat loads generated by the engines, battery pack, and the redundant actuation servos can be relatively large. This heat load needs to be dissipated for safe operational use of the UAS. In this scenario ram air cooling is used to dissipate heat to the surrounding environment. The cooling of components simply relies upon openings in the airframe to allow for air to pass through and cool the components. This approach is not very effective in managing the dissipation of heat and can reduce the reliability of components. But the complexity of designing and integrating an environmental control system is avoided.

Communications antenna constraint: This describes a preferential constraint, which again focuses on the worst case power demand scenario, with the aim of reducing power demand from the communications system. The high resolution optical camera system can produce a

relatively large data rate that is required to be transmitted back to the ground control station. The transmission power required to maintain the link margin between the platform and the ground control station for this condition can have a significant impact on the power demand. Thus, to minimise the transmission power a high gain antenna with lower polarisation losses is selected i.e. Omni-directional, circular polarisation, with a 3dBi gain. However, as previously mentioned the high gain circular polarised antenna can have a larger integration footprint on the airframe and can significantly contribute to the drag of the aircraft. For this reason, the high gain antenna is only selected for the worst case scenario.

8.1.4 MAU Model

The structuring of objectives is typically an iterative process, where the defined objectives need to be verified and agreed by the stakeholders before proceeding to the next stage. In structuring the objectives in a hierarchy, the fundamental objectives of the system can be identified. Fundamental objective can be described as objectives that qualitatively state all that is of concern in the decision context; they provide guidance for action by the decision maker, and foundations for any quantitative modelling and analysis that will follow i.e. these objectives describe the essential reason for being interested in the decision situation [62]. The objective hierarchy defined for this case study has identified five fundamental objectives, which are defined as follows:

1. **Maximise system reach:** This relates to the UAS travelling as far as possible to identify targets of interest.

2. **Maximise target coverage area:** This objective relates to the capability of the platform and the optical sensor system to cover a large area during the search phase of the mission.

3. **Maximise operational availability:** This relates to the system being able to operate satisfactorily when called upon i.e. it defines the probability that the system in the operational environment will operate effectively when called upon.

4. **Maximise the capability of the system to detect objects of interest:** This objective focuses on the payload system capabilities. Since the mission is focused towards surveillance, it is important that valuable and accurate information is available to the end-user. To achieve this objective the optical sensors needs to provide high resolution imagery to identify objects of interest.

The degree to which an objective is satisfied is measured by an attribute, which should clearly describe the objective under consideration. However, the measurement of an objective by an attribute is not always straight forward, as in some instances a direct relationship between an objective and an attribute doesn't exist. In such situations either a constructed or proxy

attribute is required to describe, at some level, the objective under the considered decision context. For the current case study the following attributes were identified for each objective:

Maximise system reach: A natural attribute to this objective is aircraft range. This attribute is dictated by the communications range of the aircraft, as defined in Section 7.4.1, and also by the aircrafts range, which is defined by the Breguet range equation. The Breguet range equation combines several aircraft characteristics, such as aerodynamics, engine performance, and available fuel mass, to determine aircraft range. This equation can be found in many aircraft design text books, and is not defined in this thesis. The reader is referred to Raymer [17] for more information on its derivation.

Maximise target coverage area: A natural attribute for this objective is coverage area. This measures the capability of the UAS to remain in the search area as long as possible, and the capability of the payload sensor to search large areas. Coverage area can be defined as follows:

The ground swath D covered by the sensor is defined as:

$$D = h \cdot [\tan(\theta_{Look} + 1/2 \cdot FOR) - \tan(\theta_{Look} - 1/2 \cdot FOR)] \tag{8.9}$$

Where, θ_{Look} is the look angle of sensor and FOR is the field of regard or the maximum field of view of the sensor. The ground coverage rate A_{Rate} of the sensor is defined as:

$$A_{Rate} = D \cdot V_{UAS} \tag{8.10}$$

Where, V_{UAS} is the velocity of the UAS during the search phase. Finally, the coverage area A can now be defined as:

$$A = A_{Rate} \cdot T_{Collect} \tag{8.11}$$

In this case $T_{Collect}$ is the collection time, or more appropriately it defines the loiter time of an aircraft during the search phase.

Maximise operational availability: A natural attribute relating to this objective is operational availability. The calculation of operational availability to a certain level of accuracy at the conceptual design stage is very difficult to achieve, unless data relating to component reliability, maintenance schedules, and logistics operations are already made available, which in many cases are not. Typically several assumptions are made to simplify the modelling process. For this case study assumption of $MTBM$ of sub-systems, such as power and propulsion, flight controls, communications, avionics, and airframe, are made based on previous UAS and manned aircraft designs. The MDT is calculated by assuming repair and replacement times for each sub-system.

Maximise the capability of the system to detect objects of interest: This fundamental objective aims to captures the payloads performance capabilities. The measurement of payload effectiveness can be defined by the sensors capability to identify targets of interest.

The measure should also include the capability of the UAS to integrate multiple, and potentially interchangeable, payload sensor systems onto the platform. Without the allowance for integrating different payload systems on-board the UAS, unforeseen future payloads with greater capability may be precluded. This is captured by measuring the additional payload mass allowance that is built into the UAS. The capture of a sensors capability to identify targets of interest is quantified by using the Johnson criteria [122]. The Johnson criteria is used as a means of determining the probability of the sensor system to detect, recognise, and identify a target based on the sensors resolution.

The assessment of the utility function is typically achieved by presenting the decision maker with a set of lottery scenarios to identify their risk attitudes to changes in the level of an attribute. However, this process can become cumbersome for a large set of system attributes. To avoid the lottery assessment process a predefined set of utility functions (risk seeking, risk neutral, and risk averse) have been made available to the decision maker to choose from. Though this process does not accurately capture the decision maker's risk attitude, it represents a general trend, which at the conceptual level is sufficient to identify value trade-offs.

The final step of this process requires combining all the utility functions, defined in Figure 8.4. The choice of this utility function depends on the verification of the independence assumptions. To examine the validity of any independence condition, several lottery scenarios are presented to the decision maker to identify if the independence condition is upheld. The verification of different independence conditions are defined in Section 2.3.5, where the focus was on the verification of additive and utility independence.

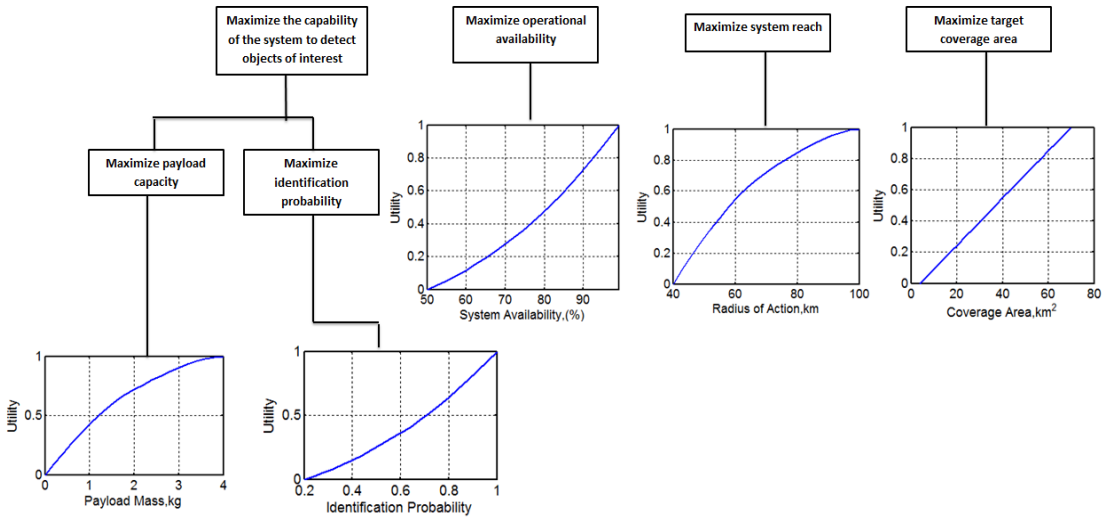


Figure 8.4: Measurement of fundamental objectives by means of measurable system attributes.

For the current case study, the set of attributes identified contain proxy attributes. The presence of proxy attributes makes the additive independence condition harder to validate, as the combined effect of the proxy attributes impact the decision maker's preference order.

For example, the decision maker's preference order is dependent on the combined effect of identification probability and coverage area. Thus, the additive independence condition does not hold, and the utility function cannot be determined by simply summing the utility functions of individual attributes. In such circumstances a less restrictive form of the independence condition, known as utility independence, can be applied to derive a multiplicative form of the utility function. The current case study makes use of the multiplicative form of the utility function to capture the overall system value. The derived scaling constants for the multiplicative model are presented in Table 8.4. For the purpose of illustrating the application of this framework, the derivation of the scaling constants and the utility functions were based on the author's judgements alone, but were later verified by the design team at the University of Southampton.

Table 8.4: Scaling constants for the multiplicative utility model.

Value trade-off of Attribute	Scaling Constant	Value
Coverage Area	k_1	0.35
Identification Probability	k_2	0.3
Payload mass capacity	k_3	0.1
Operational availability	k_4	0.25
Radius of action	k_5	0.17

8.2 Search

The *Search* phase aims to address assignment constraints by representing the problem as a CSP $(\mathbf{X}, \mathbf{D_X}, C_\mathbf{X})$; which allows the search algorithm (such as the recursive backtracking algorithm) to find feasible system architectures. This study makes use of a constraint programming package developed in Python[14]. The implementation of the Python package to solve this CSP is represented in Figure A.4 in Appendix I. In applying the search algorithm 22 different feasible system architectures were generated, all of which satisfy the specified constraints. The 22 different system architectures are defined in Table A.5 in Appendix I. If no constraints are added the number of system solutions quickly grows to 288, making the architectural exploration process unmanageable due to effort required to:

1. Parameterise each system architecture, such that system designs can be scaled up or down from its baseline design in order to satisfy metric constraints. The parameterisation of designs may vary from one system architecture to the next, requiring additional effort to parameterise each system architecture.

2. Define analytical or semi-empirical methods used to calculate the performance metrics for each system architecture. In the conceptual analysis of aircraft systems

[14] Python constraint solver package: http://labix.org/python-constraint

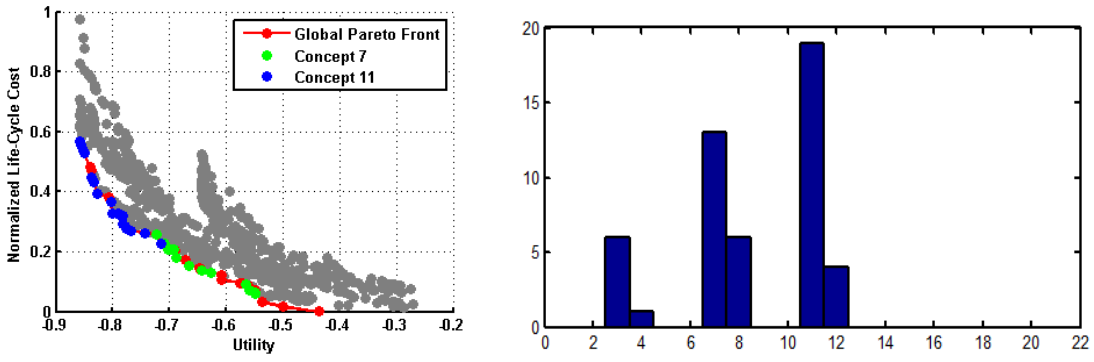
different analytical and semi-empirical methods are required to evaluate the aerodynamic and stability characteristics of different aircraft configurations.

3. Optimise each system architecture individually. The optimisation algorithm varies the values of design variables $\mathbf{x}$, in order to find an optimum solution that satisfies all the metric constraints. This optimisation loop is applied to each system architecture in order to find Pareto optimal solutions for each system architecture.

8.3 Representation

The data generated from the simulation of each system architecture can now be represented to the decision maker in manor that identifies Pareto optimal solutions and high-impact decision variables. This is achieved by presenting the data in two separate formats, (1) the fuzzy Pareto front of design solutions, and (2) the PageRank centrality measure of the architectural decision network, which identifies high-impact decision variables based on their impact on the shift of the utopia point due to changes in design alternatives within the domain of that decision variable.

