An MDP-Based Lifter Assignment Algorithm for Inter-Line Transportation in Semiconductor Fabrication

Kyohong Shin
Material Handling Automation Group, Samsung Electronics https://orcid.org/0000-0002-4243-4144
Hoon Jang
Korea University
Hae Joong Kim (haejoong.kim@gmail.com)
Material Handling Automation Group, Samsung Electronics https://orcid.org/0000-0002-3727-094X

Research Article

Keywords: manufacturing, semiconductor, automated material handling system, lifter, Markov decision process

Posted Date: June 2nd, 2021

DOI: https://doi.org/10.21203/rs.3.rs-554198/v1

An MDP-based Lifter Assignment Algorithm for Inter-line Transportation in Semiconductor Fabrication

Kyohong Shin[a] , Hoon Jang[b], and Hae Joong Kim [a]*

[a]*Material Handling Automation Group, Samsung Electronics , Hwaseong-si, Korea;*
[b]*College of Global Business, Korea University, Sejong, Korea*

[*] ***Corresponding author***: Haejoong Kim (PhD), Affiliation: Material Handling Automation Group, Samsung Electronics; Address: 1, Samsungjeonja-ro, Hwasung-si, Gyeonggi-do, Republic of Korea; Email address: haejoong.kim@gmail.com

Kyohong Shin received his M.S. and Ph.D. degree in the Department of Industrial and Systems Engineering at Korea Advanced Institute of Science and Technology. He has worked as a software engineer at Samsung Electronics since 2020. His current research interests include decision making under uncertainty, reinforcement learning and operations management in manufacturing systems. Email address: kyohong.shin@gmail.com (ORCiD https://orcid.org/0000-0002-4243-4144)

Hoon Jang is an assistant professor of College of Global Business at Korea University, Korea. His research interests are primarily in the area of complex system designs, data-driven modelling and applied optimization problems. Dr. Jang obtained his MS and PhD degree from KAIST in the Dept. of Industrial & Systems Engineering. Email address: hoonjang@korea.ac.kr (ORCiD https://orcid.org/0000-0002-5242-4900)

Hae Joong Kim received his B.S., M.S. and Ph.D. degree in Industrial Engineering from Seoul National University in 2001, 2003, and 2008, respectively. He has worked as a software engineer at Samsung Electronics since 2008. His research interests include mathematical optimization and machine learning approaches for manufacturing systems. Email address: haejoong.kim@gmail.com (ORCiD https://orcid.org/0000-0002-3727-094X)

An MDP based Lifter Assignment Algorithm for Inter-line Transportation in Semiconductor Fabrication

Abstract

As semiconductor device geometries continue to shrink, the semiconductor manufacturing process becomes increasingly complex. This usually results in unbalanced utilization of machines and decreases overall productivity. One way to resolve such a problem is to share the resource capacity between different lines divided by floors. To this end, designing an efficient lifter assignment method to more efficiently manage transfer requests (TRs) of wafer lots to different floors is required. Motivated by this, our study addresses the assignment of lifters for delivering wafer lots to different floors. Unlike previous studies, which consider the current state of the system, our study considers both the current and possible future states of the system. We formulate an optimization model based on the Markov decision process. Then, we design an efficient method as a solution using both clustering and tournament selection methods. Experiments based on historical data confirm the effectiveness of the proposed algorithm in reducing travel times and delivery delays compared to the benchmark rules in practice. Sensitivity analysis demonstrates the robustness of the proposed model as the number of TRs increased. The proposed approach is expected to yield significant economic savings in both operating costs and labor.

