Automation in Construction 71 (2016) 226–241

Contents lists available at ScienceDirect

Automation in Construction

journal homepage: www.elsevier.com/locate/autcon

A multi-objective GA-based optimisation for holistic Manufacturing, transportation and Assembly of precast construction

B. Anvari*, P. Angeloudis, W.Y. Ochieng

Centre for Transport Studies, Department of Civil & Environmental Eng., Imperial College London, SW7 2BU, United Kingdom

ARTICLE INFO

Article history:
Received 29 July 2015
Received in revised form 12 May 2016
Accepted 12 August 2016
Available online 21 August 2016

Keywords:
Extended flexible job shop modelling
Genetic algorithm
Precast construction

ABSTRACT

Resource scheduling of construction proposals allows project managers to assess resource requirements, provide costs and analyse potential delays. The Manufacturing, transportation and Assembly (MtA) sectors of precast construction projects are strongly linked, but considered separately during the scheduling phase. However, it is important to evaluate the cost and time impacts of consequential decisions from manufacturing up to assembly. In this paper, a multi-objective Genetic Algorithm-based (GA-based) searching technique is proposed to solve unified MtA resource scheduling problems (which are equivalent to extended Flexible Job Shop Scheduling Problems). To the best of the authors' knowledge, this is the first time that a GA-based optimisation approach is applied to a holistic MtA problem with the aim of minimising time and cost while maximising safety. The model is evaluated and compared to other exact and non-exact models using instances from the literature and scenarios inspired from real precast constructions.

1. Introduction

Prefabrication has been around for many decades, even centuries in the US and many European countries. However, its concept and construction practices are evolving. The Renaissance architecture and master builder Andrea Palladio standardised and prefabricated columns and stairs because of the growing demand for palaces and villas of the same style [1]. Prefabrication was then used in Europe for replacing houses that were destroyed in World War I. After World War II, there was a need for rapid and low-cost prefabricated housing for military personnel in the US [2]. Thus, there has been a continual need for prefabrication around the world for centuries, but the need is ever changing with the time and new technologies. Design for Manufacturing and Assembly (DfMA) is a simultaneous design and engineering approach where construction components are manufactured and (sub-)assembled in a factory or warehouse, before being delivered to a construction site for installation. DfMA makes use of prefabrication techniques in order to utilise construction schedule, cost, workforce, safety and quality. When optimising prefabrication, it is crucial how a project is divided into smaller parts such as a manufacturing line or an assembly line. By then combining smaller parts, larger elements can be incorporated into the building system. Haas [3] identified the driving factors being cost and schedule for adopting prefabrication in industrial projects as the most critical factors (see Fig. 1a). The results also show that DfMA techniques have a significant positive impact on safety, quality and efficiency at every stage of the project. The time and cost savings due to prefabrication is reported as 66% and 65% in American projects as shown in Fig. 1b and c.

DfMA might not always be a better choice than conventional construction, i.e., transporting structural materials to the building site and assembling on-site [5]. For instance, considerable cost overruns and project management issues have been associated with prefabrication from manufacturing up to assembly. The decision on whether prefabricating components of a building or even an entire building is often based on subjective judgment rather than a thorough analysis of consequential decisions in the MtA sectors [6,7].

Scheduling in a precast construction project is a temporary execution plan of a DfMA proposal. A project schedule reports on the time and order, in which tasks need to take place, and their allocated resources. It reflects required costs and resources to deliver the project; it can provide delay analysis to avoid exceeding the scope of the project or budgetary constraints. The schedule might highlight potential problems before they arise. Resource scheduling is an assignment problem and describes in detail when to accomplish tasks and how to utilise resources assuring the project's objectives. Scheduling requires selecting resource types (such as machineries, cranes, and workforce), determining the required number of each resource, and allocating them to simultaneously executed jobs (e.g., manufacturing a number of different components) over time

* Corresponding author.
E-mail address: b.anvari09@imperial.ac.uk (B. Anvari).

http://dx.doi.org/10.1016/j.autcon.2016.08.007
0926-5805/

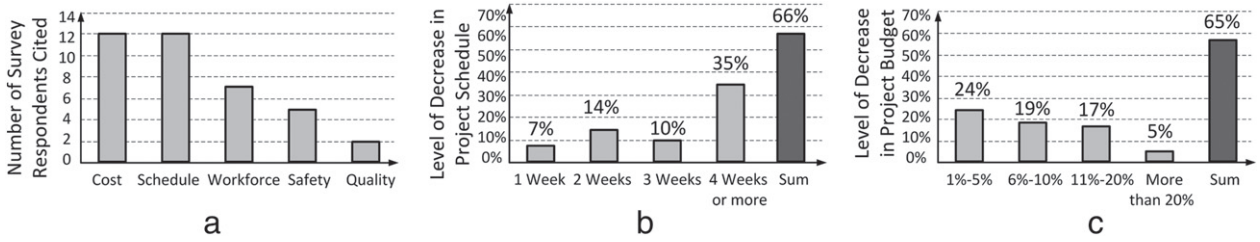


Fig. 1. (a) Comparison between the driving factors for prefabrication [3], (b) level of decrease in project schedule and (c) in project budget due to prefabrication in the US [4].

to maximise productivity subject to the constraints (e.g., limited number of workforce, early start dates, late finish date). A sufficient number and type of resources is crucial for managing demand fluctuation, procurement processes and machine failures. Each sector can process a limited number of tasks at a time; each resource can execute at most one task at a time. Resource scheduling affects and is affected by the manufacturing factory, transportation options and the construction site.

Nowadays, resource scheduling of a DfMA plan is not performed comprehensively: On the one hand, the available decision-making support tools do not cover the combined performance attributes of MtA sectors while evaluating prefabricated construction methods. On the other hand, they do not consider an optimal schedule for the combined MtA sectors before comparing conventional construction plan to prefabricated construction one. It is therefore essential to implement a decision support tool which considers all three MtA sectors as a unified system and acknowledges multiple objectives of the construction project. Hence, this paper proposes a Genetic Algorithm-based (GA-based) technique for solving a unified MtA construction system. The contributions are:

- The prefabrication scheduling problem has been considered as a holistic/unified MtA problem which needs to be solved.
- Due to the complexity of this unified MtA construction scheduling problem (which is equivalent to a complex extended Flexible Job Shop Scheduling Problem), a GA-based optimisation algorithm is applied for the first time. This non-exact model will return a good sub-optimal result in less time than exact methods.
- The presented GA-based technique is multi-objective with one dominant objective function.

To evaluate the quality of the proposed GA for solving flexible job shop problems, the model is compared with other exact and non-exact models using instances from the literature and scenarios inspired by real data from precast construction projects.

Section 2 provides a summary of the available prefabrication decision-making tools and construction scheduling models and identifies the issues of excising algorithms. The detailed description of the problem in the prefabricated construction, input/output decision variables, resource constrains, the optimisation objectives and the framework for resource scheduling a unified MtA system are described in Section 3. This leads to defining the MtA system in terms of a Resource-constrained Extended Flexible Job Shop Scheduling (REFJSS) problem with the aim of minimising the total completion time and cost. These types of problems are NP-complete problems and computationally demanding to solve [8]. Section 4 presents the MILP formulation for the REFJSS and the assumptions. Evolutionary algorithms such as the GA are suitable in finding a solution that is close to the optimal and satisfies the constraints of complex problems. In order to solve the REFJSS problem, a GA-based approach is presented in Section 5, along with a custom tool developed in C# that allows evaluating different prefabrication scenarios. The numerical results of the presented algorithm for different instances from real world scenarios and the literature are presented in Section 6. The general conclusions and future work are summarised in Section 7.

2. Background

Having presented the practical advantages of developing a decision support tool and optimising the schedule for a unified MtA system, Section 2.1 summarises the available decision-making tools for choosing a construction method. In Section 2.2, an overview of the current construction scheduling models is provided.

2.1. Decision-making tools for construction techniques

With regard to construction prefabrication, Murtaza et al. [9] developed the MOdulariz Decision EXpert (MODEX) system to help judging the feasibility and financial benefits of modular prefabrication for a power plant project. MODEX is based on a hybrid expert system, combining an Expert Decision System and a Decision Support System. It follows decision rules set by experts in its feasibility analysis and reports the cost of different degrees of prefabrication in the financial analysis. In the feasibility analysis, MODEX asks a user a series of qualitative questions regarding different factors that influence the prefabrication process. It then computes the total weighted feasibility value, applying preset relative weights, and compares this feasibility score to a pre-set threshold before making a recommendation. In the financial stage of the analysis, MODEX asks for the estimated project cost and schedule, and uses an analytical method to evaluate the cost and time savings associated with different levels of prefabrication. MODEX's recommendation making process is not transparent, however, and it is not clear how the total cost is distributed. In addition, MODEX's decision rules needs to be kept updated as relevant expertise evolves.

Murtaza and Fisher [10] developed a further model, called Neuromodex, for construction method decision-making processes. Neuromodex is based on a neural network system and uses different decision factors relating to a specific project (e.g. location, labour and environmental) as input values, forms a pattern, and then relate this input pattern with one of the output patterns (conventional, semi-prefabrication or prefabrication). In order to recognize the input patterns and produce rational and effective decisions, the neural network needs to be trained based on past modularization decisions. Neuromodex uses MODEX for this training process, with the assumption that past principles using prefabrication were correct.

Song et al. [5] also presented a decision-making framework and a computerized tool to validate the applicability of prefabrication methods in industrial projects. Their decision framework has three levels (strategic level 1, strategic 2 and tactical level). The first two

levels are considered to evaluate the feasibility of Prefabrication, Preassembly, Modularization and Off-site Fabrication (PPMOF) based on primary drivers and impediments (such as schedule, site attributes, availability of local labour and suppliers). The third level is designed to determine the cost benefits and the practicality of PPMOF. This tool provides a tactical analysis of the alternatives, and the weight given to each set is subjective.