The Pareto front of each individual system architecture is calculated and is represented on a utility versus cost plot. Here, dominated solutions are neglected as they are deemed to be sub-optimal and only non-dominated solutions on the Pareto front are taken forward for further analysis. The non-dominated solutions for all 22 system architectures are calculated and plotted on a utility-cost scale, as shown in Figure 8.5. The figure indicates overlaps between Pareto fronts of different concepts, which suggests that the performance of some system architectures are better in certain regions of the objective space than other concepts, but diminish in performance in other regions of the objective space. This overlap indicates that the global Pareto front will consist of a number of different system architectures and not just defined by one architecture.

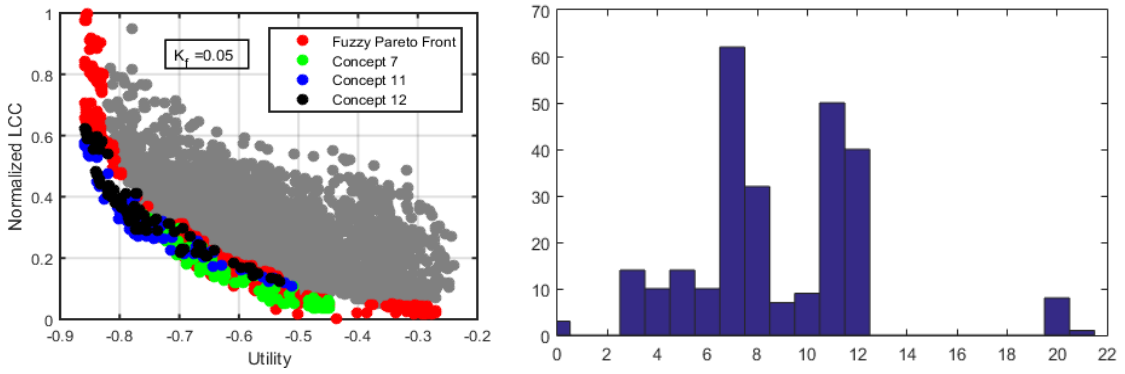


(a) (b)

Figure 8.5: Pareto optimal solutions of all 22 system architectures (a) Pareto front, (b) occurrence of each concept on the Pareto front.

The global Pareto front is determined by concatenating the Pareto fronts of individual system architectures. But, rather than eliminating all dominated solutions some of the solutions close

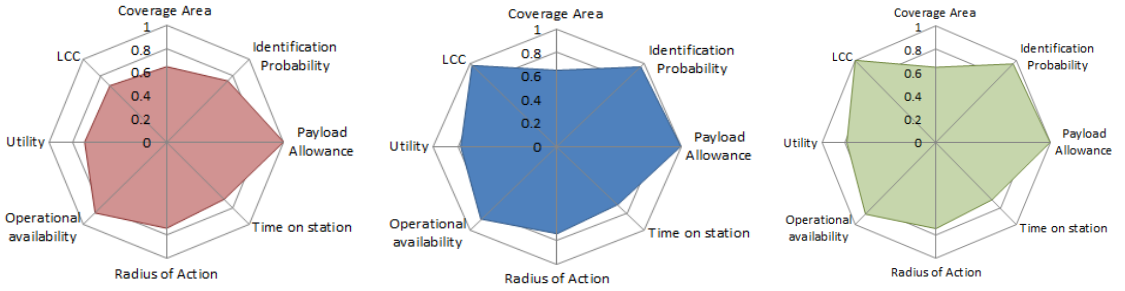
to the global Pareto front are retained, as defined by the fuzzy Pareto front. A fuzzy Pareto front represent of the system architectures is provided in Figure 8.6.



(a) (b)

Figure 8.6: Fuzzy Pareto front: (a) fuzzy Pareto front, (b) occurrence of each concept on the fuzzy Pareto front.

In comparing the number of occurrences of different system architectures on the fuzzy Pareto front it can be noted that system architectures 7, 11, and 12 dominate the fuzzy Pareto front. The output of the three concepts can be compared by capturing the system attributes used to measure the utility value at a given design point. This comparison is represented by a radar chart in Figure 8.7. In comparing the architectural layout of the three concepts, three architectural trades are identified: (1) antenna type, (2) optical sensor resolution, and (3) avionics architecture of the actuation servo mechanism.



(a) (b) (c)

Figure 8.7: Radar chart of system architectures: (a) concept 7, (b) concept 11, and (c) concept 12.

In comparing the architectural layout of the three concepts, three architectural trades are identified: (1) antenna type, (2) optical sensor resolution, and (3) avionics architecture of the actuation servo mechanism. The bladed antenna in architectures 7 and 11 is replaced with a conventional Omni-direction dipole antenna in concept 12. The impact of antenna integration on the UAS platform can be established by considering its aerodynamic impact. The aerodynamic drag can be predicted by assuming that the antenna is a circular cylinder of diameter d and length l. It is also assumed that the operating Reynolds number of the flow

across the antenna will be less than 3×10^5, which gives an approximate drag coefficient of 1.2. Using this information the antenna drag can be predicted as:

$$D_{Antenna} = 1.2 \cdot (1/2 \cdot \rho \cdot V^2) \cdot l \cdot d_{ant}, \tag{8.12}$$

In order to minimise the drag of the antenna, a bladed antenna is considered. It is assumed that the antenna is contained within a symmetric airfoil shape fairing, with a given thickness-to-chord ratio. Based on this, the drag for a bladed antenna can be predicted as:

$$D_{Antenna} = C_D(t/c, Re) \cdot (1/2 \cdot \rho \cdot V^2) \cdot l \cdot d_{ant}, \tag{8.13}$$

The drag coefficient of the airfoil is a function of the thickness-to-chord ratio and the flow Reynolds number. However, the impact of the aerodynamic faring on total aircraft drag, and thus the fuel mass, is minimal. This can be noted by comparing the LCC values for concepts 11 and 12.

The optical sensor resolution of the camera in concept 7 is lower in comparison to concepts 11 and 12, resulting in the identification probability of concept 7 to be lower compared to concepts 11 and 12. The variation in the number of detector elements in the focal plane array can have a direct impact on the ground sampling distance of the optical sensor system. For a given slant range and look angle, the image resolution can be improved by increasing the number of detector elements. However, the increase in the number of detector elements also results in a higher data rate that is required to be transmitted back to the ground control station. The increase in data rate transmitted back to the ground control station results in a higher transmission power that is required to maintain the link margin, resulting in a higher power off-take from the engine to meet the power demands. The increase in power off-take directly impacts the performance of the engine and thus the performance of the UAS, leading to higher fuel burn to complete the mission profile. This increase in fuel mass results in a heavier aircraft, which in-turn results in an increase in operating cost and unit cost for concepts 11 and 12.

The avionics architecture for the actuation servo mechanism is changed from a high-end single string simplex configuration in system architectures 11 and 12, to a low-end servo actuator in a duplex configuration with two sets of control surfaces in architecture 7. This architectural trade focuses on reliability versus the increase in mass, power, and cost. For the high-end simplex actuation mechanism the Mean Time Between Failure $(MTBF)$ is assumed to be 700 hours, in comparison to the low cost actuation mechanism, which is assumed to have a $MTBF$ of 300 hours. But the duplex system offers additional redundancy, which increases its reliability. Though the actuation servo architecture does not significantly impact the utility value, it does impact the LCC of the system. The selection of the high-end configuration leads to a higher purchase cost, resulting in a higher unit cost for concepts 11 and 12.

Based on these architectural trades it is recommended that system architectures 7, 11, and 12 be taken forward into the next phase of the design process, with concept 12 requiring further exploration to identify the potential benefits offered by this concept.

8.3.1 High-Impact Decision Variables

Given the availability of the Pareto optimal data set and the adjacency matrix A_{ij} from the DSM, centrality measure of each decision variable can be calculated. The process starts by identifying the shift in the utopia point of each individual Pareto front, of each system architecture, from the global utopia point. This is done by measuring the Euclidian distance between the two points. Before measuring Euclidian distance, it is important to first normalise the data such that it ranges from 0 to 1. Not normalising the data can result in scaling issues when measuring the Euclidian distance. The results of the utopia point shift are presented in Table 8.5.

The sensitivity to the shift in the utopia point of each decision variable can be calculated by adopting a modified version of the main effects calculation, as specified in section 6.2.5.2. Table 8.6 represents decision variable sensitivity due to changes in design alternatives within its domain. The combined effect of connectivity and sensitivity can now be measured by using PageRank centrality. These results are presented in Figure 8.8, where the size of the nodes in the network represent the relative importance of the decision variable i.e. the bigger the size the higher its impact factor. The choice of weights for the terms α and β in the PageRank centrality equation can impact the ordering of high-impact decision variables. In the UAS case study an equal weighting is applied to the network and non-network factors, wher $\alpha = 0.5$ and $\beta = 0.5$. However, further investigations are required to identify the impact of the weighting factors on the centrality results.

Table 8.5: Utopia point shift of each system architecture from the global utopia point.

System architectures	Euclidian Distance
Architecture 0	0.153
Architecture 1	0.348
Architecture 2	0.348
Architecture 3	0.128
Architecture 4	0.139
Architecture 5	0.134
Architecture 6	0.137
Architecture 7	0.133
Architecture 8	0.136
Architecture 9	0.136
Architecture 10	0.136
Architecture 11	0.105
Architecture 12	0.110
Architecture 13	0.357
Architecture 14	0.359
Architecture 15	0.353
Architecture 16	0.354
Architecture 17	0.162
Architecture 18	0.168
Architecture 19	0.155
Architecture 20	0.349
Architecture 21	0.350

Table 8.6: Decision variable sensitivity

Decision variable	Decision variable Sensitivity (DVS)
Node 5.2	0.0439
Node 7.1.2	0.0036
Node 7.2.2	0.0036
Node 8.1	0.032
Node 9.1	0.0984
Node 10.1	0.0439
Node 10.2	0.0151

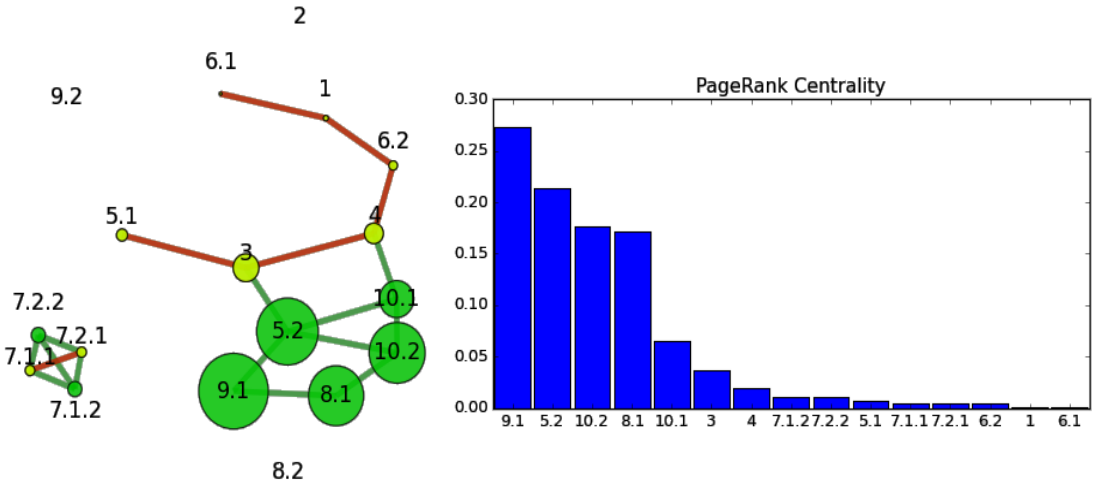


Figure 8.8: PageRank Centrality measure of decision variables.

The results indicate decision variable 9.1 to have the highest centrality measure, followed by decision variable 5.2 and 10.2. Decision variable 9.1 relates to the payload sensor system used to provide surveillance over the target area. The choice of a design alternatives relating to decision variable 9.1 has a significant impact on system properties, and also influences the compatibility of solutions of other decision variables. Thus, to improve upon the Pareto optimal solution set, it is suggested that architectural exploration of design alternatives be focused towards the payload sensor system. However, the choice of the payload sensor system can also directly impact the electrical power demand required to transmit data back to the ground control station. This interaction has been captured by decision variable 5.2, which relates to the delivery of electrical power to the payload and the communications antenna system. Thus, the exploration strategy should also be focused towards alternate means of delivering electrical power to the payload, or better management of data generated by the payload, such that only critical data is transmitted back (i.e. on-board data processing ng and storage), and the use of a high gain antenna to transmit data back to the ground control station. The representation of the avionics architecture by decision variable 10.2 mainly impacts the operational availability and maintenance cost of the system. Based on the results from PageRank centrality it would be recommended that the exploration of the architectural solution space be focused towards different means of improving system reliability. This may include a combination of both high-end (high MTBM value) and low-end (low MTBM value) system components, or a triplex avionics architecture to improve reliability with three levels of redundancy.

