PERGAMON

Engineering Applications of Artificial Intelligence 15 (2002) 105–116

Engineering Applications of
ARTIFICIAL
INTELLIGENCE

www.elsevier.com/locate/engappai

A domain knowledge based search advisor for design problem solving environments

Y.S. Ong, A.J. Keane*

Computational Engineering and Design Centre, School of Engineering Sciences, University of Southampton, Highfield, Southampton, SO17 1BJ, UK

Received 1 March 2001; accepted 1 January 2002

Abstract

This paper proposes an automatic and adaptive Domain Knowledge Based (DKB) Search Advisor for use with Design Exploration Systems (DES)—a form of design Problem Solving Environment (PSE). The advisor contains domain knowledge of search routine performance on design problems built using a knowledge modelling methodology. These help designers working on complex engineering problems to decrease the cost of design-space search and improve the quality of the resulting designs. This paper introduces this field, beginning with a view of some of the problems and inefficiencies of present design processes. This is followed by the description of a knowledge modelling methodology that may be used to build knowledge models of search routine performance on design domains. One focus of the paper is the use of machine learning to automate the process of knowledge discovery. The practicability of the DKB Search Advisor is then demonstrated with a case study taken from the aircraft wing design domain. The results presented help provide insights into the strengths and weaknesses of various optimization routines. More importantly, they also illustrate that an advisor containing knowledge of search routine performance on design domains can support design engineers in their search activities. The Search Advisor helps to decrease the cost of aircraft wing design search while at the same time increasing the quality of the resulting designs.

Keywords: Domain knowledge base; Optimization; Engineering design exploration; Knowledge modelling methodology

1. Introduction

Presently, most complex engineering design exploration is carried out manually. The design engineer uses computer aided design (CAD) tools to make a modification to the design and evaluates this by numerical simulation. He then enters a design–evaluate–redesign process and stops when he thinks that the design is adequate based on his experience and knowledge of past designs. A numerical simulation may thus be used by hand to provide a design exploration process. By contrast, Design Exploration Systems (DES) address the needs of engineers responsible for automatically establishing critical design parameters, usually during the early stages of the design process. Such systems aim to provide a controlled framework for studying the effects of parameter choices on the numerical simulations available to the designer for dealing with geometric decisions, stress analysis, performance estimation, etc.

Today, several powerful DES such as OPTIONS (Keane, 1995) and Isight (2000) have become available, most of which have the common characteristic of containing multiple sophisticated optimization and exploration techniques for design-space search. The heart of a good DES is usually a design optimization facility, which is basically a design-space search tool that is employed to automatically find good solutions to some problem (e.g., by finding the maximum of a function) by generating a collection of potential solutions to the problem and then manipulating them. Optimization is a mature technology that has been studied extensively by researchers over the last half century. Although studied for many years it has only recently been heavily used by the design community (Keane and Nair, 2001). This take-up is now happening because increases in computing power allow increasingly accurate analysis codes to be deployed in this way, see for example, the work reported by (Jameson, 1999) in a recent theme issue dedicated to optimization.

*Corresponding author. Tel.: +44-23-8059-3392; fax: +44-1703-593230.

E-mail address: andy.keane@soton.ac.uk (A.J. Keane).

0952-1976/02/$ - see front matter
PII: S0952-1976(02)00016-7

Engineering design optimization thus helps reduces the cycle time of design–evaluate–redesign iteration loops and often finds better designs by computerizing parts of this iterative process. In general, given a set of design variables D, a set of bounds and constraints B and C, constrained design optimization is the problem of determining values of D to minimize or maximize an objective function $F(D)$, subject to B and C.

In practice, however, unless one knows which optimization routines in the DES most suit the design problem in hand, the optimization method may not perform properly or achieve a truly optimum design. For example, gradient-based methods have the known advantage of their efficiency; however, they are also very sensitive to starting point selection and are more likely to stop at non-global optima than modern stochastic algorithms. Stochastic techniques on the other hand produce new design points that do not use information about the local slope of the objective function and thus are not prone to stalling in false optima. They do tend to require more analysis effort, however.[1] Moreover, some search routines might not even be capable of producing a feasible design. Sandgren (1977) applied 35 nonlinear optimization algorithms to 30 engineering design optimization problems and compared their performance. Bramlette and Cusic (1989) also compared the application and performance of different methods, including gradient based numerical optimization, to the design and manufacture of aeronautical systems. The applicability of different conventional numerical optimization methods to aircraft design has been further explored by Sobieszczanski–Sobieski and Haftka (1996). The general conclusion obtained from all these studies is that no single optimization search technique always performs well on all problems—a result that is sometimes referred to as the "no free lunch theorem".

Nevertheless, it remains common to find design engineers relying very much on their intuition, experience and knowledge of a design domain when making the choice of optimization routine to employ whenever a design search is conducted. The effect of this is that the design quality is heavily dependent on the experiences and knowledge of the design engineer and this may lead to non-optimal designs being produced at high cost due to the limited experience of novice or inexperienced designers (i.e., in reality, designers often stick to a very limited range of optimization techniques regardless of the design problem involved or the sophistication of any optimization methods suite available). It may also have a detrimental effect on design innovation by placing too much dependence on a single individual's past designs and decisions, which usually contain biases. In addition, sole reliance on experienced designers may also mean a heavy cost for professional manpower to assure maximal performance from new designs.

Few studies in the literature have addressed the problem of choice of optimization search techniques for engineering designs with much success. Most have tried to identify the optimization techniques that will work well on rather limited ranges of design test problems. Here, we propose a Knowledge Base Search Advisor for use in design search activities to address the problem. A complete Knowledge Base Search Advisor would assume the presence of two knowledge bases; a General Knowledge Base (GKB) and a Domain-Specific Knowledge Base (DKB). The GKB should contain generalized knowledge that is normally applicable for design-space search in most domains. The DKB should contain specialized knowledge that is applicable only to a particular domain or design problem area (i.e., a particular objective function or narrow classes of functions). The complete Knowledge Base Search Advisor should then provide the following capabilities: (1) it would be able to recommend overall optimization techniques for any general design problem using its GKB and (2) it would be able to refine this recommendation for each design problem using its DKB. Most importantly, the search advisor would help decrease the cost of design-space search through reducing reliance on human design experts and also help increase the quality of the resulting designs. It would also speed up the design process. In this paper, we consider only an automatic and adaptive Domain Knowledge Based (DKB) Search Advisor for use with DES. We do this, primarily, for simplicity in presenting the Search Advisor. The case of GKB Search Advisor will be addressed in detail elsewhere.