Regarding prefabrication, the Interactive Method for Measuring PRE-assembly and Standardisation (IMMPREST) toolkit was developed by Loughborough University (UK) to compare the traditional construction with the prefabricated construction [6]. This decision-making tool in fact consists of three parts (A, B and C). Part ' A' is designed to make the toolkit user friendly; part ' B' is focused on project goals and constraints in order to guide a strategic argument on prefabrication; and tool ' C' is considered to evaluate the relevant factors for prefabrication in more depth. Although IMMPREST contains an inclusive comparison between traditional and prefabrication methods, the major challenge is not having sufficient information available at the start of a project to use the toolkit.

Soetanto et al. [11] implemented a framework for selecting the structural frame of buildings. This framework requires the project members to determine evaluate seven criteria in relation to client and project objectives (e.g. physical form and space, construction process, long-term sustainability). The performance of various structural frame options in respect to the defined criteria needs to be stated, regardless of their importance. This framework is used to calculate a Performance Weighted Score (PWS) which presents the likelihood of achieving client objectives. Then, an overall PWS is stated for each structural frame option based on the seven criteria.

Luo [12] summarised a list of general prefabrication strategies, and developed a decision-making tool based on dynamic programming analysis. He evaluated prefabrication strategies based on the initial costs, schedule, quality and sustainability.

Chen et al. [13] developed a two-level Construction Method Selection Model (CMSM) to evaluate different construction methods under risk and uncertainty considerations. The Simple Multi-Attribute Rating Technique (SMART) is used at the strategic level for evaluating the feasibility of prefabrication based on the judgment of experts. The Multi-Attribute Utility Theory (MAUT), which associate attitude to uncertainty and risk, is then used at the tactical level to evaluate the appropriate level of prefabrication for the defined project according to the judgment of multi-decision makers. However, this decision support tool requires a lot of input from the decision makers and preference values need to be precise.

2.2. Scheduling models in construction applications

A schedule presents what work needs to be performed, which resources of the organization will perform the work and the time-frames in which that work needs to be performed. This consists of a list of multiple entities called jobs that need to be scheduled. Each job has an order of tasks, called operations, to go through and each operation takes a specific amount of time to finish using a particular resource. An operation is called the execution of a task by a resource. Each job can comprise of a single operation or a set of operations which must be done using shared resources. The shared resources are usually machineries, cranes, and workforce. In resource scheduling, two distinct decisions have to be made: the assignment of operations to resources (e.g. machineries, cranes, and workforce) while sharing resources, and the sequencing and timing of operations. Resource scheduling has been studied broadly because of its practical application in different fields such as production line [14,15], vehicle and crew scheduling in transit systems [16] and assembly line [17–21]. Job shop scheduling, flow shop scheduling, and flexible job shop scheduling are popularly used to model the rules which govern the MtA sectors separately. The classical Job Shop Scheduling (JSS) deals with sequencing operations of jobs on predefined resources with the aim of minimising the makespan. The JSS problems are known as one of the hardest combinational optimisation problems since resource orderings can be different for each job. In the Flow Shop Scheduling (FSS), the operation order on resources is the same for all jobs. For instance, a production-line for double-curved precast concrete panels is a job with eight operations in the following order: mould assembly, embedded placing, reinforcement fixing, concrete casting, cleaning/finishing, mould stripping, concrete curing and handling. The order of operations in this production-line is the same (fixed) for all jobs (e.g. precasting concrete panels or walls). However, the execution of each operation may require different resources in each job and certain resources may need to be shared between a number of operations in a factory. The processing time of an operation also varies using different resources. The Flexible Job Shop Scheduling (FJSS) problem is a modified version of the JSS problem where the fixed operation sequences can be processed by alternative identical or non-identical resources in parallel and not by predefined resources. A non-identical resource is the one which has the flexible capability to be set up to process more than one type of operation. The FJSS is to assign each operation to an identical or non-identical resource out of set of resources capable of performing it (a routing problem), and to sequence the job operations in order to obtain a feasible schedule which satisfies one or multiple objectives (a scheduling problem). An extension of FJSS problems allows having a set of operations with arbitrary precedence relations [22] which is similar to real problems in the precast construction.

The JSS and the FJSS are NP-complete problems [8,23,24], they are computationally demanding to solve. The approaches for solving FJSS problems are classified into two main categories: a hierarchical approach and an integrated (concurrent) approach [25]. In the former, operations are assigned to their respective resources first and after that the scheduling procedures starts. Whereas in an integrated approach the assigning and scheduling processes are made concurrently. Integrated methods are more complex and difficult to solve but can produce better results. There are two well-known classes of solution methodologies to tackle FJSS problems and fulfill the scheduling requirements of the industrial projects: mathematical programming (e.g. Mixed Integer Linear Programming (MILP)) and hybrid meta-heuristics (e.g. evolutionary algorithms).

Different MILP models for solving the FJSS and its extension is explored in many studies [22,26-28]. A MILP can find the exact solution, however, they are computationally time consuming. Evolutionary algorithms, first defined by Rosenberg [29], are optimisation methods which search scenarios iteratively over time and uses different strategies or multiple searching points to explore various solutions in order to find a non-exact optimal solution. Evolutionary algorithms do not have knowledge of the specific problem; hence they investigate many possible solutions. One type of evolutionary algorithms is the Genetic Algorithm (GA) which is adopted in this study. In general, a GA is composed of an initial population, genetic operations (e.g. crossover and mutation), and an objective function [30–32]. GAs are briefly explained and crossover, mutation and migration operators are discussed in Section 5. GAs allow combining different strategies and exploring various solutions both in the initial solution phase and in the generation phase.

As the summary in Sections 2.1 and 2.2 shows, there is no decision support tool available which does resource scheduling for the MtA sectors as a unified system and allows for comparing the optimal schedule of multiple levels of prefabrication. Scheduling a unified MtA system presents additional challenges of size, which is larger than the capabilities of the existing algorithms. In this paper, an integrated MtA system is modelled as a REFJSS problem with arbitrary precedence relations. A multi-objective Genetic Algorithm (GA) is developed to solve the REFJSS problem with the objective of

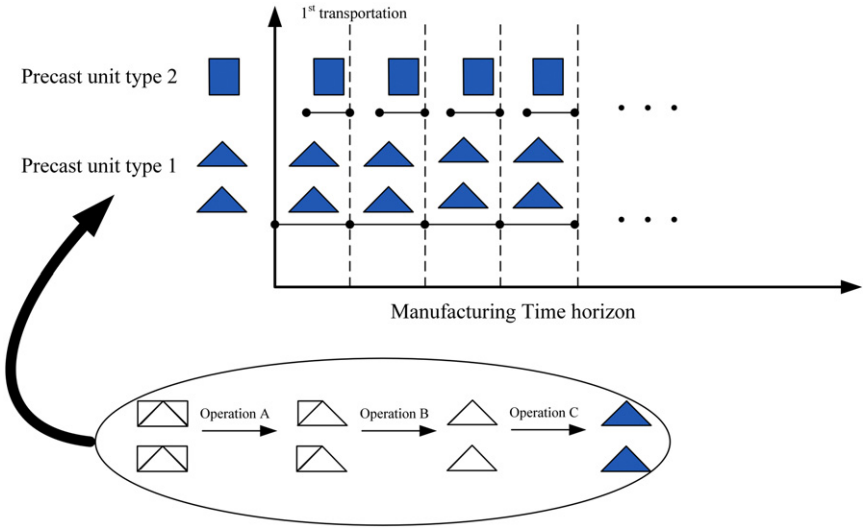


Fig. 2. An example of cyclic jobs in a manufacturing line.

minimising makespan and cost while maximising safety. Safety can be maximised by minimising the number of on-site workers on congested construction sites. The output of this GA-based REFJSS model provides an optimal allocation of resources on operations in a unified MtA system.

3. Problem description

In a unified MtA system of a prefabricated construction project, a number of different components are (semi-)prefabricated in a factory, transported to the site and assembled on-site according to the project's planning horizon *H*. These components are produced using a combination of resources. The production of the required number of each component is called a job. In the manufacturing industry, product orders are released in a cyclic manner (cyclic jobs) and delivered in batches according to the horizon of the project. An example is shown in Fig. 2 illustrating the prefabrication process for building a 3 m run of a precast component. This component is made of two precast units type 1 and one precast unit type 2 using steel moulds and prefabrication cages. Details of the manufacturing and assembly lines are shown in Tables 1 and 2. The manufacturing line is similar to a FSS problem; the assembly line is equivalent to a FJSS problem. A job is made of a set of tasks linked by precedence constraints and executed by a subset of resources. The set-up and processing times of each operation corresponding to each resource and the demand sizes for the project are stated in Tables 1 and 2.

In a unified MtA system, operations have arbitrary precedence relations which can be represented in a directed graph as illustrated in Fig. 3. Here, independent sequences of operations feed into an "assembling" operation, whereas a "disassembling" operation describes the process of any operation splitting into a number of mutually independent sequences [22]. This kind of problem is an extension of a FJSS problem with a set of operations with arbitrary precedence relations. Applying a GA-based optimisation method to this type of problem will return a good result quickly.

The multiple input, output, and optimisation process overview of a unified MtA system in a multi-objective DfMA project is summarised in Table 3. Scheduling a unified MtA system requires specifying the following inputs for different sectors (Manufacturing, transportation and Assembly) in a construction project:

- a list of product types, weights and quantities;
- a list of operations that have an effect on the overall project finish date;
- a list of available resources for each operation in the project;
- an estimation of completion time and cost for each operation using specific resources;
- the operation dependencies and strategies (e.g., predecessor or successor) in the MtA sectors for different product types and different levels of prefabrication.

In precast construction projects, the deliveries are based on the assembly strategies and the transportation options (road, rail and sea) considering the weight and size limitations. Late deliveries will cause penalties and early deliveries will contribute to holding costs. The project horizon *H* can be divided into short periods with a number of shipments from one sector to another (e.g., manufacturing sector to the transportation sector to the assembly line). Each prefabrication method may require a different set of resources as well as different delivery and assembly strategies. Thus, the precedence relation of the operations between the MtA sectors differs according to the selected prefabrication method.

