Master's Thesis of Engineering

Optimization of Short-term Scheduling in Flat Block Assembly Line using Constraint Programming and Discrete-Event Simulation

평블록 조립 공정 실행계획 최적화를 위한 시뮬레이션 기반 제약 프로그래밍 연구

August 2023

Graduate School of Engineering
Seoul National University
Naval Architecture and Ocean Engineering Major

Dong Hoon Kwak

Optimization of Short-term Scheduling in Flat Block Assembly Line using Constraint Programming and Discrete-Event Simulation

Thesis supervisor: Prof. Jong Hun Woo

Submitting a master's thesis of Engineering

August 2023

Graduate School of Engineering
Seoul National University
Naval Architecture and Ocean Engineering Major

Dong Hoon Kwak

Confirming the master's thesis written by
Dong Hoon Kwak
August 2023

Chair _____(Seal)
Vice Chair _____(Seal)
Examiner _____(Seal)

Abstract

Scheduling flat block assembly line in a shipyard is crucial for overall shipbuilding performance in terms of its high volume of workload. This problem is commonly known as the Permutation Flow-shop Scheduling Problem (PFSP) in Operation Research (OR), which has been extensively studied in various papers since the 1950s. However, existing solutions often involve simplifying real-world problems with certain assumptions, limiting their practical applicability.

In recent times, Constraint Programming (CP) has emerged as a strong alternative to exact algorithms and has been successfully applied to various PFSP problems, addressing the limitations of exact algorithms. In light of this, our study proposes a two-step optimization process to overcome these limitations. First, a novel CP algorithm is introduced to incorporate actual industrial constraints. The modelled PFSP can be categorized as a Multi-Objective PFSP with hard due date constraint (MOPFSP-hd). Next, the feasibility and objective value of the optimized solution is validated using Discrete-Event Simulation (DES).

To evaluate the performance of our proposed framework, two industrial cases are conducted. The experimental results from both cases demonstrated an improvement in makespan compared to manually planned schedule. Additionally, the solutions derived from

our proposed model are reported to be feasible, while the manually planned schedules are often infeasible either due to not satisfying industrial constraints or encountering delays. Finally, the difference between the objectives calculated from CP and DES model is analyzed quantitatively using Critical Path Method (CPM).

The proposed solution in this paper presented a practical and effective approach to address PFSP with real-world constraints. By combining CP and DES techniques, the authors demonstrated improved makespan and feasible schedules compared to traditional methods using two industrial cases.

Keyword: Optimization, Constraint programming, Discrete-event simulation, flat block assembly, shipbuilding
Student Number: 2020-21233

Table of Contents

Chapter 1. Introduction 1

Chapter 2. Theoretical Backgrounds 6
2.1. Permutation Flow-shop 6
2.1.1 Variables in the CP Model 6
2.1.2 Variables in the CP Model 10
2.2. Constriant Programming 12
2.3. Discrete-Event Simulation 14

Chapter 3. Development of Hull Block Assembly Line Model 18
3.1. Problem Definition 18
3.2. Constriant Programming Model 25
3.2.1 Variables in the CP Model 27
3.2.2 Constraints and Objectives in the CP Model 31
3.3. Discrete-Event Simulation Model 36
3.4. Optimization Strategy 40

Chapter 4. Performance Analysis 44
4.1. Case Studies 44
4.2. Result Analysis 46
4.2.1 Result Analysis in Case 1 47
4.2.2 Result Analysis in Case 2 60

Chapter 5. Conclusion 69

Bibliography 71

Abstract in Korean 75

List of Figures

[FIgure 1] An example of makespan minimization in PFSP.....7
[Figure 2] Simcomponent framework introduced by Nam et al. (2022) 16
[FIgure 3] Simple illustration of the target flat block assembly line 18
[FIgure 4] Characteristics of the operations in flat block assemblyline 19
[FIgure 5] Classification of various conditions according to two perspectives 25
[FIgure 6] Variables designed for the flat block assembly line CSP model 30
[FIgure 7] Example of operation time calculation according to work shift.....34
[FIgure 8] Simulation process and information of each class.....38
[FIgure 9] Hierarchical scheduling process using optimization and simulation model 43
[FIgure 10] Total objective according to the search time 49
[FIgure 11] Sub objective values according to solutions.....49
[FIgure 12] The critical path of solution 14 including block movement 56
[FIgure 13] Makespan comparison on potential solutions derived from CP of line B in case 1 58
[FIgure 14] Scatter plot of makespan(DES) according to makespan (CP) 64

List of Tables

[Table 1] Related studies of PFSP and uniqueness of the proposed method.....5
[Table 2] Spatial capacity of each stage.....19
[Table 3] Number of machines in machine driven operations.....20
[Table 4] Description of notations of the flat block assembly line model.....27
[Table 5] Number of variable according to the type of variable..29
[Table 6] Description of adjusted classes defined from Simcomponent.....36
[Table 7] Block composition according to the case 45
[Table 8] CSP search limit in each line.....46
[Table 9] Determined weight value of each objective 47
[Table 10] Potential solutions derived from CP of line A in case 1 48
[Table 11] Comparison between the best solution derived from CP model and manually planned schedule in case 1.....49
[Table 12] Feasibility and makespan results calculated using DES model of line A in case 1.....51
[Table 13] 32nd block from solution 7 simulation process analysis 52
[Table 14] 54th block from solution 7 simulation process analysis 52
[Table 15] Comparison between the best solution derived from DES model and manually planned schedule in case 1.....53
[Table 16] Discrimination of difference factor in the critical path 57
[Table 17] Potential solutions derived from CP of line B in case 1 58
[Table 18] Feasibility and makespan results calculated using DES model of line B in case 1 59
[Table 19] Combinations of feasible solutions from line A and B in

case 1 60
[Table 20] Potential solutions derived from CP of line A in case 2 61
[Table 21] Comparison between the best solution derived from CP model and manually planned schedule in case 2.....63
[Table 22] Feasibility and makespan results calculated using DES model of line A in case 2.....65
[Table 23] Comparison between the best solution derived from DES model and manually planned schedule in case 2.....65
[Table 24] Potential solutions derived from CP of line B in case 2 66
[Table 25] Feasibility and makespan results calculated using DES model of line B in case 2 67
[Table 26] Combinations of feasible solutions from line A and B in case 2 68

Abbreviation

B&B	*Branch and Bound*
CP	*Constraint Programming*
CPM	*Critical Path Method*
CSP	*Constraint Satisfaction Problem*
DES	*Discrete-Event Simulation*
FCFS	*First-Come-First-Served*
GA	*Genetic Algorithm*
ILS	*Iterated Local Search*
IP	*Integer Programming*
KPI	*Key Parameter Index*
MAC	*Maintaining Arc Consistency*
MILP	*Mixed-Integer Linear Programming*
MIP	*Mixed-Integer Programming*
MOPFSP	*Multi-Objective Permutation Flow-shop Scheduling Problem*
NEH	*Nawaz, Enscore, Ham*
OR	*Operation Research*
PFSP	*Permutation Flow-shop Scheduling Problem*
SA	*Simulated Annealing*

Chapter 1. Introduction

Hull block assembly process is one of the main processes in shipbuilding industry, where block parts and sub-blocks are assembled to form a larger block. The development of the ring-type erection process has allowed for the assembly of ship parts built in different shipyards, alleviating spatial constraints in the ship assembly process (Kim et al., 2005). However, from a cost and business perspective, shipbuilding companies still strive to perform shipbuilding processes within their premises as much as possible (Ahn & Kim, 2022). The hull blocks can be classified into two types based on their shape: flat blocks and curve blocks. As ship size has increased, most blocks are flat blocks (Yang et al., 2019). Therefore, scheduling flat block assembly line in shipyard is crucial for overall shipbuilding performance.

The flat block assembly line scheduling problem is considered as a typical Permutation Flow-shop Scheduling Problem (PFSP) in Operation Research (OR). Numerous papers have studied various types of PFSP using both exact algorithms and heuristics or meta-heuristic algorithms since the 1950s. For instance, Nagar et al. (1995) used the branch and bound (B&B) algorithm to solve the bi-objective PFSP with 2 machines. Framinan et al. (2003) conducted Nawaz, Enscore, Ham (NEH) heuristics to address the bi-objective PFSP with multiple machines (ranging from 5 to 25). Varadharajan and

Rajendran (2005) utilized a meta-heuristic algorithm, Simulated Annealing (SA), to optimize the bi-objective PFSP with 20 machines.

Several studies have applied the introduced algorithms on flat block assembly line in shipyards. Shie Gheun (1996) proposed a Genetic Algorithm (GA) to solve a PFSP problem in the case of block assembly shop in a shipbuilding company. Lee et al. (2009) also proposed a GA to optimize makespan on a dedicated assembly line using a simulation framework to calculate the exact makespan. Yang et al. (2019) adopted a multi-objective memetic algorithm for a parallel panel block assembly line. The authors formulated the line with fuzzy makespan, fuzzy processing time, and fuzzy due date to reflect the uncertainty of the process. However, these solutions had limitations in the model that failed to reflect real-world problems and instead simplified the problems with assumptions.

The limitations of conducting such algorithms in assembly lines in shipyard have left practitioners to schedule the plans based on their knowhow, relying on their knowledge and experiences. However, this had led to severe problems, such as a lack of consistent rules. Firstly, the problem has resulted in different qualities of schedules according to schedulers. This not only increased the company's reliance on certain schedulers but also put too much pressure on them to come up with better solutions without providing clear guidance. Secondly, it became an obstacle in the automation of scheduling (Kwak et al., 2022). Unclear strategies in scheduling have been a major obstacle to automate the process in

shipyards. It substantially increased the time spent on scheduling process whenever there was a need for a change in the schedule due to unexpected events or variations. Lastly, rule-based scheduling lacks validation of feasibility. The scheduling rules are based on the scheduler's experience and feedbacks from the shops, making it hard to guarantee the improvements in the quality of the solutions.

Recently, CP has been introduced as a competitor of exact algorithms, overcoming their limitations in handling problem complexity (Samarghandi & Behroozi, 2017). CP also offers high flexibility in formulating constraints mathematically and logically, enabling the representation of real-world constraints (Hooker, 2002). Therefore, this study aims to develop a CP model to solve the dedicated flat block assembly line with real-world constraints, referred to as "industrial constraints" in this paper. The modelled PFSP can be categorized as a Multi-Objective PFSP with hard due date constraint (MOPFSP-hd), and Discrete-Event Simulation (DES) is applied as a validation tool. Table 1 presents the related studies and their limitations, along with the aims of this study. In summary, the contributions of the paper are listed below:

1. A novel CP algorithm for MOPFSP-hd with industrial constraints are proposed. To the best of the author's knowledge, this is the first attempt to apply CP in MOPFSP-hd. Moreover, industrial constraints are not considered in most papers due to the complexity of the problem when modeled with traditional methods.

2. The proposed scheduling process is a 2-step scheduling process that integrates CP and DES model sequentially. The integration of the DES model aims to demonstrate the optimality and feasibility of the derived solution by validating it through simulation.
3. The superiority of the proposed model is demonstrated and analyzed using two actual industrial cases to assess its performance in a generalized context. The experimental results of the two cases prove the superiority of the model compared to current manually planned schedules.

The rest of the paper is organized as follows. Section 2 presents the related studies of the PFSP and the theoretical backgrounds of the proposed algorithms. In Section 3, the development of CP and DES models used to model the target flat block assembly line of this study is explained. Section 4 provides the experimental results and analysis of the two cases. Finally, the conclusion is discussed in Section 5.

Table 1 Related studies of PFSP and uniqueness of the proposed method

(IP : Integer programming, MILP : Mixed Integer Linear Programming, CP : Constraint Programming)

Related works	Multi-objectives	Due date constraint	Industrial constraints	Algorithms	Simulation	Machines	Problem size
Nagar et al. (1995)	**O (makespan + flow-time)**	×	Operational constraint (Two machine PFSP)	MP (Branch-and-bound)	×	2	10, 14
Framinan et al. (2003)	**O (makespan + flow-time)**	×	Operational constraint (General PFSP)	Heuristic (NEH)	×	5 ~ 25	5 ~ 100
Varadharajan & Rajendran (2005)	**O (makespan + flow-time)**	×	Operational constraint	Meta-heuristic (Simulated annealing)	×	20	20 ~ 100
Baker & Keller (2010)	× (tardiness)	×	Operational constraint (Single machine PFSP)	MP (Integer Programming)	×	1	40, 50
Lee et al. (2010)	× (makespan)	**O**	Operational constraint (Real factory environment)	Simulated annealing	**O**	10	140
Samarghandi & Behroozi (2014)	× (tardiness)	**O**	Operational constraint (General no-wait PFSP)	MP (MILP, CP)	×	5 ~ 40	5 ~ 20
Yang et al. (2019)	**O (makespan + agreement index)**	× (fuzzy)	Operational constraint (Real factory environment)	Memetic algorithm	×	8	20 ~ 50
This study (2023)	**O (makespan + workload balancing)**	**O**	**Operational constraint + Industrial constraint**	**CP**	**O**	**10**	**60**

Chapter 2. Theoretical Backgrounds

2.1. Permutation Flow-shop

2.1.1. Development of PFSP algorithms

Permutation flow-shop scheduling problem (PFSP) is a special type of flow-shop problem, where the processing order of jobs on the resources remains the same for each subsequent step of processing (Tseng & Stafford Jr, 2008). As a result, the job sequence plays a crucial role in determining the performance of the dedicated shop. The primary objective of PFSP is to determine the job sequence that minimizes the makespan. In industrial scenarios, however, it becomes necessary to consider various conditions and strategies, including makespan, tardiness, earliness, and idle time (Yenisey & Yagmahan, 2014).

Figure 1 presents an example of a general PFSP comprising 3 jobs and 4 operations. Each job follows a pre-defined order of operations, such as 1 → 2 → 3 → 4 in this particular case. In this example, each operation is assigned to a single machine, and there are no parallel operations. In Figure 1 (a), a Gantt chart depicts the scenario where jobs are fed to the shop in ascending order. The first job initiates its operation at $t = 1$, and the last job completes its operation at $t = 24$, resulting in a total makespan of 23 time units. On the other hand, Figure 1 (b) shows the situation where jobs are fed in descending order. The first job commences its operation at $t = 1$;

however, the last job finishes its operation between t = 22 and t = 23. This change in job sequence affects the makespan. It is worth noting that, in the absence of any other dynamic conditions, such as a stochastic operation time or re-entrant processes, the job sequence becomes the sole governing factor that influences the makespan.