Note that throughout this paper a problem domain refers to an engineering problem domain in part of which designs are being optimized. Knowledge models contain knowledge about the performance of search routines on the design domain under study, generated using the intentions of the designer; typically, this might be to finish the design search as quickly as possible, to achieve the best possible design or otherwise. The 'Best Performing Search' (BPS) strategy is the search routine that achieves the desired intention as well as possible across the entire problem domain. In most engineering domains, this will be for $<100\%$ of the problems tackled. The 'Artificial Intelligence Machine Learning Selected' (AIMLS) strategy is the search routine that the proposed knowledge based advisor recommends for the problem in hand. This will vary with the problem details.

[1] Typical gradient-based methods are Sequential Quadratic Programming; Linear Approximation; Direct Search Methods; and others (Lawrence and Tits, 1996; Schwefel, 1995; Siddall, 1982). Among the modern stochastic optimizers are Genetic Algorithms; Simulated Annealing; Evolutionary Programming and Evolution Strategies (Yin and Germay, 1993; Kirkpatrick et al., 1983).

2. Domain knowledge based search advisors

Companies usually have limited diversity of trade and thus work-scope. For example, Airbus focuses mainly on aircraft design; Rolls-Royce on engine design; while a ship building company focuses on ship design. Depending on the complexity of a domain, some search routines that may have proven to be useful in one domain might not work so well in other domains. The same reasoning applies to individual design problems within a domain. This fundamental observation is the reason why it is important for a design engineer to be supported with a DKB Search Advisor that contains specific knowledge on the performance of search routines on the design domain under study.

The proposed methodology for building the DKB Search Advisor is drawn from the experiences obtained by the Problem Solving Community. The general approach used by this community is based on "Learning from experience" techniques, where information extracted from past problem solving experiences is used to assist in solving future problems, especially those that are similar (Laird et al., 1985; Greiner and Jurisca, 1992; Rentema et al., 1997; Houstis et al., 1998). Such an approach has also been used in some engineering design applications (Surma and Braunschweig, 1996; Rasheed and Hirsh, 1997; KBSI, 1999) to improve the design process in various ways. A good overview of the approach used in design applications is also available in (EAAI, 1996). In general, "Learning from experience" techniques can be classified into two main categories that have been studied by the artificial intelligence community. Expert systems, Case-Based reasoning, Explanation-based learning, Ontology based systems and others fall into the first category, which we categorize as manual knowledge acquisition approaches. Many of the systems used in the Problem Solving Community (Laird et al., 1985; EAAI, 1996; Rentema et al., 1997; KBSI, 1999) are examples of this category. The second category is automatic knowledge acquisition, and these often employ machine learning techniques. Examples of the second category may be found in (Wolberg et al., 1994; Reich, 1996; Rasheed and Hirsh, 1997).

In this paper, machine learning is proposed to carry out the role of automatic domain knowledge acquisition from the data of past designs. The choice of an automatic knowledge acquisition approach is a consequence of the following factors:

1. There are currently no models for predicting the most appropriate search routine in general DES for design search activities. Here, with an automated knowledge capture approach for building such a model, the search process is tested first from a heuristic perspective and then refined through the observation of new data acquired for related design problems.
2. Knowledge of the applicability and strength of a search routine to a particular domain or problem is normally based on the experience of human operators, which is difficult to extract, share and model. Moreover, the capability of a given search algorithm differs even among multiple implementations of the same theory: this makes manual generalization from theory almost impossible.
3. The main criticism of manual knowledge acquisition are (1) the need for the availability of human domain experts who are willing to share their experiences and knowledge in a most unreserved manner, (2) the need for qualified knowledge engineers to perform this knowledge acquisition task, which is often difficult to model and (3) the need for the acquired domain theory to be relatively complete and consistent.
4. Design engineers may well have some intuition and qualitative rules that give rise to biases (preferences for particular search techniques). These may result in non-optimum performance or even poor designs. It is also unlikely for one designer to be able to make decisions about all aspects of a design, which may possibly be simpler for a machine.
5. A design related knowledge base has to be dynamic, so that results from new searches initiated by the design engineer generate new knowledge that can be updated into the knowledge base. The use of automatic knowledge acquisition enables this process to be accomplished automatically.
6. Machine Learning is a recent approach to knowledge elicitation often referred to as "knowledge mining" (Piatetsky-Shapiro, 1989) or "knowledge discovery": (Frawley et al., 1991). Grounded on various AI-based techniques, the approach is automatic and acquires knowledge, extracts features or identifies patterns directly from examples or databases. Machine learning is thus particularly suitable here because it is able to automate the process of generalizing past design data on the applicability of optimization search routines to different subsets of problems within a given domain.

The methodology proposed for building the DKB Search Advisor is described next.

2.1. Methodology for construction of domain knowledge bases

The proposed methodology for the construction of the DKB begins with a *Sampling* process. *Sampling* generates a sample set of possible design problems for a design domain based on the design parameters of interest. This sample set of design problems is an approximate representation of the entire problem domain. This is followed by an *Offline Simulation* process, where optimization design searches using every

single search routine available in the DES are conducted on each design problems in the sample set.[2] The data produced during *Offline Simulation* are archived for knowledge discovery and acquisition. These include the final design variables found; objective function value, total evaluation count (time taken) during the searches and violations of any kind (design problem not computable, optimization variable boundary violations and constraint violations). The simulation process can be very time-consuming if the underlying analyses used are of significant complexity—in most concept design tools this is fortunately not so. The simulated data together with those collected from previous design–evaluate–redesign iterative cycles form the archived data sources that are processed next at the *Knowledge Modelling* stage. In this process, the desired information is first derived from the archived data using Eq. (1):

$$\text{Normalized Overall Quality}_i = \{[\text{IMPS} \times \text{NECQ}_i] + [(1 - \text{IMPS}) \times \text{NOFQ}_i]\}, \quad (1)$$

where Normalized Overall Quality is a measure of the quality of a search routine j on a sampled design problem i, 'IMPS' is a user specifiable parameter that determines the balance between speed or optimal design, and 'NECQ' and 'NOFQ' are the Normalized Evaluation Count Quality and Normalized Objective Function Quality (both normalized to unity), respectively. The quantity IMPS in the equation represents the intention of the design engineer; typically, this might be to finish the design search as quickly as possible (IMPS = 1), achieve the best possible design (IMPS = 0) or otherwise.[3] In the following sections, two strategies are proposed for the generation of knowledge models using this information.