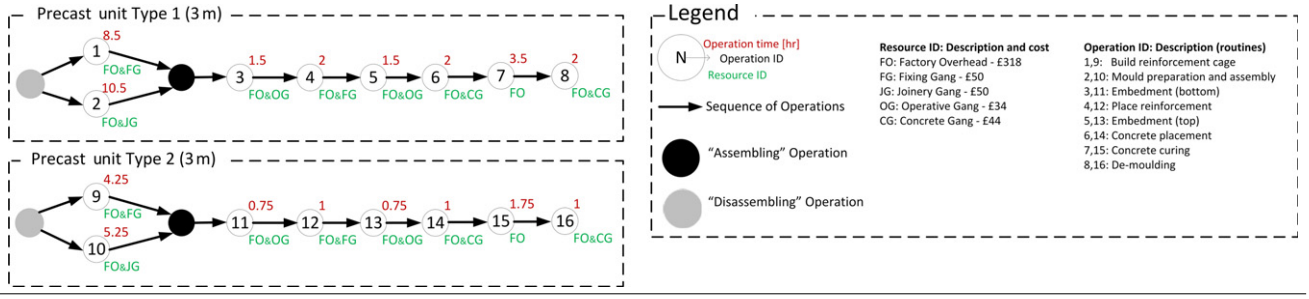
Considering a unified MtA system for optimisation, the following assumptions are made:

- All resources are available and can be set up to process more than one type of operation (non-identical resources).
- Operations can start at different times during the project's planning horizon.
- Setting up times of resources are considered.
- Resources and operations are independent from each other.
- The order of operations is predefined and fixed (precedence relations).
- A resource (or machinery) can execute one operation at a time (resources constraints).
- A started operation cannot be interrupted during its processing time on a given resource (or machinery). Thus, preemption of operations is not allowed.

Table 1
Manufacturing line for building a 3m run of a precast component formed from two precast unit type 1 and one precast unit type 2 using steel moulds and prefabrication cages.

Demand	Tasks	Operations	Resources				
			Factory Overhead (FO)	Fixing Gang (FG)	Joinery Gang (JG)	Operative Gang (OG)	Concrete Gang (CG)
			Processing time [hr]				
2	Precast unit type 1	1: Build reinforcement cage	8.5	8.5	–	–	–
		2: Mould preparation and assembly	10.5	–	10.5	–	–
		3: Embedments (bottom)	1.5	–	–	1.5	–
		4: Place reinforcement	2	2	–	–	–
		5: Embedments (top)	1.5	–	–	1.5	–
		6: Concrete placement	2	–	–	–	2
		7: Concrete curing	3.5	–	–	–	–
		8: De-moulding	2	–	–	–	2
Cost per hour			£318	£50	£50	£34	£44
1	Precast unit type 2	9: Build reinforcement cage	4.25	4.25	–	–	–
		10: Mould preparation and assembly	5.25	–	5.25	–	–
		11: Embedments (bottom)	0.75	–	–	0.75	–
		12: Place reinforcement	1	1	–	–	–
		13: Embedments (top)	0.75	–	–	0.75	–
		14: Concrete placement	1	–	–	–	1
		15: Concrete curing	1.75	–	–	–	–
		16: De-moulding	1	–	–	–	1
Cost per hour			£318	£50	£50	£34	£44

Operation dependencies for manufacturing two type of precast units

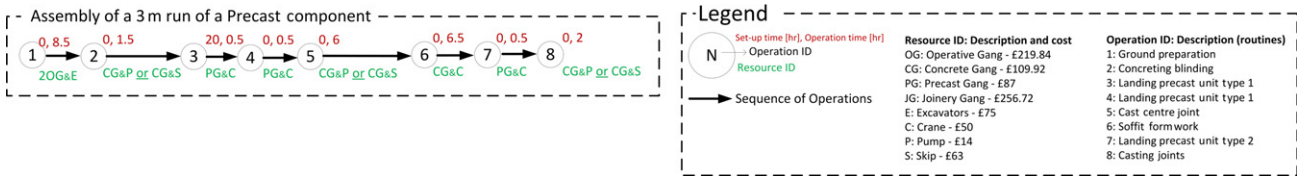


- A number of non-identical resources are available in all MtA sectors. These can be used simultaneously to process similar operations.
- Actual/fixed start dates, early start dates, and late finish dates are specified for all operations.
- Projects' due dates are identified.

Table 2
Assembly line for building a 3 m run of a precast component with the set-up and processing time possibilities.

Tasks	Operations	Resources						
		Operative Gangs (OGs)	Concrete Gang (CG)	Precast Gang (PG)	Joinery Gang (JG)	Excavators (E)	Crane (C) 1	Pump (P) OR Skip (S)
		(Set-up time [hr], processing time [hr])						
Precast component	1: Ground preparation	(0,8.5)	–	–	–	(0,8.5)	–	–
	2: Concreting blinding	–	(0,1.5)	–	–	–	–	(0,1.5) OR (0,1.5)
	3: Landing precast unit type 1	–	–	(0,0.5)	–	–	(20,0.5)	–
	4: Landing precast unit type 1	–	–	(0,0.5)	–	–	(0,0.5)	–
	5: Cast centre joint	–	(0,6)	–	–	–	–	(0,6) OR (0,6)
	6: Soffit form work	–	–	–	(0,6.5)	–	(0,6.5)	–
	7: Landing precast unit type 2	–	–	(0,0.5)	–	–	(0,0.5)	–
	8: Casting joints	–	(0,2)	–	–	–	–	(0,2) OR (0,2)
Cost per hour		£219.84	£109.92	£87	£256.72	£75	£50	£14 OR £63
On-off cost		£0	£0	£0	£0	£0	£20000	£151 OR 0

Operation dependencies for the assembling line



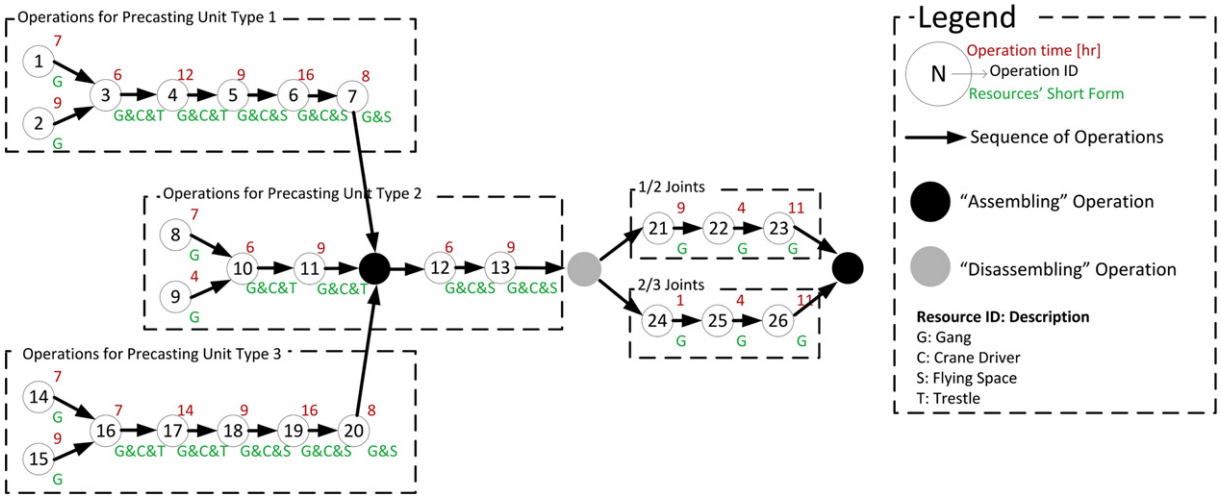


Fig. 3. A representation of operation dependencies in a unified MtA system.

- Any operation can be executed by a combination of more than one resource at a time.
- Any resource can process only one operation simultaneously.
- Product orders are released in the specified cyclic manner.

Based on these assumptions, the unified MtA system is considered to be a REFJSS problem and is modelled with the objectives of minimising time and cost while maximising safety. The mathematical description is presented in Section 4.

4. Problem formulation

A FJSS problem is considered consisting of a set of machines $M = M_1, M_2, \ldots, M_m$ available in the factory, where some machines might be identical or able to process the same operation but with a different processing time. The MtA line of each precast element consists of a chain of different operations O_h. Each operation can make use of a number of alternative machines. One machine can process only one operation at a time. The sequence of operations from one precast element to another can be the same with different processing times. The processing time P_i of each operation O_h performed by machine i is known.

The known cost of O_h by machine i is $Cost_{ih}$. Due to the size and weight of the precast elements, it is assumed that no buffer space for storing precast elements exists between operating machines (hold-while-wait constraint). The objective function can be described by minimising the makespan of the schedule f_1 and the total project cost f_2. The total project cost f_2 is calculated based on the process time Ps_{ih} of O_h on the selected machine i and the hourly rate of the machines $Cost_{ijh}$. The starting time of operation O_h is t_h. The number of operations assigned to machine i is k_i and the starting time of O_h performed by machine i in priority k is Tm_{ik}. The challenge is to determine both the assignment of machines and the sequence of operations on all the machines to minimise the objectives. Table 4 presents the notations, the MILP formulation for the generalised REFJSS and the assumptions.

The multi-objective assignment problem is optimised subject to a set of constraints. Constraint (1) determines the makespan. Constraint (2) estimates the processing time of O_{jh} on the selected machine i. Constraint (3) ensures a specified operation sequence. Each machine is only able to process one operation at a time (see constraint (4)). Constraints (5) and (6) ensure that each operation O_{jh} starts after its assigned machine is available and the previous operation O_{jh-1} is completed. Constraint (7) specifies the suitable

Table 3
Overview of multiple input and output for the optimisation process in the Manufacturing, transportation and Assembly (MtA) sectors.

Input	Optimisation process (REFJSS)	Output
Products: Types, weights, quantities	Input integration and model formulation	Scheduling: Costing the project
Operations in MtA sectors	Multi-objective Genetic Algorithm	Resource assignment Operation sequencing/planning
Required resources for MtA sectors for each operation	Evaluation	Timing the operations using the resources
Operation durations using different resources		
Operation costs using different resources		
Operation dependencies and strategies in MtA sectors		

Table 4
MILP formulation for the generalised REFJSS and the assumptions.