Keywords: manufacturing; semiconductor; automated material handling system; lifter; Markov decision process

1. Introduction

Wafer fabrication in semiconductor manufacturing is an extremely complex process. To produce an electronic or photonic circuit on a minute semiconductor wafer, a wafer lot in a semiconductor fabrication facility (FAB) is frequently transported between and within different processing areas, such as those dedicated to etching or imprinting. Many re-entrant processes in which a wafer lot enters the same group of machines repeatedly are also necessary. To manage these complex material flows, automated material handling systems (AMHSs)—sophisticated material control systems that move

materials from one machine to another—have been successfully applied in semiconductor manufacturing. In particular, an AMHS employing an overhead hoist transfer (OHT) reduces the wafer transport time dramatically and contributes to reduced cycle times and increased equipment utilization with improves on-time delivery.

As semiconductor manufacturing processes are becoming increasingly complex, and the number of production steps is increasing, a single floor line often has an inadequate capacity to cover such production needs. Therefore, FABs with multiple floors have been introduced recently [1]. At the initial FAB operation phase, the inter-floor transfers from one floor are constrained at the minimum level because the capacity of the process machines of each floor is kept balanced. However, as the product mix changes, the equipment utilization on each floor becomes unbalanced and more inter-floor transfers to share equipment resources on other floors are required. In general, the number of lifters for inter-floor transfers is determined at the minimum level because the lifter itself is expensive to set up and consumes cleanroom space. Thus, the lifters become a major bottleneck of the AMHS when the inter-floor transfers increase [2].

The typical process of inter-floor transport is shown in Figure 1a. Upon the arrival of a transfer request (TR) to transport a wafer lot from a machine on floor A to another on floor B, the AMHS allocates an OHT and a lifter connected to floor B. Then, the assigned OHT unloads the wafer lot from the source machine and moves it to the dedicated lifter. After the OHT arrives at the lifter, it loads the wafer lot into the in-buffer port of the lifter for transportation to floor B. The buffer port is where a lot waits for transport to another floor by the lifter or to the destination by an OHT afterward. The wafer lot in the in-buffer port is transported to the out-buffer port on another floor by the rack-master (Figure 1b). Finally, an OHT on floor B transports the wafer lot to the destination machine.

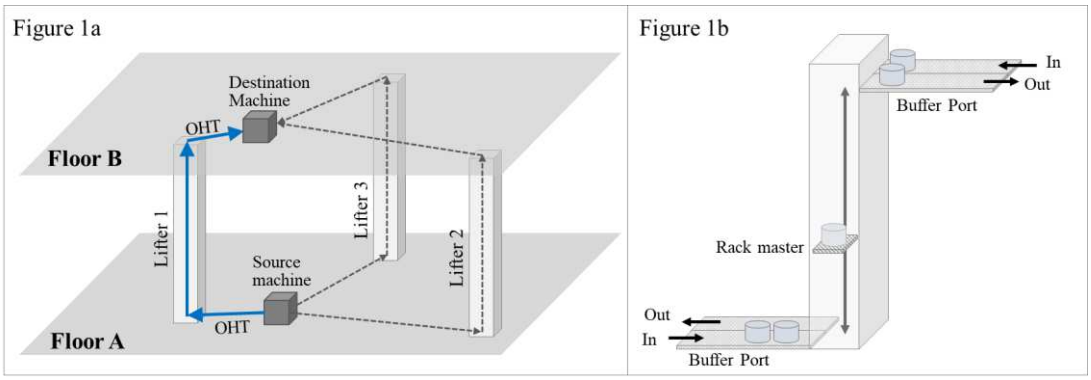

Figure 1: A typical process of the inter-line transportation in semiconductor FAB (Figure 1a); The physical structure of a lifter (Figure 1b)

Transport times between floors are much longer than on a single floor because an inter-floor transfer consists of two transfers by OHTs, a transport by a lifter, and waiting at the buffer port of the lifter. Thus, when the lifter is not selected properly, the travel time to the destination machine could be excessively long, and the shared machines would not be fully utilized on account of the transportation delays [2]. In practice, when the AMHS selects the lifter for each inter-floor transfer, dispatching rules, such as the round-robin rule or the shortest-travel-distance rule, are widely used instead of analytical models to identify the acceptable solutions in the dynamic environment of the actual scenario. The round-robin rule chooses lifters in a pre-set circular order; TRs are evenly distributed to all lifters. This approach has the advantage of keeping lifter workloads even and minimizing the variance of the waiting time in the buffer port of the lifter. It also decreases the vehicle congestion around lifters by preventing excessive traffic on specific lifters.