2.1.1. Strategy I—'Best Performing Search (BPS) strategy'

The first strategy is the BPS strategy, which attempts to recommend a single search routine that provides the best performance for the entire problem domain. It uses the information obtained via Eq. (1) to estimate the average performance of each search routine over the sampled design problems. The search routine that is recommended is obtained using Eq. (2)

$$\text{Best Performing Search Routine} = \text{Max}\{\prod_{j=1}^{p}\{\sum_{i=1}^{k}\text{Normalized Overall Quality}_{ji}\}\} \quad (2)$$

for sampled design problem $i = \{1 \ldots k\}$, where k is the total number of sampled design problems available for the domain, Search Routine $j = \{1, \ldots, p\}$, and p is the total number of search routines available in the DES.

2.1.2. Strategy II—Artificial Intelligence Machine Learning Selected (AIMLS) strategy

The second strategy proposed is the AIMLS strategy. This strategy makes use of machine learning to carry out knowledge discovery and modelling. To extract knowledge about the merits and limitations of the many optimization search routines on a design domain using machine learning, it is necessary to perform further pre-processing on the data obtained using Eq. (1). This involves the conversion of the original data into table-like datasets, such that the sampled design problems are labeled and ranked according to the optimization search routine that performs the 'best'. 'Best' here is taken to mean the search routine with the highest value of *Normalized Overall Quality* for the particular sampled design problem. Machine learning is then employed as a classifier on the pre-processed table-like datasets. It aims to generalize or learn as much as possible from the datasets, so as to accurately identify clusters of design problems from the sampled set that belong to the same class. The resulting classes represent the search routines that were observed to be 'best', among all those available in the DES. If learning is successful, generalizing from these data should be possible to aid in future design sessions by successfully recommending the most appropriate optimization technique that best matches the new design problem to be searched.

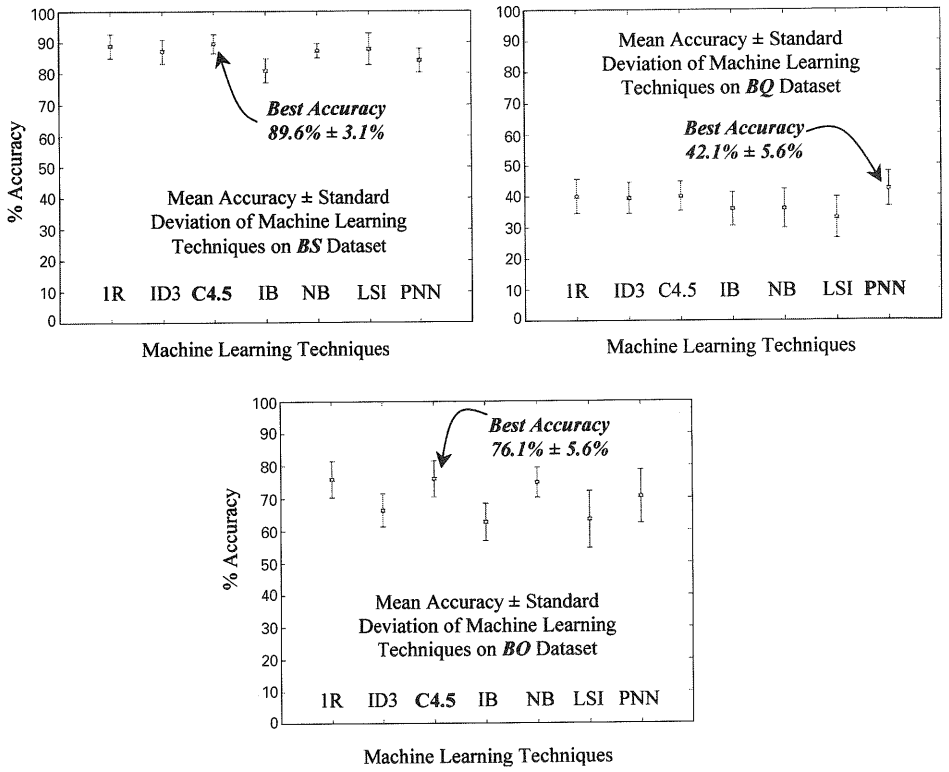
A brief survey of the many machine learning techniques described in (Aha, 1992; Paass, 1992; Deerwester et al., 1990; Holte, 1993; Langley and Sage, 1994; Quinlan, 1993) has been performed to identify a suitable learning technique for this application (see Fig. 1a for a list of the different learning techniques investigated). These techniques were evaluated according to their accuracy, standard deviation and transparency. The algorithms that have been found to be the most competitive in this application are C4.5, Naïve Bayes and Probabilistic Neural Network (Quinlan, 1993; Langley and Sage, 1994; Paass, 1992). Although most of the machine learning techniques considered here allow manual tuning, this has been considered to be too time-consuming and computationally expensive. Among the learning techniques considered, the use of a decision tree inductive learning algorithm such as C4.5 (Quinlan, 1993) is preferred because they produce reasonable classification accuracy at relatively low cost and more

[2] In all the simulations performed, the control parameters used for each search routine are based on the DES's defaults. This adoption of DES defaults is based on an analogue of designers relying on the default setting of a DES in the initial stages of most design processes. Clearly, a more sophisticated advisor might well supply advice on the choice of search routine control parameters.

[3] Many different models can be generated according to the intention of the design engineers. Alternatively, other criteria may be included into Eq. (1), such as search routine robustness etc., to better express the intentions of the design engineers.

Abbreviations	Machine Learning Techniques
1R	Simple Classifier (Holte, 1993)
ID3	Decision Trees I (Quinlan, 1993)
C4.5	Decision Trees II (Quinlan, 1993)
IB	Nearest-Neighbor (Instance-based) (Aha, 1992)
NB	Probabilistic (Naïve-Bayes) (Langley *et al*, 1994)
LSI	Information Retrieval (Latent Semantic Indexing) (Deerwester *et al*, 1990)
PNN	Neural Network (Probabilistic Neural Network) (Paass, 1992)

(a)



(b)

Fig. 1. Machine learning techniques and their abbreviations; (a) mean accuracies and standard deviation of each machine learning technique.

importantly, because they posses the ability to generate trees or rules that provide the transparency, we seek to give to designers. Designers often lack great expertise in the use of optimization methods and therefore have little confidence that extensive computational runs can produce worthwhile results as opposed to just burning up compute cycles. Therefore, when recommending search routines for design activities, it is important for the decision-making process to provide the necessary transparency. Of the many machine learning techniques available, knowledge derived in the form of decision trees or rules seems to satisfy this human-centered criterion best. Besides, human specialists can manually validate these machine-generated decision trees or rules and also use them to enhance the domain and optimization knowledge of less experienced design engineers.

