Optimal facility layout planning for AGV-based modular prefabricated manufacturing system

Chen Chen[1*], Tran Huy Duc[1], Tiong Lee Kong[1], Chen I-Ming[2], Cai Yiyu[2]

[1] School of Civil & Environmental Engineering, Nanyang Technological University, Singapore
[2] School of Mechanical & Aerospace Engineering, Nanyang Technological University, Singapore

Abstract

Cross-industry learning of the Toyota production system has inspired the precast factories in the construction industry to adopt an automated guided vehicle (AGV)-based flow production system for the manufacturing of their modular prefabricated products. Compared to the production process of automobiles, the manufacturing process of modular prefabricated products is very unbalanced leading to a large pool of queues. And additionally, after some operations, settling is needed. Hence, due to these unique features, facility layout is a crucial element that needs to be well planned in order to achieve a feasible and efficient system. In this regard, this paper proposes an approach to plan the facility layout of the investigated AGV-based modular prefabricated manufacturing system. The paper firstly gives an optimization method for the size arrangement of the workstation area and the storage area. There are two conflicting objectives in the optimization model: one is to minimize the production time and the other is to maximize the workstation utilization. A simulation based non-dominated sorting genetic algorithm is developed to solve the model. Then, the paper proposes a heuristic method to guide the placement, reshuffle, and retrieval of the modular prefabricated products in the storage area. According to the heuristic, there is no need of dedicated paths for AGVs. The storage area can be fully occupied by the work-in-progress and the AGV traveling paths are dynamically generated. And thirdly, the paper is also able to provide a suitable size of the AGV fleet which is able to accomplish the moving tasks in time. The experimental test on an industrial case shows the potential of the proposed planning approach to guide the real practice.

Keyword: facility layout, non-dominated genetic algorithm, modular prefabricated manufacturing, simulation, storage plan

1. Introduction

The Toyota production system (TPS) displaying the highest levels of manufacturing efficiency in the world automobile industry has been compared to the construction industry since the end of the twentieth century [1, 2]. The cross-industry learning practices have discovered many similarities between the prefabricated house production systems and the automobile production systems. As architects and builders are attracted by the state-of-the-art in lean production, a growing trend to move the building process from the physical site to a controlled factory environment is observed. With prefabricated construction, the building components are fabricated in the factory simultaneously with the site work. The same building codes and architectural specifications as traditional construction are followed. As such, the assembled buildings are virtually indistinguishable with their onsite-built counterparts. There are a lot of benefits from implementation of the prefabricated construction method. Quality, productivity, efficiency and safety all

[*] Corresponding author: Dr. Chen Chen, chenchen@ntu.edu.sg

can be improved [3]. Furthermore, the prefabricated construction method is recognized as playing a significant contribution to the sustainability urbanization [4]. Therefore, the use of prefabricated building components is actively promoted by a lot of government authorities. Singapore, for example, mandates the adoption of a certain amount of prefabricated products in all its residential projects on government land sale (GLS) sites (www.bca.gov.sg).

Prefabricated construction can be categorized into four categories based on the levels of manufacturing and the magnitudes of offsite assembly, they are: pre-cut, penalized, modular and manufactured [5]. Prefabricated bathroom unit (PBU) and prefabricated prefinished volumetric construction (PPVC), which are modular blocks contain finishes, fixtures and fittings and can be assembled on site in the manner of Lego blocks, are the most complete in factory finishes among the spectrum of the prefabricated products. They are now widely used in different kinds of multi-level and high-rise buildings. Facing this trend, the precast factories are stepping up efforts to upgrade their production productivity by adopting an automated guided vehicle (AGV) flow production system. Currently, the modular prefabricated products are placed at fixed positions. Workers have to find the assigned unit carrying their tools and the required materials in a maze of rows and columns. As a consequence, a lot of time is wasted in logistics before any work can commence. The new flow production method, however, divides the entire manufacturing process into many segregated operations and each operation is performed at a workstation. Then, workers stay at the fixed workstations while AGVs move the work-in-progress (WIP) around the fabrication yard.

Such an AGV-based flow production system requires a proper facility layout. A poor one can cause the entire system deadlocked. Since this is a very unbalanced production line, for example, the operation of tiling takes 48 hours while the operation of ponding only needs 1 hour, a large pool of queues will be generated. And additionally, after these operations, settling is needed. As a result, a big central storage area is needed in order to achieve a consistent production flow. Nevertheless, restricted by the plant area, the size of storage area has an upper bound. Hence, in order to reduce the dependency on the storage area, a flexible system that the operations on the workstations are dynamically adaptable to the production is much preferred. To support this flexible system, a group of workers who are capable of different operations are required. As such, the transition cost from one operation to another is minimum.

AGV-based flow production system is novel for manufacturing of modular prefabricated products. There is not yet such system in real world. This paper gives a planning approach to provide a way for the managers to design the facility layout for it. The paper firstly gives an optimization method for size arrangement of the workstation area and the storage area. There are two conflicting objectives in the optimization model: one is to minimize the production time and the other is to maximize the workstation utilization. A simulation based non-dominated sorting genetic algorithm is developed to solve the model. Then, the paper proposes a heuristic method to guide the placement, reshuffle and retrieval of the modular prefabricated products in the storage area. According to the heuristic, there is no need of dedicated paths for AGVs. The storage area can be fully occupied by the WIP and the AGV traveling paths are dynamically generated. Besides, the paper is able to provide a proper size of the AGV fleet by analyzing the worst case with the goal to accomplish all the moving tasks in time.

The remaining of this paper is arranged in this way: Section 2 is the literature review. Section 3 describes the problem and gives the mathematical formulation for the optimization model. Section 4 provides the solving algorithm of the optimization model, which is a non-dominated sorting genetic algorithm. Section 5 addresses the simulation approach for evaluation of every GA individual. Section 6 proposes the heuristic for storage plan. Section 7 gives an industrial case. And finally, Section 8 concludes this paper and recommends the directions for future research.

2. Literature review

The Toyota production system has attracted a great deal of interests from academic field since its appearance in 1970s [6-9]. It is so far the best-known example of lean. There are two distinctive features

of the TPS which are “reduction of cost through elimination of waste” and “to make full use of the worker’s capabilities”. These concepts have impacted the innovations in many other types of manufacturing industries. In 1980s, the growth of IT technology promotes the integration of “just-in-time (JIT)” with computer integrated manufacturing (CIM) in the TPS [10], resulting in a further reduction in cost and lead time whereas an improvement in quality.

In a flexible manufacturing system (FMS), the use of AGVs as an automated material handling solution is very common. Most FMS consist of three main components: automated numerically controlled machines, a material handling system, and a central control computer, representing a high level of CIM implementation. The automated guided vehicle – flexible manufacturing system (AGV-FMS) has been a hot research topic [11-15]. Scheduling of the machines and the AGVs has been thoroughly studied [13-15]. The developed scheduling models often assume sufficient input and output buffer space for queues. However, for the investigated manufacturing system in this paper, space is a constraint. Since the queues and the settling operation both need space, the storage area will easily be fully consumed and become a bottleneck in the system. In this regard, the operations on the workstations are allowed to be changeable to increase flexibility in the manufacturing system. As a consequence, the scheduling models developed for the classical AGV-FMS cannot be directly applied on the investigated system, it seems that using simulation is a better way.

The performance of an AGV-based flow production system is largely determined by its facility layout. Therefore, the problem of facility layout is a key problem to the system [16] and has been actively investigated for more than half a century [17]. Research on the facility layout usually begins with a block layout which specifies the relative location of each department inside the factory and is followed by a detailed layout which deals with the machine layout inside one department. The former is more close to the investigated problem here. Researchers have developed a pairwise exchange technique, also known as computerized relative allocation of facilities technique (CRAFT) [18] to solve it. However, CRAFT is a heuristic method and it cannot guarantee optimality because not all the combinations are assessed.