However, because the travel time of an OHT is not considered, a lifter with a long distance to travel is frequently assigned, which causes a long travel time. The shortest-travel-distance rule assigns wafer lots to the nearest lifters to reduce the travel time. However, TRs could then be concentrated on specific lifters, which increases the waiting time in the buffers and causes congestion. Thus, with such simple rules, the

operational efficiency of the AMHS is limited. For this reason, system operators manually adjust the rules according to changes in the manufacturing environment. Nevertheless, this sort of adjustment is labor intensive and can miss the correct timing.

We therefore propose a method based on the Markov decision process (MDP) to optimally assign lifters to minimize the inter-floor transportation time in dynamic manufacturing conditions. One of the biggest limitations of current practice is that the future states of lifters operating in a FAB are not considered. Therefore, undesirable situations, such as delays in transporting a wafer lot to another line, cannot be fully prevented. However, by employing MDP, which derives solutions by considering a system's future states, we can address this issue. We also evaluate the performance of the proposed approach by using an emulator with data gathered from an actual FAB.

The remainder of this paper is structured as follows. Section 2 reviews the existing literature related to this study and discusses its implications and significance. Our contributions to the existing literature are also clarified. Section 3 details the mathematical model. Section 4 describes the proposed solution algorithm based on MDP. Section 5 describes the experimental design, the results of the experiments, and their implications. Finally, Section 6 concludes our study.

2. Background

Our study addresses a lifter assignment problem that has been receiving considerable attention recently. Specifically, our objective is to increase the efficiency of AMHS and the productivity of FAB in the manufacturing domain. With this consideration, we first briefly review the research stream of AMHS, and then focus on relevant studies. For a general review on AMHS, refer to [3-6].

Modification of AMHS in operating FABs, such as relocating the machines or installing additional rails, storages, or lifters, typically results in a tremendous

production loss because it requires shut down of the partial or entire system during that period. Therefore, many studies on vehicle management, including vehicle allocation, dispatching, and routing, which are the viable options to improve productivity without the power shut down of FABs, have been conducted.

The vehicle allocation problem in the semiconductor AMHS generally involves determining the optimal fleet size to minimize delivery time. Many researchers have proposed determination of the optimal fleet size, which would fulfil a specific transfer requirement [7-10]. To solve a certain vehicle allocation problem, some studies focused on dynamically repositioning idle vehicles to appropriate locations to minimize the transport time. Kim et al. [11] and Vahdani [12] proposed an idle vehicle circulation policy to balance the number of idle vehicles among the bays. More recently, Lee et al. [13] and Schemaler et al. [14] utilized the information of future transports to respond to highly dynamic manufacturing environments.

Another important issue in managing AMHS is to design efficient vehicle dispatching rules. In general, the vehicle dispatching problem aims to find the most appropriate dispatching rules to achieve operational goals, such as minimizing vehicle waiting time. For this, many studies compared several dispatching rules in various conditions [15-18]. Several studies showed that the dispatching rules have a significant impact on the performance of the system, such as the average transfer time, waiting time, vehicle utilization, and even throughput. Some investigations introduced new dispatching rules by reassigning vehicles during the unloading travel time [19-21]. They strived to dynamically exchange the vehicle-request assignment according to the system scenario and showed that it has a significant positive effect on reducing the vehicle assign time as well as load travel time.

The third and last problem to improve AMHS operation is to design efficient vehicle routing rules. The vehicle routing problem determines the optimal vehicle routes to visit, thereby aiming to minimize the transport times of TRs. Typically, studies in vehicle routing problems have focused on finding conflict-free routes [22-25]. Another objective is to design vehicle routing decisions that consider traffic congestion, which should be avoided as much as possible [26,27].