In the methodology presented here, our main motivation towards the use of the AIMLS strategy is based on the observation that a single search routine does not always emerge as the 'best' method throughout the problem space of a single design domain. This is in contrast to the BPS strategy, which generates a knowledge model that recommends a single 'best' search routine for a complete design domain. Here, we attempt to use machine learning to generate knowledge models for predicting which search routine is 'best' when given a design problem from a familiar domain. From extensive investigations, although AIMLS is generally found to perform better than BPS, it has been noted that it can

sometimes perform poorly under cases, where there is insufficient sampled design problem data for training (i.e., search routines are too similar in performance to each other, so that more samples are required to enable a true differentiation of the 'best') and in multi-class problems (i.e., too many search routines have been seen as 'best' in a design domain). In these cases, generalization from the labeled datasets fails. Under such cases, BPS may be used instead. In effect, both strategies are used in the proposed knowledge modelling methodology. Heuristically, it is found that when the accuracy of the AIMLS generated selection is estimated to be poor (i.e. <0.7)[4] or when the performance of BPS is significantly high (>0.7), the selection proposed by the BPS is preferred over that from AIMLS.

2.1.3. Adaptability

It is important that any knowledge modelling methodology be adaptable. Here, once the DKB for a domain has been successfully generated, an *Adaptability* process is invoked repeatedly to ensure the continued usefulness of the existing knowledge base. Throughout the design process, a great deal of data is produced as a result of the design–evaluate–redesign actions conducted by designers. The Search Advisor archives all these data and indexes them according to the design problem being studied. At other times, scheduled batch jobs are started to update the DKB using information from the archive to ensure good coverage. This helps enable the Search Advisor remain relevant to current design problem domains as well as to encompass new ones.

3. Case study demonstration on transonic civil transport aircraft wing design domain

3.1. Case study description

The use of the DKB Search Advisor is next demonstrated on the domain of aircraft design, specifically the wing design of transonic transport aircraft (Keane and Petruzzelli, 2000). The design of the wings for transonic civil transport aircraft is an extremely complex task. It is normally undertaken over an extended time period and at a variety of levels of complexity. Typically, simple empirical models are use at the earliest stages of concept design, followed by ever more complex methods as the design process proceeds towards the final detailed stages. The parameters used to describe the wing design problem considered here consist of the free-stream velocity and coefficient of lift of the wing together with a small number of overall wing geometry variables. The geometry is characterized by the plan-form shape of the wing together with several span-wise functions such as twist and thickness to chord ratio. These are represented by eleven parameters (i.e., eleven optimization design variables). In order to prevent the search from driving the designs to unworkable extremes, several constraints are placed on the wings designed. These are the under-carriage bay length (which must be accommodated within the root to kink section of the wing), the fuel tank volume (which must be accommodated between the main spars within the wing), the wing weight and the pitch-up margin. A typical geometric view of such an aircraft with streamlines, the wing design variables, constraints and respective design limits are shown in Fig. 2. The goal in this problem is to design a wing with minimal drag based on empirical models.

3.2. Design Exploration system

The Design Exploration system used here is the one described in (Keane, 1995), and known as OPTIONS. Among the many different methods in OPTIONS, some are from standard libraries (Schwefel, 1995; Siddall, 1982), while others have been specially developed for the suite, based on ideas culled from the literature. The 30 optimization search routines of the OPTIONS DES used in the case study together with the abbreviations employed in the paper are listed in Table 1. For more details on the OPTIONS DES and search routines, the reader is referred to (Keane, 1995).

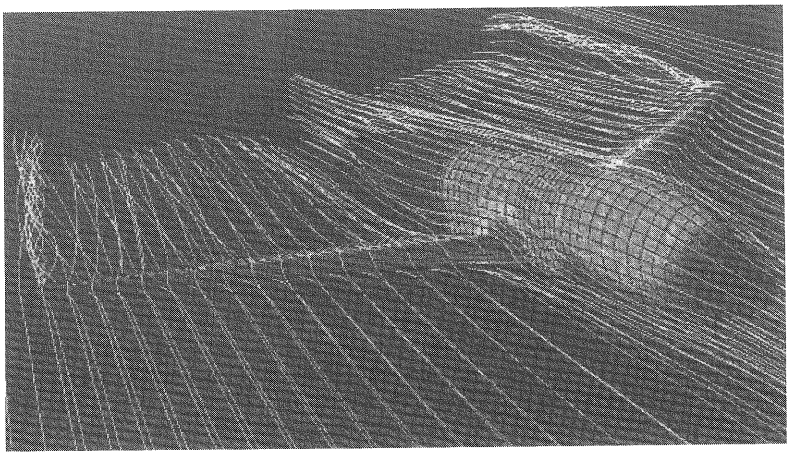
3.3. Case study results

In aircraft wing design, wings are often designed for minimal drag using a DES at given combinations of cruise height, mach number and fuel weight[5] by varying the various wing design variables. Here, a set of 729 design problems is first sampled using the Latin hypercube method (McKay et al., 1979), each defined by different cruise height, mach number and fuel weight fraction, bounded between 7500–12,000, 0.1–0.85 and 0.2–0.5 m, respectively.[6] Each sample design problem is then optimized using each search routine available in the DES. The following offline simulation data are archived; (1) Evaluation Count, (2) Minimum Wing Drag (Objective Function Value), (3) Design Boundary Violations, (4) Constraint Violations, (5) Non-Evaluable Design problems and (6) Optimized Design Variables.

[4] Accuracy estimation normalized to unity.

[5] Note that here cruise height, mach number and fuel weight are not variables to be optimized but rather considered the fixed parameters of a design problem and vary only for different problems within the aircraft wing design domain.

[6] The ranges of cruise height, mach number and fuel weight fraction were obtained from design engineers working within the aircraft wing design domain.



11 Wing Design Variable Definitions		
Lower Limit	*Upper Limit*	*Quantity (units)*
100	250	Wing Area (m^2)
6	12	Aspect Ratio
0.2	0.45	Kink position
25	45	Sweep angle (degrees)
0.4	0.7	Inboard taper ratio
0.2	0.6	Outboard taper ratio
0.1	0.18	Root thickness/chord
0.06	0.14	Kink thickness/chord
0.06	0.14	Tip thickness/chord
4.0	5.0	Tip wash (degrees)
0.65	0.85	Kink washout fraction
Four Design Constraints		
2.5		Under-Carriage bay length
	135000	Wing weight (N)
40.0		Wing volume (m^3)
	5.4	Pitch-up margin

Fig. 2. Geometric view of streamlines over a transonic civil transport aircraft with its wing design variables, constraints and respective limits shown.