Chapter 9: Conclusion and Future Work

The decision-making process in engineering design is considered to be challenging since decisions are generally non-routine, highly interconnected, and are significantly consequential on system properties. In identifying the shortfalls and gaps in the methods available in literature to aid the decision-making process, this thesis has developed a coherent framework that supports the system architecting process. The framework brings together multi-objective optimisation methodologies, in combination with centrality measures from graph theory, and a modified version of the main effects analysis from the Design of Experiments (DoE) literature to identify high-impact decision variables.

The framework is split into three phases, *Construction*, *Search*, and *Representation*. The *Construction* phase facilitates the generation of system architectures and captures the relationships governing their choice by capturing the knowledge of the decision maker. One of the central aspects of this framework is in representing the architectural decision problem as a constraint optimisation problem. The problem is further sub-divided into a constraint satisfaction problem (CSP) to handle discrete logical constraints defining the feasibility of domain values of different decision variables. However, care should be taken when defining the constraint functions, as over constraining or under constraining the architectural solution space may result in too few or too many system architectures for the framework to be of any use in aiding the decision-making process. The framework also presents novel methods in visualising data generated from the analysis of system architectures. The representation of data is categorised into two parts, one representing the Pareto-optimal solutions on a cost-utility scale, and the other representing high-impact decision variables in the constraint network. High-impact decision variables are calculated by adopting a modified version of main effects analysis, where the sensitivity to the shift in the utopia point of the Pareto front from the global utopia point due to changes in design alternatives within its domain is calculated. The decision variable sensitivity (DVS) data is then inputted into PageRank centrality to classify high-impact decision variables.

9.1 Research Questions

The objective of this research was to develop a decision support framework, specifically focused towards aiding the system architecting process by identifying the need to move from single objective decision-making towards the integration of multi-criteria decision-making

methods in aerospace systems design. This was achieved by developing methods to identify Pareto optimal system solutions and high-impact decision variables. In Chapter 1: a number of research questions and hypotheses were introduced. The answers to the research questions and the satisfaction of these hypotheses are addressed as follows:

Question 1. *How to capture and represent the knowledge of the decision maker?*

This thesis captured the knowledge of the decision maker using various representational methods. The enumeration of the architectural design space was captured using functional means analysis (FMA), which is a modification of the original morphological matrix. The interaction of constraints, specified by the decision maker, between different decision variables is captured and represented using the Design Structure Matrix (DSM). This information is then translated into a network representation via the adjacency matrix. However, the DSM does not state the explicit nature of the constraint between a set of decision variables. This is addressed by defining logical constraint statements that specify the exact nature of the constraint relationship.

Question 2. *How to encode the captured knowledge of the decision maker to represent decision variables with a set of design alternatives, a set of constraints between decision variables, and a means of removing infeasible system architectures?*

The answer to this question follows on from the previous question, in terms of capturing knowledge. The logical constraint statements described by the decision maker can be encoded into a programming environment by defining the system architecting process as a constraint satisfaction problem (CSP). CSP's offer a framework to convert a constraint problem, such as the system architecting problem, by defining a tuple of three elements, a set of variables (I.e. decision variables) $\{x_1, \ldots\ldots, x_n\}$, a set of domains- one for each variable (i.e. a set of design alternatives within each decision variable) $\{D_1, \ldots\ldots, D_n\}$, and a set of constraints C_i. This constraint problem is solved using a simple backtracking algorithm, with general purpose heuristics (see Figure 6.7).

Question 3. *How to capture the needs of system stakeholders, and translate them into a set of objective functions to allows for system optimisation?*

The value of a system is captured, prior to optimisation, using multi-attribute utility theory (MAUT). The validity of several multi-attribute decision-making (MADM) methods and value centric decision-making (VCDM) methods were described in Chapter 2: . The aim of the value model is to provide an accurate representation of stakeholder objectives in a non-commercial operating environment i.e. the operational use of the system does not result in the direct generation of revenue to the end-user. This resulted in ruling out all value models based on monetary value, such as net present value (NPV) and cost-benefit analysis (CBA). Out of the remaining, MAUT was chosen as it least violated the assumptions defined in Table A.1, and also for its ability to capture uncertainty in the decision-making process.

Question 4. *How to represent the captured information/data to the decision maker, such that optimal system solutions, and high-impact decision variables can be readily identified?*

The representation of data to the decision maker is visualised in two forms. One, representing the global Pareto (or s-Pareto front) front of system architectures on a cost-utility scale. The other representing the centrality measures of decision variables within the network, which is visualised in a graph form. Decision variables that have high centrality measure advocate for further exploration or resource allocation within its domain.

9.2 Research Contributions

The most significant contributions made by this thesis to the field of systems architecting and multi-criteria decision-making are defined as follows:

1. **Representation of the system architecting process:** The decision support framework represents the architectural decision-making process as a constraint Optimisation Problem (COP), which can be defined as $(\mathrm{CSP}, \mathbf{g})$ or $(\mathbf{X}, \mathbf{D}_{\mathbf{X}}, C_{\mathbf{X}}, \mathbf{g})$, where $\mathbf{g}$ is a set of objective functions. The result of solving this problem is a Pareto front of all the non-dominated solutions.

2. **Translating the system architecting problem into a constraint satisfaction problem:** This allows the decision support framework to address assignment constraints that relate to different decision variables. The CSP representation of the system architecting process allows the logical constraints imposed on the architectural space to be encoded into a programming environment, such that infeasible system architectures can be removed.

3. **Identifying High-Impact decision variables:** The focus of much of the research in literature has been on identifying Pareto optimal solutions from a set of system architectures. But less attention is paid towards identifying the next step in the exploration of system architectures. The decision support framework developed in this thesis proposes an exploration strategy, where the exploration of system architectures should only be carried out in domains of decision variables that have a high-impact on system properties. This is achieved by adopting centrality measures from the network analysis literature, and a modified version of the main effects analysis from the DoE literature. This thesis adopts PageRank centrality to identify high-impact decision variables that are strongly connected to other decision variables, and also have a significant impact on the shift of the utopia point.

9.3 Lessons from Implementation

In applying this framework to the UAS design case study several drawbacks were identified that need to be addressed in future work. In applying the *Construction* phase it was found that identifying the compatibility relationships between different decision variables using the

DSM was not intuitive, and required several iterations before the relationships were fully defined. This process can become time-consuming in the presence of a large number of decision variables. Thus, it can be argued that the value added in identifying the connectivity between all the decision variables is minimal, in relation to the effort required in identifying them in the first place. Therefore, the architectural network can be represented by considering only those decision variables that are involved in the architectural trade studies. The connectivity between these decision variables can be established from the logical constraint statements specified by the decision maker. This would result in ignoring the decision variables in which the decision, in regards to the choice of the design alternative, is fixed.

Other problems relate to the construction of the MAU model. The construction of the MAU model is subjective and requires a significant body of knowledge of the system operating environment. Thus, several iterations may again be required before the fundamental objectives and measurable system attributes defining the MAU model are agreed upon. However, the MAU model offers the capability to capture the decision makers risk attitude towards uncertainty in the outcomes of the system attributes, and thus capturing the uncertainty of the decision-making process. This makes the application of the MAU model very worthwhile. However, the framework does not explicitly address uncertainties in the architectural exploration process. Future work needs to account for how uncertainties in the development of a system architecture can impact the Pareto front, and thus the architectural exploration strategy.

9.4 Problems in Validation

The development of decision support tools have been vast in recent years. The most prominent of these tools include QFD, Pugh's selection method, scoring and weighting methods, AHP, MAUT, Taguchi Loss Function, and Suh's Axiomatic design. The ultimate goal of these tools, including the framework presented in this thesis, is to lead the decision maker to select the 'best' design. However, each method has a unique way of defining 'best' and the means by which to get there. The validation of these proposed frameworks is limited, as there is no agreed criteria available in literature by which to judge the proposed decision support tools to yield the correct decision. This issue is addressed in detail by Olewnik and Lewis [143], and Allen et al [144]. The problem in validation stems from the fact that quantitative analysis alone cannot validate prescriptive models [143], which tries to predict the performance or courses of action to be taken based on available information, risks, objectives, and axioms of rational behaviour [143]. The validation of such models include qualitative analysis since there are subjective elements involved, and the solutions ascertained from these methodologies do not yield a single right or wrong answer. Thus, decision support frameworks relate to relativist validation, which argues that knowledge cannot be validated in an objective way, and that subjective preferences and rules of behaviour must be considered a part of the validation process [144].

This thesis follows the validation process defined by Olewnik and Lewis [143], which defines of three elements, *logical*, *meaningful and reliable information*, and *not bias designer* .[143].

9.4.1 Logical

Testing the logic of a framework can be accomplished by using test cases for which the results are intuitive, and checking if the model results agree with intuition. This is illustrated by applying the framework to the UAS case study presented in Chapter 7. The case study presented is an illustrative example based on a real engineering design study undertaken by the University of Southampton in developing a maritime search and rescue UAS [126]. The results of the framework indicate that the exploration of the architectural solution space to be focused towards the payload sensor system, delivery of electrical power, and the avionics architecture. These results have been verified subjectively as being logical by the 2Seas design team at the University of Southampton.

However the following question still remains, *what kinds of test cases provides valid evidence of effectiveness for an intended use?*The choice of the test case should be representative of a real engineering decision problem of a large complex system. This requires modelling of all the complex interactions between multiple system elements that are present in a real system. Ideally a benchmark test case with a predefined solution that has been subject to several previously developed design frameworks would be a preferred choice for validation. But, the existence of such a test case is not available in decision-based design literature. Previous studies have resorted to test cases that are specific to the domain of interest of the author, or the funding body of research. This thesis is in no exception to this, the UAS case study presented in this thesis is a simplification of the real 2Seas UAS design problem, with simplifications made by the author to allow for the case study to be within scope of this thesis.

9.4.2 Meaningful and Reliable Information

The information that is incorporated into the model should be meaningful, in the sense that it provides insight into interdependencies among design variables, and reliable in the sense that information comes from appropriate sources [143]. The information incorporated into the framework developed in this thesis has been shown to be meaningful, as it provides insight into the interdependencies between multiple decision variables. This is achieved through the representation of the system architecting process as a constraint optimisation problem (COP). The information relating to the interdependencies is further expressed via logical constraint statements, which in-turn allows the decision maker to more openly interrogate the validity of the interdependencies.

Majority of the information incorporated into the UAS case study is reliable, in the sense that information used to enumerate the models are derived from widely recognised publications within the aerospace design domain (i.e. majority of the information was ascertained from Raymer Gundlach [17], [122]). However, information relating to the construction of the

decision problem i.e. the FMA table, and the DSM, are mainly based on the authors own knowledge of the design problem, and is only illustrative of the real 2Seas design problem. This is not to say that the information used is not reliable, however it does not paint a full picture of all the interdependencies and component level decisions that were made in the real design problem. This makes the construction of the decision problem hard to validate, and is a problem faced in many academic based research on decision support frameworks. Test cases that are applied in academic research are generally a simplification of a real design problem to illustrate the applicability of the developed framework, and to ensure that the test cases are within scope of the research project – the following references illustrate this point [63], [73], [104], [108], [130].

9.4.3 Not Bias Designer

Forcing a preference structure on the designer will influence the processes used in decision-making, which in-turn will influence the outcome. To avoid bias, the application of the method is required to be conducted in some blinded fashioned manner, such that no external factors, nor the framework itself, influence the outcome of the decision. However, in reality this is hard to achieve in engineering decision-making. Decision are not taken by an individual, but by a group of individuals in the design team, which inevitably will lead to other members of the team influencing each other's decision. Decisions are also influenced by previous experience of designing similar systems. In addition to influences from external factors, it is challenging for any decision support framework not to impose any rules and restrictions on the actions taken by the designer during the decision-making process. For example, the framework presented in this thesis imposes the use of MAUT to define the objective function. This however may not be valid in other circumstances where the preferred objective function may be NPV. Another example of this is axiomatic design, where the designer is pushed to take decisions in a manner that is in accordance with the two axioms of design, thus influencing the outcome of the process.