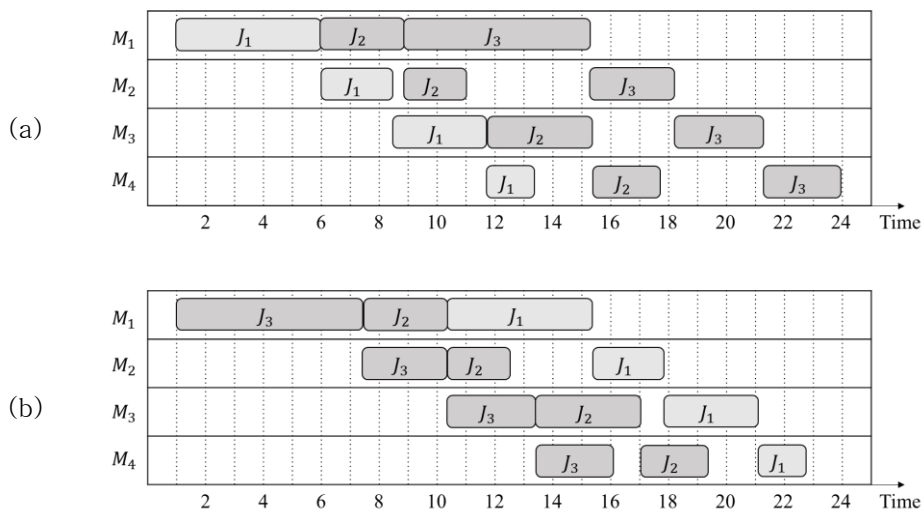

Figure 1 An example of makespan minimization in PFSP

A basic PFSP with 2 machines is initially introduced, and it is proven that the optimal sequence can be determined using a heuristic algorithm, known as Johnson's rule (Johnson, 1954). When the problem involves more than 2 machines, however, it is classified as an NP-Complete problem (Garey et al., 1976). This inherent complexity of the problem has motivated researchers not only to seek solutions using exact algorithms, but also to explore various heuristic and meta-heuristic algorithms, especially for larger-sized problems (Yenisey & Yagmahan, 2014).

Various algorithms and methodologies have been applied to solve both single and multi-objective PFSP. As mentioned above, Johnson's rule was initially used by Johnson (1954) to solve PFSP with 2 machines. Subsequent studies expanded Johnson's rule to address PFSP with multiple machines (Campbell et al., 1970; Dudek & Teuton Jr, 1964). Nawaz et al. (1983) introduced NEH heuristics, which demonstrated superior optimization capabilities compared to 15 existing heuristics, including Johnson's rule. Numerous comprehensive studies with standard PFSP have confirmed that NEH heuristic outperforms other existing heuristics (Framinan et al., 2004; Ruiz & Maroto, 2005). In addition to heuristic algorithms, various meta-heuristic algorithms, such as SA, GA, and tabu search, have been extensively explored in many studies (Chen et al., 1995; Osman & Potts, 1989; Widmer & Hertz, 1989; Zheng & Wang, 2003). Another metaheuristic algorithm called Iterated Local Search (ILS), was introduced by Stützle (1998) for PFSP and was compared with several other metaheuristics mentioned above (Ruiz & Maroto, 2005). The results showed that ILS outperformed other metaheuristic algorithms in this context.

On the other hand, there also have been many studies focusing on finding optimal solutions using exact algorithms for various PFSP. Exact algorithms encompass B&B algorithm, Integer Programming (IP), and mixed-integer programming (MIP). Nagar et al. (1995), Liao et al. (1997), and Lee and Wu (2001) utilized B&B algorithm in PFSP with two machines. Chou and Lee (1999) and Eren and Güner

(2008) implemented IP to tackle multi-objective PFSP two machine, while Selen and Hott (1986) applied MIP to bi-objective PFSP with two machine. These studies, however, predominantly focused on small-sized problems, which are often inadequate to address real-world problems effectively.

Recently, CP has emerged as a strong alternative to exact algorithms, and it has been successfully applied to various PFSP problems, addressing the limitations of exact algorithms (Rossi et al., 2006). For instance, Öztop et al. (2022) proposed and compared Mixed-Integer Linear Programming (MILP) models and a CP model for no-idle PFSP problem, and their findings indicated that the CP model outperformed the other models in terms of performance. Similarly, Karabulut et al. (2022) developed MILP and CP models for distributed PFSP with sequence-dependent setup times in small-sized problem, and the result demonstrated that the proposed CP model outperformed the MILP model in terms of solution time. These studies showcase the effectiveness and potential of CP as a viable alternative to exact algorithms in solving PFSP.

Among all research conducted in this field, very few studies have explored hard due date constraints (Samarghandi & Behroozi, 2017). "Hard due date constraint" refers to constraints where each part has its own specific due date, and has to be satisfied. In most studies, tardiness objective has been considered as an alternative to due date constraints (Blazewicz et al., 2008; Brah, 1996; Gowrishankar et al., 2001; Hunsucker & Shah, 1992), because incorporating due date

constraints into the PFSP is a challenging task (Perez-Gonzalez & Framinan, 2015).

In this study, the flat block assembly line scheduling problem can be classified as a multi-objective permutation flow-shop scheduling problem with hard due date constraint (MOPFSP-hd). This classification is supported by the following reasons:

1. Both the flat block assembly line scheduling problem and the permutation flow-shop scheduling problem share the same primary objective, which is the minimization of the makespan. However, in this study, the objective is not limited to makespan minimization; it also includes considerations for workload balancing and stock management.
2. Both problems involve ensuring that all parts (blocks) follow the same predetermined operation sequence in the shop (assembly line). Additionally, all parts strictly adhere to the First-Come-First-Served (FCFS) rule, without any skipping or re-entrant operations.
3. The problem in this study incorporates a hard due date constraint. Unlike most studies that deal with tardiness objectives, this study specifically aims to address the PFSP with a hard due date constraint.

2.1.2. Makespan in Permutation Flow-shop

In project-based industries, such as shipbuilding, the evaluation of successful projects revolves around three parameters: time, cost,

quality. Ships in the shipbuilding industry are constructed based on the specific requirements of individual contract. Makespan, in this context, serves as one of the essential Key Performance Indicators (KPIs) that reflects the lead time performance of the project. Reducing makespan directly correlates with reducing the lead time of a contract, making the minimization of makespan a critical aspect in the scheduling process. Achieving a shorter makespan is crucial to ensuring efficient project completion and meeting contractual deadlines.

In this study, makespan is further analyzed to distinguish the results obtained from the proposed CP and DES model. One of the most widely used analysis method for makespan is the Critical Path Method (CPM). The critical path represents the sequence of tasks that determine the minimum time required for the entire process. CPM calculates both critical and non-critical tasks through forward and backward procedures. The detailed steps of these procedures are outlined below,

Forward Procedure

Step 1.

Set time t = 0

Set $S_j' = 0$ *and* $C_j' = p_j$ *for each job* j *that has no predecessors*

Step 2.

Compute inductively for each job j

$$S_j' = \max_{\{all\ k \to j\}} C_k'$$

$$C_j' = S_j' + p_j$$

Step 3.

The makespan is $C_{max} = max(C'_1, \dots, C'_n)$.

STOP

where C'_j is earliest possible completion time of job j, S'_j is earliest possible starting time, p_j is processing time, and $\{all\ k \rightarrow j\}$ is the set of all jobs that are predecessors of job j. The backward procedure is,

Backward Procedure

Step 1.

Set time $t = C_{max}$.

Set $C''_j = C_{max}$ *and* $S''_j = C_{max} - p_j$ *for each job* j *that has no successors*

Step 2.

Compute inductively for each job j

$$C''_j = \min_{\{j \rightarrow all\ k\}} S''_k$$

$$S''_j = C''_j - p_j$$

Step 3.

Verify that $\min(S''_1, \dots, S''_n) = 0$.

STOP

where C''_j is latest possible completion time of job j, S''_j is latest possible starting time, and $\{j \rightarrow all\ k\}$ is the set of all jobs that are successors of job j.

2.2. Constraint Programming

Constraint Programming (CP) is recognized as a powerful technology for solving practical problems represented in the form of a Constraint Satisfaction Problem (CSP) (Rossi et al., 2006). The CSP is defined by three main elements:

- Variable: $\mathrm{X} = \{x_1, \dots, x_n\}$,

- Domain: $\mathrm{D} = \{d_1, \ldots, d_n\}$,
- Constraint: $\mathrm{C} = \{c_1, .., c_m\}$

where x_i represents the decision variable of CSP, d_i represents the possible range of each variable, and c_j restricts the possible range of variables by specifying relationships between the variables.

The primary objective of CP is to find a feasible solution to a given CSP (Hooker, 2002). However, it is now extensively employed as an optimization tool by adapting an objective function. Constraint Programming, in essence, refers to the computer implementation of an algorithm designed for solving CSPs (Brailsford et al., 1999). Some well-known algorithms in CP include backtracking, forward checking, and Maintaining Arc Consistency (MAC) as mentioned in Brailsford et al. (1999).

The three algorithms all rely on tree search techniques. The process of searching involves assigning values to variables, which are within the defined domain and satisfy all constraints. In other words, selecting a branch in the tree corresponds to specifying the value of the next variable.

The distinction between the algorithms lies in the search process when the domain of the next branch variable no longer exists. The backtracking method selects a different branch if the selected variable, along with previously selected variables, violates any related constraints. On the other hand, the forward checking and MAC consider the relative constraints among the variables of the current branch, past branches, and the next branch. As a result, the algorithm

backtracks the branch when a single related variable fails to have a feasible solution. Although forward checking and MAC may take longer to calculate due to considering all related variables, they effectively increase the search efficiency by not exploring the infeasible branches. The difference between forward checking and MAC is that MAC adopts chaining-arc-consistency, which checks constraints for the changed next branch's variable, thereby further reducing the domain of variables and detecting failures more quickly. Hooker (2002) exposits detailed descriptions of search techniques and classifies them according to theories of search.

In this study, the CSP of hull block assembly line is modeled, and CP is employed to search for an optimal solution. The detailed model is explained in Section 3.2.

2.3. Discrete Event Simulation

Simulation is a virtual environment that replicates the operation of a real-world process or system over time (Banks, 2005). It is commonly utilized as a virtual experiment environment because of its flexibility, controllability, time-effectiveness, and cost effectiveness (Robinson, 2014). Simulation is categorized into two types: DES and continuous simulation, based on the manner in which state changes are handled (Banks & Carson, 1986). In a DES model, the state changes occur only at specific discrete points known as event times, while continuous simulation model allows for state changes to occur

continuously and smoothly over time. The continuous simulation, however, require a substantial amount of calculation time compared to DES.

DES has proven to be an effective testbed in various studies, providing a reflection real-world factory operation. For example, Rogers and Brennan (1997) utilized DES as an experimental testbed to evaluate alternative control architectures. Brennan and William (2000) developed a simulation testbed to assess a distributed multi-agent control architecture for holonic manufacturing systems. Huynh et al. (2020) presented a DES model of a manufacturing process to generate data for calculating production KPIs. This model served as the baseline for the factory performance management, and was further utilized to optimize production flow. Lee et al. (2009) employed DES to model a panel block assembly line in a shipyard for calculating the makespan of scheduled block sequences using a GA. These examples demonstrate the versatility and practicality of using DES as a valuable tool in the evaluation, management, and optimization of various industrial processes.

In this study, a specialized DES framework for factory simulation is utilized to model the assembly line. Figure 2 illustrates the Simcomponent framework introduced by Nam et al. (2022). The framework consists of several modules, each serving a specific function in the simulation process.

1. Adapter Module: the adapter module is responsible for data preprocessing, converting raw data into a format suitable for

simulation data input.

2. Modeler Module: The modeler module serves as a connection between the preprocessed data and simulation. In ensures that the data is correctly integrated into the simulation model
3. Simulation Module: The simulation module defines the components of modeled simulation. It includes the DES kernel, where the actual simulation is performed.
4. Analyzer Module: After the simulation is completed, the analyzer takes charge of the post-processing step. It calculates KPIs that represent the performance of the simulation.
5. Reporter Module: The reporter module is responsible for saving the simulation result and calculated KPIs in a separate file, making them easily accessible for analysis and reporting.

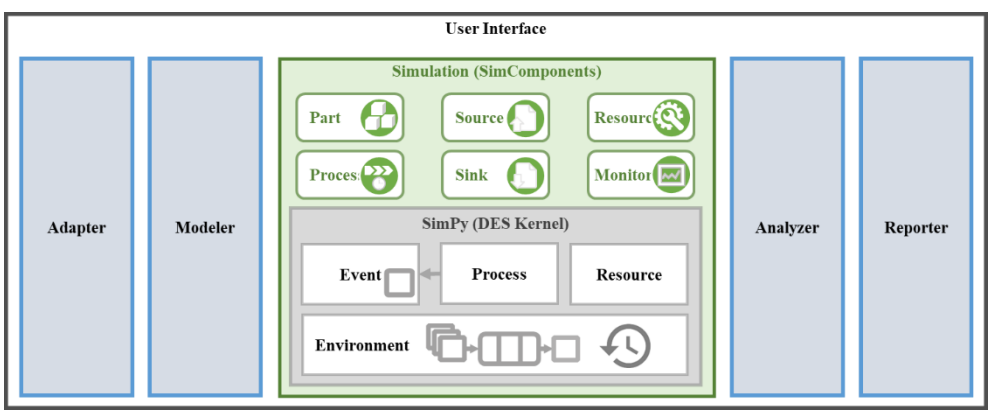

Figure 2 Simcomponent framework introduced by Nam et al. (2022)

The simulation module comprises several components, each representing a crucial aspect of the designed factory. The components are Part, Source, Resource, Process, Sink, and Monitor.

The part class represents the part that flows in the designed factory. It contains all the part information that simulation requires. The source class generates part in pre-defined or random sequence in specific time. The process class defines detailed process of part in dedicated operation, and the resource class determines the machine number in the process. The sink class eliminates the part from the system if needed. Lastly, the monitor class stores all the event logs and information that user assigns. This information is sent to analyzer for further analysis when the simulation is completed. Specific classes defined to model the flat block assembly line is mentioned in Section 3.3.

Chapter 3. Development of Hull Block Assembly Line Model

3.1. Problem Definition

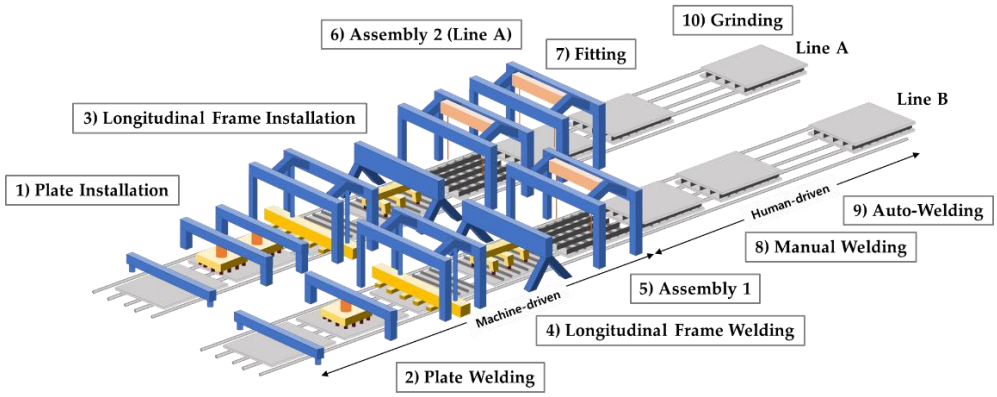

Figure 3 Simple illustration of the target flat block assembly line

Shipbuilding process involves the assembly of multiple hull blocks, and each hull block is composed of several sub-blocks, with most of them being flat blocks (Yang et al., 2019). This characteristic of shipbuilding introduces the possibility of the flat block assembly becoming a bottleneck process. Figure 3 provides an illustration of the target flat block assembly line that is the focus of this research. The assembly line comprises two lines; the Grand-assembly line (Line A) and the Unit(Sub)-assembly line (Line B). Line A consists of 10 operations, while Line B has 9 operations, excluding assembly 2. As all blocks entering each line follow the same order of operations, the flat block assembly line can be classified as a permutation flow-shop scheduling problem. Hence, the main objective of the flat block

assembly line scheduling problem is to minimize the makespan and prevent it from becoming a bottleneck in the shipbuilding process.