The normalized average design search performances[7] (i.e., search efficiency and quality) of each search routine working on the aircraft wing design domain are summarized in Fig. 3. Each abbreviation in the figure represents an optimization search routine available in the OPTIONS DES. The percentage measure of each search routine's robustness across the sampled design problems are summarized in Fig. 4. From these figures, it is evident that no single routine always generates the best performance on each problem in the design domain, irrespective of design speed, quality or robustness. These results support the conclusions drawn in the literature and mentioned in the introduction.

As previously discussed, different forms of knowledge models representing the intention of the design engineer can be generated from the archived data sources. In this case study, we illustrate three common knowledge models often desired by design engineers. By setting IMPS in Eq. (1) to 0.5, the information necessary for Balanced-Overall (BO) knowledge modelling can be obtained. The Best-Speed (BS) model is obtained by having IMPS set to 1.0, while setting IMPS to 0.0 enables the Best-Quality (BQ) model to be generated.

3.3.1. *Knowledge modelling strategy I: Best Performing Search (BPS) strategy*

Using BPS, the various results of the DKB Search Advisor can be read directly from the information summarized in Fig. 3c. The routine giving the best speed is seen to be Successive Linear Approximation (AP) (Siddall, 1982)—the BS knowledge model emphasizes the importance of speed and therefore ignores the final result of the wing drag values as long as the final searched design parameters are feasible. Simulated Annealing (SA) (Kirkpatrick et al., 1983) gives the best quality by providing the most-optimal designs on average across the entire design domain. Finally, the BO model balances speed and quality and the best

[7] This may be achieved using Eq. (2).

Table 1
The 30 optimization search routines employed from the OPTIONS DES

Abbreviations	30 Optimization search routines from OPTIONS DES
AP	Method of successive linear approximation by Siddall (1982)
AD	Adaptive random search by Siddall (1982)
BC	Bit climbing algorithm (Keane, 1995)
CO	Complex strategy of box by Schwefel (1995)
DA	Davidon–Fletcher–Powell strategy by Siddall (1982)
DF	Davidon–Fletcher–Powell strategy by Schwefel (1995)
DH	Dynamic hill-climbing algorithm (Keane, 1995)
DO	Design of experiments based optimizer (Keane, 1995)
DP	Davis, Swan and Campey with Palmer orthogonalizational by Schwefel (1995)
DS	Davis, Swan and Campey, with Gram–Schmidt orthogonalization by Schwefel (1995)
EP	Evolutionary programming (Keane, 1995)
ES	Evolution strategy based on the earlier work of Bäck et al. (Keane, 1995)
FI	Repeated one-dimensional Fibonacci search by Schwefel (1995)
FL	Fletcher's 1972 method by Siddall (1982)
GA	Genetic algorithm based on clustering and sharing (Keane, 1995)
GO	Repeated one-dimensional Golden section search by Schwefel (1995)
HO	Hooke and Jeeves direct search by Schwefel (1995)
JO	Jacobson and Oksman method by Siddall (1982)
LA	Repeated one-dimensional Lagrangian interpolation search by Schwefel (1995)
MM	Schwefel's multi-membered evolution strategy by Schwefel (1995)
NU	Powell routine in the Numerical Recipes cookbook (Keane, 1995)
PB	Population-based incremental learning algorithm (Keane, 1995)
PD	Powell direct search method by Siddall (1982)
PO	Powell's strategy of conjugate directions by Schwefel (1995)
RO	Rosenbrock's rotating co-ordinated search by Schwefel (1995)
SA	Simulated annealing (Keane, 1995)
SE	Hooke and Jeeves direct search by Siddall (1982)
SI	Simplex strategy of Nelder & Meade by Schwefel (1995)
SM	Simplex strategy of Nelder & Meade by Siddall (1982)
2M	Schwefel's two-membered evolution strategy by Schwefel (1995)

The abbreviations for each search routine used in the paper are as shown in alphabetical order.

overall routine is found to be the Powell Direct (PD) search routine (Siddall, 1982).

3.3.2. Knowledge modelling strategy II: AI machine learning selected strategy (AIMLS)

In AIMLS, unlike BPS, the information available for learning undergoes further data processing as table-like datasets, before the knowledge models are built. The sampled aircraft wing design searches are labeled and ranked according to the search routine that performs 'best'. So for example, in the case of the best-speed model, 'best' represents the search routine that provides a feasible design within the shortest period of time. Here, the following statistics were obtained from the datasets:

- Five routines rank as 'Best Speed'. Even though there are 30 optimization search routines available in the OPTIONS DES, across the entire sampled wing design domain of 729 sets of parameters, only five search routines ever rank as 'Best Speed'. They are AP, PD, PO, FL and LA and form 56.0%, 21.7%, 16.6%, 4.7% and 1.0% of the BS dataset. Table 2 shows a portion of the pre-processed BS dataset.
- Seven search routines rank as 'Best Quality' and these are SM, PD, SI, 2M, AD, CO and SA, with respective percentages 28.2%, 18.4%, 15.1%, 12.5%, 9.2%, 8.9% and 7.7% in the BQ dataset.
- Finally, six search routines rank as 'Best Overall'. These are AP, PD, PO, FL, LA and 2M, with respective percentages of 49.1%, 28.4%, 14.1%, 4.7%, 2.2% and 1.5%, respectively, in the BO dataset.

The C4.5 induction algorithm is next used to extract the knowledge models from these datasets in the form of decision trees or rules. Fig. 5 shows one such decision tree generated by the C4.5 induction algorithm from the BS dataset. The accuracy estimation and standard deviation of the different machine learning techniques when applied on the datasets are also summarized in Fig. 1b. It is evident that the C4.5, Naïve Bayes and Probabilistic Neural Network are generally most competitive in this application.