The *not bias* criteria is by no means met by the developed framework, or the applied case study in this thesis. The means to satisfy this criteria is not clearly defined in any literature known to the author. The validation of the *no bias* criteria remains at the time of writing an open ended research question in the field of validating decision support tools [145].

9.5 Recommendations for Future Work

Although several aspects of the decision-making process have been addressed in this thesis, research within this areas is in its infancy and several aspects of the system architecting process have not been addressed in this thesis. These are sub-categorised into:

1. Improvements in knowledge capture and reuse.
2. Incorporation of a domain specific meta-language.
3. Uncertainty in architectural decision-making.

4. Framework Validation.

9.5.1 Knowledge Capture and Reuse

The construction of the decision problem can be further improved by developing a user interface for the decision support framework. By developing a user interface the speed and effectiveness of encoding the decision support framework can be improved. At present, the entry of data into the decision support framework is based on spreadsheets and input files in MATLAB and Python. The development of a user interface should allow data to be inputted and stored in a relational database. This will allow the user to query the decision variables and check for consistency. In addition, outputs of the simulations models can be stored in a database format to allow the decision maker to query subsets of data, and allow for a more flexible representation of data.

A means of developing such a user interfaces can be achieved through the development of an ontology. Ontologies define a common vocabulary to share information in a domain, such that domain knowledge can be reused, assumptions are explicit, and knowledge can be shared, analysed, and interpreted using a common syntax. Providing this capability in turn will allow the decision support framework to widely expand the decision network into a large number of nodes and connections, and can be shared easily between several users. This should allow design knowledge from various design teams to be captured and queried. For an introduction to ontology development, and its applicability in sharing and reusing knowledge, the reader is referred to Noy et al. [95].

9.5.2 Meta-Language

The use of a common meta-language to construct the decision network, and identify interrelationships between decision variables will aid the efficiency and transparency of constructing the decision support framework. Two such meta-languages, AoS and OPN, were discussed in section 4.6. At present the enumeration of network components, such as connectivity, domain of alternatives, and logical constraints are achieved through a series of systems engineering methods. The inclusion of all such methods into one common methodology will offer a great benefit in the ease of defining the network structure. It is recommended that future work be focused towards integrating a formal meta-language, such as AOS or OPN, into the developed decision support framework. Having a common meta-language will allow the constructs of the network to be understood, and its validity questioned by multiple users.

9.5.3 Uncertainty

At present the developed framework does not explicitly address uncertainty in architectural decision-making. Though the backbone to address this issue has already been incorporated in the framework via the use of MAUT. From a systems architecting point of view Crawley et

al. [3] defines uncertainty from the view of system flexibility. Flexibility defines the capability of the system to interface with future system elements that are not yet present in the original version. The need for flexibility requires the architecting process to model future usage of the system, uncertainties in the operating environment, regulations, and future end user needs [3]. Previous studies such as TDN and epoch-era analysis [64], [104] have adapted the Markovian Decision Process (MDP) to develop frameworks that account for operational uncertainties and technology development uncertainties. These uncertainties are addressed by making changes to a system architecture in the future, such that the system remains adaptable. However, frameworks such as TDN and epoch-era analysis lack the means of generating an initial set of candidate system architectures and candidate design alternatives that should be taken forward for further consideration. The decision support framework developed in this thesis could be used as a front end for TDN and epoch-era analysis in order to identify a set of viable system architectures.

9.5.4 Validation

The problems associated with validating a decision support tool has already been addressed in Section 9.4. It is recommended that future work focus on defining a validation framework prior to, or during the course of developing the decision support framework. Special attention should also be paid to the test cases used to validate the framework, as this plays a critical role in increasing confidence of the decision maker in applying the framework to address real design problems.

Appendices

Appendix A Value Model Assumptions

Table A.1: Value model assumptions to guide the selection of an appropriate value model [39].

	Net Present Value (NPV)	Cost-Benefit Analysis (CBA)	Multi-Attribute Utility Theory (MAUT)
Definition of value	Value is discounted cash flow (monetised).	Value is discounted cash flow of net benefits (monetised).	Value is an aggregation of a set of benefits relative to their net cost (non-monetised).
Source of Value	Value is not derived from any source other than revenue.	Value is derived from multiple benefits and costs.	Value is derived from multiple benefits and costs.
Market Prediction	Cash flow and discount rate.	Cash flow and discount rate.	
	Extensive and quantitative predications can be made about the systems future financial markets, revenue and pricing structure, demand functions, etc.	Extensive and quantitative predications can be made about the systems future financial markets, revenue and pricing structure, demand functions, etc. and also about the derivation of the monetary value of system benefits.	
	Mutual additive (preferential) independence - stakeholder(s) absolute preference for a given attribute is independent of the respective values of all other system attributes.	Mutual additive (preferential) independence - stakeholder(s) absolute preference for a given attribute is independent of the respective values of all other system attributes.	Mutual utility independence - stakeholder(s) relative preference between two values for a given attribute is independent of the respective values of all other attributes; absolute preference for one attribute is dependent on the respective values of all other attributes.
	Stationary assumption - stakeholder preferences do not change over time.	Stationary assumption - stakeholder preferences do not change over time	Stationary assumption - stakeholder preferences do not change over time.
	Stakeholders make decisions under certainty - they have perfect foresight into all present and future events pertaining to the value of system attributes.	Stakeholders make decisions under certainty - they have perfect foresight into all present and future events pertaining to the value of system attributes.	Stakeholders make decisions under uncertainty - they do not have perfect foresight into all present and future events pertaining to the value of system attributes.
		Multiple stakeholder preferences cannot be aggregated - nonexistence of a social welfare function.	Multiple stakeholder preferences cannot be aggregated - nonexistence of a social welfare function.
	Cash flow and/or discount rate are discrete in time and also potentially held as constants.	Monetised benefits, costs, and/or discount rate are discrete in time and also potentially held as constants.	
		Monetisation of benefits.	
	Truncation of information regarding distribution of costs (monetised).	Truncation of distribution of costs and benefits (monetised).	Truncation of distribution benefits into a single metric.
	Value is cardinal metric.	Value is cardinal metric.	Ordered comparison of benefit (non-ratio cardinal) and cost is assumed a proxy value.

Appendix B Power Iteration Method

The power iteration method is a simple procedure for computing approximate values of the eigenvalue of an $n \times n$ matrix $\mathbf{A}$ that is largest in absolute value i.e. the dominant eigenvalue. A dominant eigenvalue can be defined as follows:

Let $\lambda_1, \lambda_2, \ldots, \lambda_n$ be the eigenvalues of an $n \times n$ matrix $\mathbf{A}$. λ_1 is called the dominant eigenvalue of A if

$$|\lambda_1| > |\lambda_i|, \quad i = 2, \ldots, n. \tag{A.1}$$

The eigenvector corresponding to λ_1 are called the dominant eigenvector of A.

Now based on the assumption that the matrix $\mathbf{A}$ has a dominant eigenvalue, the power method works by initially making an approximation of the eigenvector $\mathbf{x_0} (\neq \mathbf{0})$ with n components and iteratively solving for the dominant eigenvector until the iteration converges. This procedure can be defined as follows:

$$\begin{aligned} \mathbf{x_1} &= \mathbf{A x_0} \\ \mathbf{x_2} &= \mathbf{A x_1} = \mathbf{A}(\mathbf{A x_0}) = \mathbf{A^2 x_0} \\ \mathbf{x_3} &= \mathbf{A x_2} = \mathbf{A}(\mathbf{A^2 x_0}) = \mathbf{A^3 x_0} \\ &\vdots \\ \mathbf{x}_k &= \mathbf{A x}_{k-1} = \mathbf{A}(\mathbf{A}^{k-1}\mathbf{x_0} = \mathbf{A}^k \mathbf{x_0} \end{aligned} \tag{A.2}$$

For large powers of k and by scaling the vector at each step of the sequence of the iteration (i.e. divide the vector $\mathbf{x}$ its absolute largest component, ensuring that the components of the vector have absolute values less than or equal to 1), we can obtain a good approximation of the dominant eigenvector of $\mathbf{A}$.

Given the eigenvector, the corresponding eigenvalue can now be calculated by applying the Rayleigh quotient:

$$\lambda_q = \frac{\mathbf{x^T A x}}{\mathbf{x^T x}} \tag{A.3}$$

The error in the predicted dominant eigenvalue can be computed by setting $\lambda_q = \lambda - \epsilon$, where ϵ is the error of λ_q. This given to be:

$$|\epsilon| \leq \delta = \sqrt{\frac{(\mathbf{Ax})^\mathbf{T}\mathbf{Ax}}{\mathbf{x^T x}} - \lambda_q^2} \tag{A.4}$$

This method can be modified to approximate other eigenvalues through the use of the deflation procedure. For a more in-depth review of the power iteration method and other iterative schemes of calculating eigenvalues and their corresponding eigenvectors, the reader is referred to Kreyszig [146].

Appendix C A Brief Overview of Brent's Method

Brent's method is considered to be a hybrid method, which makes use of the bisection method in combination with inverse quadratic interpolation to find roots of univariate functions. The application of the inverse quadratic interpolation is similar to the secant method, where a linear interpolation is used to approximate the function. Instead of linear interpolation, here we apply a higher order interpolation using estimates for the zero and find the zero of the interpolating polynomial to get the next approximation. However, rather than solving for the roots of the interpolating polynomial, an inverse quadratic interpolant is applied i.e. we can interpolate x as a function of $y = f(x)$, the inverse of which can be defined as $x = g(y)$. The new update estimate of the root can be found by setting $g(0)$.

Brent's method uses the Lagrange interpolating polynomial of degree 2 to generate an approximate of the function f. Given three points x_1, x_2, and x_3 and their function values $f(x_1)$, $f(x_2)$, and $f(x_3)$, an approximation of f can be computed using the following interpolation formula:

$$x = \frac{[y - f(x_1)][y - f(x_2)]x_3}{[f(x_3) - f(x_1)][(f(x_3) - f(x_2)]} + \frac{[y - f(x_2)][y - f(x_3)]x_1}{[f(x_1) - f(x_2)][f(x_1) - f(x_3)]} + \frac{[y - f(x_3)][y - f(x_1)]x_2}{[f(x_2) - f(x_3)][f(x_2) - f(x_1)]} \tag{A.5}$$

Setting the value of y to zero will give the update estimate of the root:

$$x = x_2 + \frac{P}{Q},$$

where $$P = S[T(R - T)(x_3 - x_2) - (1 - R)(x_2 - x_1)$$

$$Q = (T - 1)(R - 1)(S - 1)$$

with $$R \equiv \frac{f(x_2)}{f(x_3)}$$

$$S \equiv \frac{f(x_2)}{f(x_1)}$$

$$T \equiv \frac{f(x_1)}{f(x_3)} \tag{A.6}$$

This process is repeated with x_2 replaced by the new approximation, x_1 is replaced with the old x_2, and x_3 is replaced with the old x_1. Similar to the Newton Raphson method, the inverse quadratic method may run the risk of diverging away from the root. Brent's method counters this risk by maintaining brackets on the root and checks where the next update value is going. If the update goes outside the brackets, then Brent's methods switches to bisection method.

Appendix D Link Budget Table

Table A.2: Link budget Table.

		Value
Transmitter	Component line loss, $L_{T,Line}$	−1.0 dB
	Pointing loss, $L_{T,Point}$	0 dB
	Radome loss, $L_{T,Radome}$	0 to −0.5 dB
	Transmitter Power, P_T	−
Propagation	Free space loss, $20\ \log_{10}(\lambda/4\pi R)$	−
	Atmospheric absorption, $L_{P,Atm}$	−0.01 dB/km
	Precipitation absorption, $L_{P,Precip}$	−0.006 dB/km
Receiver	Peak antenna gain, G_R	20 dBi
	Polarisation loss, $L_{R,Polar}$	0 to −0.5 dB
	Pointing loss, $L_{R,Point}$	0 dB
	Component line losses, $L_{R,Line}$	−1.0 dB
	Spreading implementation loss, $L_{R,spread}$	0 dB
Noise	Thermal noise density, kT	−174.0 dB/Hz
	Noise bandwidth, BW	−
	Noise figure, NF	−5 dB

The values chosen for the losses and gains in the table above are representative of the information ascertained from Gundlach [122] and from the authors own assumptions. Thus, the communication system represented is only illustrative and is not representative of any specific real world system. For more information in regards to the signal losses and system noise, the reader is referred to Gundlach [122]. It can be noted that some of the values in the above table are left blank, as they are dependent variables that change in value based on other variables (e.g. the distance travelled by the RF signal, R). It can also be noted that the values for radome loss, $L_{T,Radome}$, and polarisation loss, $L_{R,Polar}$, are provided over a range as they are depended on the choice of antenna architecture. For example, if the antenna has no aerodynamic fairing covering it, than the polarisation loss will be 0 dB. The polarisation loss, in the case of linear polarisation, is proportional to the square of the cosine of the misalignment angle θ_{Align}, which is given as:

$$L_{R,Ploar} = 20\log_{10}[\cos(\theta_{Align})] \tag{A.7}$$

The atmospheric and precipitation propagation losses are dependent on the frequency of the RF signal and the atmosphere itself. Higher fidelity analysis would require detailed atmospheric models to predict the absorption and attenuation of the signal. For simplicity, it

is assumed that the signal losses are linear in distance, at a given frequency. For the case study presented in this thesis, the transmitting frequency band is assumed to be L-band, with a frequency range of 1-2 GHz.