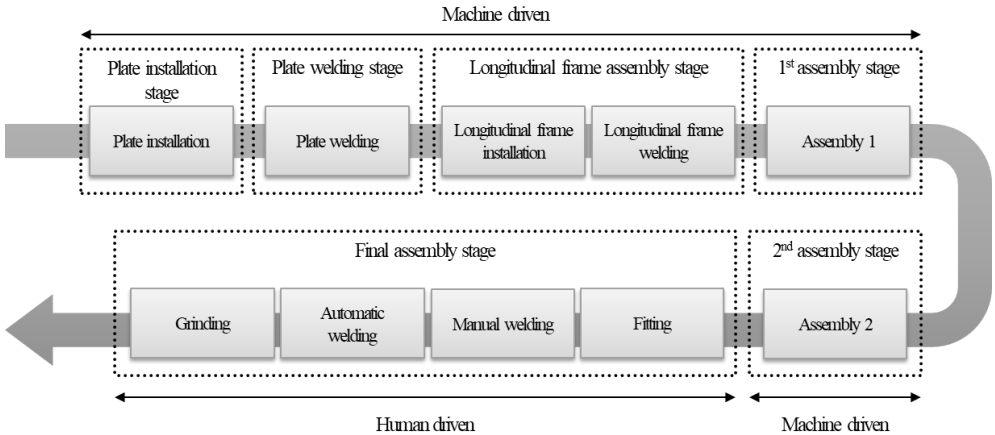

Figure 4 Characteristics of the operations in flat block assembly line

Figure 4 provides a detailed description of the characteristics of each operation in flat block assembly line. The operations and their sequence in the process are listed within a gray box, along with their respective names. Each operation is carried out at a specific location on the line, referred to as a "stage". The dotted box represents the spatial constraint, where operations within the same box share the same stage. As the assembly line operates, there is a limited capacity for blocks to be considered. To address this capacity constraint, the sum of lengths of blocks in each stage is limited to the stage length. The spatial capacity of each stage is shown in Table 2.

Table 2 Spatial capacity of each stage

Stage	Capacity (m)	
	Line A	Line B
Plate installation	30	30
Plate welding	64	64
Longitudinal frame assembly	53	53

1st assembly	31.5	24
2nd assembly	32	-
Final assembly	131	160

The operations can be classified into two categories based on the level of automation: machine-driven operations and human-driven operations. Machine driven operations are mostly automated, with required machines regulating the capacity of the operation. Table 3 provides the number of machines for each machine-driven operation. All operations, except for plate welding, have only one machine, indicating that each operation can process a single block. However, in the plate welding process, there are 3 and 2 machines in each line, respectively, and only two machines could be assigned to one block due to spatial limitation. As a result, the operation in Line B could be considered as a single-machine process, while Line A should be considered as a parallel machines process. The machine assignment in Line A follows a strict logic to minimize the makespan, adhering to the FCFS rule.

Table 3 Number of machines in machine driven operations

Stage	Machine	
	Line A	Line B
Plate installation	1	1
Plate welding	3	2
Longitudinal frame installation	1	1
Longitudinal frame welding	1	1
1st assembly	1	1
2nd assembly	1	-

The worker-driven operations rely on the work schedule to

determine their process time. The schedule is divided into two shifts: day and night, and the number of workers varies for each shift. The total man-hours required for each operation are provided based on task types and materials, and the process time can vary depending on the start time of the operation, as the number of workers is determined by the work schedule. In some operations, no workers are assigned during the night shift. The daily working hours are typically 10 hours for the day shift and 8 hours for the night shift. Although an operation is machine driven, there is still a need for workers to be assigned to manage the process. As a result, the operation time of all operations follows the work schedule, and the availability of workers during different shifts can impact the process time for each operation.

From a scheduling perspective, several sequential rules are applied to ensure operational efficiency and achieve business objectives. The first rule is the "set block" rule. Since a ship hull is axisymmetric, port and starboard blocks are identical and referred as "set blocks". These blocks are assembled simultaneously in a berth and involve similar tasks. To optimize operational efficiency and stock management, it is advantageous to process these block set at the same time. Therefore, these set blocks should be scheduled consecutively.

The second rule pertains to nonconsecutive blocks that should be kept separate within the sequence. These block sets require a significant amount of operation time. By keeping these blocks

separate, workload balancing is achieved not only in the assembly line, but also in subsequent processes.

Finally, there are scheduled blocks and operating blocks in the factory. These block sequence and factory conditions should be carefully considered in the operational scheduling step, as the feasibility of the completed schedule depends on their proper evaluation and integration. By incorporating these sequential rules, the scheduling process aims to enhance overall operational performance and ensure the successful execution of the flat block assembly line.

The block sequence scheduling process involves three distinct objectives. The first objective is the minimization of makespan. This objective focuses on reducing the total lead time and cost by minimizing the makespan and optimizing operation utilization. By achieving a shorter makespan, the overall production process becomes more efficient, leading to cost savings and timely project delivery.

The second objective is workload balancing between the two teams in the assembly line. In each operation, there are two teams, each assigned to blocks alternatively. The objective is to evenly distribute the workload between these two teams. This balancing mechanism complements the "set block" rule mentioned earlier, as set blocks typically have a comparable workload. Similarly, set blocks with substantial amount of workload should be placed apart by an even number of cells in the sequence, further enhancing workload

balance.

The third objective is related to due date objective. Blocks with earlier due dates are prioritized and positioned ahead in the sequence. This objective aligns with the principles of lean manufacturing, aiming to minimize block stock in the shipyard and prevent delays in subsequent processes. By giving priority to blocks with imminent due dates, the production process becomes more responsive and adaptive to changing project requirements.

In the scheduling process for the assembly line, the makespan minimization objective is applied to both line A and line B. However, line B does not consider the other two objectives, namely workload balancing and due date prioritization. The reason for this difference is that unit and sub-assembly blocks in line B have similar workloads for most block types. Additionally, their sizes are relatively small compared to grand assembly blocks processed in line A. As a result, stock management is not as important for line B as it is for line A. Considering these factors, the optimization model for line A considers all three objectives: makespan minimization, workload balancing between the two teams in the line, and due date prioritization to achieve lean manufacturing principles. On the other hand, the optimization model for line B only focuses on makespan minimization, as the other objectives are not as relevant for this specific line.

In this study, the goal is to solve the real-world problem of flat block assembly line scheduling by considering all the conditions that schedulers encounter in the factory and scheduling process. To

achieve this, two categories are classified from the various conditions: general PFSP conditions and industrial conditions. Figure 5 shows the classification of various conditions according to general PFSP or industrial conditions.

The general PFSP conditions include constraints and objectives commonly found in typical PFSPs, such as process time, operation capacity, FCFS rule, due date, and makespan minimization. These conditions form the foundation of the problem and are considered in the optimization model.

On the other hand, the industrial conditions are specific to the flat block assembly line in the shipbuilding industry and include spatial arrangement, shift calendar, set block, separate block, and scheduled block rules, workload balancing, and due date priority. These conditions are critical in reflecting the real-world complexities of the assembly line and are incorporated into both the optimization model and the simulation model.

However, due to the computational complexity of considering all industrial conditions in the optimization model, some simplifications are made to improve search efficiency. These simplifications are then validated in simulation model, which provides a more detailed and accurate representation of the assembly line's performance under real-world conditions. By comparing the result between the optimization model and the simulation model, the effectiveness of the proposed scheduling approach in solving the actual industrial problem is ensured.

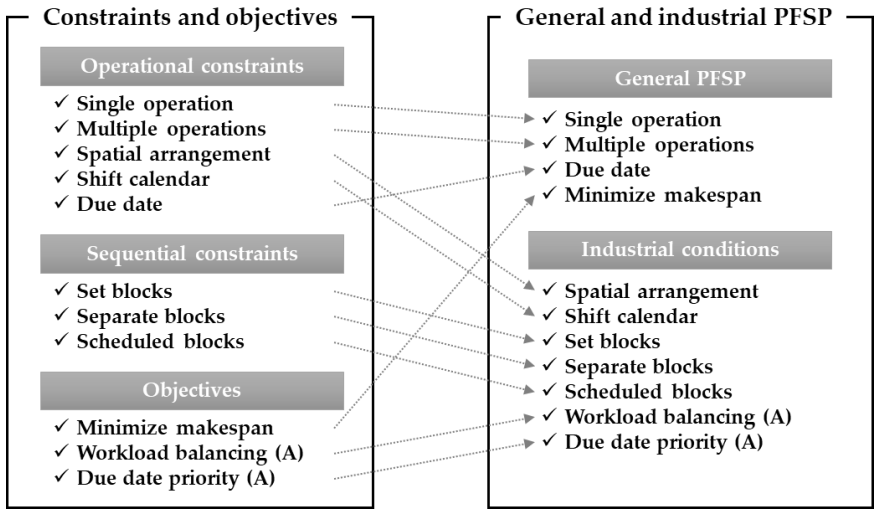

Figure 5 Classification of various conditions according to two perspectives

3.2. Constraint Programming Model

In the CP model, some conditions are simplified to improve the efficiency of the search speed. Three specific simplifications are made in this study.

First, block movement is not considered in the CP model. Block movement time depends on the sequence and process time of each block, making it a dynamically determined variable that adds complexity to the model. Due to this complexity, block movement is not explicitly modeled in the CP model.

Second, parallel machine process in plate welding process (Line A) is simplified as a single machine process with 3 machines. In line A, the plate welding process involves 3 welding machines, and each block can be processed using 2 machines simultaneously. However, defining resource variables and assigning process times of each

resource dynamically significantly decrease the search efficiency of the CP model. To mitigate this, a simplification is applied by assuming that all machines can be used for a single block at the same time until the operation is finished. As a result, the operation time (p_{ij}) of the plate welding is divided by 3 in the CP model.

Third, the dynamic operation time of assembly 1 and 2 (Line A) is also simplified for computational efficiency. In the actual assembly line (line A), the operation time of assembly 1 and 2 is dynamically determined during the processing to minimize the makespan. Specifically, the assembly 1 and 2 operation can be divided into 3 sub-tasks: assembly 1, assembly 2, and assembly 1 or 2. The operation time of assembly 1 or 2 is dynamically assigned to assembly 1 or assembly 2 stage, ensuring minimization of the makespan. The CP model, however, this dynamic condition is simplified by dividing the total operation time of assembly 1 or 2 equally between assembly 1 and assembly 2.

By applying these simplifications, the CP model can still capture the general characteristics of the assembly line and produce feasible solutions while reducing the complexity of the optimization problem. However, it is important to note that these simplifications may introduce some differences between the results obtained from the CP model and the DES model. As such, the DES model is used to validate and compare the results, ensuring that the simplifications made in the CP model do not significantly impact the quality of the solutions. The analysis of these differences provides valuable insights into the

performance of the proposed scheduling approach for the flat block assembly line.

In The CP model, the problem is formulated as a CSP. The variables, domains, constraints, and objective functions are defined to represent the scheduling problem for the flat block assembly line. The notations used in the developed CSP are described in Table 4.

Table 4 Description of notations of the flat block assembly line model

Symbol	Description	Symbol	Description
I	Set of blocks	N_j	Machine (worker) number of operations j
I_i^p	Set of blocks which block due date$> d_i$	N_B	Number of blocks
i	Block number	MH_i	Man-hour of block i
j	Operation number	B_t^k	Set of blocks in stage k in time t
k	Stage number	L_i	Length of block i
λ	Weight value of objective	L_k	Length of stage k
m	Sequence number	d_i	Due date of block i
α	Cell distance parameter	$S(J_{ij})$	Start time of J_{ij}
i, i'	Set block	$F(J_{ij})$	Finish time of J_{ij}
J_{ij}	Interval variable of operation j in block i	$N(J_{ij})_t$	Number of jobs (J_{ij}) in time t
A_{ik}	Interval variable of stage k in block i	Var_{Seq}	Sequential variable
T_{ij}	Length of interval variable (J_{ij})	Var_{seq}^{ref}	Sequential variable that order of variable is predefined
p_{ij}	Total operation time of process j in block i	$Cost$	Total man-hour assigned to the team

3.2.1. Variables in the CP Model

Indeed, the search efficiency of CP heavily depends on problem modeling and the number of variables considered in the CSP. Minimizing the number of variables while still capturing all the

relevant constraints and objectives is essential for improving the search efficiency. In the context of the flat block assembly line, three types of variables are required in the CSP model: operational variable (J_{ij}), spatial variable (A_{ik}), and sequential variable (Var_{seq}). J_{ij} represent the jobs (blocks) and their assignments to specific machines at specific times. They capture the initial times of the operations, processing times, and completion times. A_{ik} represent the spatial arrangement of the blocks in the assembly line. They determine which stage the block is located, considering the spatial constraints and capacity limitations of each stage. Var_{seq} represent the block sequence and capture the order in which the blocks are processed on the assembly line. They ensure that the sequence of blocks follows the specified rules and objectives, such as the set block rule and separate block rule.

Figure 6 provides an overview of the designed variables required, and Table 5 summarizes the total number of variables in the model. Each block has 10 operation variables representing the start and finish time of each operation. To efficiently manage these parameters, the "Interval_var" in ILOG CPLEX Optimizer is used. Additionally, there are 6 spatial variables on each block representing the start and finish time of stage occupation. These variables also utilize the "Interval_var" to effectively model the spatial arrangement of blocks in the assembly line. A sequence variable and reference sequence variable is also defined to implement sequential constraints such as set blocks, separate blocks, and scheduled blocks. However,

the sequence variable itself cannot represent the order of specific blocks. Therefore, a reference sequence variable is introduced, which contains a predefined order of blocks to represent the order number of blocks in the sequence variable.

In the developed CSP model, the domains of variables are not explicitly defined, but instead, it is constrained by various operational constraints such as FCFS rule and due date constraint. These constraints define the allowable values for the variable dynamically, ensuring that solutions satisfy the specific operational requirements. For most variables in the model, the initial domain is defined as $[0, max\ date]$, where $max\ date$ represents the maximum possible time or deadline in the scheduling problem. The scheduled blocks and operating blocks, however, entering time into the assembly line is known in advance. In the model, this information is used to assign the entering time as the start time of the first operation variable, which corresponds to the plate installation.