4. Performance comparisons

In assessing the DKB Search Advisor, it is useful to develop standards for comparison. The following

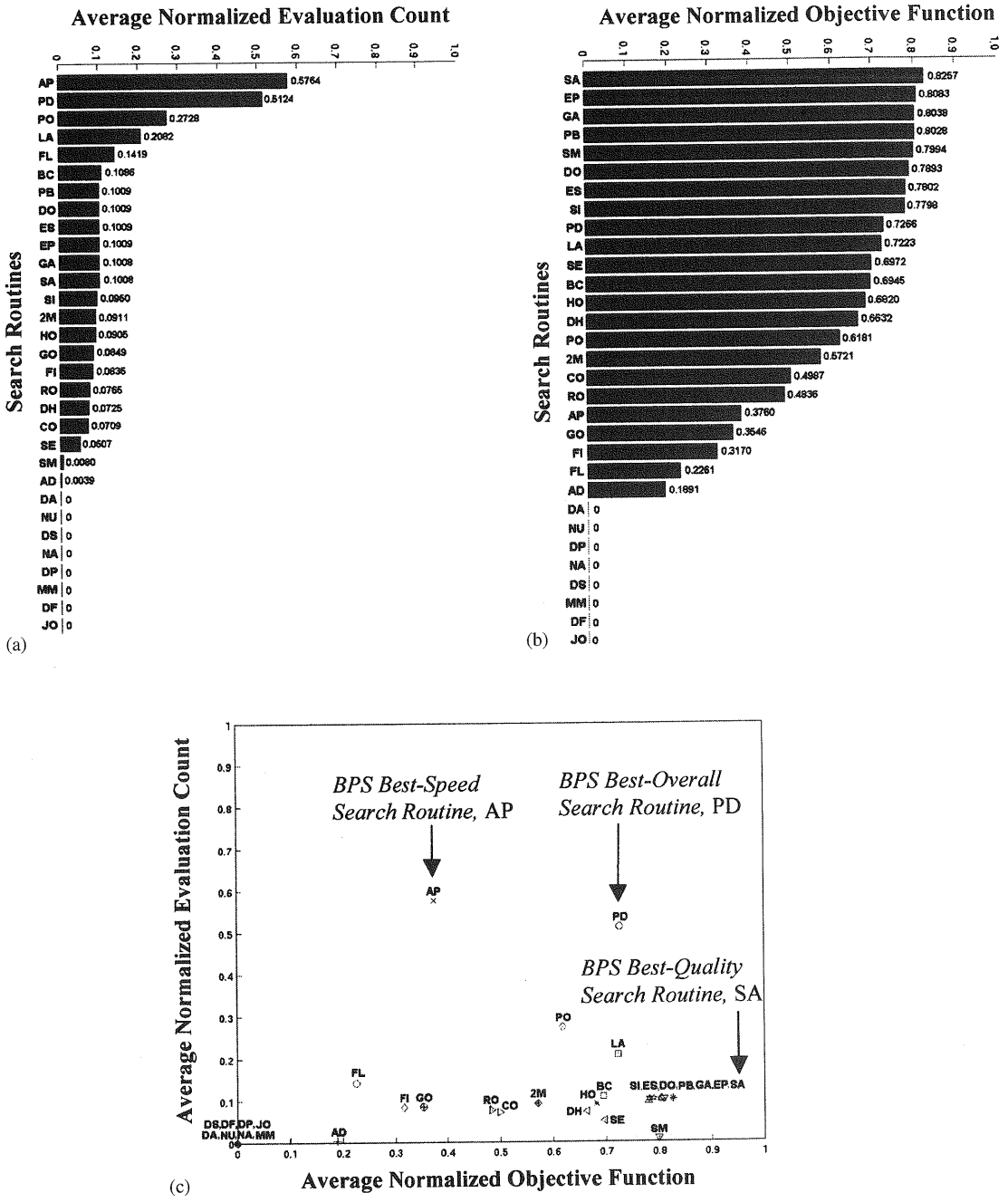


Fig. 3. Average performance of each Search Routine in the OPTIONS DES across the set of 729 Sampled Aircraft Wing Design Problems (larger values indicates faster searches or better designs). (a) Average Normalized Objective Function with rankings in descending order; (b) Average Normalized Evaluation Count with rankings in descending order; (c) A Plot of Average Normalized Evaluation Count against Average Normalized Objective Function for each Search Routine.

approaches, termed 'Common Designer Strategies' (CDS), have been derived from analogues to typical designers' behaviors (both novices and optimization experts) when working on design problems using a DES that requires design engineers to choose the optimization search methods manually. Five potential strategies have been identified:

- The simplest and most basic strategy adopted by a design engineer (usually a novice) is (CDS1) 'Random Guessing'. This just means randomly choosing optimization routines from those available in the Optimization engine to search on each new design problem.
- The second strategy (CDS2) is the notion of always using the same randomly selected optimization

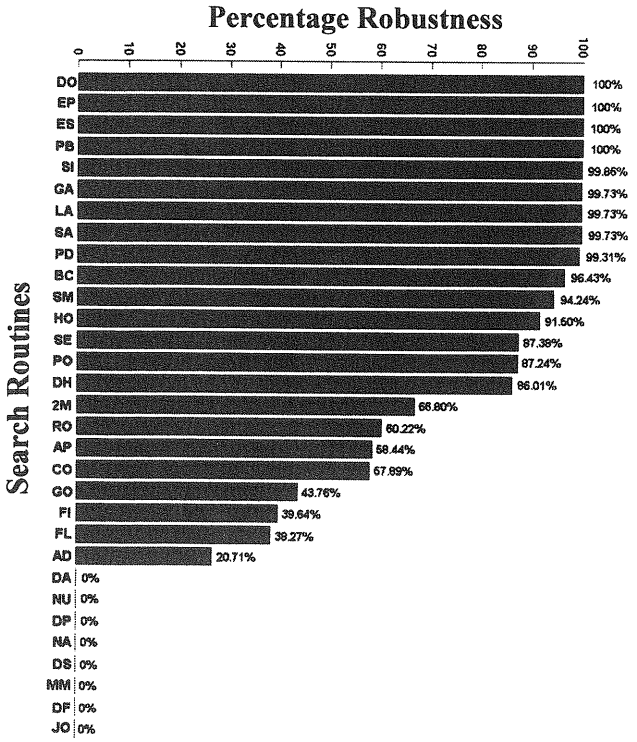


Fig. 4. Robustness measure of each Search Routine in the OPTIONS DES across the set of 729 Sampled Aircraft Wing Design Problems.

Table 2
Processed best-speed table-like dataset

Mach number (0.1–0.85)	Fuel weight fraction (0.2–0.5)	Cruise height (7500–12,000)	'Best-speed' search routine
0.35	0.225	8625.0	AP
0.725	0.425	10125.0	PD
		⋮	
0.1	0.45	7500.0	PO
0.85	0.275	8250.0	FL
0.22	0.475	10875.0	LA

routine, given that it was able to successfully generate a feasible design the first time it was used in a domain.