Appendix E Model Validation

Table A.3: UAS mass analysis comparison.

	2Seas UAS, kg	Calculated data, kg	% Difference
Wing assembly	3.59	4.72	31.6
Empennage	1.82	1.06	-41.5
Tail booms	0.56	0.60	7.8
Under carriage	2.15	3.11	44.5
Inner wing assembly	1.87	2.53	35.3
Avionics	4.48	3.79	-15.4
Propulsion System (x2)	5.35	5.5	3.5
Nacelles	2.6	1.65	-36.5
Payload	5	5.7	14
Fuel	6	8.25	37.5
Empty mass	22.418	23.00	2.6
Dry mass	27.418	28.70	4.7
GTOM	33.418	36.95	10.6

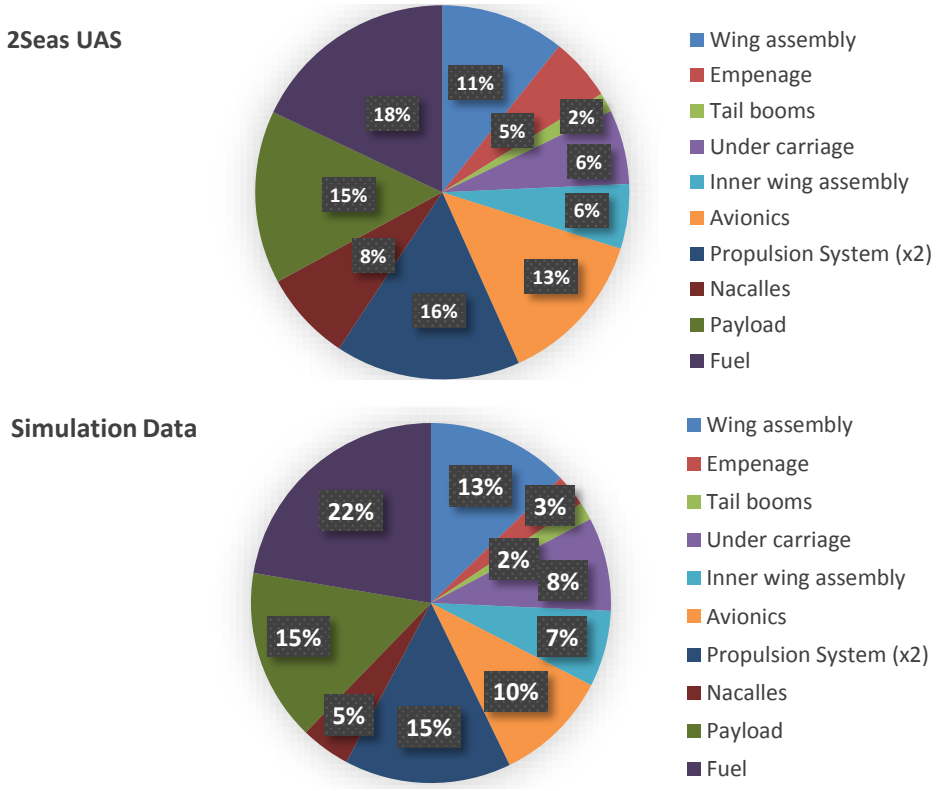


Figure A.1: UAS mass breakdown.

Appendix F Expected Improvement Plots

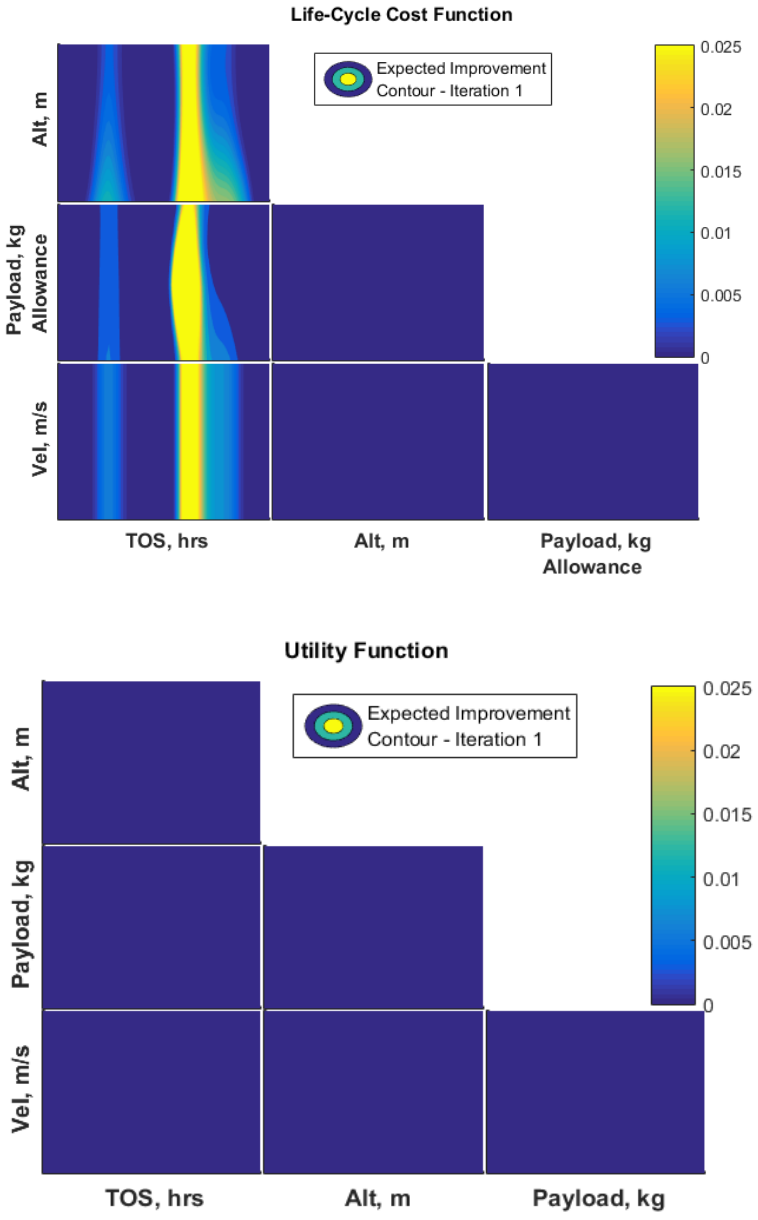


Figure A.2: Expected Improvement contour plots of the LCC and utility function. Each tile shows a contour of $E[I(\mathbf{x})]$ versus two of the four variables, with the remaining two held at baseline values.

Appendix G Functional Decomposition

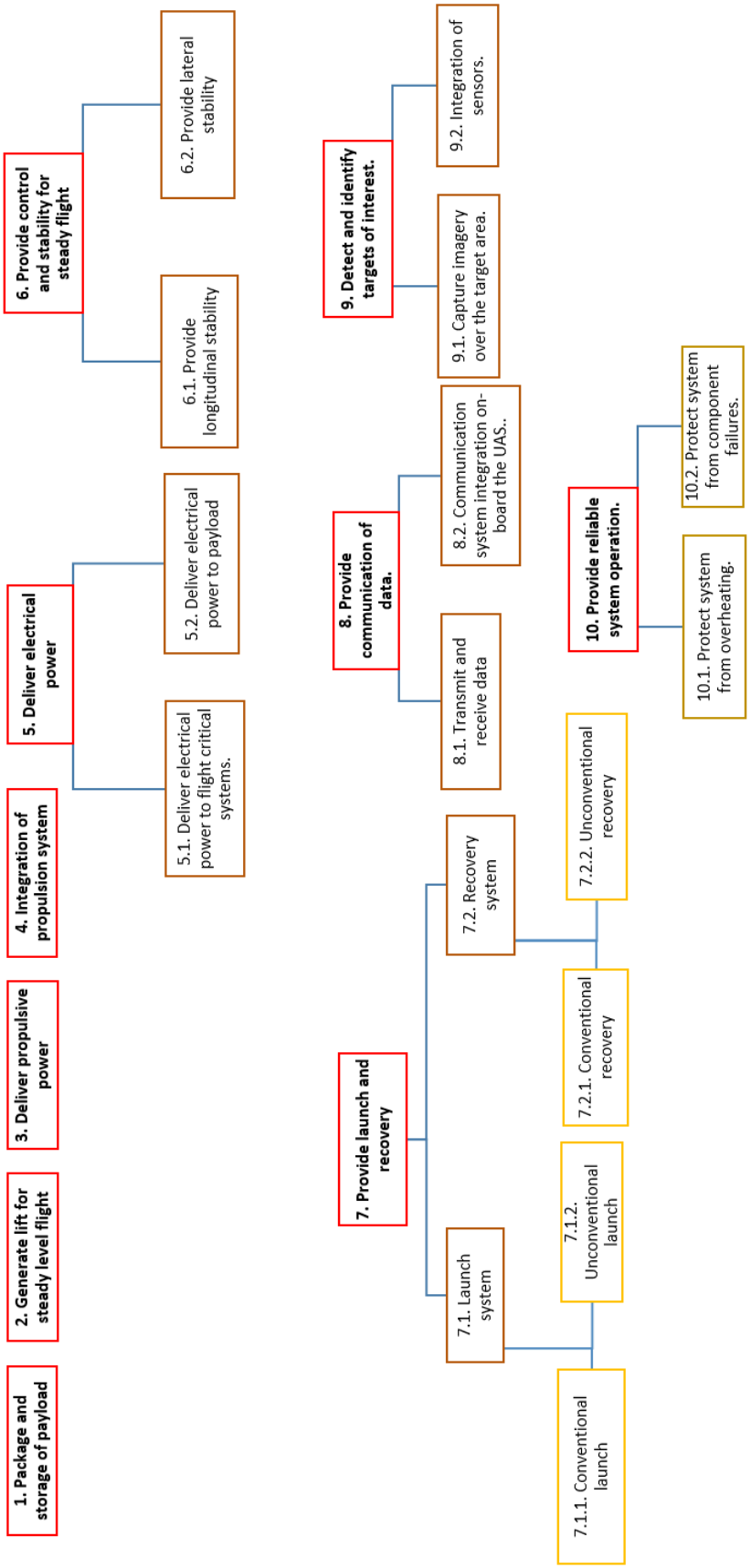


Figure A.3: Functional Decomposition of the UAS.

Appendix H Functional Means Analysis

Table A.4: Functional means analysis tables for the UAS case study.