Table 5 Number of variables according to the type of variable

Variable type	Number of variables
Operation	$10N$
Space	$6N$
Sequence	1
Reference	$N+1$
Total	$17N+1$

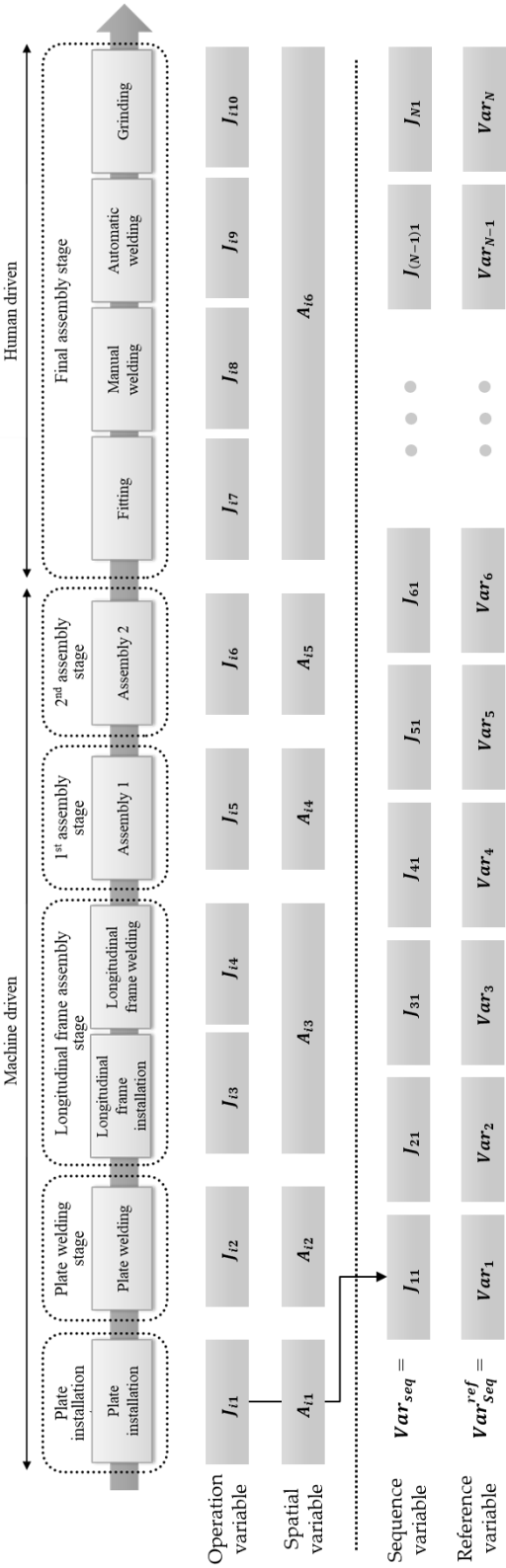

Figure 6 Variables designed for the flat block assembly line CSP model

3.2.2. Constraints and Objectives in the CP Model

Constraints and objectives in this case are explained in Figure 5. They are mathematically and logically formulated using variables defined in the Section 3.2.1. The formulations of the constraints and objectives are shown below:

$$Minimize\left(\lambda_1 Obj_{makespan} + \lambda_2 Obj_{balance} + \lambda_3 Obj_{duedate}\right) \quad (1)$$

s.t.

$$Obj_{makespan} = F_M - S_M \quad (2)$$

$$Obj_{balance} = |Cost_{odd} - Cost_{even}| \quad (3)$$

$$Obj_{duedate} = \sum_{i_1}\sum_{i_2}\left(S(J_{i_1 1}) - S(J_{i_2 1}) > 0\right),\ \forall i_2 \in I_{i_1}^{D} \quad (4)$$

$$\begin{cases} S_M = argmin_{S(J_{ij})}\left(J_{ij}\ \forall(i,\ j)\right) \\ F_M = argmax_{F(J_{ij})}\left(J_{ij}\ \forall(i,\ j)\right) \end{cases} \quad (5)$$

$$\begin{cases} Cost_{odd} = \sum\left(\left(S(J_{i1}) = S\left(Var_m^{ref}\right)\right) \times MH_i\right), m \in \{m | m = 2n-1, n \in \mathbb{N}\} \\ Cost_{even} = \sum\left(\left(S(J_{i1}) = S\left(Var_m^{ref}\right)\right) \times MH_i\right), m \in \{m | m = 2n, n \in \mathbb{N}\} \end{cases} \quad (6)$$

$$T_{ij} = F(J_{ij}) - S(J_{ij}) = \begin{cases} p_{ij}/N_j \\ f(p_{ij}, N_j) \end{cases} \quad (7)$$

$$N(J_{ij})_t \le 1,\ \forall i \in I \quad (8)$$

$$F(J_{ij}) \le S(J_{i(j+1)}),\ \forall(i,j) \quad (9)$$

$$\left\{\left(S(J_{i_1 1}) - S(J_{i_2 1})\right) \times \left(S(J_{i_1 j}) - S(J_{i_2 j})\right)\right\} > 0,\ \forall j \in \{2, 3, \dots, 10\} \quad (10)$$

$$\sum_{(i,j)\ \in\ B_t^k}^{(i,j)} L_i \le L_k \quad (11)$$

$$\begin{cases} S(A_{ik}) \le S(J_{ij_1}) \\ F(J_{ij_2}) \le F(A_{ik}) \end{cases},\ \forall\ (j_1, j_2,\ k) \in \{(1,\ 1,\ 1),\ (2,\ 2,\ 2), \dots,\ (7,\ 10,\ 6)\} \quad (12)$$

$$F(A_{ik}) = S(A_{i(k+1)}) \quad (13)$$

$$\begin{cases} Line\ A\ : F(A_{i6}) \le d_i,\ \forall i \in I^A \\ Line\ B: \begin{cases} F(A_{i6}^B) \le S(A_{i4}^A) \\ F(A_{i5}^B) \le d_i \end{cases},\ \forall i \in I^B \end{cases} \quad (14)$$

$$Previous(J_{i1}, J_{i'1}) \quad (15)$$

$$Isomorphism\left(Var_{seq},\ Var_{seq}^{ref}\right) \tag{16}$$

$$\begin{cases} \left(S(J_{i1}) = S\left(Var_{m}^{ref}\right)\right) + \left(S(J_{i'1}) = S\left(Var_{m+2\alpha+1}^{ref}\right)\right) = 0 \\ or \\ \left(S(J_{i1}) = S\left(Var_{m}^{ref}\right)\right) + \left(S(J_{i'1}) = S\left(Var_{m+2\alpha+1}^{ref}\right)\right) = 2 \end{cases}, \tag{17}$$

$$\forall m \in \{1,\ 2,\ \ldots,\ N_B - 2(\alpha+1)\},\ \alpha \in \{1,\ 2,\ 3\}$$

The objective function (1) and relative sub-objective functions (2) ~ (6) ensure that the search algorithm aims for makespan minimization, workload balancing, and stock management. The coefficient λ can vary based on the scheduling strategy. Equations (2) and (5) represent the makespan minimization objective function, which is calculated as the difference between the minimum start time and the maximum finish time of all operations.

Equation (3) and (6) is the workload balancing objective function. As discussed in Section 3.1, there are two teams in each operation that take turns receiving block assignments. Therefore, the total workload of each team can be calculated using equation (6). This equation is formulated as a logical expression representing a boolean value that returns 1 if true and 0 otherwise. It' s expressed as a bracket $\left(S(J_{i1}) = S\left(Var_{m}^{ref}\right)\right)$.

The stock management function is formulated as equation (4). Individual block (i_1) counts all blocks (i_2) that have longer due date but are placed earlier in the sequence. Minimizing this value aims to arrange the blocks according to their due dates. Thus, this objective function adds all values of individual blocks to be considered in equation (1).

The constraints are classified as operational constraints and sequential constraints. The operational constraints are listed in equations (7) to (14), and sequential constraints are listed in equations (15) to (17). The operational constraints define the relationship between operation variables and spatial variables, and they consist of single operation constraints, constraints between operations and including stage, and constraints between the stages.

The single operation constraints are formulated as equation (7) and (8). Equation (7) determines the operation time according to operation type. For machine-driven operations, the operation time is defined as the total operation time (p_{ij}) divided by the number of machines (N_j) in the operation. As discussed in Section 3.2, all operations are considered to have only one block capacity, which forbids a parallel operation (Equation (8)). The operation time of a worker-driven operation is also defined similarly, but the number of workers differs according to shift time. Therefore, the calculation of the operation time according to the work shift schedule is presented as a function f in the formulation. Figure 7 shows an example of operation time calculation according to the work shift. Let's assume there is an operation with total operation time of 28 hours. The operation starts at 12 with 5 workers assigned, and workers are switched at 16 due to the work shift schedule. The remaining hour of workload is 8 hours with 4 workers, so the operation finishes at 18. This constraint is applied using intensity function, which is embedded function of "Interval_var" in ILOG CPLEX Optimizer.

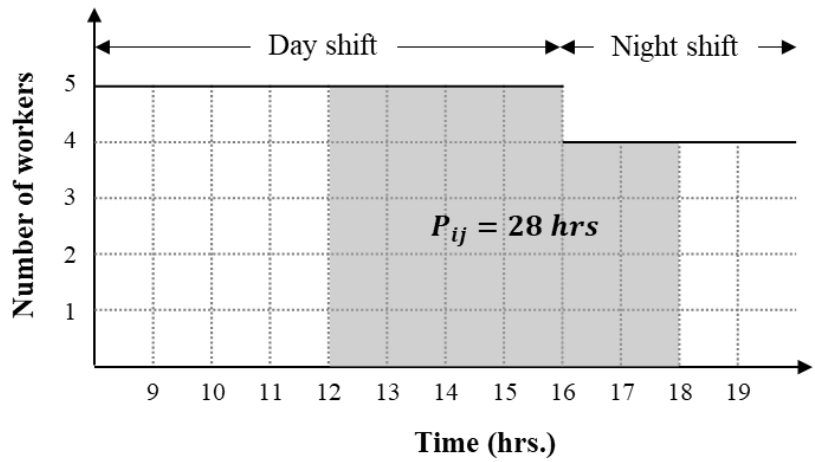

Figure 7 Example of operation time calculation according to work shift

The constraints between multiple operations and a stage are formulated as equation (9) to (12). Equation (9) defines the operation sequence of each block. Since the line can be categorized as a permutation flow shop, all blocks have the same operation sequence. Equation (10) regulates the block dispatching method on the operations as the FCFS rule. This ensures that the blocks are processed in the order they arrive at the operation. Equation (11) constrains the spatial capacity of the available blocks in a stage by length. In this model, blocks are assumed to be placed in a row, so the sum of the block lengths $(\sum_{(i,j) \in B_t^k}^{(i,j)} L_i)$ in a stage should not exceed the stage capacity (L_k). Equation (12) ensures the existence of a block in a stage until all operations in the stage are finished. There are single operation-single stage relationships, such as plate installation, plate welding, and assembly, and multiple operations-single stage relationships, such as longitudinal frame stage and final assembly stage. The notation (j_1, j_2, k) represents the relationships.

The constraints between the stages are formulated as equation (13) and (14). Equation (13) ensures the existence of blocks in the line, meaning that there should not be any gaps in time between the spatial variables in a block. This constraint guarantees that the blocks move continuously through the line without any interruptions. Equation (14) represents the due date constraint. The line A and line B have different due date constraints, because some blocks from line B are assembled with blocks on line A at assembly operation. Therefore, line A regulates the block due date with individually determined due dates, whereas line B discretizes the block if it is an input of line A or not, and regulates the due date accordingly.

The sequence constraints are shown in equations (15) to (17). Equation (15) uses "*Previous*" function, that is an embedded constraint in ILOG CPLEX Optimizer utilized with sequence variable. This constraint ensures that block i is scheduled previous to block i' in the sequence. The set block constraint is designed using this constraint. To design separate block constraint, there needs a distance variable that represents a distance between two blocks in a sequence should be available to express the even cell distance. To achieve this, a reference variable is designed to indirectly represent the cell distance. Equation (16) is the isomorphism constraint between the sequence variable and the reference variable, which creates a one-to-one correspondence relation between the set of two interval variables. The sequence of interval variables in the reference variable is known, and it indirectly addresses the sequence

of interval variables in the sequence variable using a boolean value, $\left(S(J_{i1}) = S\left(Var_m^{ref}\right)\right)$. Equation (17) shows the separate block constraint. This constraint ensures that block i and block i' should have cell distance in one of 2, 4, and 6 between them.

3.3. Discrete-Event Simulation Model

The simulation model is built using the Simcomponent framework, as shown in Figure 2, which was introduced by Nam et al. (2022). To simulate the assembly line, seven classes of factory simulation components are adjusted to fit the specific problem. Table 6 provides a description of these classes defined for the assembly line. All classes, except the process class, have analogous purposes and structures with the classes in Simcomponent. The process class is further divided into two classes: the operation class serves as a workspace where operations are processed, and the stage class functions as a physical line where spatial movement is considered.

Table 6 Description of adjusted classes defined from Simcomponent

Classes	Adjustment	Description
Part	Block	Contains block information
Source	Source	Creates block object and feed to first stage
Process	Operation	A workspace where operation is processed
	Stage	Physical line where spatial movement is considered
Resource	Worker	Controls the number of workers based on shift calendar
Sink	Sink	Eliminates finished objects from the line when delivered

Monitor	Monitor	Records all events occurred in a scenario

The simulation process involving the classes are shown in Figure 8. Initially, a block object is created by the source class, based on either a predetermined schedule or a block sequence generated from the optimization model. This block object contains essential block specifications and status data that are tracked throughout the simulation process. The block is then stored in a designated storage area and awaits its turn to move to the next stage storage. Within the stage class, there are two storage areas: “storage (in)” and “storage (out)”, which are used to control the movement of blocks. The “storage (in)” assesses the available space in the stage, comparing the stage length with the lengths of blocks currently present. When sufficient space is available, the “storage (in)” either receives a new block from the source storage or retrieves a block from the storage (out) of the previous stage. Subsequently, the block is transferred to the first operation storage within the stage once the operation becomes available. Upon completing the last operation in the stage, the block is moved to “storage (out)”, from where it can proceed to the next stage or be eliminated from the line, depending on whether all processes have been completed. Even the block is moved to the operation stage, the stage storage still take account to the block length until the block is transferred to next stage. Therefore, block occupation is promised during all processes. Ultimately, the blocks moved to the sink storage are eliminated from

the line, signifying the end of their simulation journey.