An alternative approach is to convert it into a quadratic assignment problem (QAP) [19]. But the classical QAPs assume all the departments are of equal size, moreover, the modified QAPs for unequal-area facility layout (UA-FLP) are often found inefficient [20]. Although some encouraging results by using meta-heuristics have been achieved recently [21-23], as the UA-FLP model is based on slicing tree structure that the initial rectangle is divided completely either in horizontal or vertical direction from one side to the other and the division is recursively continued with the newly generated rectangles, a lot of departments will be generated. So, the developed UA-FLP models are not very proper for designing the facility layout of the investigated manufacturing system in which there are only two departments: workstation and storage. Therefore, this paper is going to propose a new bi-objective optimization model for the problem.

When it comes to multiple objective optimization problems, a number of evolutionary multiobjective optimization algorithms (MOEAs) have been proposed and successfully used in a wide range of real-world applications [24]. In these MOEAs, the concept of Pareto front is adopted. On one Pareto front, a solution dominates another if it is better in at least one and not worse in all other objectives. However, there are some difficulties in applying these MOEAs, and the major of them are summarized in the following: (1) the convergence speed is severely deteriorated when the number of objectives increases; (2) the number of solutions required for approximating the entire Pareto front is exponentially increased; and (3) it is difficult to visualize the solutions [25].

There are standard solvers available which are bio-inspired from natural evolution, immune systems, and swarm intelligence, such as non-dominated sorting genetic algorithm II (NSGA-II), strength Pareto evolutionary algorithm version 2 (SPEA2), S-metric selection evolutionary multiobjective optimization algorithm (SMS-EMOA), multiple objective particle swarm optimization (MOPSO), and multiobjective evolutionary algorithm based on decomposition (MOEA/D) [26]. All these algorithms have their advantages and disadvantages respectively. Generally speaking, the Pareto-based approaches follow a straightforward design principle and require only a few parameters but they do not guarantee convergence. The indicator-based approaches are possible to assess the convergence but they spend more computational

cost. And the decomposition-based methods provide a very flexible framework for algorithm design but they require some a priori knowledge of the position of the Pareto front in the objective space.

For bi-objective optimization problems especially when the two objectives are conflicting, non-dominated sorting genetic algorithm is more suitable. Deb et al. proposed it firstly in 1994 [27] and a revised one, namely NSGA-II in 2002 [28]. NSGA-II is a fast and elitist MOEA. It can sort out a complete set of Pareto fronts in a single run. Therefore, NSGA-II is used in this paper to solve the proposed bi-objective optimization model. Compared to its previous version, NSGA-II has a reduced computation complexity. The successive population is produced by selecting better solutions from a combination set which consists of parents and children. Also, the non-dominated solutions are further ranked by calculating their crowding distances which are the densities of solutions surrounding a particular solution in the population.

3. Problem description and mathematical formulation

This section provides the problem formulation. The scope of modular prefabricated fabrication addressed in this paper is after erection of structures. The manufacturing steps involve different trades in activities such as wall, floor, ceiling, door, waterproofing, finishes, sanitary fixtures and hardware, associated plumbing, ventilation and electrical works. The characteristics of the scheduling problem are described:

(1) Jobs (or modular prefabricated products) will not be allowed to linger at the workstations when their operations on the workstations have been finished;
(2) the operation time is deterministic;
(3) AGV fleet is sufficiently large that AGVs are always available whenever they are needed;
(4) Facility breakdown can be neglected;
(5) There is adequate manpower to perform all the operations.

The mathematical expression is given in the following:

Notations

S	plant size which is represented by the quantity of total slots available in the plant
y	quantity of storage slots, $y = S - x$, x is the decision variable representing the quantity of workstation slots
T	total production time
U	total workstation utilization
C	capacity of the manufacturing system
V	production volume which is represented by the number of modular prefabricated products that are required to fabricate
m, n	indices for workstation slots, $1, \dots, x$
u	indices for storage slots, $1, \dots, y$
p	indices for jobs currently in the manufacturing system, $1, \dots, C$
o	indices for operations
$next(o, p)$	the following operation of operation o according to the manufacturing process of job p
$first(p)$	index for first operation of job p
$last(p)$	index for last operation of job p
θ	count of finished job, increment of 1 if the last operation of a job has been finished
t	simulation clock time
$E(t)$	time of the first movement in the jobs after time t
$l(o, p)$	start time of operation o of job p
$f(o, p)$	finish time of operation o of job p
$TP(o)$	processing time of operation o

$TS(o)$	settling time of operation o
$TT(m,n)$	traveling time of AGV to move the job from/to workstation m to/from workstation n
$TT(m,u)$	traveling time of AGV to move the job from/to workstation m to/from storage slot u
$TT(0,m)$	traveling time of AGV to move the job from/to loading/unloading area to/from workstation m
$TT(0,u)$	traveling time of AGV to move the job from/to loading/unloading area to/from storage slot u

Decision variable

x	quantity of workstation slots

$$\alpha_{m,o,t} = \begin{cases} 1 & \text{if workstation } m \text{ is performing operation } o \text{ at time } t \\ 0 & \text{otherwise} \end{cases}$$

$$\beta_{m,p,t} = \begin{cases} 1 & \text{if workstation } m \text{ is occupied by job } p \text{ at time } t \\ 0 & \text{otherwise} \end{cases}$$

$$\gamma_{u,p,t} = \begin{cases} 1 & \text{if storage slot } u \text{ is occupied by job } p \text{ at time } t \\ 0 & \text{otherwise} \end{cases}$$

Objectives

minimize T	Eq. (1)
Maximize U	Eq. (2)

There are two objectives in the optimization model: to minimize the production time and to maximize the workstation utilization. The total production time T is obtained when the number of finished jobs θ is equivalent to the required production volume V. The increment of parameter θ takes place when the simulation clock time reaches $t = f(last(p), p)$.

The total workstation utilization U is the ratio of the total working time of the workstations to the total production time T. The total working time can be calculated using the formula: $\sum_t \sum_m \alpha_{m,o,t} \times (E(t) - t)$, in which the simulation clock time is advanced from t to $E(t)$ at each time. $E(t)$ represents the time of the first movement in the jobs after time t. The state of the manufacturing system does not alter between t and $E(t)$.

A job can move only when it has finished its current operation and the position for its following operation is available. The start time of an operation for a job $l(o,p)$ is estimated using the following formula:

If $o = first(p)$, $l(o,p) = TT(0,m)$ when $\beta_{m,p,t} = 1$
Otherwise,
  If $\beta_{m,p,t} = 1$
    If $\beta_{n,p,E(t)} = 1$, $l(next(o,p),p) = f(o,p) + TT(m,n)$
    Otherwise, $l(next(o,p),p) = f(o,p) + TT(m,u)$ when $\gamma_{u,p,E(t)} = 1$
  Otherwise,
    If $\beta_{n,p,E(t)} = 1$, $l(next(o,p),p) = f(o,p) + TT(m,u)$
    Otherwise, $l(next(o,p),p) = f(o,p)$

Which means if the current operation is the first operation of a job, then its start time on the workstation is the addition of the traveling time (from the loading area to the workstation) to the project start time which is usually assumed 0. Otherwise, the start time of the next operation is the addition of the traveling time to the finish time of the current operation.

Also, the finish time of an operation for a job $f(o,p)$ is estimated using the following formula:

If $o = last(p)$
  If $\beta_{m,p,t} = 1$, $f(o,p) = l(o,p) + TP(o) + TT(0,m)$
  Otherwise, $f(o,p) = l(o,p) + TS(o) + TT(0,u)$ when $\gamma_{u,p,t} = 1$
Otherwise,
  If $\beta_{m,p,t} = 1$, $f(o,p) = l(o,p) + TP(o)$
  Otherwise,
    If $TS(o) > 0$, $f(o,p) = l(o,p) + TS(o)$
    Otherwise, $f(o,p) = l(o,p)$

Which means if the current operation is the last operation of a job, then the operation finish time is the addition of the operation start time to the operation time (processing time if the job is currently on a workstation or settling time if the job is currently on a storage slot), and also to the traveling time (from the current location to the unloading area). Otherwise, if the job is currently on a workstation, the operation finish time is the addition of the processing time to the processing start time. On the other hand, if the job is currently on a storage slot, then if settling is needed, the settling finish time is the addition of the settling time to the settling start time. However, if settling is not needed, the job will be in queue, so the operation finish time is still the operation start time.