Indices	
m	Total number of independent machines
i	Machine index where $i = \{1, 2, \ldots, m\}$
n	Total number of independent operations
h	Operation index where $h = \{1, 2, \ldots, n\}$
k	A set of operations assigned to each machine where $k = \{1, 2, \ldots, k_i\}$
Sets	
M	A set of machines $M = \{M_1, M_2, \ldots, M_m\}$
Parameters	
k_i	The number of assigned operations to machine i
a_{ih}	Describes the capable machine set M_h for operation O_h where $a_{ih} \in \{0,1\} = \begin{cases} 1, \text{ if machine } i \text{ is capable of performing operation } O_h \\ 0, \text{ otherwise} \end{cases}$
p_{ih}	Processing time of O_h if performed on machine i
$Cost_{ih}$	Cost of O_h if performed on machine i
L	A large number
Decision variables	
$C_{\max}$	Maximum completion time of a schedule
y_{ih}	$y_{ih} \in \{0,1\} = \begin{cases} 1, \text{ if machine } i \text{ is selected for performing operation } O_h \\ 0, \text{ otherwise} \end{cases}$
x_{ihk}	$x_{ihk} \in \{0,1\} = \begin{cases} 1, \text{ if operation } O_h \text{ is performed on machine } i \text{ in priority } k \\ 0, \text{ otherwise} \end{cases}$
t_h	The start time for processing O_h
Tm_{ik}	The start of working time for machine i in priority k
k_i	The number of assigned operations to machine i
Ps_{jh}	Processing time of operation O_{jh} after selecting a machine
Objective	
Minimize $(f_1 = C_{\max}, \quad f_2 = \sum_i y_{ih} \times p_{ih} \times Cost_{ih})$	
Subject to	
(1)	$C_{\max} \geq t_h + Ps_h$
(2)	$\sum_i y_{ih} \times p_{ih} = Ps_h$
(3)	$t_h + Ps_h \leq t_{h+1}$ where $h = \{1, 2, \ldots, n-1\}$
(4)	$Tm_{ik} + Ps_h \times x_{ihk} \leq Tm_{ik+1}$ where $k = \{1, 2, \ldots, k_i - 1\}$
(5)	$Tm_{ik} \leq t_h + (1 - x_{ihk}) \times L$
(6)	$Tm_{ik} + (1 - x_{ihk}) \times L \geq t_h + Ps_h$
(7)	$y_{ih} \leq a_{ih}$
(8)	$\sum_j \sum_h x_{ihk} = 1$
(9)	$\sum_i y_{ih} = 1$
(10)	$\sum_k x_{ihk} = y_{ih}$
(11)	$t_h \geq 0$
(12)	$Ps_h \geq 0$
(13)	$Tm_{ik} \geq 0$

machines for each operation. Constraint (8) assigns each operation to machines and identifies the sequence of operations on the machines. Each operation can be performed on one machine only with the priorities defined in constraints (8), (9), and (10). As the FJSS problem is NP-hard, solving the above model, which is an extended FJSS problem, is also NP-hard. In the next section, a multi-objective GA is presented for solving the problem efficiently.

5. Genetic algorithm for REFJSS

A GA mimics the natural process of evolution over a period of time. It uses string coding of variables (chromosome encoding), randomly generates a set of possible solutions (initial population) to the problem, ranks possible solutions of the population (generation) after calculating the fitness function for each one (chromosome evaluation), keeps the best solutions and uses these to generate new possible solutions (genetic operators). A GA iteratively applies genetic operators (such as crossovers and mutations) to change the current population to a new population. By repeating chromosome evaluation and applying genetic operators, either an acceptable solution will be found or the GA will iterate a given number of cycles.

The critical elements of GAs are the chromosome definition, the design of genetic operators, and the mechanism for population management to decide which chromosomes are selected in each population for applying the genetic operators. Also, a fitness function is defined in GAs to measure the quality of each chromosome. After randomly choosing two chromosomes (parent set), a crossover operator (or marriage) combines the genes of the parent set to generate off-springs. The chromosomes with lower fitness value are then replaced with off-springs generated from crossover of more fit chromosomes. To avoid local optima during the search process, a mutation operator makes a perturbation to the genes of off-springs. The chromosome encoding and decoding method, objective function, initial population generation, genetic operators of the proposed GA for solving REFJSS problems are presented in Sections 5.1, 5.2, 5.3 and 5.4, respectively.

$$\Big(\overbrace{(O_{16},O_{17},\ldots,O_{13})}^{C},\overbrace{(O_{14},O_{1},\ldots,O_{23})}^{G[1]},\overbrace{(O_{9},O_{15},\ldots,O_{26})}^{G[2]},\overbrace{(O_{18},O_{19},O_{5},O_{6},O_{20},O_{7},O_{12},O_{13})}^{S},\overbrace{(O_{16},O_{17},O_{3},O_{4},O_{10},O_{11})}^{T}\Big)$$

Fig. 4. The chromosome structure for the assignment and sequencing of the initial solution for example in Fig. 3.

5.1. Chromosome encoding and decoding

A new modified encoding scheme based on the work by Falkenauer and Bouffoix [33] is described and utilised in this paper. A chromosome/solution is composed of a number of sub-strings corresponding to the number of resources. Each sub-string represents the sequence of the operations processed on a resource. Fig. 3 shows an example of a unified MtA system which is an extended FJSS problem with 26 operations and 4 non-identical resources (G, C, S, and T). Two Gangs (G) are available during the horizon of this project. Some operations require a combination of different resources to be executed. For instance, operation ID 1 can be executed using one Gang (G) while operation ID 3 needs to be executed using three resources (G, C, and T) at the same time. The processing time of each operation corresponding to the resources is presented in Fig. 3. The chromosome structure for the assignment and sequencing of the initial solution is illustrated in Fig. 4. The length of the chromosome is the sum of assigned operations to all resources/machines ($\sum_{i=1}^{m} k_i$). In this example, the length of the chromosome is 52.

In the presentation of Fig. 4, a chromosome is composed of five sub-strings, one for each resource (C,G[1],G[2],S,T) - and a sub-string ($O_{16}, O_{17}, O_3, O_4, O_{10}, O_{11}$) represents the sequences of the operations

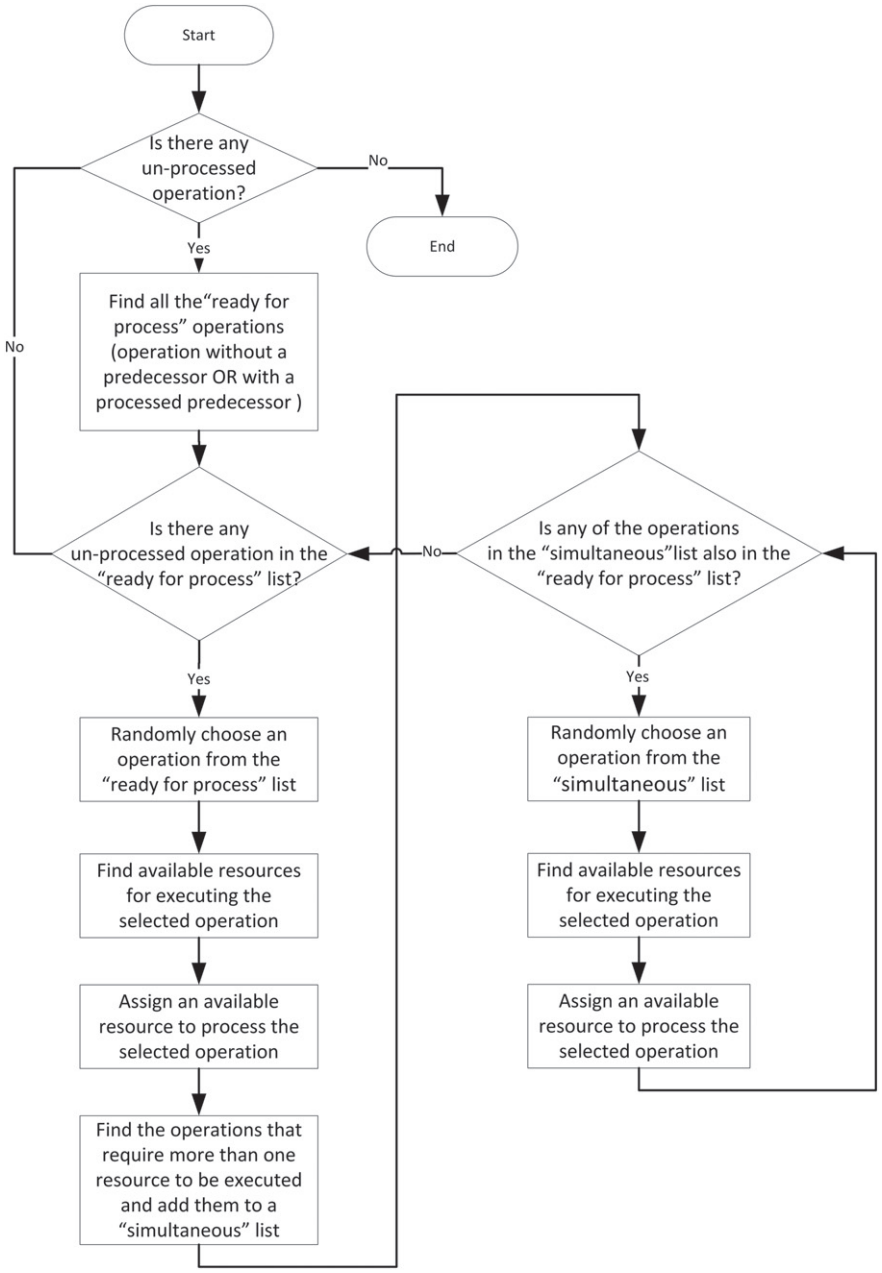


Fig. 5. Chromosome encoding procedure.