More recently, studies on improving AMHS have expanded their scope to include storage or lifter allocation. This expanded scope is inevitable owing to the fact that many FABs are starting to use multiple lines (floors) to increase their production capacity. Kim et al. [28] and Siebert et al. [29] focused on lot targeting to improve the material flows through the storage locations, while Lee et al. [30] proposed a machine learning approach to select the best dispatching rule for storage allocation. They showed that the storage allocation significantly affects the performance of the AMHS.

In terms of prior research, to the best of our knowledge, so far only three studies address the lifter assignment problem for inter-line transfers in semiconductor manufacturing facilities [1, 31, 32]. The first study to introduce the lifter assignment problem for inter-line transportation is by Jimenez et al. [31]. They suggested four rules for the selection of the rail for TRs to be sent to a stocker on the same floor (intra-line transportation), and four rules for the selection of the lifter to be sent to another floor (inter-line transportation). Based on simulation experiments in which the ratio of the intra-line to inter-line transportation volume was changed, they determined a rule that considers the number of waiting lots on lifters. This rule showed good performance when the inter-line transportation was increased, whereas when the inter-line transportation was decreased, considering both the travel time and the number of waiting lots was recommended. The other two studies were conducted relatively

recently. Na et al. [1] presented the lowest utilization, the integer programming model, and the shortest-expected-arrival-time (SEAT) rule as methods of selecting a lifter. They proved that the SEAT method decreases the average delivery time by 8.9% compared to the round-robin method used in practice. Lee et al. [32] proposed operation policies to improve the efficiency of lifter operations in material handling of semiconductor lines. Their policies involve the specific and practical operational decisions, such as the number of virtual ports, activating the alternative storage, and using a shelf extraction procedure by rack maters. However, they did not suggest an algorithm or mathematical model to generate the most optimal policies.

Our study differs from the above research in two respects. First, the prior studies that provided heuristic approaches were empirically shown to yield a reasonable performance; however, they did not provide an optimal solution algorithm that accounts for the stochastic nature of the problem. Also, they only considered the current state, whereas ours can consider both the current and the future states, thus making it possible to derive better solutions for AMHS management.

Based on the literature review above, this field has been studied quite extensively. However, we believe that our study contributes to it in three ways. First, our optimization model is the first to consider stochastic dynamics in the lifter-assignment problem. To the best of our knowledge, all the studies on the lifter-assignment problem [1, 31, 32], as well as the rules currently in actual use on the line, consider only the current state. Our proposed model considers not only the current state but also the states that will occur in the future. Second, we propose an algorithm that can be applied in a real-world setting. In general, MDP models have difficulty finding an optimal solution when the target system is large or complex. We solve this limitation by using a clustering technique to simplify the representation of the source machine

with similar travel-time distributions and by grouping multiple lifters to partition the problem. Lastly, we propose a framework with autonomous control that can serve as the basis for establishing a smart factory. Since operators adjust the rules empirically according to changing operating conditions, there is a large variation in operational efficiency depending on the operator's expertise, and it is very labour-intensive. In contrast, the proposed framework enables autonomous control by automatically updating the model.

3. Mathematical Model

We consider a lifter-assignment problem in which the delivery time is to be minimized. When a TR is generated and wafer lots should be transported to another line, the sequence of events occurs, as shown in Figure 2, from the perspective of the departure floor. We define assign time as the time from the moment of the TR’s arrival until the OHT arrives at the source machine and loads wafer lots; travel time as the time required between unloading wafer lots from the source machine and loading them into the buffer port of the lifter; waiting time as the time it takes for wafer lots to be picked up by the rack-master after being loaded into the lifter's buffer port; and cycle time as the time until the rack-master transports the lot to another floor and returns to the departure floor. Finally, the delivery time is the sum of the travel time and waiting time.