Constraints

There are two constraints. The first constraint limits the number of the jobs in the manufacturing system. New jobs cannot enter the system when the total number of the jobs in the system reaches the system capacity. The second constraint is the relationship constraint which indicates that one job can only occupy one location which is either in the workstation area or in the storage area.

$$\sum_m \sum_p \beta_{m,p,t} + \sum_u \sum_p \gamma_{u,p,t} \leq C, \forall t \qquad \text{Eq. (3)}$$

$$\sum_m \beta_{m,p,t} \times \left(1 - \sum_u \gamma_{u,p,t}\right) + \left(1 - \sum_m \beta_{m,p,t}\right) \times \sum_u \gamma_{u,p,t} = 1, \forall t, p \qquad \text{Eq. (4)}$$

4. NSGA-II

4.1. Non-dominated sorting

The two objectives in the optimization model are conflicting: minimizing the production time will produce a solution with more workstations whereas maximizing the workstation utilization will give a solution with less workstations. As such, no single solution exists that can simultaneously optimize both objectives. There are a set of multiple solutions. NSGA-II is employed to sort out the complete set of Pareto fronts.

In NSGA-II, for the solutions on the same Pareto front, they are further ranked by their crowding distances. The crowding distance of solution i is calculated by measuring the average distance of its neighboring solutions $i-1$ and $i+1$ on either side of the point along each of the objectives (f_1 and f_2), as shown in Figure 1. The crowding distance is the perimeter of the cuboid. The boundary solutions 1 and N on the front have infinite crowding distances. The solutions having the same objective values may have different crowding distances because they belong to two different cuboids. If three or more solutions share the same objective values, every solution except the two boundaries along each objective will be assigned with a crowding distance of 0. A solution is considered better than another if it has a lower Pareto rank or the same Pareto front then a higher crowding distance.

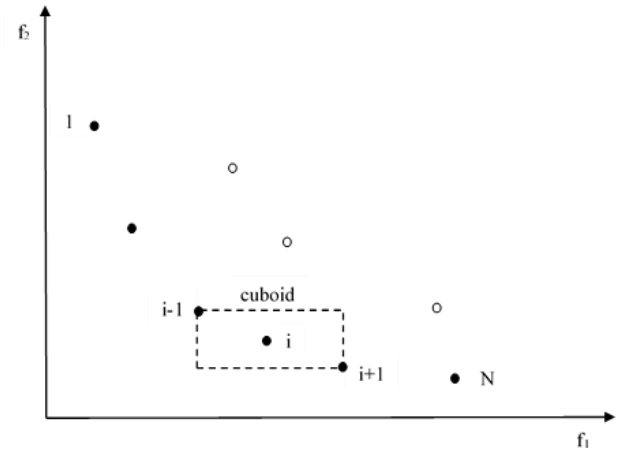

Figure 1. Crowding distance cuboid (points marked in filled circles are solutions of the same Pareto front)

4.2. Chromosome format and genetic operators

Notations

$MaxIter$	maximum iteration
$PopSize$	population size
P_b	binary value probability
P_c	crossover probability
P_m	mutation probability

Binary chromosome

The chromosome is designed in a binary form. Each digit in the chromosome points to a location in the fabrication yard and the binary digit itself indicates this is a workstation slot (1) or a storage slot (0). So, the length of the chromosome is the size of the fabrication yard which is made by the workstation area and the storage area. Figure 2 gives a sample. The fabrication yard has totally 20 slots, among which slots 3, 5, 6, 7, 12, 17 and 20 are workstations and the rest are for storage.

Fabrication yard	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
Chromosome	0	0	1	0	1	1	1	0	0	0	0	1	0	0	0	0	1	0	0	1

Figure 2. GA chromosome

The individuals in the initial population are randomly created. However, in order to produce a more feasible solution, the chance for a slot to be assigned to a workstation is controlled by a value, herein called binary value probability P_b. A smaller value of P_b will produce a solution with less workstations and vice versa. Usually the value of P_b is chosen inversely proportional to the size of the plant. For example, for plant size of 300, P_b is set to a value of 0.1, and for plant size of 3000, P_b is set to a value of 0.01.

Selection

A tournament selection operator based on the rank and crowded distance is introduced in the parent selection for crossover. The tournament size is 2.

Crossover

Single-point crossover method is introduced and it happens with a given probability P_c.

Mutation
Mutation operator performs on the child solutions after crossover. It alters one digit in the chromosome by changing 1 to 0 or 0 to 1. Also, mutation happens with a given probability P_m.

Measures to further improve GA efficiency
Firstly, as iteration continues, the diversity among the individuals will lose. The populations will become more and more homogeneous. This will lower down the search capability of the developed GA. Furthermore, when the population size $PopSize$ is less than the number of solutions on the Pareto front, the obtained solutions will be incomplete. Hence, to overcome these difficulties, the duplicated individuals in the population will be replaced by randomly generated ones.

And secondly, the objective values of an individual are obtained from simulation, and a full-run of simulation takes a lot of computational efforts. Therefore, in order to save computational cost, for those repeated individuals, it is desired to use the results which have previously been obtained. Then a pool is created to keep the objective values for different individuals which are found during iterations. Simulation is only run one time for those new individuals and the obtained objectives are added to the pool. As such, in the future, when the same individuals are found again their objective values can be simply retrieved from the pool.

4.3. Pseudo codes

Here gives the pseudo codes that are implemented in this NSGA-II. The first pseudo code is the iteration procedure for NSGA-II. The second one is the procedure for non-dominated sorting. And the third one is the procedure for calculating the crowding distance.

Procedure: NSGA-II
1: $Iter = 0$, create the initial population
2: Calculate the objective values for each individual in the population
3: Perform a complete non-dominated sorting for the initial population
4: While the size of child population $< PopSize$, repeat step 5 to step 8
5: Tournament selection of parents for crossover
6: Crossover
7: Mutation
8: Add the children to the child population
9: Construct a combination set by combining the parent population with the child population
10: Calculate the objective values for each individual in the population
11: While the size of the population for the next generation $< PopSize$, repeat step 12 to step 14
12: Non-dominated sorting of the combination set to find the Pareto front
13: Add the individuals on the Pareto front to the population for the next generation
14: Remove these individuals from the combination set
15: $Iter = Iter + 1$
16: While $Iter < MaxIter$, repeat from step 4 to step 15

Procedure: Non-dominated sorting
1: To obtain a Pareto front
2: To find the best individual I_b
3: Start from the first individual I_1 in the population, let $I_b = I_1$
4: Compare I_b successively with other individual I_i in the population
5: If I_i is better than I_b in both objectives, replace I_b by I_i
6: Add I_b to the solution list of the Pareto front, and remove it from the population
7: To find all the other solutions on this Pareto front

8: Successively compare the individual I_i in the population with I_b
9: If I_i is better than I_b in either of the objectives, add it to the solution list of the Pareto front and remove it from the population
10: To sort the solutions on the same Pareto front
11: Calculate the crowding distance for each solution in the solution list
12: Rank the solutions from larger crowding distance to lower
13: Add the sorted solution list of the Pareto front to the sorted population
14: While the size of the sorted population $< PopSize$, repeat Step 1 to Step 13 to create the next Pareto front

Procedure: Calculation of the crowding distance
1: Order the solutions on the same Pareto front according to one of the objectives
2: The values of the crowding distances of the boundary solutions 1 and N are assigned with ∞
3: For any other solutions among 2 to $N-1$, measure the perimeter of the cuboid made by its neighbors, and the value of its crowding distance is the obtained perimeter

5. Simulation approach

Simulation provides useful knowledge about the dynamic behavior of the manufacturing system. The main objective of conducting a rule-based simulation in this work is to obtain the production time and the workstation utilization. The simulation algorithm is embedded in the GA algorithm for calculation of the objectives for every GA individual. Figure 3 describes the flowchart of the proposed simulation algorithm.