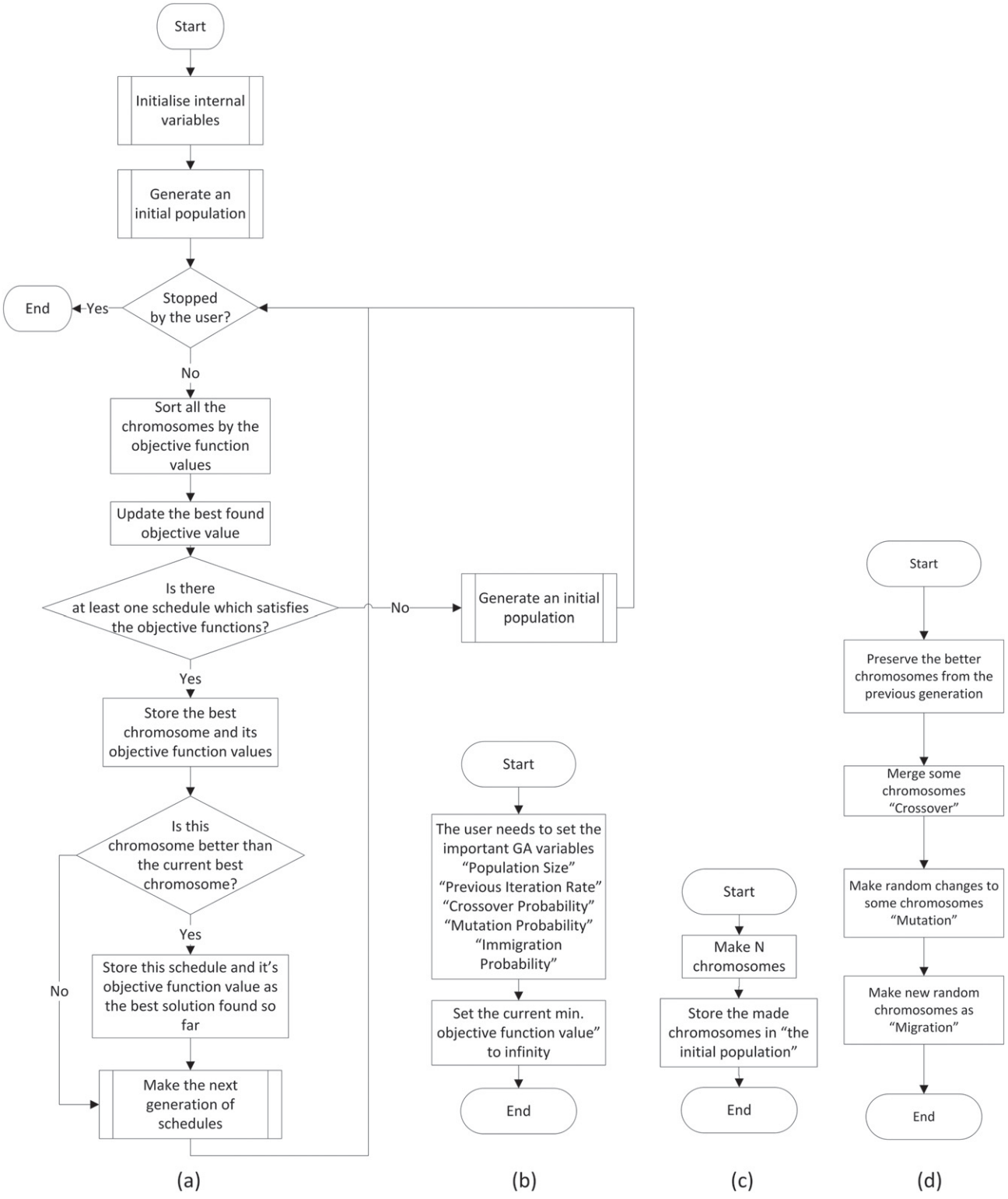


Fig. 6. The GA assignment and sequencing procedures: (a) overall algorithm, (b) parameter initialisation, (c) generating the initial population and (d) sub-sequential iteration algorithm.

processed on resource T. A feasible solution must meet the precedence relations shown in Fig. 3. Thus, operation 3 must be processed before operation 4, operation 10 must be processes before operation 11, and operation 16 must be processes before operation 17. All predecessor operations of each operation have to be completed before this operation can start. Any solution that does not satisfy the predecessor-successor relations is not feasible. Unlike the method proposed by Falkenauer and Bouffoix [33], the assignment of operations to resources is based on the precedence relations for generating the initial population. Hence, only feasible schedules are produced for the initial population. As shown in Fig. 5, operations without any predecessors or with completed predecessors are added to a list

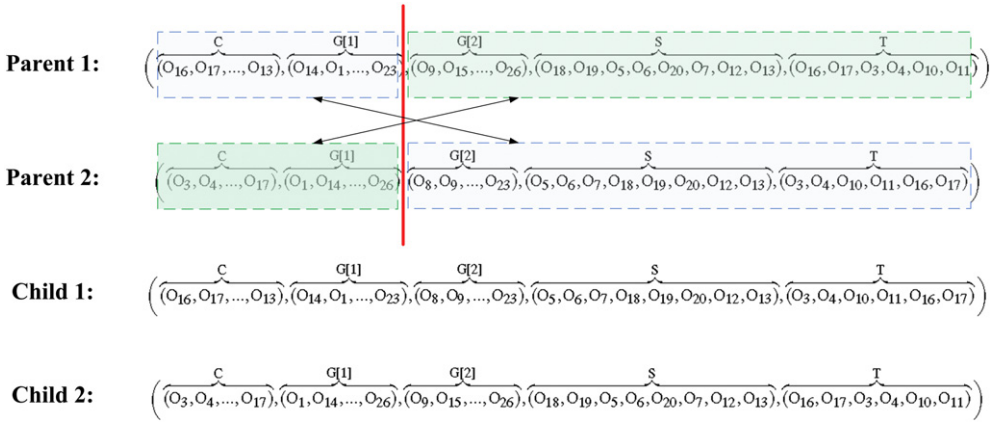


Fig. 7. Crossover operator.

called "ready for process". An operation is randomly selected from the "ready for process" operation list for resource assignment. The available resources that are suitable to execute the selected operation are identified with one of them being assigned. Operations that require more than one resource are added to a "simultaneous" list. The same sequential steps applied to the "ready for process" operation list are followed here until all the operations are assigned to their required set of resources.

The decoding method by Falkenauer and Bouffoix [33] is adopted to achieve feasible solutions. In this method, operations are selected based on their sequence in each sub-string and scheduled sequentially at the earliest possible time by considering the precedence relations. The ones that do not satisfy the precedence relations are scheduled at a later time that satisfies the predecessor constraints. The scheduling process also considers actual/fixed start dates, early start dates, late finish dates, and a maximum number of on-site workers per day. In this way, infeasible chromosomes are changed to feasible ones in the decoding process and, therefore, valid solutions are created. It is evident that the operations which require more than one resource can only be processed when all the required resources are available.

5.2. Objective functions

The evaluation criteria for prefabrication are cost, time and safety. The dominate objective function is minimising the completion time of a project f_1. The second objective function is minimising the total project cost f_2 while utilising resource allocation. The number of workers on the construction site has a significant impact on safety in precast construction projects as mentioned earlier. Hence, the number of on-site workers per day is constrained. The two objective functions f_1 and f_2 are computed for all chromosomes in each generation. The chromosomes are then ranked according to f_1. Solutions with identical completion times are ordered according to their total project cost f_2.

5.3. Generating initial population subject to objective functions

The overall structure of the proposed GA is shown in Fig. 6. Firstly, an initial population is generated of N feasible chromosomes. These chromosomes are then ranked according to the evaluation criteria (see Section 5.2). A number of the fittest chromosomes are preserved and transferred into the next generation. The preserved chromosomes remain eligible for selection as parents when breeding the remainder of the next generation. At each step, the crossover and mutation operators define new chromosomes by preserving the assignment property of the parent chromosome/s and changing the sequence of operations in the set of operations assigned to resources (see Figs. 7 and 8).

5.4. Genetic operators

As mentioned, a number of genetic operators such as the crossover, mutation and migration operators are applied here. The resource based crossover operator proposed by Qing-dao-er-ji and Wang [34] is used in this paper. An example is given in Fig. 7: Suppose parent 1 and parent 2 are selected to create two offsprings, the crossover operator randomly divides the sub-strings of the parents into two resource based sets and swaps the genes in the sub-strings of each set. Hence, the children inherit the sequence of operations processed on resources from their parents. After applying the crossover operator to parent 1 and parent 2, it can be seen in Fig. 7 that child 1 inherits the operation sets (with the same order) of resource C and resource G[1] from parent 1 while resource G[2], resource S and resource T obtained the operation sets (with the same order) from parent 2. The symmetric process is applied to create child 2. The new children that are created using the crossover operator will be then decoded to get a feasible schedule.

The mutation operator by Qing-dao-er-ji and Wang [34] is adopted to get a feasible schedule. First, a single chromosome is chosen to create a new offspring. Then, the mutation operator randomly

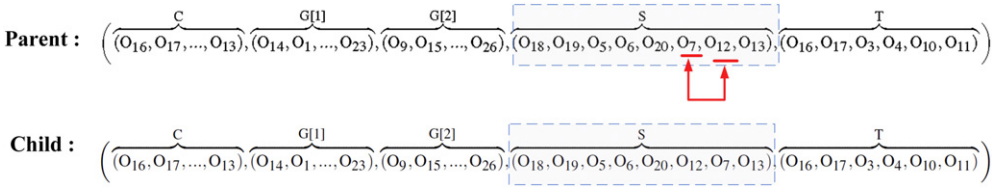


Fig. 8. Mutation operator.