Function	Means	Alternatives				
1. Package and store payload and other system components.	Fuselage Arrangement	Conventional tube shape	Blended wing body (BWB)	Detachable pod		
2. Generate lift for steady level flight.	Fixed Wing	High wing	Mid Wing	Low wing	Variable sweep geometry.	
3. Deliver propulsive power.	Reciprocating engines	Twin 2 stroke engines	Twin 4 stroke engines			
	Fuel cells	Solid-oxide fuel cells	Regenerative fuel cells			
	Solar power	Multi-crystalline silicon cell	Mono-crystalline gallium arsenide cell	Thin film silicon cell		
4. Integration of propulsion system.	Propeller arrangement	Fuselage pusher	Fuselage tractor	Wing pusher	Wing tractor	
5.1. Deliver electrical power to flight critical systems.	On-board power generation systems.	Engine Generator	Separate battery source	Engine Generator + external propeller driven generator	Engine Generator + Fuel cells	
5.2. Deliver electrical power to payload.	On-board power generation systems.	Engine generator	Separate battery source	Engine Generator + external propeller driven generator	Engine Generator + Fuel cells	
6.1. Provide longitudinal stability.	Horizontal stabilisers.	Conventional	T-tail	V-tail	Canard	Joined-wing
6.2. Provide lateral stability.	Vertical stabilisers.	Winglets	Twin vertical surfaces	V-tail	Joined wing	Single vertical surface.
7.1.1. Conventional launch		Tail-dragger	Quadricycle	Bicycle	Tricycle	

7.1.2. Unconventional launch		Rail launch	Tensioned Line Launch	Ground-vehicle launch	Air tow	*null*
7.2.1 Conventional recovery		Tail-dragger	Quadricycle	Bicycle	Tricycle	
7.2.2.Unconventional recovery		Skid and belly recovery	Net recovery	Cable-assisted recovery	Parachute	*null*
8.1. Transmit and receive data	On-board antenna	Omni-Dipole, vertically polarisation., Gain 2dBi	Omni- circular polarisation, Gain 3dBi	Bladed Antenna, Omni -Dipole, vertical polarisation, Gain 2dBi		
8.2. Communication system integration. On-board the UAS.	Antenna location	Lower fuselage / wing surface	Upper surface, ahead or behind the wing	Winglet/wingtip integrated	Antenna can be separated from the upper fuselage by a faired post.	
9.1. Capture imagery over the target area.	Optical Camera system	Resolution: 1080 by 1080.	Resolution 640 by 480.	Resolution: 1280 by 720.		
9.2. Integration of sensors.	EO/IR ball	EO/IR ball pan tilt	Mounted rigidly on airframe	EO/IR ball pan tilt roll	EO/IR ball roll tilt gimbals	
10.1. Protect system from overheating.	Environmental control systems (ECS).	No ECS systems	Ram air cooling of engine components	Fans		
10.2 Protect system from component failures.	Avionics architecture	Simplex	Duplex with two sets of control surfaces	Triplex		

Appendix I Viable System Architectures

```
import os
os.chdir('C:\python-constraint-1.2')
import constraint as con
# Define architectural functional data in dictionary format
funcData = {'0.0':[1],
            '1.0':[1],
            '2.0':[1],
            '3.0':[1],
            '4.0':[1],
            '5.0':[1,2],
            '6.0':[1],
            '7.0':[1],
            '8.0':[1],
            '9.0':[1,2],
            '10.0':[1],
            '11.0':[1,2],
            '12.0':[1,2,3],
            '13.0':[1],
            '14.0':[1,2,3],
            '15.0':[1],
            '16.0':[1,2],
            '17.0':[1,2]
            }
# Locations of
# Add variables into the CSP problem
functions = funcData.keys()# Extract function keys from dictionary
problem = con.Problem()# Constraint Problem

for i in range(len(functions)):
    problem.addVariable(functions[i],funcData[functions[i]])

# Define constraint functions
def electricalPower(a,b,c):
    if (a==3 and b==2):
        return c !=1
    else:
        return c!=2

def launchandrecovery(a,b):
    if (a==1):
        return b!=2
    elif(a==2):
        return b!=1

def ECS(a,b,c):
    if (a==2 and b==2):
        return c !=1
    else:
        return c!=2

def comms(a,b,c):
    if (a==3 and b==2):
```

```
        return c !=1 and c !=3
    else:
        return c!=2
# ----- Add constraint functions to the constraint problem-----------
# Deliver electrical power constraint
problem.addConstraint(electricalPower,['14.0','17.0','5.0'])
# Launch and Recovery constraint
problem.addConstraint(launchandrecovery,['9.0','11.0'])
# Environmental control system constraint
problem.addConstraint(ECS,['5.0','17.0','16.0'])
# Communication antennta constraint
problem.addConstraint(comms,['14.0','17.0','12.0'])

# Solve CSP
print len(problem.getSolutions())
```

Figure A.4: Python implementation of the CSP algorithm for the UAS case study.

The numbers in the rows indicate the order in which design alternatives appear in the rows of Table A.4 (excluding the greyed out design alternatives).

Table A.5: Architectural solutions generated from the constraint-satisfaction search algorithm.

System Architectures	Function 5.2	Function 7.1.2	Function 7.2.2	Function 8.1	Function: 9.1	Function 10.1	Function 10.2
Architecture 0	1	1	1	1	2	1	2
Architecture 1	1	2	2	3	2	1	2
Architecture 2	1	2	2	1	2	1	2
Architecture 3	1	2	2	1	1	1	2
Architecture 4	1	1	1	3	2	1	2
Architecture 5	1	2	2	3	3	1	1
Architecture 6	1	2	2	1	3	1	1
Architecture 7	1	1	1	3	1	1	1
Architecture 8	1	1	1	1	1	1	1
Architecture 9	2	1	1	2	3	2	2
Architecture 10	2	2	2	2	3	2	2
Architecture 11	1	1	1	3	3	1	1
Architecture 12	1	1	1	1	3	1	1
Architecture 13	1	2	2	3	2	1	1
Architecture 14	1	2	2	1	2	1	1
Architecture 15	1	1	1	3	2	1	1
Architecture 16	1	1	1	1	1	1	2
Architecture 17	1	2	2	3	1	1	1
Architecture 18	1	2	2	1	1	1	1
Architecture 19	1	2	2	3	1	1	2
Architecture 20	1	1	1	3	1	1	2
Architecture 21	1	1	1	1	2	1	1

Appendix J Objective Hierarchy

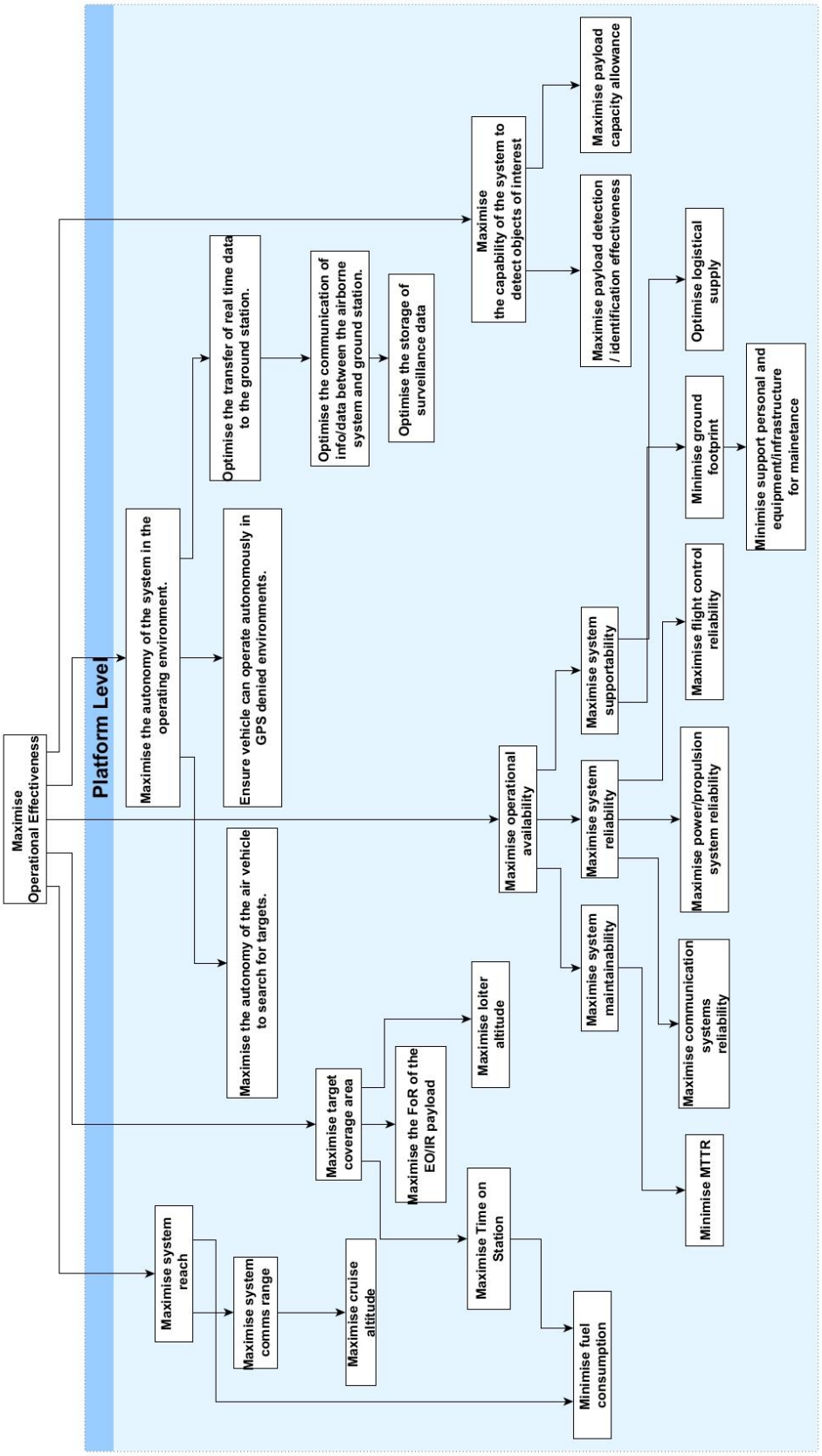


Figure A.5: Means-Ends objective hierarchy.

References

[1] "NASA Systems Engineering Handbook," 2007. [Online]. Available: ntrs.nasa.gov/archive/nasa/casi.ntrs.nasa.gov/20080008301.pdf. [Accessed: 14-Jul-2015].

[2] J. Leonard, "Systems Engineering Fundamentals.," 1999. [Online]. Available: www.dtic.mil/cgi-bin/GetTRDoc?AD=ADA372635. [Accessed: 14-Jul-2015].

[3] E. Crawley, O. De Weck, S. Eppinger, C. Magee, J. Moses, W. Seering, J. Schindall, D. Wallace, and D. Whitney, "The Influence of Architecture in Engineering Systems," Boston, Masschusetts, 2004.

[4] G. Hazelrigg, *Systems Engineering: An Approach to Information-Based Design*. New Jersey: Prentice Hall, Inc., 1996.

[5] M. Maier and E. Rechtin, *The Art of Systems Architecting, Third Edition*. Taylor & Francis, 2009.

[6] G. Pahl, K. Wallace, L. Blessing, W. Beitz, and F. Bauert, *Engineering Design: A Systematic Approach*. Springer London, 2013.

[7] S. Burge, "Function Means Analysis (FMA) alias Morphological Analysis," 2006. [Online]. Available: www.burgehugheswalsh.co.uk/uploaded/.../FMA-Tool-Box-V1.1.pdf. [Accessed: 14-Jul-2015].

[8] D. Raymer, "Enhancing Aircraft Conceptual Design Using Multidisciplinary Optimization," Swedish Royal Institute of Technology (KTH), Stockholm, Sweden, 2002.

[9] W. Simmons, "A Framework for Decision Support in Systems Architecting," Massachustts Institute of Technology, 2008.

[10] L. Page, S. Brin, R. Motwani, and T. Winograd, "The PageRank Citation Ranking: Bringing Order to the Web.," Stanford InfoLab, Nov. 1999.

[11] "The Design Society." [Online]. Available: https://www.designsociety.org/publications/1/books_proceedings. [Accessed: 23-Jul-2015].

[12] H. Jimenez and D. Mavris, "A Framework for Collaborative Design in Engineering Education," in *45th AIAA Aerospace Sciences Meeting and Exhibion*, 2007, pp. 1-9.

[13] A. Keane and P. Nair, *Computational Approaches to Aerospace Design*. Chichester: John Wiley & Sons Ltd., 2005.

[14] N. Keith, "Decision Making Strategies for Probabilistic Aerospace Systems Design," Georgia Institute of Technology, 2006.

[15] P. Sen and J. Yang, *Multiple Criteria Decision Support in Engineering Design*. London: Springer London, 1998.

[16] D. Howe, *Aircraft Conceptual Design Synthesis*. London: Professional Engineering Publishing Limited, 2000.

[17] D. Raymer, *Aircraft Design: A Conceptual Approach*, 4th ed. Reston, Virigina: American Institute of Aeronautics and Astronautics, 2006.

[18] C. Dickerson and D. Mavris, *Architecture and Principles of Systems Engineering*. Boca Raton, Florida: Auerbach Publications, 2009.

[19] Y. Li, "An Intelligent , Knowledge-based Multiple Criteria Decision Making Advisor for Systems Design," Georgia Institute of Technology, 2007.