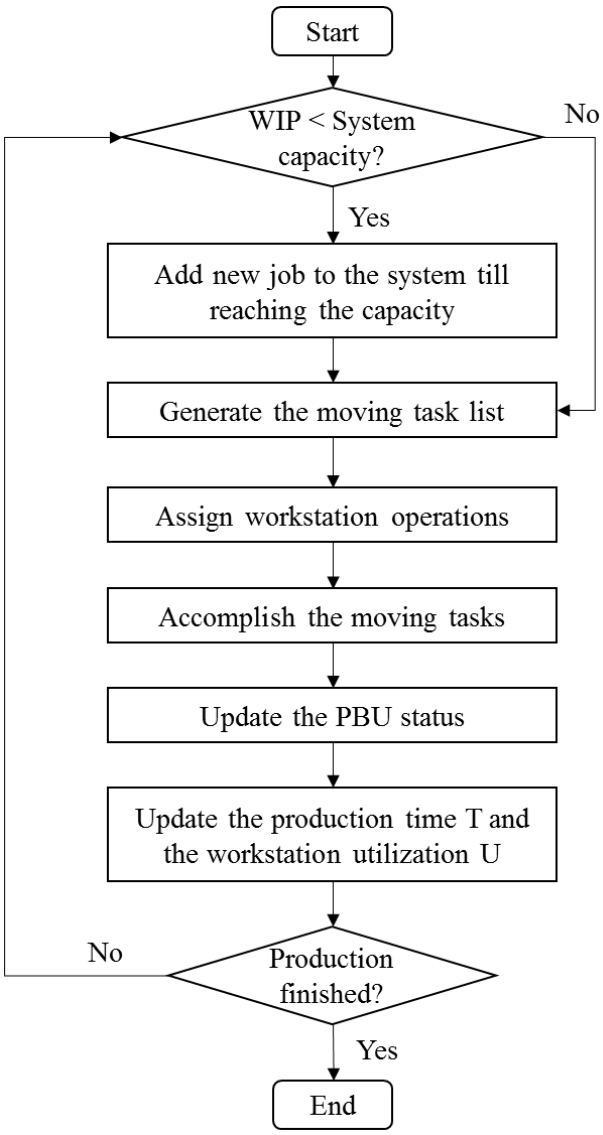

Figure 3. Flowchart of the simulation algorithm

New jobs enter the manufacturing system only when the current WIP are less than the system's capacity. In this paper, the capacity is assumed to be the size of the storage area. The assumption is based on the observation that the system will be deadlock-prone if no more storage slot is available.

The assignment of operations to the workstations is following the pre-given rules. These rules are associated with the storage plan and will be explained in the next section.

6. Heuristic for storage plan

The distinguishing features of the investigated AGV-based manufacturing system justify the need of a great number of storage slots. In order to increase the level of usage of the storage area, a heuristic is designed. By using this heuristic, there are no areas dedicated for AGV traveling. Hence, the space for paths is saved and the storage area can be maximally used. Figure 4 illustrates the general idea of the heuristic.

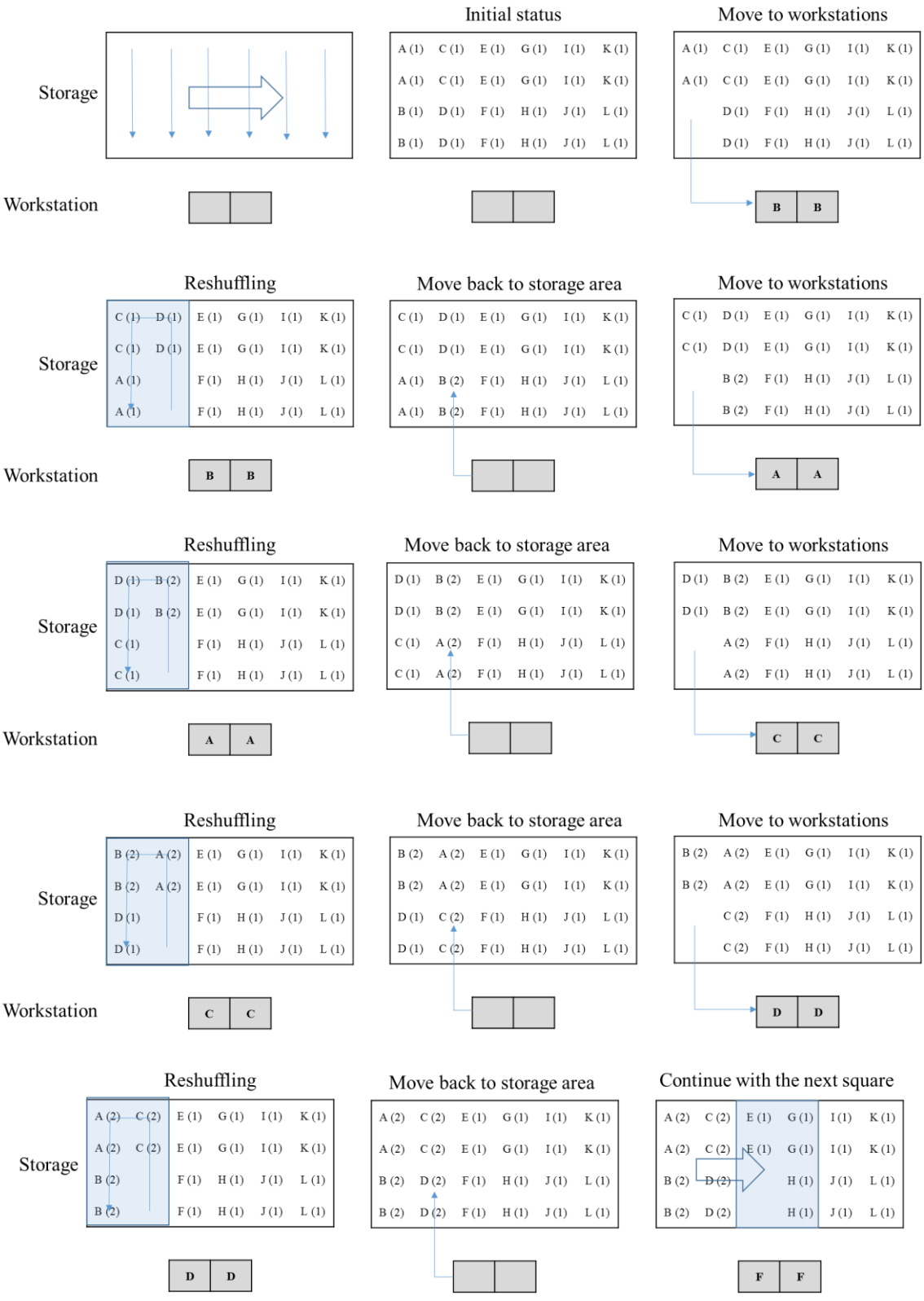

Figure 4. Storage plan

At first, jobs continually enter the manufacturing systems till the storage area is used up. The jobs are arranged in columns in an order that from the farthest slots to the workstations to the nearest. The columns are then arrayed from either left to right or right to left. The storage area is divided into many blocks. Figure 5 shows how to define these blocks in different scenarios.