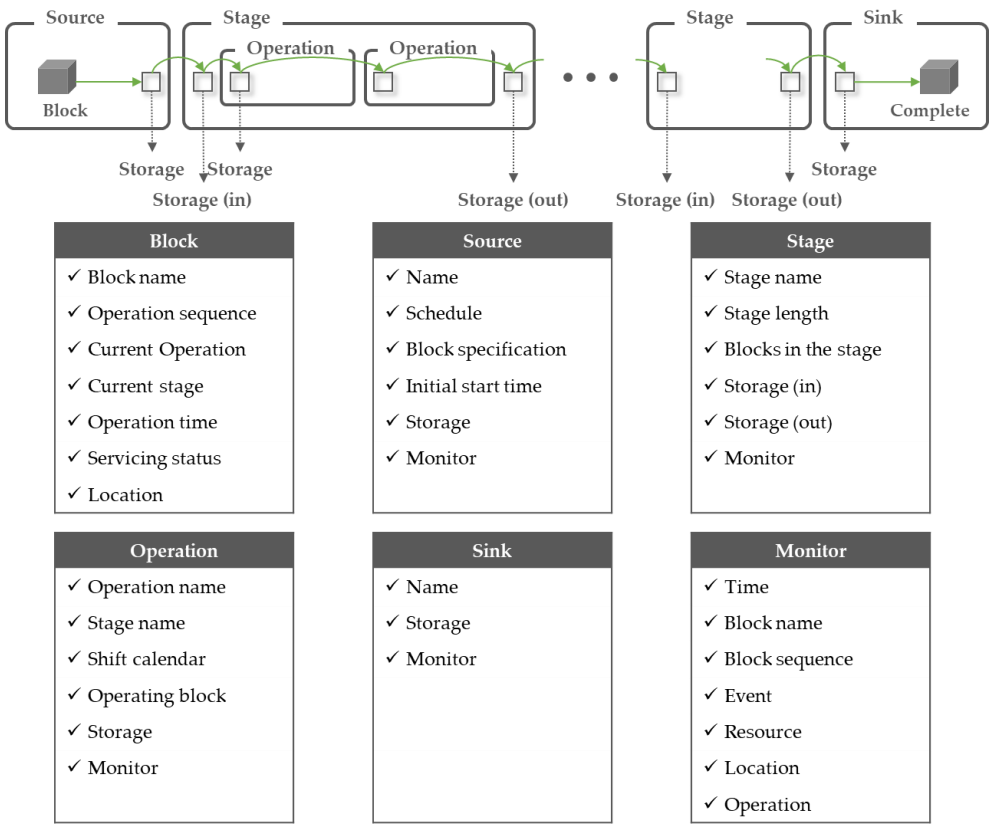

Figure 8 Simulation process and information of each class

As stated in Section 3.2, the CSP model simplifies or does not consider three important aspects for search efficiency: block movement, parallel operation in plate welding, and dynamic operation time in assembly 1 and 2. However, the simulation model aims to accurately reflect the reality and considers all processes without simplification. Consequently, the three processes are also incorporated in the simulation model based on the actual system's functioning principles.

In the proposed simulation model, the block movement is considered within the stage class. Each block object is assigned

location information as depicted in Figure 8. This information is updated when a block enters or leaves the stage. Upon entering the stage, the block is placed in the farthest available position. The block movement speed within the line is set at 1m/min. Conversely, when a block leaves the stage, all blocks in the stage move simultaneously to create space for the incoming block from the preceding stage. Even operating blocks stops their processes, moves first, and then resume their operations at the new position.

The simulation model appropriately incorporates the parallel operation in the plate welding within the process class by utilizing multiple resources. In this model, both machines and workers are represented as resources in the process class. Since all welding machines have the same specifications, a simple assignment rule based on the FCFS principle is adopted. The simulation ensures that no block can cut in line, meaning machines are assigned to blocks in the order of their arrival. Specifically, two machines are always assigned to the first block in the queue. If there are other blocks in line A, the second block takes the remaining machine. In line B, where there are only two machines, all machines are assigned to the first block.

Lastly, the dynamic operation time in assembly 1 and 2 in line A is considered using a simple rule. The subtasks of the assembly operation are distinguished as assembly 1, assembly 1 or 2, and assembly 2. While assembly 1 and assembly 2 tasks are exclusively processed in designated stage, the assembly 1 or 2 task can be

processed in either one. To minimize makespan and prevent the assembly operation from being a bottleneck, this task is processed in assembly 2 whenever it is available. This strategy avoids the situation where assembly 2 is idle while assembly 1 is working when the block could be processed in assembly 2 instead. In this scenario, it is advantageous to process the block in the assembly 1 within assembly 2, thus optimizing the overall production flow and reducing makespan.

3.4. Optimization Strategy

The optimization of the flat block assembly line sequence covers both grand assembly and unit (sub) assembly lines, as these two lines are interconnected and have a precedence relationship that impacts the overall factory environment.

In the shipbuilding industry, scheduling is typically done based on backward planning, where the scheduling of preceding steps is determined using the due date from subsequent steps to ensure smooth logistics and timely completion. In case of the flat block assembly line, most sub and unit blocks processed in line B serves as inputs for line A blocks. As a result, the block sequence schedule for line A must be established before scheduling the blocks in line B.

However, the operation scheduling process is not just about determining the sequence for undecided blocks; it also needs to consider the real-time shop conditions of already decided blocks.

This means the established schedule should be robust and considers the dynamic nature of the shop floor, as optimization models may overlook certain factors. To address this issue and evaluate the feasibility of the optimized schedule, this study proposes a simulation model (as described in Section 3.3). This simulation model is connected with the optimization model to simulate the actual production process and evaluate the performance of the optimized schedule.

Figure 9 illustrates the hierarchical scheduling process that integrates the optimization model and simulation model. The process begins with target blocks of line A being fed to the optimization model, which then searches for the optimal sequence based on the objective function. During the search, the optimization model generates feasible solutions while continuously updating the objective function.

Once the search is complete, the optimization model produces feasible solutions with the least makespan, and these solutions are then validated using the simulation model. However, for this validation, the workload balancing and due date objective are not considered, as these objectives are solely related to the block sequence.

Next, the simulation result of the feasible solution with the least makespan are delivered to line B, where the due dates of blocks are adjusted accordingly. The scheduling process for line B follows the same order as the optimization process for line A. If there is no feasible solution found initially, the feasible solution with the second

smallest makespan from line A is delivered to line B for further iterations. This iterative process continues until a set of feasible solutions for both line A and B are obtained.

The scheduling process is considered complete when the set of feasible solutions from both lines is combined and announced as the optimal solution for the given case. This hierarchical approach allows for the integration of optimization and simulation techniques, ensuring that the proposed schedule is both feasible and capable of achieving the minimum makespan.

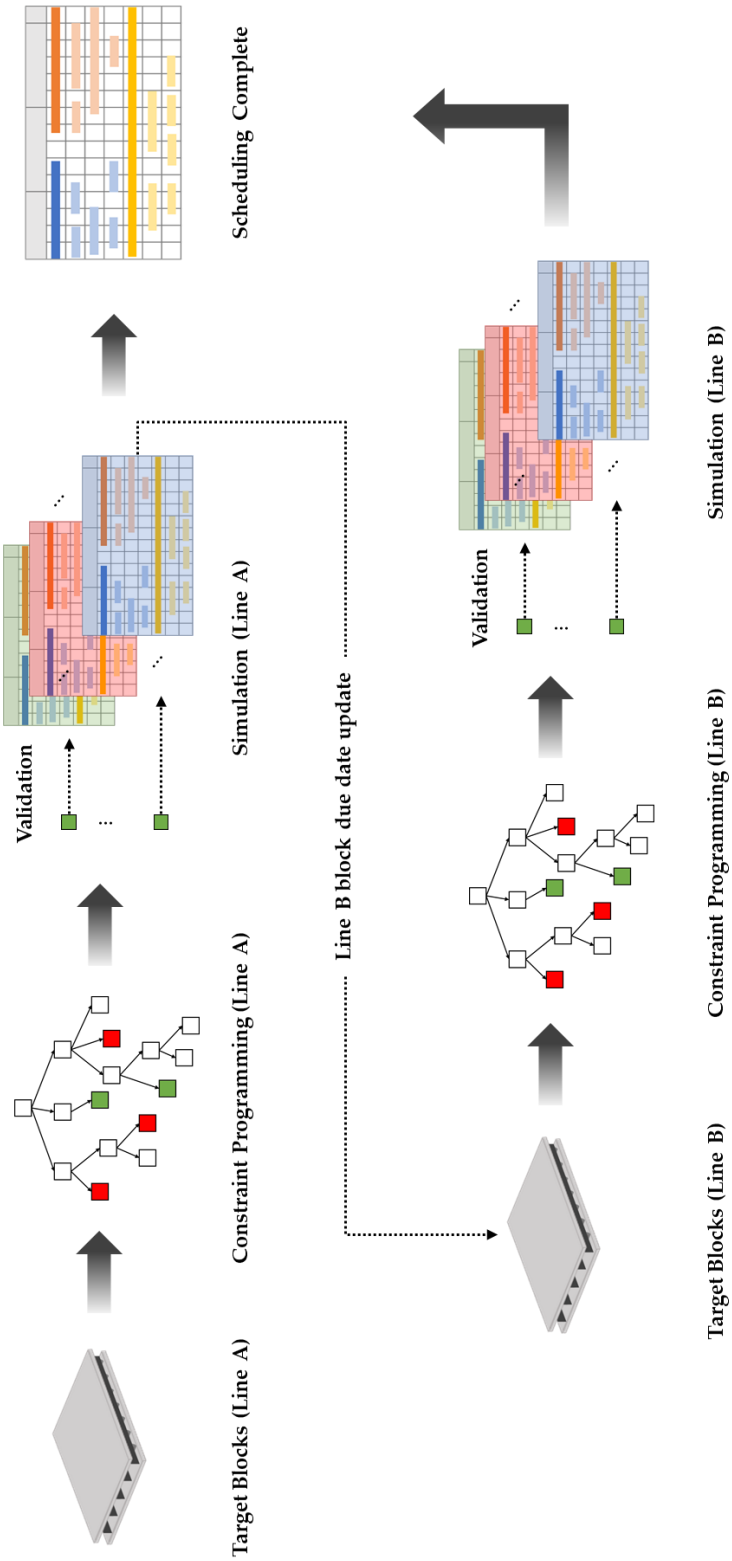

Figure 9 Hierarchical scheduling process using optimization and simulation model

Chapter 4. Performance Analysis

4.1. Case Studies

Target schedule is 2 weeks of blocks in flat block assembly line from an actual shipyard in South Korea. Two cases are experimented to validate the performance of the proposed framework. Each case is optimized following the optimization strategy mentioned in Section 3.4, and the result is compared with the manually planned schedule by actual schedulers.

Table 7 shows the block composition according to the case. Case 1 line A has 69 total 69 blocks with 3 scheduled blocks, 17 sets of set blocks, 5 sets of separate blocks, and 22 individual blocks. Individual blocks refer to blocks that doesn't fit either set blocks and separate blocks, mostly consists of hull center blocks that are not accompanied by an axisymmetric block. According to the schedulers, these blocks contribute to effectively easing workload balancing in scheduling perspective, since these blocks doesn't have any sequential constraints. Line B blocks has 40 blocks with 4 scheduled blocks, 14 sets of set blocks, 2 separate blocks, and 4 individual blocks. Compared to line A, line B has fewer separate blocks, because there aren't many types of blocks that require heavy workload in unit and sub assembly lines.

Table 7 Block composition according to the case

Case	Line	Total blocks	Scheduled blocks	Set blocks	Separate blocks	Individual blocks
1	A	69	3	34	10	22
	B	40	4	28	4	4
2	A	60	4	30	10	16
	B	61	3	42	0	16

Similarly, case 2 has total 60 and 61 blocks in each line, and line A has 4 sets of blocks, and line B doesn't have any separate blocks. The separate block constraint defined in Section 3.2 is complex compared to other sequential constraints, formed with logical constraints and comparison operator. This decreases the search speed significantly. Therefore, line A optimization usually require significantly more time to search for the optimal solution than line B. In addition, grand assembly is known to be the bottleneck process among all assembly processes (unit, sub, and grand), so it is more critical to find a solution with smaller makespan in line A than B in limited optimization time. Accordingly, search limit for each line is defined within search time, and number of solutions. Table 8 shows the CSP search limit in each line. The Line A optimization has an hour limit of search time, whereas the line B has only 20 minutes. Also, the line B optimization has additional search constraint, the number of solutions. It is because the line B solution is derived to validate the feasibility of the line A solution. The line B solution has an impact on the feasibility of the solution for line A, necessitating that the operation scheduling considers the shop condition of preceding

processes, which involves unit and sub assembly.

Table 8 CSP search limit in each line

Line	Search time (s)	Number of solutions
A	3,600	-
B	1,200	3

The experiment is conducted using IBM ILOG CPLEX Optimization studio 20.1.0 in virtual environment of Python 3.10.9 and docplex 2.25.236, and DES using Simpy 4.0.1 on a computer with an Intel Core i7-13700 2.10 GHz, 16 cores with 24 threads CPU and 64 GB RAM.

4.2. Result Analysis

The two cases mentioned in Section 4.1 are scheduled following optimization strategy in Section 3.4. The result is analyzed quantitatively, and the makespan difference between CP and DES is analyzed using CPM.

The weights of the objectives in CP model is defined according to the interviews with practitioners. Table 9 shows the determined weight values of each objective. It shows that ratio of each objective is 1:1:0.01. This is due to the relative importance of each objective; makespan, workload balancing, and due date in descending order. The unit of makespan and workload balancing is minute and hour accordingly. In other words, even the weight value is same, the sensitivity and value of makespan is larger than workload balancing.

Thus, the weight effectively shows the relative importance between the objectives.

Table 9 Determined weight value of each objective

Weight	Value
λ_1	1
λ_2	1
λ_3	0.01

4.2.1. Result Analysis in Case 1

From this section, the outputs of each step in the optimization strategy from Figure 9, and quantitative analysis of the results are presented on two cases. The quantitative analysis consists of objective function comparison, makespan comparison between CP result and simulation result, feasibility analysis and relative blocks in individual solutions, and quantitative analysis of makespan difference using CPM.

Table 10 shows the potential solutions derived from CP of line A in the 1st case. The search time limit follows Table 8. Total 27 solutions were found during the search. Figure 10 illustrates the updated objective values during the search time. The first feasible solution found from CP model is shown to have small improvement compared to the manual, and the best (27th) has improved 15.84%, with 15,630.11 as the total objective according to Table 11.

Although overall objective has continuously improved, each sub-objective may not have improved in parallel. One sub-objective could be sacrificed to improve other objectives, resulting overall objective

improvement. This is called as "trade-off". Figure 11 clearly shows the trade-off between makespan and workload balancing. Update from solution 14 to solution 15 clearly had big change in both objectives. The workload balancing and due date were sacrificed to achieve dramatic decrease of makespan. Note that the weight of due date objective is 0.01, so the value of due date objective is negligible.