[20] D. DeLaurentis, "A probabilistic approach to aircraft design emphasizing stability and control uncertainties," Georgia Institute of Technology, 1998.

[21] H. Simon, *The New Science of Management Decision*. New Jersey: Prentice Hall, Inc., 1977.

[22] A. Marquez and C. Blanchar, "A Decision Support System for evaluating operations investments in high-technology business," *Decis. Support Syst.*, vol. 41, no. 2, pp. 472-487, Jan. 2006.

[23] E. Turban, J. Aronson, and T.-P. Liang, *Decision Support Systems and Intelligent Systems*, 7th Edition. New Jersey: Pearson Education inc., 2005.

[24] D. Power and R. Sharda, "Model-driven decision support systems: Concepts and research

directions," *Decis. Support Syst.*, vol. 43, no. 3, pp. 1044–1061, Apr. 2007.

[25] V. Sauter, *Decision Support Systems for Business Intelligence*, 2nd Edition. Hoboken, N.J.: John Wiley & Sons Ltd., 2014.

[26] "IEEE Standard Glossary of Software Engineering Terminology," 1990. [Online]. Available: ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=2238. [Accessed: 14-Jul-2015].

[27] J. Miller and J. Mukerji, "MDA Guide Version 1.0," 2003. [Online]. Available: http://www.omg.org/mda/mda_files/MDA_Guide_Version1-0.pdf. [Accessed: 14-Jul-2015].

[28] G. Tzeng and J. Huang, *Multiple Attribute Decision Making: Methods and Applications*. Boca Raton, Florida: CRC Press, 2011.

[29] R. de Neufville, *Applied Systems Analysis: Engineering Planning and Technology Management*. New York, USA: McGraw-Hill Inc, 1990.

[30] O. De Weck and D. Chang, "Quantitative Assessment of Technology Infusion in Communications Satellite Constellations," in *21st International Communications Satellite Systems Conference and Exhibit*, 2003, pp. 1–12.

[31] D. Mavris and N. Borer, "Formulation of a Multi-Mission Sizing Methodology for Competing Configurations," in *42nd AIAA Aerospace Sciences Meeting and Exhibit*, American Institute of Aeronautics and Astronautics, 2004.

[32] K. Deb, *Multi-Objective Optimization Using Evolutionary Algorithms*. New York, NY, USA: John Wiley & Sons, 2001.

[33] R. Steuer, *Multiple Criteria Optimization: Theory, Computation, and Application*. New York, NY, USA: Wiley Subscription Services, Inc., 1989.

[34] A. Forrester, A. Sobester, and A. Kean, *Engineering Design via Surrogate Modelling*. Chichester, West Sussex: John Wiley & Sons Ltd., 2008.

[35] D. Brown, V. Tech, and L. Thomas, "Reengineering the Naval Ship Concept Design Process", From Research to Reality," in *Ship Systems Engineering Symposium, ASNE*, 1998.

[36] A. Singh and C. Dagli, "Multi-objective Stochastic Heuristic Methodology for Tradespace Exploration of a Network Centric System of Systems," in *IEEE SysCon 2009 —3rd Annual IEEE International Systems Conference.*, 2016, no. APRIL 2009.

[37] A. M. Ross and D. E. Hastings, "The Tradespace Exploration Paradigm," in *INCOSE international Symposium. Vol. 15. No. 1*, 2005, pp. 1706–1718.

[38] A. Ross, G. O'Neill, D. Hastings, and D. Rhodes, "Aligning Perspectives and Methods for Value-Driven Design," in *AIAA SPACE 2010 Conference & Exposition*, 2010, pp. 1–30.

[39] "Merriam-Webster Dictionary." [Online]. Available: http://www.merriam-webster.com/dictionary/value. [Accessed: 30-Aug-2013].

[40] P. Collopy, "Economic-based distributed optimal design," *AIAA Pap.*, vol. 4675, 2001.

[41] O. Brown, P. Eremenko, and P. Collopy, "Value-Centric Design Methodologies for Fractionated Spacecraft: Progress Summary from Phase 1 of the DARPA System F6 Program," in *AIAA SPACE 2009 Conference Exposition*, 2009, no. September, p. AIAA 2009-6540.

[42] G. Richardson, J. Penn, and P. Collopy, "Value-Centric Analysis and Value-Centric Design," in *AIAA SPACE 2010 Conference & Exposition*, 2010, no. September, pp. 1–11.

[43] P. Collopy, "Aerospace system value models: A survey and observations," *AIAA Proceedings.[np]. 14–17 Sep*, pp. 1–18, 2009.

[44] M. Price, S. Raghunathan, and R. Curran, "An integrated systems engineering approach to aircraft design," *Prog. Aerosp. Sci.*, vol. 42, no. 4, pp. 331–376, Jun. 2006.

[45] J. Cheung, J. Scanlan, J. Wong, J. Forrester, H. Eres, P. Collopy, P. Hollingsworth, S. Wiseall, and S. Briceno, "Application of value-driven design to commercial aero-engine systems," *10th AIAA Aviat. Technol. Integr. Oper. ATIO Conf.*, no. September, pp. 1–21, 2011.

[46] W. Chen, C. Hoyle, and H. Wassenaar, *Decision-Based Design: Integrating Consumer Preferences into Engineering Design*. London, U.K.: Springer Science & Business Media, 2012.

[47] T. Saaty, "How to make a decision: The Analytic Hierarchy Process," *Eur. J. Oper. Res.*, vol.

48, no. 1, pp. 9–26, 1990.

[48] E. Triantaphyllou and S. H. Mann, "Using the Analytic Hiearchy Process For Decision Making In Engineering Aapplications: Some Challenges," *Int. J. Ind. Eng. Appl. Pract.*, vol. 2, no. 1, pp. 35–44, 1995.

[49] R. Keeney and H. Raiffa, *Decisions with Multiple Objectives: Preferences and Value Tradeoffs*. Cambridge, UK: Cambridge University Press, 1993.

[50] R. Keeney, *Value-Focused Thinking: A Path to Creative Decision Making*. Cambridge, MA: Harvard University Press, 1992.

[51] C. Hwang and K. Yoon, *Multiple attribute decision-making: methods and applications : a state-of-the-art survey*. Berlin, Germany: Springer-Verlag, 1981.

[52] H. Shih, H. Shyur, and E. Lee, "An extension of TOPSIS for group decision-making," *Math. Comput. Model.*, vol. 45, no. 7–8, pp. 801–813, Apr. 2007.

[53] E. Karsak, "Fuzzy multiple objective programming framework to prioritize design requirements in quality function deployment," *Comput. Ind. Eng.*, vol. 47, no. 2–3, pp. 149–163, Nov. 2004.

[54] L. Chan and M. Wu, "Quality Function Deployment: A Comprehensive Review of Its Concepts and Methods," *Qual. Eng.*, vol. 15, no. 1, pp. 23–35, Sep. 2002.

[55] M. Eres, M. Bertoni, M. Kossmann, and J. Scanlan, "Mapping Customer Needs to Engineering Characteristics : An Aerospace Perspective for Conceptual Design," *J. Eng. Des.*, no. 1–24, 2014.

[56] D. Dawson and R. Askin, "Optimal new product design using quality function deployment with empirical value functions," *Qual. Reliab. Eng. Int.*, vol. 15, no. 1, pp. 17–32, 1999.

[57] L. Vanegas and A. Labib, "A Fuzzy Quality Function Deployment (FQFD) model for deriving optimum targets," *Int. J. Prod. Res.*, vol. 39, no. 1, pp. 99–120, 2001.

[58] Z. Ayağ and R. özdem[idot]r, "An analytic network process-based approach to concept evaluation in a new product development environment," *J. Eng. Des.*, vol. 18, no. 3, pp. 209–226, 2007.

[59] W. Song, X. Ming, and Z. Wu, "An integrated rough number-based approach to design concept evaluation under subjective environments," *J. Eng. Des.*, vol. 24, no. 5, pp. 320–341, May 2013.

[60] M. Woolley, J. Scanlan, and W. Eveson, "Optimising the Development of a Medical Device Using Formal Engineering Design Techniques and the CODA-System," in *Page 1 7th International Conference on Concurrent Enterprising 27-29*, p. 2001.

[61] N. Belavendram, *Quality By Design: Taguchi Technique for Industrial Experimentation*. London, U.K.: Prentice Hall, 1995.

[62] R. Keeney, *Value-Focused Thinking: A Path to Creative Decisionmaking*. Cambridge, MA: Harvard University Press, 1992.

[63] A. Ross, N. Diller, D. Hastings, and J. Warmkessel, "Multi-Attribute Tradespace Exploration in Space System Design," in *53 rd International Astronautical Congress The World Space Congress*, 2002, pp. 0–14.

[64] A. Ross and D. Rhodes, "Using Natural Value-Centric Time Scales for Conceptualizing System Timelines through Epoch-Era Analysis," in *INCOSE International Symposium*, 2008.

[65] M. Fitzgerald and A. Ross, "Mitigating Contextual Uncertainties with Valuable Changeability Analysis in the Multi-Epoch Domain," in *SysCon2012 -- IEEE International Systems Conference*, 2012, pp. 19–23.

[66] M. Fitzgerald and A. Ross, "Sustaining lifecycle value: Valuable changeability analysis with era simulation," *2012 IEEE Int. Syst. Conf. SysCon 2012*, pp. 1–7, Mar. 2012.

[67] R. Burden and D. Faires, *Numerical Analysis*, 9th Edition. Boston, Masschusetts: Cengage Learning, 2010.

[68] D. Kahneman and A. Tversky, "Prospect theory: An analysis of decision under risk," *Econom. J. Econom. Soc.*, pp. 263–291, 1979.

[69] K. Arrow, A. Sen, and K. Suzumura, *Handbook of Social Choice and Welfare*, Vol 1. London, U.K.: Elsevier Science, 2010.

[70] W. Megginson, S. Smart, and B. Lucey, *Introduction to Corporate Finance*. London, U.K.:

Cengage Learning EMEA, 2008.

[71] B. Needles, M. Powers, and S. Crosson, *Principles of Accounting*. Mason, USA: South-Western Cengage Learning, 2008.

[72] S. Castagne, R. Curran, and P. Collopy, "Implementation of Value-Driven Optimisation for the Design of Aircraft Fuselage Panels," *Int. J.ProductionEconomics*, 2009.

[73] C. Justin, S. Briceno, and D. N. Mavris, "A COMPETITIVE AND REAL-OPTIONS FRAMEWORK FOR THE ECONOMIC ANALYSIS OF LARGE AEROSPACE PROGRAMS," in *28TH INTERNATIONAL CONGRESS OF THE AERONAUTICAL SCIENCES*, 2012, pp. 1–12.

[74] A. Boardman, D. Greenberg, A. Vining, and D. Weimer, *Cost-Benefit Analysis: Concepts and Practice*, 4th Edition. Prentice Hall, 2010.

[75] M. Ferraro, D. Gorissen, J. Scanlan, A. Kean, E. Quaranta, B. Schumann, J. Van Schaik, and M. Bolinches, "Towards Value-Driven Design of a small, low-cost UAV," in *53rd AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference*, 2012.

[76] J. Nickel, A. Ross, and D. Rhodes, "Comparison of Project Evaluation Using Cost-Benefit Analysis and Multi- Attribute Tradespace Exploration in the Transportation Domain," in *Second International Symposium on Engineering Systems*, 2009.

[77] C. Mattson, A. Mullur, and A. Messac, "Minimal Representation of Multiobjective Design Space Using a Smart Pareto Filter," in *9th AIAA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, American Institute of Aeronautics and Astronautics, 2002.

[78] A. Ching-Lai Hwang, *Multiple Objective Decision Making — Methods and Applications*. Berlin: Springer Berlin Heidelberg, 1979.

[79] D. Jones and M. Tamiz, *Practical Goal Programming*. Berlin, Germany: Springer Science & Business Media, 2010.

[80] I. Das and J. Dennis, "Normal-Boundary Intersection: A New Method for Generating the Pareto Surface in Nonlinear Multicriteria Optimization Problems," *SIAM J. Optim.*, vol. 8, no. 3, pp. 631–657, Mar. 1998.

[81] A. Messac and C. Mattson, "Normal Constraint Method with Guarantee of Even Representation of Complete Pareto Frontier," *AIAA J.*, vol. 42, no. 10, pp. 2101–2111, Oct. 2004.