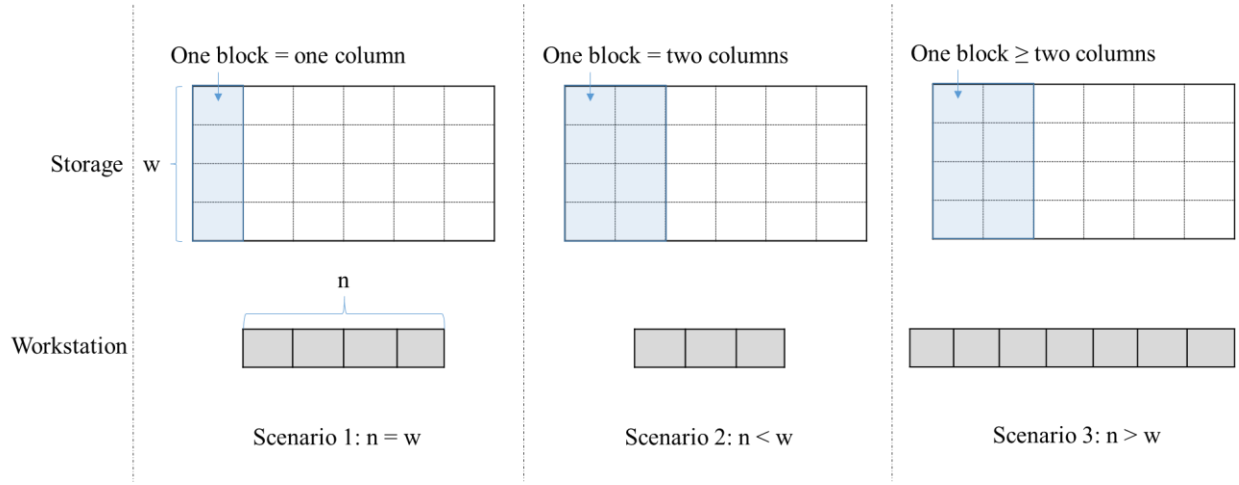

Figure 5. Design of blocks in the storage area

From Figure 5, one can easily see that the size of one block is merely determined by the number of the workstations and the breadth of the storage area if the longitude is the interface to connect with the workstations. In the first scenario, the number of the workstations is equivalent to the number of the slots in the latitude of the storage area. Every column serves as one block. Jobs on the entire block enter the workstations at one time. No shuffling is needed in this scenario. In the second scenario, the number of the workstations is less than the number of the slots in the latitude of the storage area. One block consists of two columns. Beginning from the nearest slot to the workstations at one end (leftmost if from left to right or rightmost if from right to left) in the block, the jobs continuously enter the workstations in a way of an inverse sequence that they enter the storage area. During the workstation processing time, the locations of the remaining jobs in the block are shuffled. Every successive jobs step forward till the vacant slots are filled. In the third scenario, the number of the workstations is bigger than the number of the slots in the latitude of the storage area. One block consists of at least two columns depending on the number of workstations. The jobs on the first column enter the workstations in a way of an inverse sequence that they enter the storage area. However, for the following columns, in order to minimize the shuffling times, the entering sequence of the jobs to the workstations may be the same as the way they enter the storage area since the previous columns can be used as AGV traveling paths. After the jobs have finished their workstation operations, they return to the storage area for either settling or waiting. Then after all the jobs in the block have completed their workstation operations, the above-described procedure continues with the next block. When the jobs in the current block are not enough to supply the entire workstations, the vacant workstations will be supplemented with the jobs in the following block. Figure 6 shows the idea.

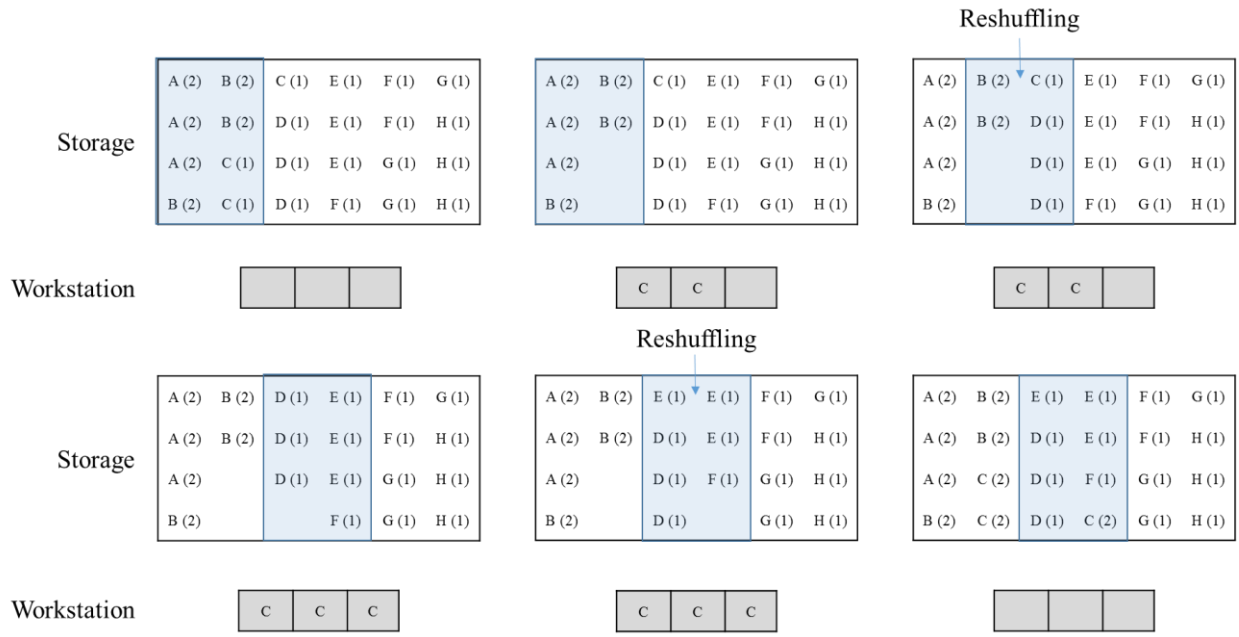

Figure 6. The scenario that the workstations are supplied by the jobs in the following block

The assignment of operations to the workstations is based on the blocks. The sequence of the jobs to enter the workstations determines the operations on the workstations in turn. It is likely that all the workstations will perform the same operations for a while. Using this heuristic, shuffling happens only in the scenarios that the number of the workstations is less or more than the number of the slots in the latitude of the storage area and within every block. In addition, the job shuffling times is related to the size of every block. An appropriate AGV fleet size can be roughly estimated from the worst case that ensures the shuffling being accomplished within the workstation processing time so that no delay due to the lack of AGVs is expected. Multiple AGVs may work in parallel yet their routings should not conflict. Simulation could be the best tool for deciding the exact AGV fleet size.

7. Case Study

The developed algorithms are coded in C# and run using a personal computer with 8GM RAM and 2.10 GHz processor.

A typical precast factory layout is shown in Figure 7. In the given sample, there are three covered yards dedicated for this AGV-based manufacturing system of PBU. The sizes of them are: Yard 1 & 2 - 24 meter × 110 meter and Yard 3 - 24 meter × 72 meter, totally 7,008 square meters. The dimension of a PBU is 3.6 meter × 4.6 meter. Figure 8 provides a proposed slot plan in this fabrication yard. The yard can have a total of 366 slots.