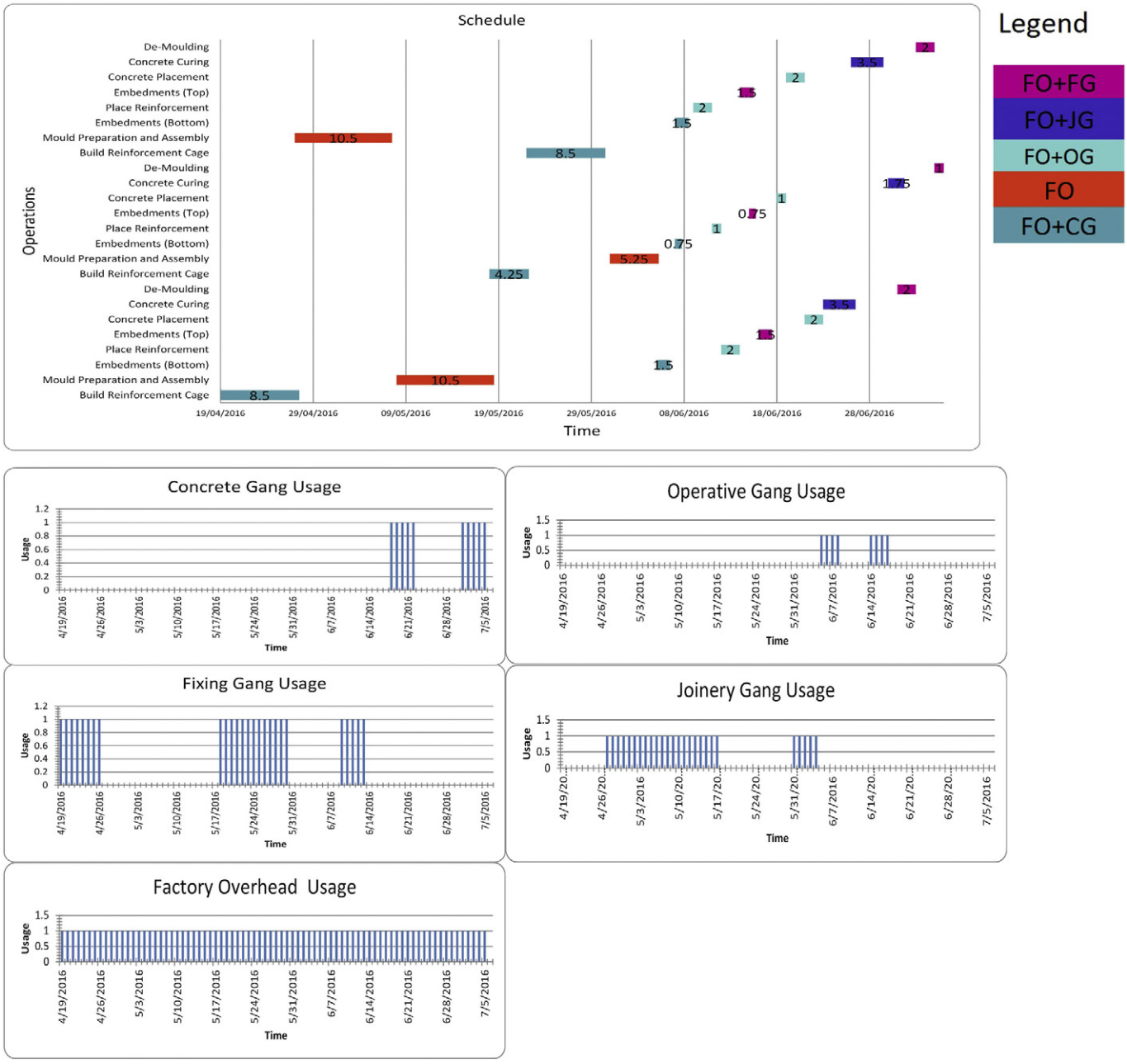


Fig. 9. Schedule obtained for the manufacturing line in Table 1 using the proposed GA-based algorithm.

selects a sub-string from the selected chromosome and changes the position of two operations within the sub-string of the parent chromosome considering the precedence relations (see Fig. 8). This operator preserves the assignment property of the parent chromosome and changes the sequence of operations. The new child will be then decoded to get a feasible schedule.

The migration operator generates a number of new chromosomes as explained in Section 5.3.

The genetic operators can be stopped by the user, if an acceptable solution which satisfies the total project's time and the project's cost is found. In this case, the corresponding schedule including the utilised number of on-site workers over time are reported as the output. Otherwise, the algorithm will continue searching for a better solution.

6. Case studies

To assess the performance of the developed GA, a solution for a number of precast construction scenarios has been determined. Our GA-based algorithm is applied to

1. a FSS and FJSS problem presented in Tables 1 and 2: The FSS problem consists of two cyclic jobs using 5 machines (FO, FG, JG, OG, and CG) and 8 operations for each job. The FJSS problem is made of one job using 9 machines (2 × OG, CG, PG, JG, E, C, P, and S) and 8 operations.
2. an extended FJSS problem shown in Fig. 3: This scenario is made of 5 machines (2 × G, C, S, and T) and 26 operations.

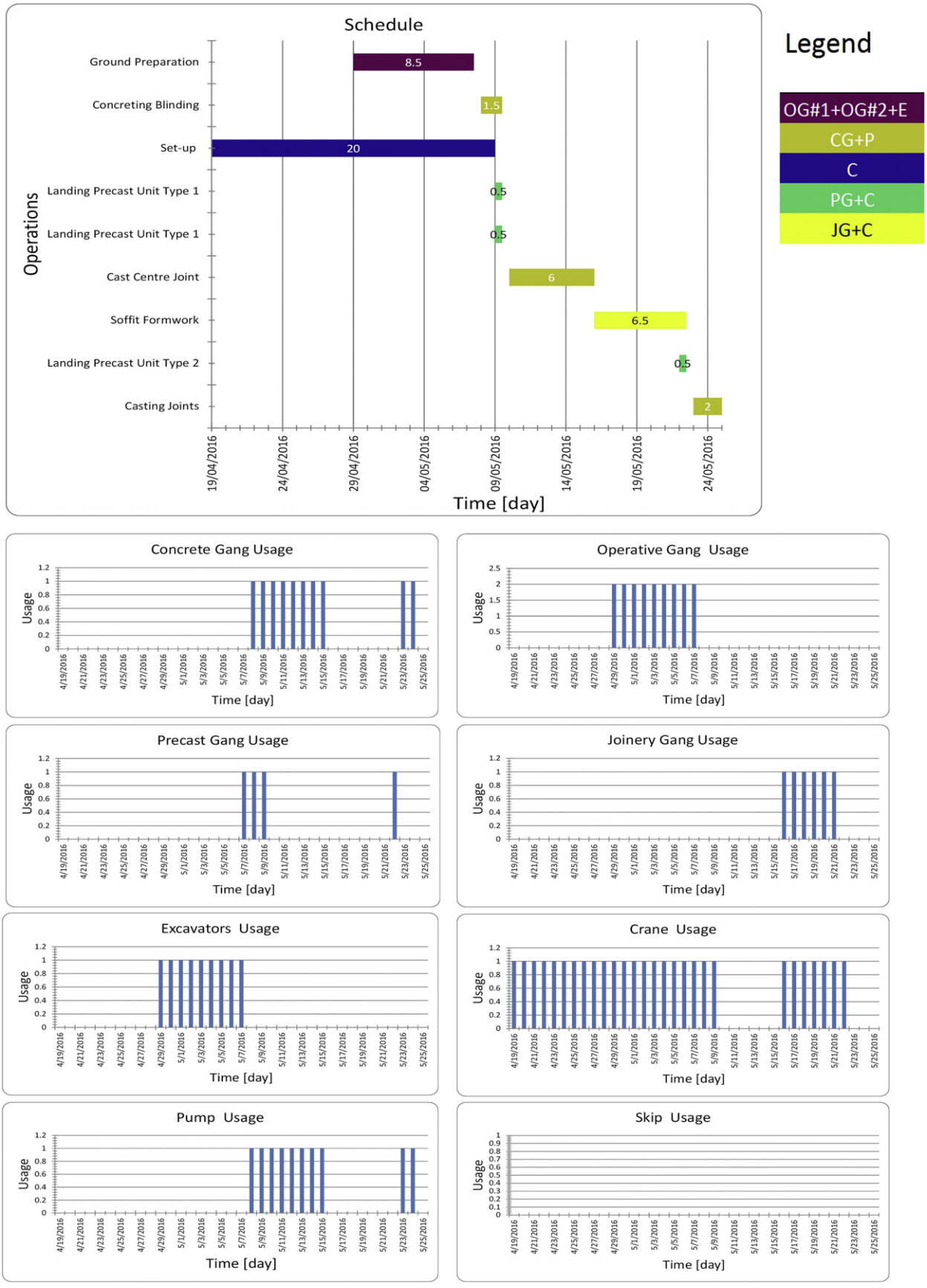


Fig. 10. Schedule obtained for the assembly line in Table 2 using the proposed GA-based algorithm.