[82] D. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*, 1st ed. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1989.

[83] N. Srinivas and K. Deb, "Muiltiobjective Optimization Using Nondominated Sorting in Genetic Algorithms," *Evol. Comput.*, vol. 2, no. 3, pp. 221–248, Sep. 1994.

[84] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *Evol. Comput. IEEE Trans.*, vol. 6, no. 2, pp. 182–197, Apr. 2002.

[85] L. SCHMIT and B. FARSHI, "Some Approximation Concepts for Structural Synthesis," *AIAA J.*, vol. 12, no. 5, pp. 692–699, May 1974.

[86] J. LA Schmit and H. Miura, "Approximation concepts for efficient structural synthesis," Los Angeles, CA, 1976.

[87] I. Voutchkov and A. Keane, "Multi-Objective Optimization Using Surrogates," *Comput. Intell. Optim.*, vol. 7, pp. 155–175, 2010.

[88] D. MacKay, *Information Theory, Inference and Learning Algorithms*. Cambridge, U.K.: Cambridge University Press, 2003.

[89] G. Box and N. Draper, *Empirical model-building and response surfaces*. Wiley, 1987.

[90] D. Krige, "A Statistical Approach to Some Basic Mine Valuation Problems on the Witwatersrand," *J. Chem. Metall. Min. Soc. South Africa*, vol. 52, no. 6, pp. 119–139, Dec. 1951.

[91] J. Bonnans, J. Gilbert, C. Lemarechal, and C. A. Sagastizábal, *Numerical Optimization: Theoretical and Practical Aspects*. Springer, 2006.

[92] S. Han, "A Globally Convergent Method for Nonlinear Programming," *J. Optim. Theory Appl.*, vol. 22, no. 3, pp. 297–309, 1977.

[93] M. D. Powell, "A fast algorithm for nonlinearly constrained optimization calculations," in *Numerical analysis*, Springer, 1978, pp. 144–157.

[94] W. Engler, P. Biltgen, and D. Mavris, "Concept Selection Using an Interactive

Reconfigurable Matrix of Alternatives (IRMA)," in *45th AIAA Aerospace Sciences Meeting and Exhibit*, American Institute of Aeronautics and Astronautics, 2007.

[95] N. Noy and D. McGuinness, "Ontology development 101: A guide to creating your first ontology," *Development*, vol. 32, pp. 1–25, 2001.

[96] D. Allemang and J. Hendler, *Semantic Web for the Working Ontologist: Effective Modeling in RDFS and OWL*. Elsevier Science, 2011.

[97] I. Sanya and E. Shehab, "A framework for developing engineering design ontologies within the aerospace industry," *Int. J. Prod. Res.*, no. July, pp. 1–27, 2014.

[98] S. Ahmed, S. Kim, and K. Wallace, "A Methodology for Creating Ontologies for Engineering Design," *J. Comput. Inf. Sci. Eng.*, vol. 7, no. 2, pp. 132–140, Oct. 2006.

[99] L. van Ruijven, "Ontology for Systems Engineering," in *Procedia Computer Science*, 2013, vol. 16, pp. 383–392.

[100] M. Darlington and S. Culley, "Investigating ontology development for engineering design support," *Adv. Eng. Informatics*, vol. 22, no. 1, pp. 112–134, 2008.

[101] S. Lim, Y. Liu, and W. Lee, "A methodology for building a semantically annotated multi-faceted ontology for product family modelling," *Adv. Eng. Informatics*, vol. 25, no. 2, pp. 147–161, 2011.

[102] S. Russell and P. Norvig, *Artificial Intelligence: A Mordern Approach*, 3rd Edition. New Jersey: Pearson Education inc., 2010.

[103] O. Daniel, A. Ross, D. Rhodes, and C. Link, "Evaluating system change options and timing using the epoch syncopation framework.," in *New Challenges in Systems Engineering and Architecting Conference on Systems Engineering Research (CSER)*, 2012, pp. 22–30.

[104] M. Silver and O. De Weck, "Time-Expanded Decision Networks : A Framework for Designing Evolvable Complex Systems," *Syst. Eng. Vol. 10*, vol. 10, no. 2, pp. 167–186, 2007.

[105] M. Lin, "Multi-objective Constrained Optimization for Decision Making and Optimization for System Architectures," Massachusetts Institute of Technology, 2010.

[106] M. Ehrgott and X. Gandibleux, "A survey and annotated bibliography of multiobjective combinatorial optimization," *OR Spectrum*, vol. 22, pp. 425–460, 2000.

[107] D. Rayside and H. Estler, "A Spreadsheet-like User Interface for Combinatorial Multi-objective Optimization," in *Proceedings of the 2009 Conference of the Center for Advanced Studies on Collaborative Research*, 2009, pp. 58–69.

[108] W. Simmons, B. Koo, and E. Crawley, "Architecture Generation for Moon-Mars Exploration Using an Executable Meta-Language," in *AIAA SPACE 2005*, 2005, pp. 1–17.

[109] W. Simmons, B. Koo, and E. Crawley, "Architecture Generation for Moon-Mars Exploration Using an Executable Meta-Language," in *Space 2005*, American Institute of Aeronautics and Astronautics, 2005.

[110] W. Simmons, B. Koo, and E. Crawley, "Space Systems Architecting using Meta-Languages," in *56th International Astronautical Congress of the International Astronautical Federation, the International Academy of Astronautics, and the International Institute of Space Law*, American Institute of Aeronautics and Astronautics, 2005.

[111] S. Catanzaro, "Multi-stakeholder quantitative analysis of sustainability for value delivery systems," Massachusetts Institute of Technology., 2006.

[112] B. Koo, W. Simmons, and E. Crawley, "Algebra of Systems: A Metalanguage for Model Synthesis and Evaluation," *Syst. Man Cybern. Part A Syst. Humans, IEEE Trans.*, vol. 39, no. 3, pp. 501–513, May 2009.

[113] M. Newman, *Networks: An Introduction*, New York. Oxford University Press Inc., 2010.

[114] R. Smaling and O. De Weck, "Assessing Risks and Opportunities of Technology Infusion in System Design," in *AIAA-2004-4553, 10th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, 2004, vol. 10, no. 1, pp. 1–25.

[115] X. Zhang, S. Tong, H. Eres, M. Kossmann, and K. Wang, "A value-focused approach for establishing requirements specification of commercial aircraft," *Proc. Inst. Mech. Eng. Part G J. Aerosp. Eng.*, Dec. 2013.

[116] X. Zhang, G. Auriol, H. Eres, C. Baron, and M. Kossmann, "Understanding Airlines ' Value Perceptions for Value-Based Requirements Engineering Of Commercial Aircraft," in *22nd*

Annual INCOSE International Symposium, 2012.

[117] "Wolfram Math World." [Online]. Available: http://mathworld.wolfram.com/BrentsMethod.html. [Accessed: 18-Sep-2013].

[118] R. Brent, *Algorithms for Minimization Without Derivatives*. New Jersey: Dover Publications, 2013.

[119] R. Smaling, "System architecture analysis and selection under uncertainty," Massachusetts Institute of Technology, 2005.

[120] R. Purshouse, P. Fleming, C. Fonseca, S. Greco, and J. Shaw, *Evolutionary Multi-Criterion Optimization: 7th International Conference, EMO 2013, Sheffield, UK, March 19-22, 2013. Proceedings*. Springer Berlin Heidelberg, 2013.

[121] A. Utturwar, S. Rallabhandi, D. DeLaurentis, and D. Mavris, "A Bi-level Optimization Approach for Technology Selection," in *9th AIAA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, American Institute of Aeronautics and Astronautics, 2002.

[122] J. Gundlach, *Designing Unmanned Aircraft Systems: A Comprehensive Approach*. Manassas, Virginia: American Institute of Aeronautics and Astronautics, 2012.

[123] *Defence Committee The contribution of Unmanned Aerial Vehicles to ISTAR capability*. House of Commons Defence Committee, 2008.

[124] "Joint Doctrine Note 3 / 10 Unmanned Aircraft Systems : Terminology , Definitions and Classification," 2010. [Online]. Available: https://www.gov.uk/government/uploads/system/uploads/attachment_data/file/432646/20150427-DCDC_JDN_3_10_Archived.pdf. [Accessed: 14-Jul-2015].

[125] "Remote Control: Remotely Piloted Air Systems - current and future UK use - Defence Committee," *www.parliament.uk*, 2014. [Online]. Available: http://www.publications.parliament.uk/pa/cm201415/cmselect/cmdfence/611/61104.htm. [Accessed: 31-Jul-2015].

[126] B. Schumann, M. Ferraro, A. Surendra, J. Scanlan, and H. Fangohr, "Better Design Decisions through Operational Modeling during the Early Design Phases," *J. Aerosp. Inf. Syst.*, 2013.

[127] "Unmanned Aircraft System Operations in UK Airspace - Guidance," 2012. [Online]. Available: https://www.caa.co.uk/cap722. [Accessed: 14-Jul-2015].

[128] A. Vahidi and A. Eskandarian, "Research advances in intelligent collision avoidance and adaptive cruise control," *IEEE Trans. Intell. Transp. Syst.*, vol. 4, no. 3, pp. 143-153, 2003.

[129] L. Blake and M. Long, *Antennas: Fundamentals, Design, Measurement*. Raleigh, NC: SciTech Publishing, 2009.

[130] M. Buonanno, "A Method for Aircraft Concept Exploration Using Multicriteria Interactive Genetic Algorithms," Georgia Institute of Technology, 2005.

[131] A. Hahn and N. Langley, "Vehicle Sketch Pad : A Parametric Geometry Modeler for Conceptual Aircraft Design," in *48th AIAA Aerospace Sciences Meeting*, 2010, no. 1, pp. 1-11.

[132] J. Anderson, *Fundamentals of Aerodynamics*, 5th Edition. Boston, MA, USA: McGraw-Hill Education, 2010.

[133] P. Fortescue, G. Swinerd, and J. Stark, *Spacecraft Systems Engineering*. Wiley, 2011.

[134] L. Cavagna, S. Ricci, and L. Travaglini, "NeoCASS: An integrated tool for structural sizing, aeroelastic analysis and MDO at conceptual design level," *Prog. Aerosp. Sci.*, vol. 47, no. 8, pp. 621-635, Nov. 2011.

[135] J. Leachtenauer and R. Driggers, *Surveillance and Reconnaissance Imaging Systems: Modeling and Performance Prediction*. Artech House, 2001.

[136] "2008 NASA Cost Estimating Handbook," 2008. .

[137] R. Hess and H. Romanoff, "Aircraft airframe cost estimating relationships: all mission types," Santa Monica CA, 1987.

[138] J. Cherwonik, "Unmanned Aerial Vehicle System Acquisition Cost Estimating Methodology," 2004.

[139] "Unmanned Aerial Vehicle Reliability Study," 2003. [Online]. Available: https://www.uvsr.org/Documentatie UVS/Publicatii-internationale/ReliabilityStudy-

2003.pdf. [Accessed: 14-Jul-2015].

[140] D. Jones, M. Schonlau, and W. Welch, "Efficient Global Optimization of Expensive Black-Box Functions," *J. Glob. Optim.*, vol. 13, no. 4, pp. 455–492, 1998.

[141] G. Box, *Statistics for Experimenters: Design, Innovation, and Discovery.*, Second Edi. Wiley, 2009.

[142] E. Torenbeek, *Synthesis of Subsonic Airplane Design: An Introduction to the Preliminary Design of Subsonic General Aviation and Transport Aircraft, with Emphasis on Layout, Aerodynamic Design, Propulsion and Performance*. Springer, 1982.

[143] A. Olewnik and K. Lewis, "Research and Applications On Validating Engineering Design Decision Support Tools," *Concurr. Eng. Res. Appl.*, vol. 13, no. 2, 2005.

[144] K. Pedersen, R. Bailey, J. Allen, and F. Mistree, "VALIDATING DESIGN METHODS & RESEARCH: THE VALIDATION SQUARE," in *ASME Design Engineering Technical Conferences*, 2000, pp. 1–12.

[145] D. Frey and C. Dym, "Validation of design methods: Lessons from medicine," *Res. Eng. Des.*, vol. 17, no. 1, pp. 45–57, 2006.

[146] E. Kreyszig, *Advanced Engineering Mathematics*. John Wiley & Sons, 2010.

[147] N. Chriss, *Black-Scholes and Beyond: Option Pricing Models*. Chicago, USA: McGraw-Hill Inc, 1997.