Table 10 Potential solutions derived from CP of line A in case 1

No.	Total objective	Makespan	Workload balancing	Due date	Search period (s)
1	17,682.03	17,674	7	103	455.8
2	17,674.64	17,667	7	64	573.3
3	16,808.66	16,801	7	66	870.1
4	16,610.11	16,602	7	111	889.7
5	16,602.36	16,594	7	136	1,034.3
6	16,602.34	16,594	7	134	1,140.3
7	16,602.02	16,593	7	202	1,791.1
8	16,601.98	16,593	7	198	1,868.2
9	16,601.94	16,593	7	194	1,875.8
10	16,601.92	16,593	7	192	2,641.2
11	16,590.48	16,581	7	248	2,644.3
12	16,589.98	16,581	7	198	2,697.7
13	16,589.42	16,581	7	142	2,703.2
14	16,580.30	16,572	7	130	2,860.0
15	16,137.78	16,021	115	178	2,880.3
16	16,114.18	15,997	115	218	2,888.0
17	16,110.08	15,993	115	208	2,896.1
18	16,062.02	15,945	115	202	2,935.3
19	16,061.90	15,945	115	190	2,954.7
20	16,061.86	15,945	115	186	2,972.8
21	15,832.21	15,715	115	221	3,011.0
22	15,721.23	15,604	115	223	3,042.3
23	15,721.21	15,604	115	221	3,106.3

24	15,700.19	15,615	83	219	3,115.8
25	15,676.19	15,591	83	219	3,120.2
26	15,630.19	15,545	83	219	3,121.1
27	15,630.11	15,545	83	211	3,126.6
Manual	18,573.12	17,924	647	212	-

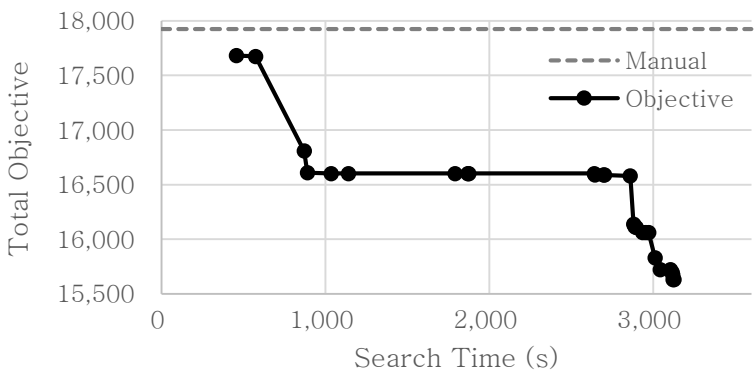

Figure 10 Total objective according to the search time

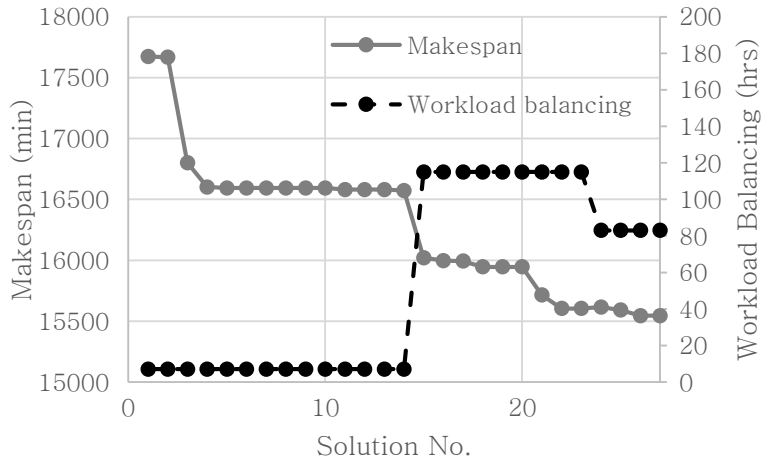

Figure 11 Sub objective values according to solutions

Table 11 Comparison between the best solution derived from CP model and manually planned schedule in case 1

Solution	Total objective	Makespan	Workload balancing	Due date	Sequential constraint
Manual	18,573.12	17,924	647	212	63.6%
Best (27)	15,630.11	15,545	83	211	100%
Improvement (%)	**15.84**	**13.27**	**87.17**	**0.47**	**36.36**

The improvement of individual objective is also presented in Table 11. It is shown that improvement of makespan, workload balancing, and due date objective is 13.27%, 87.17%, and 0.47% respectively. Note that the manual schedule didn't satisfy the sequential constraints. This is because of two reasons. First, sequential constraint is one of many rules in current scheduling process. Other rules consist of due date priorities, workload balancing, etc. There is no strict standard for the priority of each rule, and it also differs from person to person. This is one of the main reasons why mathematical and algorithmic approach is developed in this research to ensure the consistency of the schedules, not dependent on schedulers. The second reason is there could be emergency blocks that are transferred from other shops due to various circumstances. Schedulers reschedule these blocks, acknowledging the potential risk of violating established rules. These blocks are considered as scheduled blocks in the proposed model.

The following step is to validate the solutions using DES model. The key indices that the model validates are makespan change and feasibility of the derived solutions. Table 12 shows the validation result of 15 solutions from CP model. Last 12 solutions are omitted, because none are feasible. The average makespan difference between CP and DES model is 6.73%. The simulation result criticizes its justification, because only 6 out of 27 solutions are feasible. This is due to the increase of makespan that leads to some blocks to fail

to meet the due date.

Table 12 Feasibility and makespan results calculated using DES model of line A in case 1

No.	Makespan (CP)	Makespan (Sim)	Difference (%)	Delayed	Search period (s)
1	17,674	18,570.79	4.83	N	455.8
2	17,667	18,814.97	6.10	N	573.3
3	16,801	18,462.09	9.00	N	870.1
4	16,602	18,108.07	8.32	Y	889.7
5	16,594	18,107.56	8.36	N	1,034.3
6	16,594	18,088.56	8.26	N	1,140.3
7	16,593	18,087.79	8.26	Y	1,791.1
8	16,593	18,087.79	8.26	Y	1,868.2
9	16,593	17,981.74	7.72	Y	1,875.8
10	16,593	17,970.74	7.67	Y	2,641.2
11	16,581	17,498.15	5.24	Y	2,644.3
12	16,581	17,404.85	4.73	Y	2,697.7
13	16,581	17,435.30	4.90	Y	2,703.2
14	16,572	17,255.12	3.96	N	2,860.0
15	16,021	17,202.56	6.87	Y	2,880.3
⋮	⋮	⋮	⋮	⋮	⋮
Manual	17.924	18,894.12	5.14	Y	-

Detailed makespan difference analysis is conducted with example blocks from solution 7 in Table 13 and Table 14. The 7^{th} solution is determined to be infeasible solution due to 2 blocks, 32^{nd} and 54^{th} block. According to Table 13, 32^{nd} block started 107.87 minutes later in DES than CP model. This difference is the accumulated result of preceding blocks' delay. The total lead time for this block in DES model is 2,046.83 minutes, which is 740.83 minutes longer than CP model. This directly affected the block to fail to meet the due date.

Therefore, although there is 750 minutes of buffer between finish and due date in CP model, the DES result shows 98.7 minutes of delay.

The 54th block in Table 14 shows more dramatic difference in start time. This is because the block is placed back of the sequence, resulting more delays compare to the 32nd block. Even though CP and DES model show similar lead time, only 179.72 minutes of difference, the delay of start time critically affected the delay of finish time. Accordingly, the block has a delay of 326.03 minutes.

The two blocks, 32nd and 54th block, are the primary factor contributing to the infeasibility of the solution. All the other infeasible solutions have same delays from different blocks. This result emphasizes the limitation of CP model and the necessity of validation tool, in this research is DES model. The feasible solution, however, also has difference between two models. The quantitative analysis of feasible solution is conducted in below.

Table 13 32nd block from solution 7 simulation process analysis

32nd	Start	Finish	Lead time	Due date (Delay)
CP	11,804	13,110	1,306	13,860
DES	11,911.87	13,958.7	2,046.83	13,958.7
Δ	**107.87**	**848.7**	**740.83**	**-98.7**

Table 14 54th block from solution 7 simulation process analysis

54th	Start	Finish	Lead time	Due date (Delay)
CP	15,759	18,366	2,607	19,200
DES	16,739.31	19,526.03	2,786.72	19,526.03
Δ	**980.31**	**1,160.03**	**179.72**	**-326.03**

The best solution derived from DES model is 14th solution. The

comparison between the solution and manually planned schedule is shown in Table 15. The simulation result criticizes makespan improvement has increased from 7.54% to 8.68% compared to CP model. The workload balancing objective is dramatically improved by 99%, the workload difference between teams is almost negligible. The due date objective has also improved by 38.68%. This means less blocks are sequenced reversely. Most importantly, this solution is viable in actual shop condition with no delay.

Table 15 Comparison between the best solution derived from DES model and manually planned schedule in case 1

Solution	Makespan (CP)	Makespan (Sim)	Workload balancing	Due date	Delayed
Manual	17,924	18,894.23	647	212	Y
Best (14)	16,572	17,255.12	7	130	N
Improvement (%)	**7.54**	**8.68**	**99**	**38.68**	**-**

The CPM is conducted to analyze the makespan difference of the selected solution. This method finds the critical path regarding all processes and movements, and distinguishes the time according to difference factors, which are exclusions and simplifications in CP model. Figure 12 illustrates the Gantt chart and critical path of solution 14. The processes and movement are colored in different colors, and the critical path is labelled as blue. The specific discrimination of difference factor in the critical path is analyzed. The different factors between CP and DES model are block movement, parallel process in plate welding operation, and dynamic operation time in assembly.

Additional two factors are shift change and unit error. First, shift change in post assembly operations could make the difference. The post assembly operations have dynamic operation time due to shift change. For an example, no workers are assigned during the night shift for manual welding and grinding operation. If grinding operation of one block is not finished in day shift, it should wait till the next day shift. This difference is analyzed as shift change. Next is unit error. The CP model can only calculate integer units. Real values below the decimal point are excluded, however, the DES model consider them all. Thus, this difference is due to unit difference between the models, and it is named as "unit error".

Table 16 shows the discrimination of difference factor in the critical path of 14^{th} solution. The makespan of DES model represents the lead time of critical path, and the makespan of CP model represents the lead time of same critical path derived from the CP model. The critical path in DES is not critical path is CP model, since the makespan of CP model is 16,572 minutes from Table 15, and the lead time of the critical path is 15,538. This means that difference factors has changed the critical path.

When closely examining each factor, the block movement is only considered in DES model. Specifically, 1,733 minutes are the block movement in the critical path in DES model, and none in CP model. The parallel operation has increased the operation time in DES model, since only 2 machines are assigned simultaneously, whereas 3 machines are assigned in CP model. Assembly 1 and 2 is calculated

separately and there were 316.02 and 44.11 minutes of difference respectively. It is due to dynamic operation time mentioned above. According to the result, there were more blocks operated tasks in the Assembly 1 than Assembly 2, since there were blocks already operating in Assembly 2. Lastly, shift change and unit error are also examined to have 535 and 12.8 minutes of difference respectively. The total time difference of all factors are 1,717.12 minutes of increment from original lead time from CP model, which is 15,538 minutes. Therefore, total lead time of this path is 17,255.12 minutes.

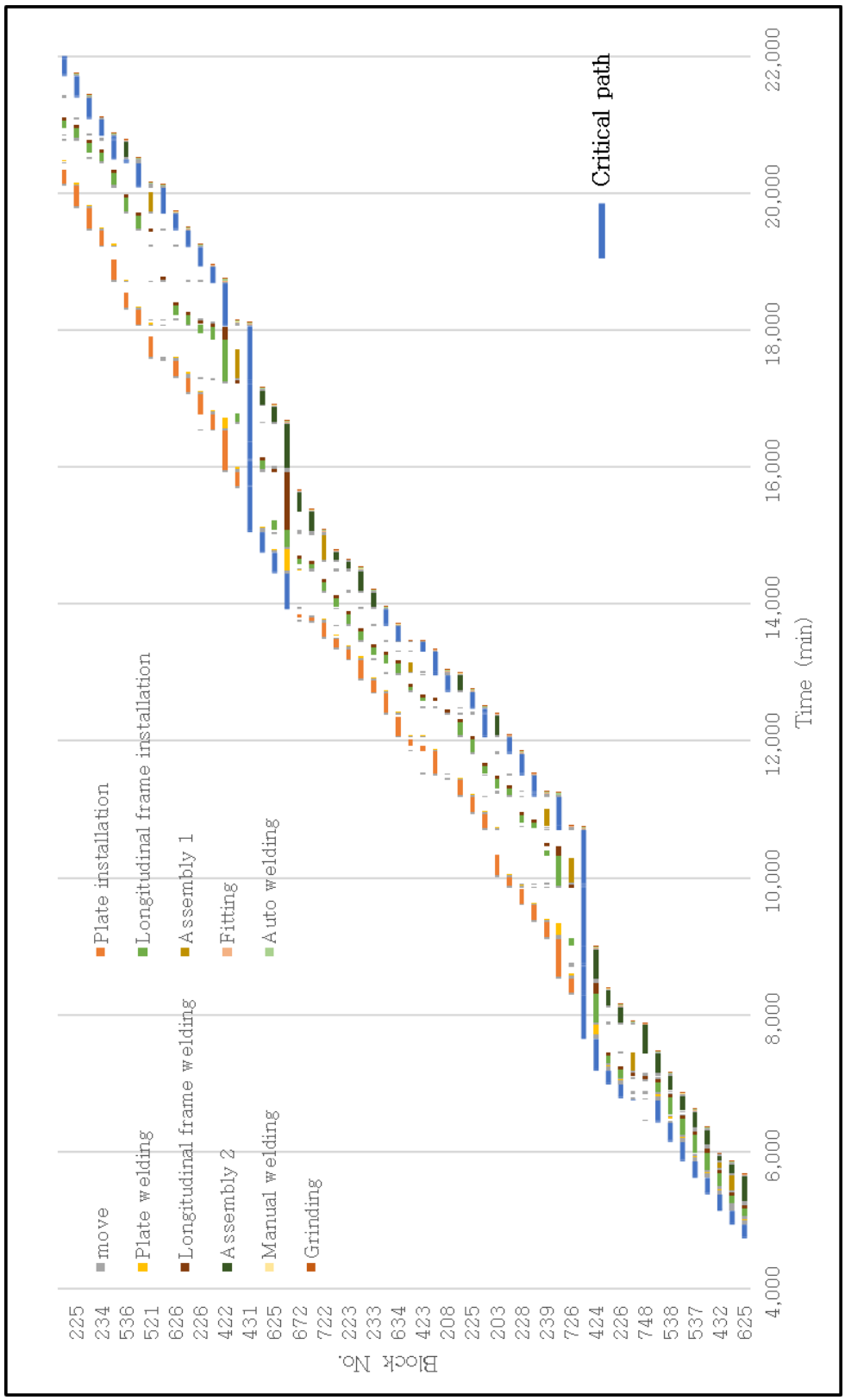

Figure 12 The critical path of solution 14 including block movement

Table 16 Discrimination of difference factor in the critical path

Factor	CP	Δ	DES
Makespan	15,538	-	-
Movement	-	1,733	1,733
Plate weld	465	234.41	699.41
Assembly 1	809	316.02	1125.02
Assembly 2	6710	-44.11	6665.89
Shift change	535	-535	0
Unit error	-12.8	+ 12.8	0
Total	**15,538**	**1,717.12**	**17,255.12**

The next step of proposed method of scheduling is optimization of line B. By the nature of backward scheduling, most blocks in the line B should inherit the due date from the line A schedule. Since there are 6 feasible solutions in line A, the optimization of line B inheriting the due date from each solution in line A is conducted with both time and number of solution limit noted in Table 8.

Table 17 presents the potential solutions derived from CP of line B. All solutions from line A have found 3 solutions in line B respectively. In all cases, the first solution is found about 10 minutes after the algorithm is initiated, and last solution is found close to 1,200 seconds. The makespan comparison on potential solutions are illustrated in Figure 13. Unlike the optimization result in line A, derived makespan is larger than manual plan except 2 solutions, and even these solutions have only 4 minutes of difference. This, however, is further analyzed in DES model to validate the feasibility of the derived solutions and manual plan.