- The third strategy (CDS3) is to utilize an optimization method that is thought to be most robust, e.g., the Evolutionary Programming (EP) optimization method is often regarded to be very robust and is thus chosen here.
- The fourth strategy (CDS4) is an analogue to optimization method favoritism that may be displayed by a designer. Here, the Genetic Algorithm (GA) is chosen to be the design engineers' favorite.
- The final strategy identified (CDS5), is utilizing a optimization method that has generally been accepted as having the ability to provide a design within the shortest time, e.g., Successive Linear Approximation (AP) is often regarded as the fastest available method.

These five strategies are compared to the search advisor system in Table 3. The performance statistics shown in the table are based on design searches using one-third of the 729 sample design problems as the validation set, and using domain knowledge models derived from the remaining samples as the training set. Judging from these results, the Search Advisor manages to generate significant improvement in design search performance when compared to any of the traditional 'CDS'.

Finally, it should be noted that learning from previous use of search routines is, of course, nothing novel. It is just an analogue to designers' actions in deriving qualitative rules from their experiences on past designs and using this knowledge to aid them in making decisions on new designs. Here, the novelty lies in our attempt to model and improve this process using machine over human learning which is often tedious, biased and error-prone.

5. Conclusion

In this paper, we propose a DKB Search Advisor that supports design engineers in search activities. Results from a case study where the Search Advisor is applied to an existing engineering design problem domain are presented. From these results, we have shown that a DES supported by a DKB Search Advisor possesses the advantage of improving the design process in terms of both speed and design quality. The DKB Search Advisor also helps to reduce the reliance of design processes on optimization domain experts by ensuring that designers require minimum knowledge of optimization techniques. It helps eliminate any human biases such as favoritism and makes full use of the data generated by designers during design–evaluate–redesign studies, which are otherwise often discarded. The results presented also support the conclusions of many published papers that no single search technique is best throughout a given design domain.

Acknowledgements

This work was funded by the University Technology Partnership for Design, which is a collaboration between BAE SYSTEMS, ROLLS-ROYCE and the Universities of Cambridge, Sheffield and Southampton. The authors wish to thanks the anonymous referees for their valuable comments on the early versions of the paper.

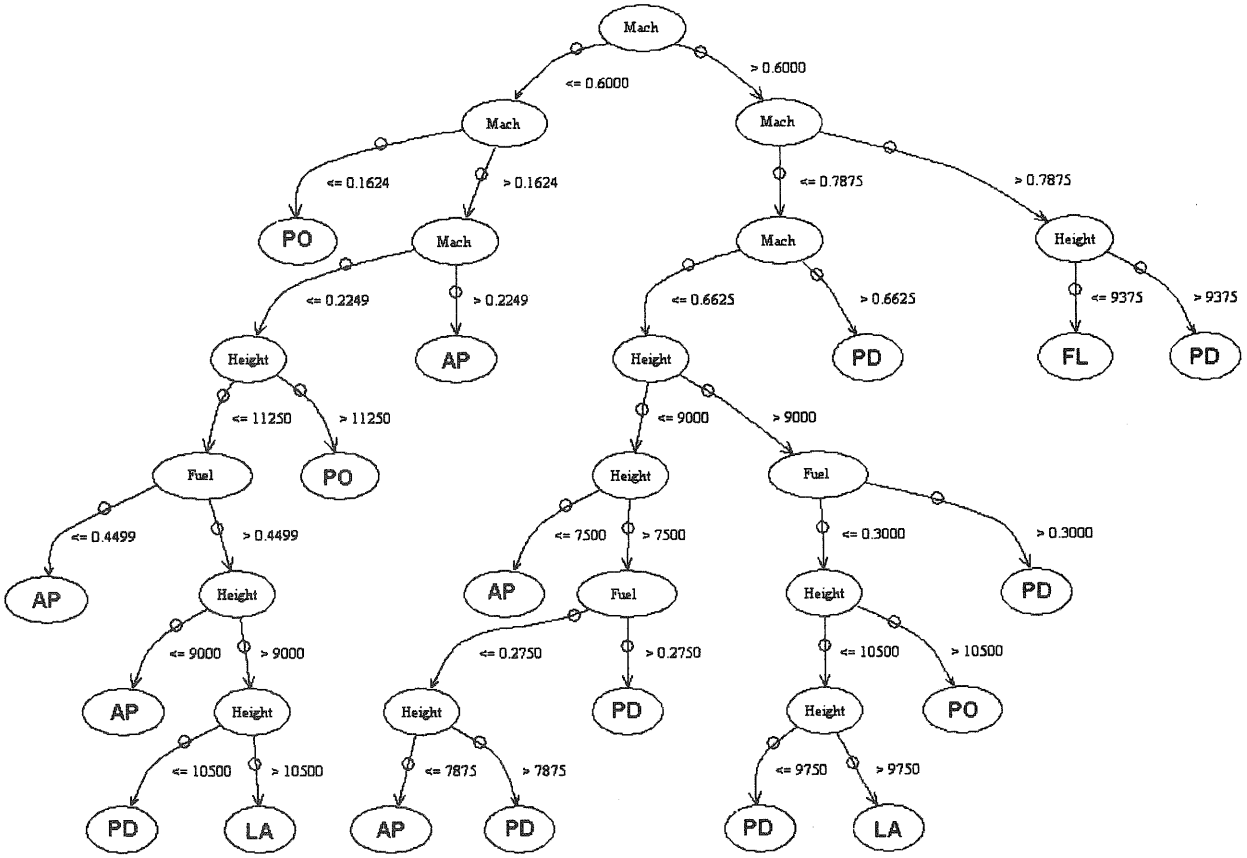


Fig. 5. The decision tree produced by C4.5 when trained on the BS dataset. Each design problem is defined by the values of mach number, fuel weight fraction and cruise height.

Table 3
Normalized performance measures of the design searches using the OPTIONS DES supported by the DKB Search Advisor validated on 243 unseen problems from the domain, in comparison with the respective 'Common Designers Strategies' CDS 1–5

Strategy for conducting search design	Estimated search performance on the aircraft wing design domain		
	Best-Speed model	Best-Quality model	Best-Overall model
CDS1	0.1007	0.4199	0.2496
CDS2	0.1351	0.5166	0.3155
CDS3 (EP)	0.0981	0.6782	0.3882
CDS4 (GA)	0.0981	0.6740	0.3861
CDS5 (AP)	0.5908	0.2978	0.4443
Domain knowledge-based search advisor	0.9447	0.7017	0.7191

The nearer the value is to 1, the better is the performances of each strategy in conducting design search.