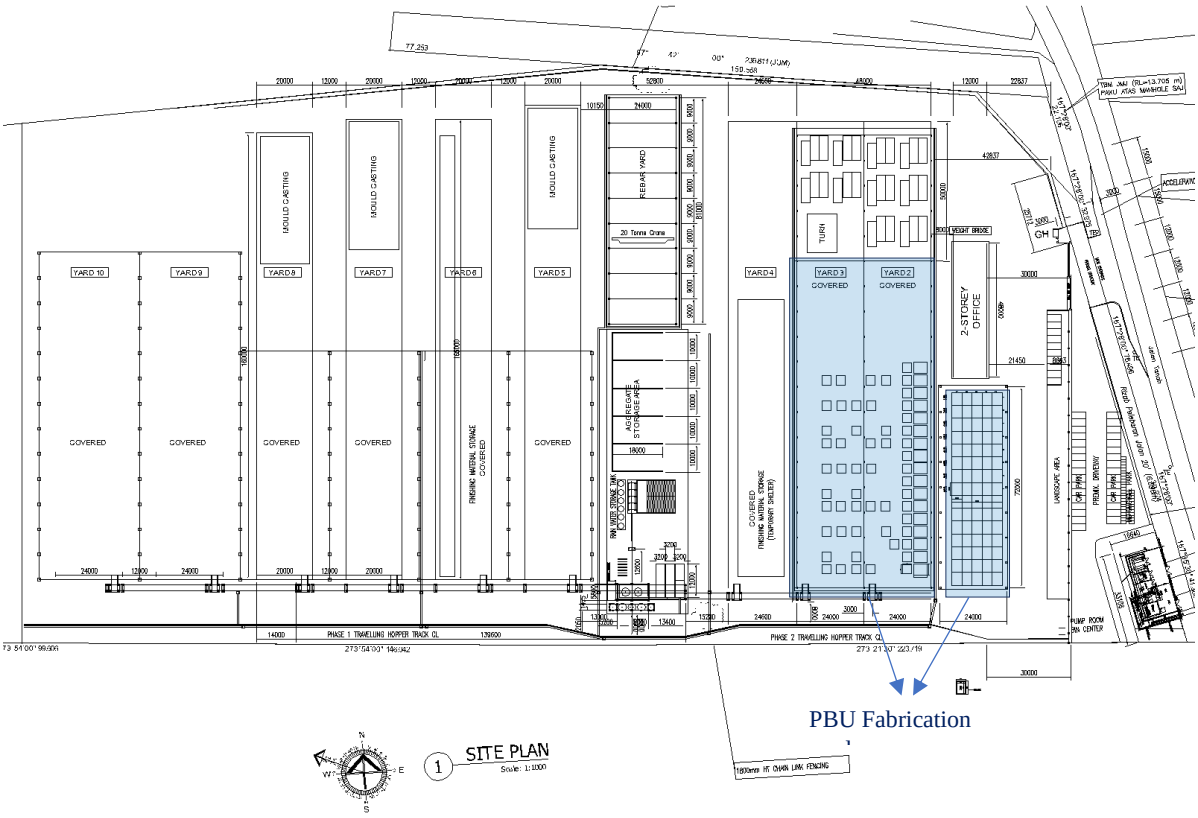

Figure 7. A precast factory layout

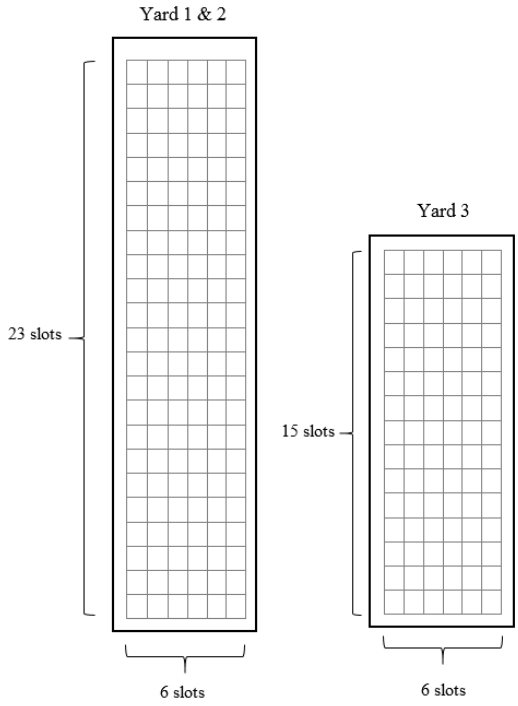

Figure 8. Slots plan in the yard

In the proposed AGV-based manufacturing system, each PBU is placed on a flat-bed platform and then AGVs hoist and move these platforms around the fabrication yard. Figure 9 gives a sample of the platform and the AGV. The AGVs are designed free-ranging that their preferred traveling paths are software programmed. As such, the system can be easily scalable to meet different project's requirements. The speeds of AGVs moving towards different directions are usually different. The speed in side-ways is less than the speed in front-back.

There is no necessary to unload the PBUs from their flat-bed platforms for the workstation operations. The workstations are dedicated areas where there are workers and materials. When AGVs move the platforms to the workstation areas, workers can climb on the platforms to conduct the necessary operations. Cranes are only used in emergency situations such as in the case of facility break-down or when the manufacturing system encounters deadlocks.

Figure 9. Platform and AGV

The production details are described in Table 1. A project consisting of six product types is studied. The production volume is 335 units. The manufacturing processes of each product type are based on four elementary operations: P1 - Ceiling frame, P2 - Plumbing and electrical, P3 - Tiling, and P4 - Sanitary fitting and shower screen.

Table 1. Description of the production

		Time (hours)		Process sequence					
		Process	Settle	M01	M02	M03	T01	T02	T03
P1	Ceiling frame	3	0	1	1	1		1	1
P2	Plumbing & Electrical	6	0	2	2	2	1		2
P3	Tiling	24	12	3	3	3	2	2	3
P4	Sanitary fitting & Shower screen	7	3	4	4	4	3	3	4
			Quantity	15	20	85	121	6	88

The performance of a GA depends on many numerical parameters, they are: initial population type, population size, maximum iteration number, crossover probability, and mutation probability. Due to the problem-specific nature of GA, it is difficult to draw a universal parameter setting for all problems. The selection of a good combination of these parameters has to rely on an extensive sensitivity analysis. In this

regard, the study conducted a sensitivity analysis first with the hope to obtain insights about an appropriate GA parameter setting for the investigated problem.

It is obvious that the population size and the maximum iteration number have positive effect on the solution as larger values of them increase the search space. However, trade-offs should be made between the solution quality and the computational costs. The previous study recommends that the population size is chosen between one and two string length of a chromosome [29]. The plant sizes studied here are: $S = 100, 200, 300$ and 400. The ranges of parameters are as follows: binary value probability P_b = 0.1 to 0.4 in steps of 0.1, crossover probability P_c = 0.6 to 0.9 in steps of 0.1, and mutation probability P_m = 0.1 to 0.4 in steps of 0.1. The maximum iteration $MaxIter$ = 1000 and the population size $PopSize$ = 500. For every problem instance, the algorithm was run multiple times independently to ensure the best results were found. It took about several minutes to run one case one time. Table 2 summarizes the results of the sensitivity analysis.

Table 2. Effect of binary value, crossover and mutation probabilities on the obtained number of optimal solutions on the Pareto front

Binary value	Crossover	Mutation	Number of optimal solutions			
P_b	P_c	P_m	100 slots	200 slots	300 slots	400 slots
0.1	0.6	0.1	17	32	24	30
		0.2	17	32	24	30
		0.3	17	32	24	30
		0.4	17	32	24	30
	0.7	0.1	17	32	24	29
		0.2	17	32	24	28
		0.3	17	32	24	30
		0.4	17	32	24	30
	0.8	0.1	17	32	24	27
		0.2	17	32	24	27
		0.3	17	32	24	27
		0.4	17	32	24	26
	0.9	0.1	17	29	24	27
		0.2	17	29	24	27
		0.3	17	29	24	25
		0.4	17	29	24	30
0.2	0.6	0.1	17	32	24	27
		0.2	17	32	24	26
		0.3	17	32	24	28
		0.4	17	32	24	28
	0.7	0.1	17	32	24	27
		0.2	17	32	24	26
		0.3	17	32	23	28
		0.4	17	32	23	26
	0.8	0.1	17	32	24	24
		0.2	17	32	21	25
		0.3	17	32	22	23
		0.4	17	32	23	24
	0.9	0.1	17	32	18	24
		0.2	17	30	18	22
		0.3	17	28	17	22
		0.4	17	29	20	22
0.3	0.6	0.1	17	32	24	22
		0.2	17	32	16	24

			0.3	17	32	14	22
			0.4	17	32	16	21
		0.7	0.1	17	32	15	19
			0.2	17	31	16	20
			0.3	17	32	14	18
			0.4	17	32	15	22
		0.8	0.1	17	30	8	18
			0.2	17	30	9	18
			0.3	17	32	8	17
			0.4	17	31	9	16
		0.9	0.1	17	27	6	14
			0.2	17	31	7	14
			0.3	17	28	6	11
			0.4	17	27	7	11
0.4		0.6	0.1	17	32	19	14
			0.2	17	32	19	14
			0.3	17	32	22	14
			0.4	17	32	19	13
		0.7	0.1	17	30	15	14
			0.2	17	31	15	14
			0.3	17	31	19	14
			0.4	17	31	14	16
		0.8	0.1	17	28	9	9
			0.2	17	29	12	11
			0.3	17	25	15	11
			0.4	17	26	13	11
		0.9	0.1	17	25	9	9
			0.2	17	23	6	9
			0.3	17	26	7	9
			0.4	17	25	7	7

One can see from Table 2 that the obtained number of optimal solutions follow a pattern for different values of P_b, P_c and P_m. When P_b and P_c have smaller values, more optimal solutions are obtained. Therefore, a combination of $P_b = 0.1$, $P_c = 0.6$ is recommended. However, there is no obvious pattern for different values of P_m. The complete optimal solutions on the Pareto front for the case are listed in Table 3. Totally 35 optimal solutions were obtained. The associated Pareto front pattern is shown in Figure 10.