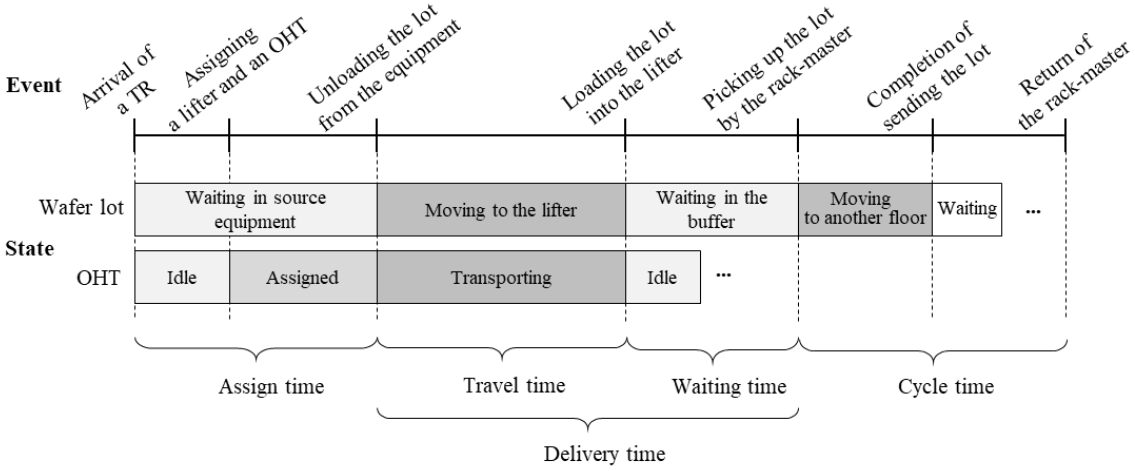

Figure 2: Event flow and time definition of inter-line transportation

As noted, to consider both the cost of the decision and the long-term effect on the AMHS system, we mainly formulated this problem using MDP. MDP is a modelling technique that takes into account the immediate rewards (or costs) of current decisions and their effect on the entire system under uncertainty [33]. MDP is typically used to model sequential decision-making problems.

An MDP model consists of *system state (s), decision (a), cost (C(s,a,s')), and state transition probability (p(s,a,s'))*. The system is in one of the possible states in the state set S. At each decision epoch, the decision maker observes the current state *s* of the system and chooses an action a from a set of possible actions *A(s)*. When the selected action is performed, the current system state changes to a new state s' according to the transition probability *p(s,a,s')*. The decision maker receives a cost *C(s,a,s')* when certain actions are performed, or certain conditions are satisfied.

Using these quantities, the decision to minimize the value function of the state is derived. The value function is formulated using Bellman's equation:

$$V(s) = \min_{a \in A(s)} \sum_{s'} p(s,a,s') \times \{C(s,a,s') + \gamma V(s')\}$$

where γ is a discount factor satisfying $0 \leq \gamma < 1$. Finally, a policy that maps each state to the optimal action is derived.

The lifter assignment problem can be viewed as a sequential decision-making problem because it is necessary to determine which lifter to send the TR to at the moment the TR is generated. Therefore, we developed an MDP model that aims to minimize the expected delivery time. The components of the proposed MDP model are now herein detailed.

System state. In the proposed model, the system state is defined as $S = (TR_1, \ldots, TR_j, \ldots, TR_L, b_1, \ldots, b_j, \ldots, b_L)$, where TR_j is the number of TRs proceeding to lifter j, b_j is the number of TRs waiting in the buffer of lifter j, and L is the number of lifters.

Transition probability. The system state transitions to other states upon the occurrence of three events (*Arrival of a TR* event, Loading the lot (into the lifter) event, and *Picking up the lot by the rack-master* event). Among these, when the *Arrival of a TR* event occurs, the lifter should be determined. An example of the state transition diagram is shown in Figure 3.