Table 17 Potential solutions derived from CP of line B in case 1

No.	Makespan	Start time	Finish time	Search periods (s)
1 (1)	8,166	4,923	13,089	787.3
2 (1)	8,123	4,923	13,046	1,086.0
3 (1)	8,114	4,923	13,037	1,172.3
1 (2)	8,166	4,923	13,089	674.0
2 (2)	8,124	4,923	13,047	730.2
3 (2)	8,123	4,923	13,046	738.6
1 (3)	8,166	4,923	13,089	727.5
2 (3)	8,123	4,923	13,046	825.7
3 (3)	8,118	4,923	13,041	1,014.8
1 (5)	8,166	4,923	13,089	864.6
2 (5)	8,123	4,923	13,046	893.3
3 (5)	8,114	4,923	13,037	1,071.1
1 (6)	8,166	4,923	13,089	593.2
2 (6)	8,123	4,923	13,046	667.2
3 (6)	8,118	4,923	13,041	904.4
1 (14)	8,166	4,923	13,089	599.4
2 (14)	8,123	4,923	13,046	682.6
3 (14)	8,118	4,923	13,041	1,101.8
Manual	8,118	4,923	13,041	-

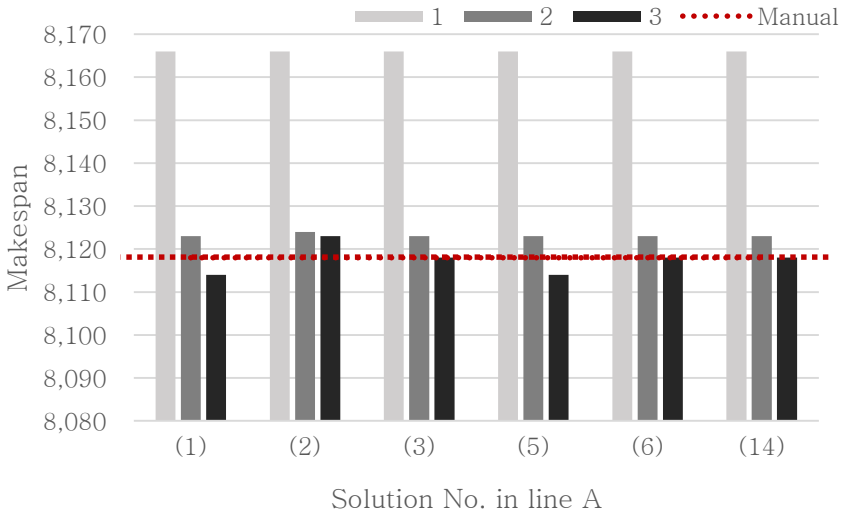

Figure 13 Makespan comparison on potential solutions derived from CP of line B in case 1

Table 18 shows the validation result of feasibility and makespan using DES model from line B solutions. The validated makespan from DES model is larger makespan in CP, which is mostly contributed from the additional time of block movement. The average makespan increment is 12.12%, and all solutions showed larger makespan than manual plan. However, only 7 solutions out of 18 are examined to be feasible, and manual plan is also infeasible. As derived optimization solutions in line B inherited the individual block due date from line A simulation result, manual plan in line B also inherited from simulation result of manual plan in line A.

Table 18 Feasibility and makespan results calculated using DES model of line B in case 1

No.	Makespan (CP)	Makespan (DES)	Delayed	Difference (%)
1 (1)	8,166	9,140.11	N	11.93
2 (1)	8,123	9,114.86	Y	12.21
3 (1)	8,114	9,121.25	Y	12.41
1 (2)	8,166	9,140.11	N	11.93
2 (2)	8,124	9,144.56	N	12.56
3 (2)	8,123	9,114.86	Y	12.21
1 (3)	8,166	9,140.11	N	11.93
2 (3)	8,123	9,114.86	Y	12.21
3 (3)	8,118	9,107.18	Y	12.18
1 (5)	8,166	9,140.11	N	11.93
2 (5)	8,123	9,114.86	Y	12.21
3 (5)	8,114	9,104.25	Y	12.20
1 (6)	8,166	9,140.11	N	11.93
2 (6)	8,123	9,114.86	Y	12.21
3 (6)	8,118	9,102.18	Y	12.12
1 (14)	8,166	9,140.11	N	11.93
2 (14)	8,123	9,114.86	Y	12.21

3 (14)	8,118	9,109.18	Y	12.21
Manual	8,118	9,076.18	Y	11.80

The last step is to decide the optimal solution based on the results from line A and B. Table 19 shows the derived combinations of feasible solutions from both lines. All feasible combinations are included in the table. The objective of the proposed optimization method is to obtain feasible schedule that minimizes makespan on line A regarding the shop conditions. According to Table 19, the optimal solution combination is 14^{th} solution and 1^{st} (14) solution respectively. This combination has minimum makespan of line A and also has a feasible line B schedule. Note that if there were no feasible line B solution for 14^{th} solution, 6^{th} solution from line A and 1^{st} (6) solution from line B is the next best combination.

Table 19 Combinations of feasible solutions from line A and B in case 1

Solution No. (A)	Solution No. (B)	Makespan (A, DES)	Makespan (B, DES)	Total makespan
1	1 (1)	18,570.79	9,140.11	27,710.9
2	1 (2)	18,814.97	9,140.11	27,955.1
	2 (2)	18,814.97	9,144.56	27,959.5
3	1 (3)	18,462.09	9,140.11	27,602.2
5	1 (5)	18,107.56	9,140.11	27,247.7
6	1 (6)	18,088.56	9,140.11	27,228.7
14	1 (14)	17,255.12	9,140.11	26,395.2

4.2.2. Result Analysis in Case 2

The Section 4.2.1. presented the overall optimization strategy and quantitative analysis in each step using various methods like CPM.

This section follows same step but is more focused on the optimized schedule with second case mentioned in Table 7. The composition of case 2 is more complex with 20 more blocks in line B. There are a smaller number of blocks and separate block sets in line A, which refers to looser condition than the first case. In line B, however, there are 20 more blocks. This case is typical assembly line condition in the factory. There is usually same amount or a greater number of blocks in line B compared to line A. The main objective of this case is to test the performance of the proposed algorithm in normal conditions.

The first step is optimization of line A blocks using CP model. Table 20 shows the potential solutions of line A derived from CP model. 49 solutions are found during 3,600s of search time.

Table 21 shows the comparison between the best solution and manually planned schedule. The total objective has improved 11.52%, and each sub objective has improved 5.11%, 99.9%, and −17.15% respectively. In both cases, workload balancing objective values have dramatically improved. This could be inferred that the schedulers could not consider this objective, rather only controlled with the heuristic rules such as set blocks and separate blocks. Furthermore, sequential constraints are also not satisfied in manual plan.

Table 20 Potential solutions derived from CP of line A in case 2

No.	Total objective	Makespan	Workload balancing	Due date	Search period (s)
1	14,106.24	13,905	199	224	112.8
2	14,099.44	13,898	199	244	229.2

3	14,090.8	13,889	199	280	248.9
4	14,087.78	13,886	199	278	261.1
5	13,986.63	13,901	83	263	268.7
6	13,983.71	13,898	83	271	272.1
7	13,914.63	13,907	5	263	277.6
8	13,914.62	13,907	5	262	335.4
9	13,914.52	13,907	5	252	363.9
10	13,896.82	13,889	5	282	495.7
11	13,896.77	13,889	5	277	532.0
12	13,894.24	13,886	5	324	560.7
13	13,894.22	13,886	5	322	568.3
14	13,893.83	13,886	5	283	578.3
15	13,893.72	13,886	5	272	827.3
16	13,893.62	13,886	5	262	879.0
17	13,885.56	13,878	5	256	1,018.0
18	13,342.06	13,334	5	306	1,079.8
19	13,342.05	13,334	5	305	1,090.0
20	13,341.96	13,334	5	296	1,138.0
21	13,324.62	13,316	5	362	1,142.4
22	13,301.28	13,293	5	328	1,144.8
23	13,294.24	13,286	5	324	1,148.6
24	13,294.23	13,286	5	323	1,217.9
25	13,293.98	13,286	5	298	1,255.5
26	13,293.95	13,286	5	295	1,505.6
27	13,293.74	13,286	5	274	1,538.2
28	13,293.73	13,286	5	273	1,612.0
29	13,289.72	13,286	1	272	1,826.6
30	13,289.7	13,286	1	270	1,874.6
31	13,289.68	13,286	1	268	2,012.6
32	13,289.66	13,286	1	266	2,088.5
33	13,289.62	13,286	1	262	2,108.4
34	13,258.62	13,255	1	262	2,245.8
35	13,243.66	13,240	1	266	2,557.4
36	13,243.63	13,240	1	263	2,598.0

37	13,243.59	13,240	1	259	2,670.9
38	13,243.55	13,240	1	255	2,692.7
39	13,243.53	13,240	1	253	2,753.8
40	13,243.52	13,240	1	252	2,766.1
41	13,229.37	13,207	19	337	2,799.0
42	13,229.35	13,207	19	335	2,807.0
43	13,229.31	13,207	19	331	2,814.1
44	13,211.14	13,207	1	314	2,826.6
45	13,211.1	13,207	1	310	2,974.3
46	13,211.09	13,207	1	309	2,977.2
47	13,211.08	13,207	1	308	2,987.6
48	13,211.06	13,207	1	306	2996.4
49	13,211.01	13,207	1	301	3273.8
Manual	14,931.57	13,918	1,011	257	-

Table 21 Comparison between the best solution derived from CP model and manually planned schedule in case 2

Solution	Total objective	Makespan	Workload balancing	Due date	Sequential constraint
Manual	14,931.57	13,918	1,011	257	85%
Best (49)	13,211.01	13,207	1	301	100%
Improvement (%)	**11.52**	**5.11**	**99.90**	**-17.15**	**15**

The derived solutions are then validated in the DES model. Table 22 organizes the validation result regarding the feasibility and makespan. Out of 49 solutions, 4 solutions are concluded to be feasible. From 11th solution, all solutions are all infeasible, thus they are not listed in this table. The average makespan increment rate in DES model compared to CP is 11.38%.

The individual makespan value of solutions that are feasible is more interesting, because it turned out that the 1st solution has the

minimum makespan, which is 15,151.95 minutes. The 5th and 6th solution also had reversed rankings in makespan, but both had larger value than the first. This is an extreme example where CP result shows opposite result from DES model. It infers that although CP opt for optimal solution, lack of reality reflected in the model could change the result dramatically. This is critical reason why DES model is required to fully guarantee the optimal solution in real-world problem. Figure 14 shows the scatter plot of the makespan derived from DES model according to the makespan from CP model. The graph shows that there is a positive correlation between two models, which infers that the CP model partially reflects the real shop condition.

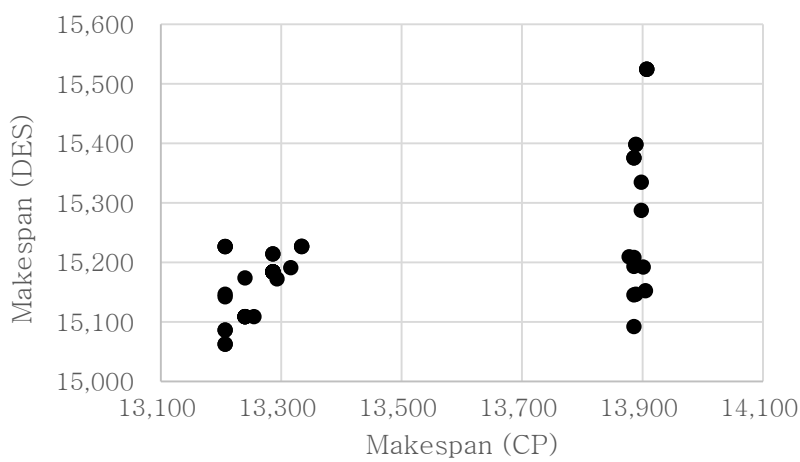

Figure 14 Scatter plot of makespan (DES) according to makespan (CP)

Table 23 shows the comparison between the best solution derived from DES model and manually planned schedule. This result again criticizes the necessity of DES model to validate the makespan. Even though, the CP model concluded that there were only 0.14%

improvement in makespan, the DES model, however, showed that the solution improved the makespan by 4.35%.

Table 22 Feasibility and makespan results calculated using DES model of line A in case 2

No.	Makespan (CP)	Makespan (DES)	Difference (%)	Delayed	Search period (s)
1	13,905	15,151.95	8.23	N	112.8
2	13,898	15,334.31	9.37	N	229.2
3	13,889	15,145.90	8.30	Y	248.9
4	13,886	15,207.74	8.69	Y	261.1
5	13,901	15,191.78	8.50	N	268.7
6	13,898	15,286.78	9.08	N	272.1
7	13,907	15,524.03	10.42	Y	277.6
8	13,907	15,524.47	10.42	Y	335.4
9	13,907	15,524.47	10.42	Y	363.9
10	13,889	15,397.77	9.80	Y	495.7
⋮	⋮	⋮	⋮	⋮	⋮
Manual	13,918	15,981.34	12.91	N	-

Table 23 Comparison between the best solution derived from DES model and manually planned schedule in case 2

Solution	Makespan (CP)	Makespan (Sim)	Workload balancing	Due date	Delayed
Manual	13,918	15,981.34	1,011	257	N
Best (6)	13,898	15,286.78	83	271	N
Improvement (%)	**0.14**	**4.35**	**91.79**	**-5.45**	**-**

The next step is to optimize the line B block schedule according to the 4 feasible solutions. The potential solutions derived from CP model of line B is listed in Table 24 with the details. In this case, all 3 solutions are found within a minute. The results, however, show longer makespan than manually planned schedule. Two conditions could have contributed to this result. First, the number of solution

limit could have terminated the algorithm before more improved solutions could be found. Since, CP algorithm is first developed to find feasible solution in complex operation research problems, first few feasible solutions could be very different from the optimal solution. Secondly, the manually planned schedule is infeasible solution. This schedule does not fully satisfy set block constraints. It achieves 85.7% (36/44) satisfaction of the constraint, and it is not enough to be feasible, since it is a constraint. In other words, the manually planned schedule is not in feasible space in the perspective of CP model. Thus, the minimum makespan could be larger than the manual one.

Table 24 Potential solutions derived from CP of line B in case 2

No.	Makespan	Start time	Finish time	Search periods (s)
1 (1)	11,833	1,084	12,917	17.8
2 (1)	11,787	1,084	12,871	37.8
3 (1)	11,783	1,084	12,867	48.0
1 (2)	11,833	1,084	12,917	17.6
2 (2)	11,820	1,084	12,904	61.9
3 (2)	11,787	1,084	12,871	67.2
1 (5)	11,833	1,084	12,917	17.2
2 (5)	11,820	1,084	12,904	37.9
3 (5)	11,775	1,084	12,859	40.5
1 (6)	11,833	1,084	12,917	17.2
2 (6)	11,820	1,084	12,904	47.8
3 (6)	11,780	1,084	12,864	53.9
Manual	11,772	1,084	12,856	-

The validation result of line B solutions using DES model is

mentioned in Table 25. All solutions including manual plan has met the due date, resulting no delay. Note that no delay is not same with feasible. A feasible solution must have met due date for all blocks, and satisfied all constraints. In this table, only delay is analyzed, but manual plan has failed to satisfy set block constraints as mentioned above. The derived solutions, however, satisfy all constraints, thus they are all feasible solutions.