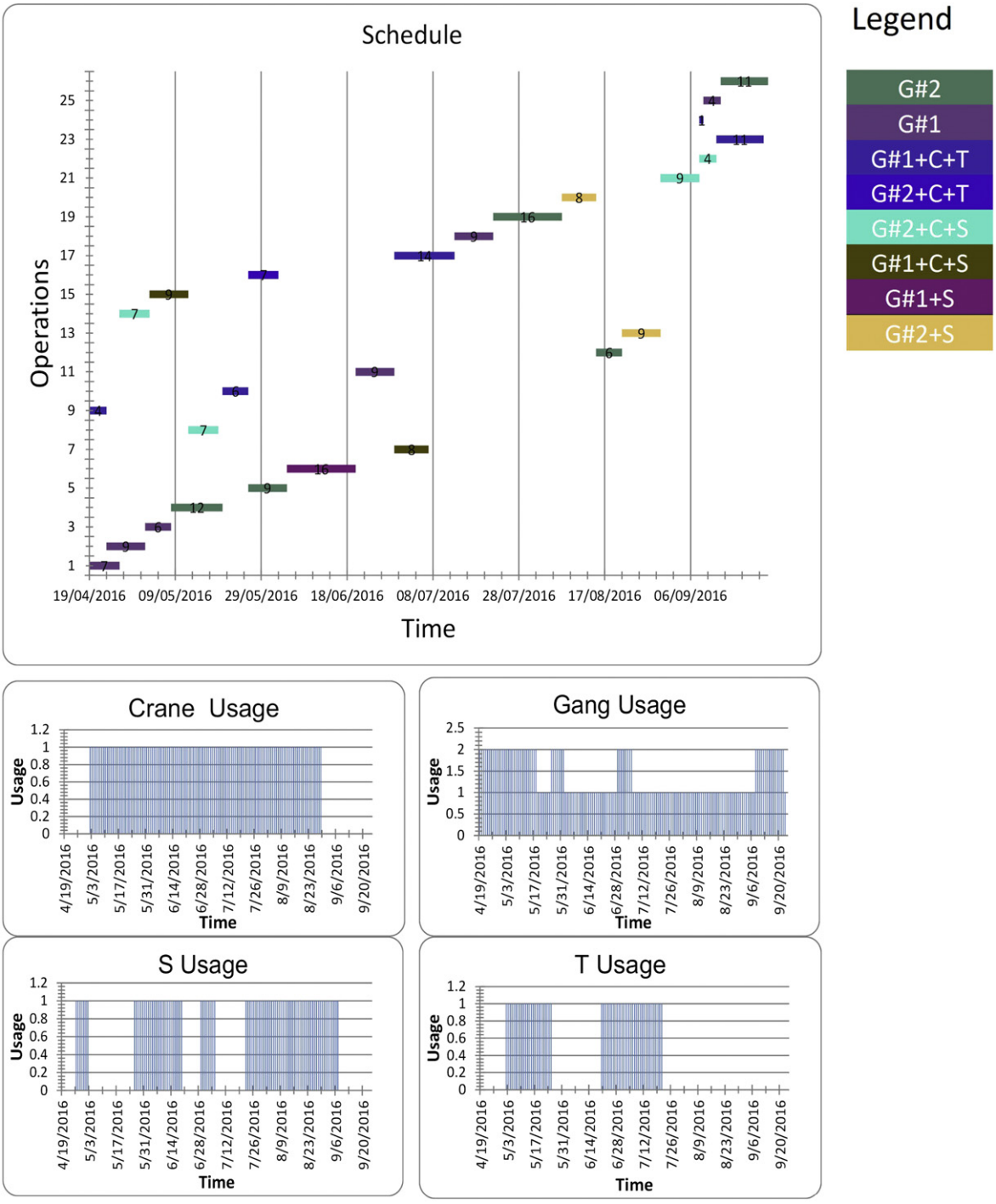


Fig. 11. Schedule obtained for the conceptual exercise of a unified MtA system in Fig. 3 using the proposed GA-based algorithm.

3. a number of FJSS benchmarking problems from the literature: Solutions to several sets of FJSS problem instances designed by Fattahi et al. [26] and Brandimarte [25] have been reproduced. The data sets consist of 10 small problems, 10 medium problems and 10 large problems. The small problems are composed of 2–4 jobs using 2–5 machines and performing 2–5 operations for each job. The medium problems consist of 5–12 jobs, 6–8 available machines and 3–4 operations for each job. The large problems have 10–20 jobs using 4–15 machines and executing 5–15 operations for each job.

A list of algorithm parameter values for having the least computational demand is reported below:

Population Size for the FSS, FJSS and extended FJSS instances: 10
Population Size for Small and Medium Size Benchmarking Instances: 100
Population Size for Large Size Benchmarking Instances: 5000
Previous Iteration Rate: 60%
Crossover Probabilities: 23%

Table 5
Comparison of the developed GA with other algorithms on 10 small-sized Flexible Job Shop Scheduling (FJSS) problem instances and 10 medium-sized FJSS problem instances from Fattahi et al. [21].

Instance							GA	MILP			CP	Tabu	Annealing	Heuristic	Best Solution
Name	n	m	$h_{jmin} - h_{jmax}$	meq	$proc$	LB		Fattahi et al. [26]	OOY [27]	Birgin et al. [22]	BG [28]			dev (%)	dev (%)
SFJS1	2	2	2 – 2	2	24–65	66	66	66	66	66	66	66	66	0%	0%
SFJS2	2	2	2 – 2	2	21–71	107	107	107	107	107	107	107	107	0%	0%
SFJS3	3	2	2 – 2	2	43–135	212	221	221	221	221	221	221	221	0%	0%
SFJS4	3	2	2 – 2	2	54–152	331	355	355	355	355	355	390	355	0%	0%
SFJS5	3	2	2 – 2	2	21–71	107	119	119	119	119	119	137	119	0%	0%
SFJS6	3	3	3 – 3	3	17–170	310	320	320	320	320	320	320	320	0%	0%
SFJS7	3	5	3 – 3	3	62–214	397	397	397	397	397	397	397	397	0%	0%
SFJS8	3	4	3 – 3	3	24–65	216	253	253	253	253	253	253	253	0%	0%
SFJS9	3	3	3 – 3	3	17–100	210	210	210	210	210	210	215	215	0%	0%
SFJS10	4	5	3 – 3	3	62–214	427	516	516	516	516	516	617	516	0%	0%
												Average performance		0%	0%
MFJS1	5	6	3 – 3	3	47–214	396	468	470	468	468	468	548	488	+4.09%	0%
MFJS2	5	7	3 – 3	3	47–214	396	459	484	446	446	446	457	478	+4.15%	−2.91%
MFJS3	6	7	3 – 3	3	62–320	396	466	564	466	466	466	606	599	+22.20%	0%
MFJS4	7	7	3 – 3	3	62–247	496	569	684	564	554	554	870	703	+19.06%	−2.70%
MFJS5	7	7	3 – 3	3	62–250	414	539	696	514	514	514	729	674	+20.02%	−4.86%
MFJS6	8	7	3 – 3	3	62–320	469	708	786	635	634	634	816	856	+13.23%	−11.67%
MFJS7	8	7	2 – 3	4	47–250	619	965	1433	935	879	931	1048	1066	+1.91%	−9.78%
MFJS8	9	8	2 – 3	4	40–257	619	992	1914	905	884	884	1220	1328	+18.68%	−12.21%
MFJS9	11	8	2 – 5	4	40–268	764	1169	2908	1192	1037	1070	1124	1148	−4.00%	−9.25%
MFJS10	12	8	2 – 5	4	40–357	944	1369	4960	1276	1251	1208	1737	1546	+11.44%	−13.32%
												Average performance		+11.08%	−6.67%

n: number of jobs.
m: number of machines/resources.
$h_{jmin} - h_{jmax}$: minimum and maximum number of operations per job.
meq: maximum number of equivalent machines/resource per operation.
$proc$: minimum and maximum processing time per operation.
LB: Lower Bounds.
MILP: Mixed Integer Linear Programming.
GA: Genetic Algorithm.
OOY: Ozguven, Ozbakir and Yavuz.
CP: Constraint Programming.
BG: Behnke and Geiger.
dev: deviation.

Mutation Probabilities: 10%
Migration Probabilities: 7%

The population size of 100 is used in this paper for running the small and medium size instances based on the study by Roeva et al. [35]. Looking at Table 6, the best makespan for large size instances is achieved by the GA-based work of Pezzella et al. [36]. Hence, the same population size of 5000 is used here for these large size instances. The listed crossover, mutation and migration values are found most effective from computational experiments.

The results for the FSS and FJSS problems are reported in Figs. 9 and 10. The results calculated by the GA-based algorithm are identical to the actual manufacturing and assembly schedules.

In Fig. 11, the results of the conceptual exercise of a unified MtA system shown in Fig. 3) is illustrated using the proposed GA-based algorithm. It can be seen that the proposed algorithm is capable of solving extended FJSS problems which have higher degree of flexibility to capture real world scenarios.

Table 5 summarises the results obtained by the developed GA in comparison with results from the literature with respect to small and medium size instances. Starting from the left, the columns specify the instance name, the number of jobs (n), the number of machines (m) for each example, and the Lower Bound (LB). In the eighth column, the best makespan gained within ten minutes time using the developed GA is shown. These results can be directly compared to the MILP results of the work by Fattahi et al. [26], Ozguven et al. (OOY) [27], Birgin et al. [22], and Behnke and Geiger(BG) [28]. Table 5 also shows the results using the Constraint Programming (CP) model by Behnke and Geiger [28] as well as the tabu and annealing algorithms. The last two columns present the relative deviation (dev) of the best known solution by other heuristics and MILPs to our GA algorithm. The relative deviation is defined as $dev = [(MK_{best} - MK_{GA})/MK_{best}] \times 100\%$, where MK_{best} is the best makespan gained by other algorithms and MK_{GA} is the makespan achieved by the developed GA. In comparison to current available heuristic algorithms, Table 5 shows that our GA performs well for small problems and outperforms by +9.41% for medium instances. However, our GA deviates by −6.67% in medium instances from an exact solution determined by MILPs. With regard to the 10 large FJSS problem instances by Brandimarte [25], Table 6 presents our results compared to the results by the tabu algorithms reported by Brandimarte [25] and Mastrolilli and Gambardella (MG) [37] (MG), the results by the CP model presented by Behnke and Geiger [28], and the results by the GA algorithms proposed by Pezzella et al. [36], Chen et al. [38], Jia et al. [39] and Ho and Tay (HT) [40] using Composite Dispatching Rules (CDRs). The arrangement of the data in Table 6 is equivalent to Table 5. In summary, our proposed algorithm underperforms on average of about 15% in comparison to current available algorithms. The proposed structure for a chromosome allows having a set of operations with arbitrary relations and, therefore, is able to solve complex extended FJSS problems. However, it requires more time for finding a good solution compared to available non-exact methods for solving FJSS problems. It should be considered that reported results are returned within 10 min. Thus, the proposed algorithm has a higher degree of flexibility to capture real world scenarios in comparison to other non-exact algorithms.

Table 6
Comparison of the developed GA with other algorithms on 10 Flexible Job Shop Scheduling (FJSS) problem instances from Brandimarte [25].

Instance							GA	Tabu		CP	GA				GA	
Name	n	m	$h_{jmin} - h_{jmax}$	meq	$proc$	LB		Brandimarte [25]	MG [37]	BG [28]	Pezzella et al. [36]	Chen et al. [38]	Jia et al. [39]	HT [40]	dev (%)	dev (%)
Mk01	10	6	5 – 7	3	1–7	36	43	42	40	40	40	40	40	41	−7.5%	−7.5%
Mk02	10	6	5 – 7	6	1–7	24	29	32	26	27	26	29	28	29	−11.53%	−11.53%
Mk03	15	8	10 – 10	5	1–20	196	196	211	204	204	204	204	204	204	+3.92%	+3.92%
Mk04	15	8	3 – 10	3	1–10	48	70	81	60	60	60	63	61	67	−16.67%	−16.67%
Mk05	15	4	5 – 10	2	5–10	168	176	186	173	174	173	181	176	176	−1.73%	−1.73%
Mk06	10	15	15 – 15	5	1–10	33	81	86	58	59	63	60	62	68	−35%	−44.64%
Mk07	20	5	5 – 5	5	1–20	133	153	157	144	143	139	148	145	148	−10.07%	−10.07%
Mk08	20	10	5 – 15	2	5–20	523	545	523	523	523	523	523	523	523	−4.20%	−4.20%
Mk09	20	10	10 – 15	5	5–20	299	375	369	307	307	311	308	310	328	−22.07%	−22.14%
Mk10	20	15	10 – 15	5	5–20	165	287	296	198	214	212	212	216	231	−35.37%	−44.94%
													Average performance		−14.02%	−15.95%

n: number of jobs.
m: number of machines/resources.
$h_{jmin} - h_{jmax}$: minimum and maximum number of operations per job.
meq: maximum number of equivalent machines/resource per operation.
$proc$: minimum and maximum processing time per operation.
LB: Lower Bounds.
GA: Genetic Algorithm.
CP: Constraint Programming.
MG: Mastrolilli and Gambardella.
BG: Behnke and Geiger.
HT: Ho and Tay.
GA dev: deviation from the best makespan obtained by other GAs.
dev: deviation.