References

Aha, D.W., 1992. Tolerating noisy, irrelevant and novel attributes in instance-based learning algorithms. International Journal of Man–Machine Studies 36 (1), 267–287.

Bramlette, M., Cusic, R., 1989. A comparative evaluation of search methods applied to parametric design of aircraft. In: Proceedings of the Third International Conference on Genetic Algorithms. Morgan Kaufmann, San Mateo, CA.

Deerwester, S., Dumais, S.T., Furnas, G.W., Landauer, T.K., Harshman, R., 1990. Indexing by latent semantic analysis. Journal of the American Society for Information Science 41 (6), 391–407.

EAAI, 1996. Special section on AI in design applications. Engineering Application of Artificial Intelligence 9 (4).

Frawley, W.J., Pietetsky-Shapiro, G., Matheus, C.J., 1991. Knowledge discovery in databases: an overview. In: Piatetsky-Shapiro, G., Frawley, W.J. (Eds.), Knowledge Discovery in Database. MIT Press, Cambridge, MA, pp. 1–30.

Greiner, R., Jurisca, I., 1992. A statistical approach to solving the EBL utility problem. In: Proceedings of National Conference Artificial Intelligence (AAAI).

Holte, R.C., 1993. Very simple classification rules perform well on most commonly used datasets. Machine Learning 11, 63–90.

Houstis, E.N., Rice, J.R., Ramakrishnan, N., Drashansky, T., Weerawarana, S., Houstis, C.E., 1998. MultiDisciplinary PSEs

for Computational Science. In: Selkowits, M. (Ed.), Advances in Computers (The Engineering of Large Systems), Vol. 46. Academic Press, New York, pp. 401–438.

Isight, 2000. Engineous Software.http://www.engineous.com/isight.html.

Jameson, A., 1999. Re-engineering the design process through computation. Journal of Aircraft 36 (1), 36–50.

KBSI, 1999. An Intelligent Assistant for Optimization Modelling. Available: http://www.kbsi.com/Research/PlanningScheduling/oma.html.

Keane, A.J., 1995. The OPTIONS Design Exploration System, http://www.soton.ac.uk/~ajk/options /welcome.html.

Keane, A.J., Nair, P.B., 2001. Problem Solving Environments in Aerospace Design. J. Advances in Engineering Software 32 (6), 477–487.

Keane, A.J., Petruzzelli, N., 2000. Aircraft wing design using GA-based multi-level strategies. Proceedings of the Eighth AIAA/USAF/NASSA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, A.I.A.A., Long Beach, 2000.

Kirkpatrick, S., Gelatt, C., Vecchi, M., 1983. Optimization by simulated annealing. Science 220, 671–680.

Laird, J., Newell, A., Rosenbloom, P., 1985. Soar: an architecture for general intelligence. Artificial Intelligence 33 (1), 1–64.

Langley, P., Sage, S., 1994. Induction of selective Bayesian classifiers. In: Proceedings of the Tenth Conference on Uncertainty in Artificial Intelligence. Morgan Kaufmann, Seattle, WA, pp. 399–406.

Lawrence, C.T., Tits, A.L., 1996. Nonlinear equality constraints in feasible sequential quadratic programming. Optimization Methods and Software 6, 265–282.

McKay, M.D., Beckman, R.J., Conover, W.J., 1979. A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. Technometrics 21 (2), 239–245.

Piatetsky-Shapiro, G., 1989. Workshop on knowledge discovery in real databases. In: International Joint Conference of Artificial Intelligence.

Paass, G., 1992. Probabilistic reasoning and probabilistic neural networks. International Journal of Intelligent Systems 7, 47–59.

Quinlan, J.R., 1993. C4.5: programs for machine learning. Morgan Kaufmann, Los Altos, CA.

Rasheed, K., Hirsh, H., 1997. Using case-based learning to improve genetic-algorithm-based design optimization. In: Proceedings of the Seventh International Conference on Genetic Algorithms. Morgan Kaufmann, Los Altos, CA.

Reich, Y., 1996. Modeling engineering information with machine learning artificial intelligence for engineering design. Analysis and Manufacturing 10 (2), 171–174.

Rentema, D., Jansen, F., Netten, B., Vingerhoeds, R., 1997. An AI-Based Support Tool for the Conceptual Design of Aircraft. Computer Aided Conceptual Design, Lancaster International Workshop on Engineering Design, pp. 47–56.

Sandgren, E., 1977. The utility of nonlinear programming algorithms. Ph.D. Thesis, Technical Report, Purdue University.

Schwefel, H.P., 1995. Evolution and Optimum Seeking. Wiley, New York.

Siddall, J.N., 1982. Optimal Engineering Design: Principles and Applications. Marcel Dekker, New York.

Sobieszczanski-Sobieski, J., Haftka, R.T., 1996. Multidisciplinary aerospace design optimization: Survey of recent developments. In: 34th AIAA Aerospace Sciences Meeting and Exhibit, Reno, NV, AIAA-96-0711.

Surma, J., Braunschweig, B., 1996. Case-base retrieval in process engineering: supporting design by reusing flowsheets. Engineering Application of Artificial Intelligence 9 (4), 385–391.

Wolberg, W.H., Street, W.N., Mangasarian, O.L., 1994. Machine learning techniques to diagnose breast cancer from fine-needle aspirates. Cancer Letters 77, 163–171.

Yin, X., Germay, N., 1993. A fast genetic algorithm with sharing scheme using cluster methods in multimodal function optimization. In: Albrecht, R.F., Reeves, C.R., Steele, N.C. (Eds.), Proceedings of the International Conference on Artificial Neural Nets and Genetic Algorithms. Springer, Innsbruck, pp. 450–457.

Y.S. Ong is currently with the Computational Engineering and Design Centre, attached to the BAE SYSTEMS/ROLLS-ROYCE University Technology Partnership for Design, in the School of Engineering Sciences, at the University of Southampton. He received his Bachelor and Master Degrees from Nanyang Technology University (Singapore). His research interests include Machine Learning, Evolutionary Optimization, Information Technology and Engineering Design.

A.J. Keane is a professor in the School of Engineering Sciences, at the University of Southampton. He is the Director of the Southampton Regional E-Science Centre, the Computational Engineering and Design Centre (CEDC), and the Southampton arm of the BAE SYSTEMS/ROLLS-ROYCE University Technology Partnership for Design. His research interests lie in two main areas, Evolutionary Optimization and Structural Dynamics.