Table 3. Optimal solutions on the Pareto front for the case

	366 slots			
	Workstation slots x	Storage slots y	Production time T	Workstation utilization U
1	1	365	13009	99.94%
2	2	364	6510	99.85%
3	3	363	4343	99.79%
4	4	362	3259	99.73%
5	5	361	2610	99.62%
6	6	360	2176	99.58%
7	7	359	1868	99.43%
8	9	357	1453	99.42%
9	10	356	1309	99.32%

10	12	354	1092	99.21%
11	13	353	1012	98.82%
12	17	349	774	98.81%
13	20	346	660	98.49%
14	21	345	630	98.27%
15	25	341	530	98.12%
16	26	340	510	98.05%
17	30	336	443	97.83%
18	32	334	416	97.66%
19	33	333	404	97.52%
20	34	332	393	97.30%
21	42	324	319	97.04%
22	43	323	312	96.91%
23	44	322	306	96.56%
24	46	320	293	96.46%
25	47	319	289	95.72%
26	55	311	247	95.70%
27	58	308	235	95.39%
28	59	307	232	94.98%
29	62	304	224	93.61%
30	71	295	197	92.95%
31	82	284	171	92.72%
32	92	274	156	90.59%
33	93	273	155	90.19%
34	112	254	129	89.98%
35	115	251	126	89.72%

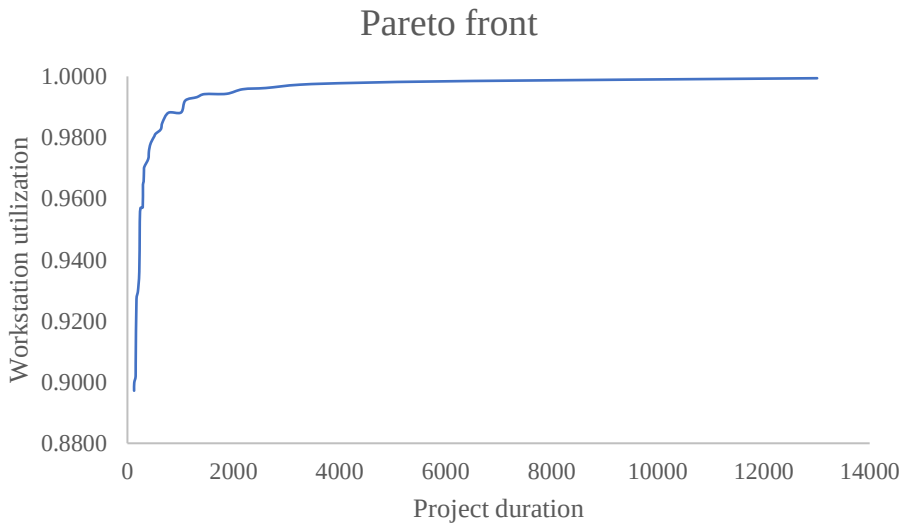

Figure 10. Obtained Pareto front

An appropriate number of workstations can be selected from the optimal solutions based on the project targets and the real factory factors. A trade-off is made between production time and workstation utilization. If time is the key concern, solution with a larger number of workstations will be chosen. On the other hand, if manpower is in shortage, then solution with a smaller number of workstations will be chosen. When the

facility layout has been confirmed, the corresponding production plan can be derived. Furthermore, a preferred AGV fleet size can also be roughly estimated using the developed approach. Assuming the side-way speed of an AGV is 0.6 meter/second, the front-back speed is 1 meter/second, and the hoist speed is 0.2 meter/second, then the time that is required to complete a set of shuffling work is able to be calculated. The minimum number of AGVs is obtained when the AGV fleet can accomplish the set of shuffling tasks within the workstation processing time. The worst scenario is analyzed. Since the amount of shuffling tasks is decided by the breadth of the storage area, it is preferred a narrower space which will have a longer interface with the workstations.

For instance, the number of workstations is supposed to be chosen as 13. Figure 11 gives a proposed facility layout plan, in which all the workstations are placed in one row in Yard 3. Then Yard 1 and 2 are both unmanned yard. The numbers of slots in the latitude of the storage area are 6 (in Yard 1 and 2) and 4 (in Yard 3) respectively. The worst cases are to shuffle 18 slots within 3 hours and to transport 13 jobs from/to the workstations to/from the farthest slots in the storage area. Then, a fleet of at least 2 AGVs is suggested for this case.

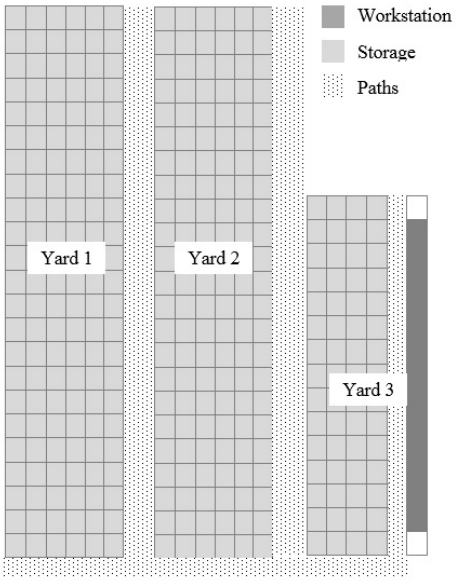

Figure 11. The proposed facility layout for the case

8. Conclusions and further research

As there are big variations among the processing times, the investigated AGV-based modular prefabricated manufacturing system is a very unbalanced production line. A great amount of queues will be accumulated in the system. In addition, some operations require settling afterwards. In this regard, it is expected there is a large storage space. The proposed facility layout planning approach aims to balance the workstation area and the storage area with a hope to reduce the wastes in space. The problem is formulated into a bi-objective optimization model with objectives of minimizing the production time and maximizing the workstation utilization and solved using NSGA-II and simulation approach. A heuristic is designed to guide the placement, reshuffle and retrieval of the jobs in the storage area. With the heuristic, the storage area can be fully used without reserving dedicated space as AGV traveling paths. The heuristic also guides the assignment of operations on the workstations. The minimum AGV fleet size can be roughly estimated by analyzing the worst reshuffle case. The outcomes of the proposed planning approach eventually include an optimal arrangement of the number of the workstations, a feasible production schedule, as well as a suitable

AGV fleet size. The results of the case study show the proposed approach is a powerful and efficient planning method for real practice.

The amount of shuffling times is determined by the number of workstations and the breadth of the storage area. A narrower storage space shape is more preferred. The ideal case is that the number of slots in the latitude of the storage area is the same as the number of the workstations. As such, no shuffling will be needed. All the workstations simultaneously perform the same operation unless there are no more jobs for this operation. The number of the jobs in the system is a fixed number which represents the capacity of the manufacturing system. Seeing the system will become deadlock-prone when the storage area is used up, therefore, the size of storage area is deemed as the system's capacity. Then, new jobs will not enter the manufacturing system until the number of the jobs currently in the system drops below the system's capacity.

The simulation in this study is based on some assumptions and these assumptions could limit its practical feasibility. For example, (1) the operation time may not be deterministic in the real world; (2) size of AGV fleet has limitation practically; (3) facility may breakdown; (4) eight-hour day and so on. Even though, in spite of these limitations, the proposed simulation is still good enough as it is mainly used for fitness calculation in the optimization algorithm at this moment. Nevertheless, an enhanced flexible simulation that addresses the above-mentioned issues will be developed in the future for the purpose of verification and validation of the proposed planning solutions.