7. Conclusions and future work

This paper contributes to the development of a heuristic method for the holistic Manufacturing, transportation and Assembly (MtA) resource scheduling problem. The holistic/unified MtA system of precast construction projects is defined like a Resource-constrained Extended Flexible Job Shop Scheduling (REFJSS) problem. A multi-objective Genetic Algorithm (GA) has been developed to optimise cost and time associated in different precast construction techniques. Hence, it allows constraining the number of on-site workforce per day from congested construction site. Using this multi-objective GA-based optimisation model, the most advantageous solution for different levels of prefabrication is determined and compared with respect to overall time and cost. The performance of the developed algorithm was evaluated using results of 30 small, medium and large problem instances reported in the literature. It can be concluded that our GA algorithm outperforms available heuristics in small and medium size instances. However, this GA-based algorithm requires further improvements to outperform the current available scheduling algorithms in large size FJSS instances. However, the developed algorithm has a higher degree of flexibility to capture real world scenarios in comparison to other algorithms. The proposed model is capable of solving complex extended FJSS problems with arbitrary precedence relationships among their operations. The optimisation architecture explored in this paper allows introducing further objectives that are essential to be optimised as part of future work. In future work, the proposed model will be combined with exact models. Hence, the flexibility of the presented GA-based algorithm will remain; at the same time, the performance with respect to large size FJSS instances might improve. Also, a Pareto frontier based on the non-dominant objective functions will be calculated to solve complex extended FJSS problems using the proposed method.

Acknowledgments

This research is funded by Innovate UK [101425] and the Engineering and Physical Sciences Research Council (EPSRC) [grant No. EP/L504683/1] whose support is gratefully acknowledged. The authors also express appreciation to the Laing O'Rourke's team especially Adam Locke, Corin Walford, John Hornsby, Andrew Jackson, Chan Deryck, Alex Heward, James Thorpe, and Tom Gilpin for their assistance. This publication is supported by multiple datasets, which are in the "Problem Description" section of this paper and openly available at: http://www.ime.usp.br/~cris/fjs/benchmark/. Please contact portec@imperial.ac.uk for any further inquiries regarding the data.

References

[1] J. Hilgeman, A Prefabricated Framing and Enclosure System: Economy, Flexibility, and Applications, Master's thesis. University of Washington. 2004.
[2] J. Gibbons, Technology, trade, and the U.S. residential construction industry, Congress of the US Special Report, 1986.
[3] C. Haas, Prefabrication and preassembly trends and effects on the construction workforce, Tech. Rep. Issue 14 of Report (University of Texas at Austin. Center for Construction Industry Studies), Center for Construction Industry Studies. 2000.
[4] SmartMarket Report: McGraw-Hill Construction, Prefabrication and modularization: increasing productivity in the construction industry, Tech. Rep., 2011.
[5] J. Song, W.R. Fagerlund, C. Haas, C. Tatum, J. Vanegas, Considering prework on industrial projects, J. Constr. Eng. Manag. 131 (6) (2005) 723–733.
[6] C. Pasquire, A. Gibb, N. Blismas, What Should You Really Measure if You Want to Compare Prefabrication With Traditional Construction? 2005.
[7] C. Pasquire, G. Connolly, Leaner construction through off-site manufacturing, Proceedings IGLC, (Gramado, Brazil), 2002.
[8] M. Garey, D. Johnson, R. Sethi, The complexity of flow shop and job shop scheduling, Math. Oper. Res. 1 (1976) 117–129.
[9] M. Murtaza, D. Fisher, M. Skibniewski, Knowledge-based approach to modular construction decision support, J. Constr. Eng. Manag. 119 (1) (1993) 115–130.
[10] M. Murtaza, D. Fisher, Neuromodex - neural network system for modular construction decision making, J. Comput. Civ. Eng. 8 (2) (1994) 221–233.
[11] R. Soetanto, A. Dainty, J. Glass, A. Price, A framework for objective structural frame selection, In Proceedings of the Institution of Civil Engineers-Structures and Buildings, 2008.
[12] Y. Luo, Decision Support for Prefabrication Strategy Selection on Building Systems, (Ph.D. thesis). The Pennsylvania State University. 2008.
[13] Y. Chen, G. Okudan, D. Riley, Decision support for construction method selection in concrete buildings: prefabrication adoption and optimization, Autom. Constr. 19 (2010) 665–675.
[14] S. Leu, S. Hwang, GA-based resource-constrained flow-shop scheduling model for mixed precast production, Autom. Constr. 11 (2002) 439–452.

[15] C. Ko, S. Wang, Precast production scheduling using multi-objective genetic algorithms, Expert Syst. Appl. 38 (2011) 8293–8302.
[16] K. Haase, G. Desaulniers, J. Desrosiers, Simultaneous vehicle and crew scheduling in urban mass transit systems, Transp. Sci. 35 (2001) 286–303.
[17] C. Lee, T. Cheng, B. Lin, Minimizing the makespan in the 3-machine assembly–type flowshop scheduling problem, Manag. Sci. 39 (1993) 616–625.
[18] C. Potts, S. Sevast'Janov, L.V. Wassenhove, C. Zwanevel, The two-stage assembly scheduling problem: complexity and approximation, Oper. Res. 43 (2) (1995) 346–355.
[19] A. Hariri, C. Potts, A branch and bound algorithm for the two-stage assembly scheduling problem, Eur. J. Oper. Res. 103 (1997) 547–556.
[20] M. Yokoyama, Flow-shop scheduling with setup and assembly operations, Eur. J. Oper. Res. 187 (2008) 1184–1195.
[21] P. Fattahi, S. Hosseini, F. Jolai, R. Tavakkoli-Moghaddam, A branch and bound algorithm for hybrid flow shop scheduling problem with setup time and assembly operations, Appl. Math. Model. 38 (2014) 119–134.
[22] E. Birgin, P. Feofiloff, C. Fernandes, E. de Melo, M.I. Oshiro, D. Ronconi, A MILP model for an extended version of the flexible job shop problem, Optim. Lett. 8 4 (2013) 1417–1431.
[23] R. Vaessens, Generalized Job Shop Scheduling: Complexity and Local Search, (Ph.D. thesis). Technische Universiteit Eindhoven. 1995.
[24] D. Williamson, L. Hall, J. Hoogeveen, C. Hurkens, J. Lenstra, S. Sevastianov, D. Shmoys, Short shop schedules, Oper. Res. 45 (1997) 288–294.
[25] P. Brandimarte, Routing and scheduling in a flexible job shop by tabu search, Ann. Oper. Res. 41 (1-4) (1993) 157–183.
[26] P. Fattahi, M. Mehrabad, F. Jolai, Mathematical modeling and heuristic approaches to flexible job shop scheduling problems, J. Intell. Manuf. 18 (2007) 331–342.
[27] C. Ozguven, L. Ozbakir, Y. Yavuz, Mathematical models for job-shop scheduling problems with routing and process plan flexibility, Appl. Math. Model. 34 (2010) 1539–1548.
[28] D. Behnke, M.J. Geiger, Test Instances for the Flexible Job Shop Scheduling Problem With Work Centers, Helmut-Schmidt-University, Logistics Management Department, Hamburg, Germany, 2012.
[29] R. Rosenberg, Simulation of genetic populations with biochemical properties, Math. Biosci. 8 (1-2) (1970) 1–37.
[30] A.H. Wright, Genetic algorithms for real parameter optimization, Found. Genet. Algoritm. 1 (1991) 205–218.
[31] M. Melanie, An Introduction to Genetic Algorithms, 1999. Cambridge, Massachusetts London, England
[32] M. Gen, R. Cheng, Genetic Algorithms and Engineering Optimization, John Wiley & Sons, Inc., New York, 2000.
[33] E. Falkenauer, S. Bouffoix, A genetic algorithm for job shop, Proceedings of the 1991IEEE International Conference on Robotics and Automation, 1991.
[34] R.Q. dao-er ji, Y. Wang, A new hybrid genetic algorithm for job shop scheduling problem, Comput. Oper. Res. (2012).
[35] O. Roeva, S. Fidanova, M. Paprzycki, Influence of the population size on the genetic algorithm performance in case of cultivation process modelling, Proceedings of the 2013 Federated Conference on Computer Science and Information Systems, 2013, pp. 371–376.
[36] F. Pezzella, G. Morganti, G. Ciaschetti, A genetic algorithm for the flexible job-shop scheduling problem, Comput. Oper. Res. 35 (2008) 3202–3212.
[37] M. Mastrolilli, L. Gambardella, Effective neighbourhood functions for the flexible job shop problem, J. Sched. 3 (1999) 3–20.
[38] H. Chen, J. Ihlow, C. Lehmann, A genetic algorithm for flexible job-shop scheduling, Proceedings of the IEEE International Conference on Robotics and Automation, 1999, pp. 1120–1125.
[39] H. Jia, A. Nee, J. Fuh, Y. Zhang, A modified genetic algorithm for distributed scheduling problems, J. Intell. Manuf. 14 (2003) 351–362.
[40] N. Ho, J. Tay, GENAce: an efficient cultural algorithm for solving the flexible job-shop problem, Evol. Comput. 2 (2004) 1759–1766.