Table 25 Feasibility and makespan results calculated using DES model of line B in case 2

No.	Makespan (CP)	Makespan (DES)	Delayed	Difference (%)
1 (1)	11,833	12,213.36	N	3.2
2 (1)	11,787	12,154.69	N	3.1
3 (1)	11,783	12,147.35	N	3.1
1 (2)	11,833	12,213.36	N	3.2
2 (2)	11,820	12,131.00	N	2.6
3 (2)	11,787	12,154.69	N	3.1
1 (5)	11,833	12,213.36	N	3.2
2 (5)	11,820	12,051.5	N	2.0
3 (5)	11,775	11,922.00	N	1.2
1 (6)	11,833	12,213.36	N	3.2
2 (6)	11,820	11,980.67	N	1.4
3 (6)	11,780	11,947.79	N	1.4
Manual	11,772	12,078.49	N	2.6

As the final step, final combinations of the feasible solutions from line A and B could be derived as in Table 26. Since, the primary goal is to minimize the makespan of line A, the optimal solution is combination of 1st solution and 3 (1) solution in line A and B respectively. The optimal solution, however, could vary depending on

assembly line strategy. For example, if line B is overloaded now, 3 (5) solution that has minimum makespan in line B could be optimal solution. In the perspective of minimizing makespan of all lines, 3 (5) solution with 3rd solution in line A has the minimum total makespan among all combinations.

Table 26 Combinations of feasible solutions from line A and B in case 2

Solution No. (A)	Solution No. (B)	Makespan (A, DES)	Makespan (B, DES)	Total makespan
1	1 (1)	15,151.95	12,213.36	27,365.31
	2 (1)	15,151.95	12,154.69	27,306.64
	3 (1)	15,151.95	12,147.35	27,299.30
2	1 (2)	15,334.31	12,213.36	27,547.67
	2 (2)	15,334.31	12,131.00	27,465.31
	3 (2)	15,334.31	12,154.69	27,489.00
5	1 (5)	15,191.78	12,213.36	27,405.14
	2 (5)	15,191.78	12,051.5	27,243.28
	3 (5)	15,191.78	11,922.00	27,113.78
6	1 (6)	15,286.78	12,213.36	27,500.14
	2 (6)	15,286.78	11,980.67	27,267.45
	3 (6)	15,286.78	11,947.79	27,234.57

Chapter 5. Conclusion

In this study, a two-step optimization process is proposed to address the limitations of existing PFSP solutions. The first step involved developing a novel CP algorithm for MOPFSP-hd that incorporates actual industrial constraints, making it more practical for real-world problems. The second step involved validating the feasibility and objective value of the optimized solution using DES.

Two case studies were conducted to evaluate the superiority of the proposed model. The experimental results from both cases demonstrated an improvement in makespan compared to manually planned schedules. Furthermore, the solutions derived from the proposed model were reported to be feasible, while the manually planned schedules often encountered delays or did not satisfy the industrial constraints. This highlights the significance of using simulation to validate the derived solutions. The analysis of the difference between the objective calculated from CP and DES model showed a difference of 2 ~ 12%, and CPM was used to identify the factors influencing this difference.

Although, the proposed method demonstrated superior performance to existing scheduling method, there are still opportunities for further improvement. One potential area of improvement is the analysis of weight values assigned to each objective function. Multi-objective optimization problems often lack

a unified measuring unit between sub-objective functions. While time and cost units are commonly used as unified units in schedule optimization problems, more studies are needed to quantify the importance of each objective and assign appropriate weights to each sub-objective function.

In conclusion, the proposed solution in this paper presented a practical and effective approach to address PFSP with real-world constraints. By combining CP and DES techniques, the authors demonstrated improved makespan and feasible schedules compared to traditional methods. Further investigations into weight assignment for objective functions could enhance the quality of the optimization results.

Bibliography

Ahn, N., & Kim, S. (2022). A Mathematical Formulation and a Heuristic for the Spatial Scheduling of Mega-Blocks in Shipbuilding Industry. *Journal of Ship Production and Design*, *38*(04), 193-198.

Banks, J. (2005). *Discrete event system simulation*. Pearson Education India.

Banks, J., & Carson, J. S. (1986). Introduction to discrete-event simulation. Proceedings of the 18th conference on Winter simulation,

Blazewicz, J., Pesch, E., Sterna, M., & Werner, F. (2008). Metaheuristic approaches for the two-machine flow-shop problem with weighted late work criterion and common due date. *Computers & operations research*, *35*(2), 574-599.

Brah, S. A. (1996). A comparative analysis of due date based job sequencing rules in a flow shop with multiple processors. *Production Planning & Control*, *7*(4), 362-373.

Brailsford, S. C., Potts, C. N., & Smith, B. M. (1999). Constraint satisfaction problems: Algorithms and applications. *European journal of operational research*, *119*(3), 557-581.

Brennan, R. W., & William, O. (2000). A simulation test-bed to evaluate multi-agent control of manufacturing systems. 2000 Winter Simulation Conference Proceedings (Cat. No. 00CH37165),

Campbell, H. G., Dudek, R. A., & Smith, M. L. (1970). A heuristic algorithm for the n job, m machine sequencing problem. *Management science*, *16*(10), B-630-B-637.

Chen, C.-L., Vempati, V. S., & Aljaber, N. (1995). An application of genetic algorithms for flow shop problems. *European journal of operational research*, *80*(2), 389-396.

Chou, F.-D., & Lee, C.-E. (1999). Two-machine flowshop scheduling with bicriteria problem. *Computers & Industrial Engineering*, *36*(3), 549-564.

Dudek, R. A., & Teuton Jr, O. F. (1964). Development of m-stage decision rule for scheduling n jobs through m machines. *Operations Research*, *12*(3), 471-497.

Eren, T., & Güner, E. (2008). The tricriteria flowshop scheduling problem.

The International Journal of Advanced Manufacturing Technology, *36*, 1210-1220.

Framinan, J., Leisten, R., & Rajendran, C. (2003). Different initial sequences for the heuristic of Nawaz, Enscore and Ham to minimize makespan, idletime or flowtime in the static permutation flowshop sequencing problem. *International Journal of Production Research*, *41*(1), 121-148.

Framinan, J. M., Gupta, J. N., & Leisten, R. (2004). A review and classification of heuristics for permutation flow-shop scheduling with makespan objective. *Journal of the Operational Research Society*, *55*(12), 1243-1255.

Garey, M. R., Johnson, D. S., & Sethi, R. (1976). The complexity of flowshop and jobshop scheduling. *Mathematics of operations research*, *1*(2), 117-129.

Gowrishankar, K., Rajendran, C., & Srinivasan, G. (2001). Flow shop scheduling algorithms for minimizing the completion time variance and the sum of squares of completion time deviations from a common due date. *European journal of operational research*, *132*(3), 643-665.

Hooker, J. N. (2002). Logic, optimization, and constraint programming. *INFORMS Journal on Computing*, *14*(4), 295-321.

Hunsucker, J., & Shah, J. (1992). Performance of priority rules in a due date flow shop. *Omega*, *20*(1), 73-89.

Huynh, B. H., Akhtar, H., & Li, W. (2020). Discrete event simulation for manufacturing performance management and optimization: a case study for model factory. 2020 9th International Conference on Industrial Technology and Management (ICITM),

Johnson, S. M. (1954). Optimal two-and three-stage production schedules with setup times included. *Naval research logistics quarterly*, *1*(1), 61-68.

Karabulut, K., Öztop, H., Kizilay, D., Tasgetiren, M. F., & Kandiller, L. (2022). An evolution strategy approach for the distributed permutation flowshop scheduling problem with sequence-dependent setup times. *Computers & operations research*, *142*, 105733.

Kim, H., Lee, S.-S., Park, J. H., & Lee, J.-G. (2005). A model for a simulation-based shipbuilding system in a shipyard manufacturing process. *International Journal of Computer Integrated Manufacturing*, *18*(6), 427-441.

Kwak, D. H., Cho, Y. I., Choe, S. W., Kwon, H. J., & Woo, J. H. (2022). Optimization of long-term planning with a constraint satisfaction

problem algorithm with a machine learning. *International Journal of Naval Architecture and Ocean Engineering*, *14*, 100442.

Lee, K., Shin, J. G., & Ryu, C. (2009). Development of simulation-based production execution system in a shipyard: a case study for a panel block assembly shop. *Production Planning & Control*, *20*(8), 750-768.

Lee, W.-C., & Wu, C.-C. (2001). Minimizing the total flow time and the tardiness in a two-machine flow shop. *International Journal of Systems Science*, *32*(3), 365-373.

Liao, C. J., Yu, W., & Joe, C. (1997). Bicriterion scheduling in the two-machine flowshop. *Journal of the Operational Research Society*, *48*(9), 929-935.

Nagar, A., Heragu, S. S., & Haddock, J. (1995). A branch-and-bound approach for a two-machine flowshop scheduling problem. *Journal of the Operational Research Society*, *46*, 721-734.

Nam, S.-H., Oh, S.-H., Yoon, H.-C., Cho, Y.-I., Cho, K.-Y., Kwak, D.-H., & Woo, J. H. (2022). Development of Des Application for Factory Material Flow Simulation With Simpy. 2022 Winter Simulation Conference (WSC),

Nawaz, M., Enscore Jr, E. E., & Ham, I. (1983). A heuristic algorithm for the m-machine, n-job flow-shop sequencing problem. *Omega*, *11*(1), 91-95.

Osman, I. H., & Potts, C. (1989). Simulated annealing for permutation flow-shop scheduling. *Omega*, *17*(6), 551-557.

Öztop, H., Tasgetiren, M. F., Kandiller, L., & Pan, Q.-K. (2022). Metaheuristics with restart and learning mechanisms for the no-idle flowshop scheduling problem with makespan criterion. *Computers & operations research*, *138*, 105616.

Perez-Gonzalez, P., & Framinan, J. M. (2015). Assessing scheduling policies in a permutation flowshop with common due dates. *International Journal of Production Research*, *53*(19), 5742-5754.

Robinson, S. (2014). *Simulation: the practice of model development and use*. Bloomsbury Publishing.

Rogers, P., & Brennan, R. W. (1997). A simulation testbed for comparing the performance of alternative control architectures. Proceedings of the 29th conference on Winter simulation,

Rossi, F., Van Beek, P., & Walsh, T. (2006). *Handbook of constraint*

programming. Elsevier.

Ruiz, R., & Maroto, C. (2005). A comprehensive review and evaluation of permutation flowshop heuristics. *European journal of operational research*, *165*(2), 479-494.

Samarghandi, H., & Behroozi, M. (2017). On the exact solution of the no-wait flow shop problem with due date constraints. *Computers & operations research*, *81*, 141-159.

Selen, W. J., & Hott, D. D. (1986). A mixed-integer goal-programming formulation of the standard flow-shop scheduling problem. *Journal of the Operational Research Society*, *37*, 1121-1128.

Shie Gheun, K. (1996). A Production Schedule with Genetic Algorithm in Block assembly shop. *Journal of The Korean Operations Research and Management Science Society*, *13*(1), 1-12.

Stützle, T. (1998). *Applying iterated local search to the permutation flow shop problem*.

Tseng, F. T., & Stafford Jr, E. F. (2008). New MILP models for the permutation flowshop problem. *Journal of the Operational Research Society*, *59*(10), 1373-1386.

Varadharajan, T., & Rajendran, C. (2005). A multi-objective simulated-annealing algorithm for scheduling in flowshops to minimize the makespan and total flowtime of jobs. *European journal of operational research*, *167*(3), 772-795.

Widmer, M., & Hertz, A. (1989). A new heuristic method for the flow shop sequencing problem. *European journal of operational research*, *41*(2), 186-193.

Yang, Z., Liu, C., Zhang, S., & Shi, J. (2019). A multi-objective memetic algorithm for a fuzzy parallel blocking flow shop scheduling problem of panel block assembly in shipbuilding. *Journal of Ship Production and Design*, *35*(02), 170-181.

Yenisey, M. M., & Yagmahan, B. (2014). Multi-objective permutation flow shop scheduling problem: Literature review, classification and current trends. *Omega*, *45*, 119-135.

Zheng, D.-Z., & Wang, L. (2003). An effective hybrid heuristic for flow shop scheduling. *The International Journal of Advanced Manufacturing Technology*, *21*, 38-44.

Abstract in Korean

평면 블록 조립 라인의 스케줄링은 조선소의 전체 조선 건조 성능에 있어서 높은 작업량 때문에 매우 중요하다. 이 문제는 운영 연구에서 잘 알려진 Permutation Flow-shop Scheduling Problem(PFSP)으로서 1950년대부터 다양한 논문에서 광범위하게 연구되었다. 그러나, 기존의 해결책들은 실제 문제를 단순화하고 특정 가정들을 포함시키는 경우가 많아서 실제 문제들에 대한 적용이 제한되고 있다.

최근 제약 프로그래밍(CP)이 수리적 알고리즘들에 대한 강력한 대안으로 등장하며, 수리적 알고리즘의 한계를 성공적으로 극복하는 연구 사례들이 등장하고 있다. 이러한 배경에 따라, 본 연구는 기존 알고리즘들의 한계를 극복하기 위해 두 단계 계획 수립 최적화 프레임워크를 제안한다. 먼저, 실제 산업 제약 조건을 반영하는 새로운 CP 알고리즘을 소개한다. 이 PFSP는 Multi-Objective PFSP with hard due date constraint (MOPFSP-hd)으로서 분류될 수 있다. 다음으로, 도출된 최적화 해들의 타당성과 목적 함수 값은 이산 사건 시뮬레이션(DES)을 통해 검증한다.

제안한 스케줄링 프레임워크의 성능을 평가하기 위해 두 개의 산업 사례를 대상으로 계획이 수행되었다. 두 사례에 대한 실험 결과는 일관적으로 현재 실행 계획에 적용되는 휴리스틱 룰에 기반하여 작성된 계획에 비해 makespan이 개선되었음을 보여준다. 또한, 본 연구에서 제안한 프레임워크를 통해 도출된 해들은 산업 제약 조건을 모두 만족시키고 실현 가능한 반면에, 기존의 방법으로 수립된 해는 제약을 불만족시키거나 지연이 발생하여 실현 불가능한 해임을 DES를 통해

확인할 수 있었다. 마지막으로, CP와 DES 모델로부터 계산된 목적 함수 사이의 차이는 임계 경로 분석법 (CPM)을 사용하여 분석하였다.

본 연구에서 제시한 해결책은 현실적인 산업 환경에서의 PFSP를 다루기 위한 실질적이고 효과적인 방법을 소개했다. 기존의 한계를 극복하기 위해 CP와 DES 기법을 결합하여 기존의 방법들보다 개선된 makespan을 가지는 실현 가능한 계획을 도출 할 수 있음을 두 개의 산업 사례를 활용하여 보였다.

주요어 : 최적화, 제약 프로그래밍, 이산 사건 시뮬레이션, 평블록 조립 공정, 선박 건조 과정
학번 : 2020-21233