Besides, further research focuses can also be placed on the analysis of the material handling costs. It is possible to determine the exact locations of the facilities when the material handling cost reaches the minimum. An automated material handling system will be designed and integrated with the current AGV-based manufacturing system.

Acknowledgement

The work was supported by the National Research Foundation of Singapore under Grant [NRF IIP project M4098029]. The authors would like to thank the industrial parner of this project - Excel Precast Pte Ltd for their financial support, valuable practical recommendations and guidance, and a lot of time spent on discussions. Without their help, it is impossible for the authors to design a production system with practical values for the industry. In addition, the authors would also like to thank the anonymous reviewers whose comments and suggestions helped to improve the layout and presentation of this paper.

References

[1] D. M. Gann, Construction as a manufacturing process? Similarities and differences between industrialized housing and car production in Japan, Construction Management and Economics. 14 (5) (1996) 437-450, https://doi.org/10.1080/014461996373304.

[2] A. Crowley, Construction as a manufacturing process: Lessons from the automative industry, Computers and Structures. 67 (5) (1998) 389-400, https://doi.org/10.1016/S0045-7949(97)00147-8.

[3] X. Zhai, R. Reed and A. Mills, Factors impeding the offsite production of housing construction in China: an investigation of current practice, Construction Management and Economics. 32 (1-2) (2014) 40-52, https://doi.org/10.1080/01446193.2013.787491.

[4] J. Hong, Q. Shen, Z. Li, B. Zhang and W. Zhang, Barriers to promoting prefabricated construction in China: A cost-benefit analysis, Journal of Cleaner Production. 172 (2018) 649-660, https://doi.org/10.1016/j.jclepro.2017.10.171.

[5] A. G. F. Gibb, Off-site fabrication: prefabrication, pre-assembly and modularisation, Whittles Publishing, 1999, (ISBN: 9780470378366).

[6] Y. Sugimori, K. Kusunoki, F. Cho and S. Uchikawa, Toyota production system and Kanban system Materilization of just-in-time and respect-for-human system, International Journal of Production Research. 15 (6) (1977) 553-564, https://doi.org/10.1080/00207547708943149.

[7] E. Lander and J. K. Liker, The Toyota Production System and art: making highly customized and creative products the Toyota way, International Journal of Production Research. 45 (16) (2007) 3681-3698, https://doi.org/10.1080/00207540701223519.

[8] H. Sakai and K. Amasaka, Demonstrative verification study for the next generation production model: application of the advanced Toyota Production System, Journal of Advanced Manufacturing Systems. 7 (2) (2008) 195-219, https://doi.org/10.1142/S0219686708001577.

[9] J. Li, Continuous improvement at Toyota manufacturing plant: applications of production systems engineering methods, International Journal of Production Research. 51 (23-24) (2013) 7235-7249, https://doi.org/10.1080/00207543.2012.753166.

[10] Y. Monden, Toyota Production System: An integrated approach to just-in-time, 4th ed., CRC Press, Taylor & Francis Group, 2011, (ISBN: 9781439820971).

[11] G. C. Vosniakos and B. J. Davies, Simulation study of an AGV system in an FMS environment, The International Journal of Advanced Manufacturing Technology. 3 (4) (1988) 33-46, https://doi.org/10.1007/BF02601832.

[12] F. G. Maughan and H. J. Lewis, AGV controlled FMS, International Journal of Production Research. 38 (17) (2000) 4445-4453, https://doi.org/10.1080/00207540050205217.

[13] B. S. P. Reddy and C. S. P. Rao, A hybrid multiobjective GA for simultaneous scheduling of machines and AGVs in FMS, International Journal of Advanced Manufacturing Technology. 31 (5-6) (2006) 602-613, https://doi.org/10.1007/s00170-005-0223-6.

[14] A. G. Babu, J. Jerald, A. N. Haq, V. M. Luxmi and T. P. Vigneswaralu, Scheduling of machines and automated guided vehicles in FMS using differential evolution, International Journal of Production Research. 48 (16) (2010) 4683-4699, https://doi.org/10.1080/00207540903049407,

[15] U. A. Umar, M. K. A. Ariffin, N. Ismail and S. H. Tang, Hybrid multiobjective genetic algorithms for integrated dynamic scheduling and routing of jobs and automated-guided vehicle (AGV) in flexible manufacturing systems (FMS) environment, International Journal of Advanced Manufacturing Technology. 81 (9-12) (2015) 2123-2141, https://doi.org/10.1007/s00170-015-7329-2.

[16] A. Hammad, O. Salem, M. Hastak and M. Syal, Decision support system for manufacturing housing facility layout, Journal of Architectural Engineering. 14 (2) (2008) 36-46, https://doi.org/10.1061/(ASCE)1076-0431(2008)14:2(36).

[17] H. Hosseini-Nasab, H. Fereidouni, S. M. T. F. Ghomi and M. B. Fakhrzad, Classification of facility layout problems: a review study, International Journal of Advanced Manufacturing Technology. 94 (1-4) (2018) 957-977, https://doi.org/10.1007/s00170-017-0895-8.

[18] G. C. Armour and E. S. Buffa, A heuritic algorithm and simulation approach to relative location of facilities, Management Science. 9 (2) (1963) 294-309, https://doi.org/10.1287/mnsc.9.2.294.

[19] T. C. Koopmans and M. Beckmann, Assignment problems and the location of economic activities, Econometrica. 25 (1) (1957) 53-76, https://doi.org/10.2307/1907742.

[20] Y. A. Bozer and R. D. Meller, A reexamination of the distance-based facility layout problem, IIE Transactions. 29 (7) (1997) 549-560, https://doi.org/10.1023/A:1018597329398.

[21] K. Y. Wong, Applying ant system for solving unequal area facility layout problems, European Journal of Operational Research. 202 (3) (2010) 730-746, https://doi.org/10.1016/j.ejor.2009.06.016.

[22] G. Aiello, G. L. Scalia and M. Enea, A non dominated ranking Multi Objective Genetic Algorithm and electre method for unequal area facility layout problems, Expert Systems with Applications. 40 (12) (2013) 4812-4819, https://doi.org/10.1016/j.eswa.2013.02.026.

[23] J. Liu, H. Zhang, K. He and S. Jiang, Multiobjective particle swarm optimization algorithm based on objective space division for the unequal-area facility layout problem, Expert Systems With Applications. 102 (2018) 179-192, https://doi.org/10.1016/j.eswa.2018.02.035.

[24] A. L. Jaimes and C. A. C. Coello, Many-objective problems: Challenges and methods, in Springer Handbook of Computational Intelligence. (2015) 1033-1046, https://doi.org/10.1007/978-3-662-43505-2_51.

[25] H. Ishibuchi, N. Tsukamoto, and Y. Nojima, Evolutionary many-objective optimization: A short review, Proceedings of 2008 IEEE Congress on Evolutionary Computation. (2008) 2424-2431, https://doi.org/10.1109/CEC.2008.4631121.

[26] T. M. Emmerich, A. H. Deutz, A tutorial on multiobjective optimization: fundamentals and evolutionary methods, Natural Computing. (2018) 1-25, https://doi.org/10.1007/s11047-018-9685-y.

[27] N. Srinivas and K. Deb, Multiobjective optimization using nondominated sorting in genetic algorithms, Evolutionary Computation. 2 (3) (1994) 221-248, https://doi.org/10.1162/evco.1994.2.3.221.

[28] K. Deb, A. Pratap, S. Agarwal and T. Meyarivan, A fast and elitist multiobjective genetic algorithm: NSGA-II, IEEE Transactions on Evolutionary Computation. 6 (2) (2002) 182-197, https://doi.org/10.1109/4235.996017.

[29] J. T. Alander, On optimal population size of genetic algorithms, Proceedings of CompEuro 92. (1992) 65-70, https://doi.org/10.1109/CMPEUR.1992.218485.