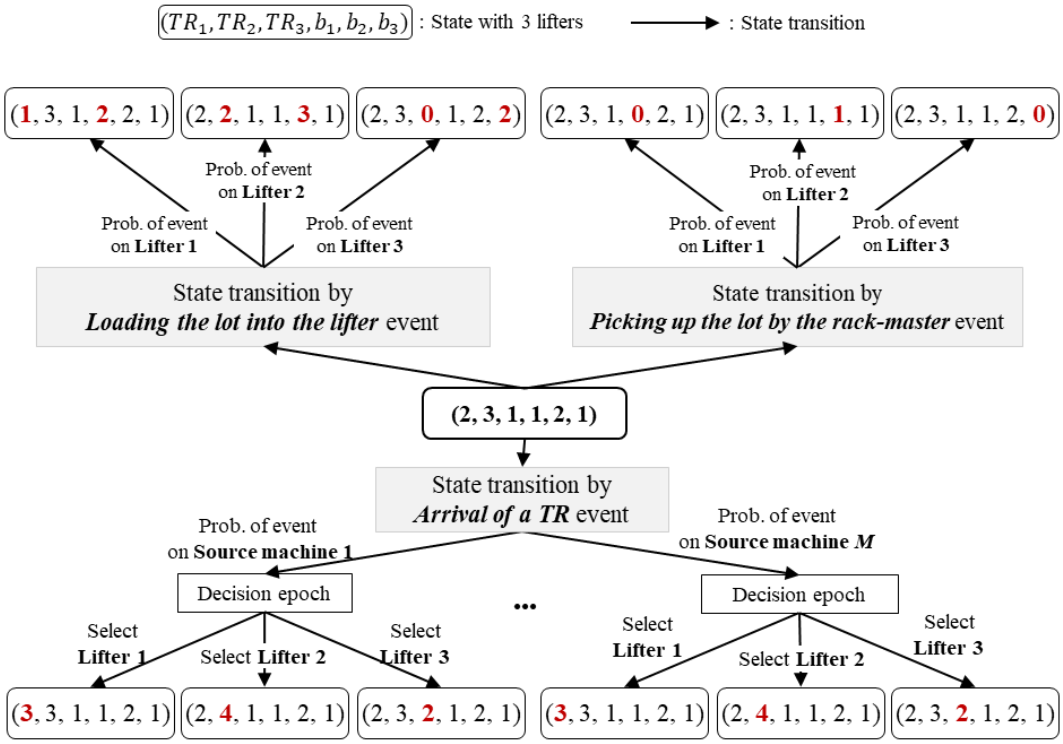

Figure 3: Example of state-transition of the system with three lifters

Based on the results of the preliminary analysis that fits historical data into some mathematical distributions, we assume that the inter-arrival times for all events follow an exponential distribution. Under this assumption, the event generation process of this model follows the Poisson process. Therefore, the system state transition probability is given by the arrival rate of each event divided by the sum of all rates [33, 34],

$$\rho = \sum_{i=1}^{M} \lambda_i + \sum_{j=1}^{L} TR_j \times \mu_j + \sum_{j=1}^{L} 1_{\{b_j>0\}} \times w_j$$

where λ_i denotes the inter-arrival rate of TR from source machine i; $\frac{1}{\mu_j}$ denotes the travel time to the lifter j; and $\frac{1}{w_j}$ represents the cycle time of the lifter j. The first term is the rate at which TRs arrive from each of the M source machines, and the second term is the rate at which OHTs arrive at each lifter and load lots into them. The last term is the rate at which the lifter's rack-master transports the lot to another floor and returns

to the departure floor. This is the reciprocal of the cycle time. The indicator function $1_{\{b>0\}}$ takes a value of 1 when there is at least one waiting TR in the lifter and a value of 0 when there is no waiting TR. Finally, the transition probabilities for the three events are as follows:

- Arrival of a TR (source machine i) $= \frac{\lambda_i}{\rho}, i \in \{1, \dots, M\}$,
- Loading the lot (into the lifter j) $= \frac{TR_j \times \mu_j}{\rho}, j \in \{1, \dots, L\}$,
- Picking up the lot by the rack-master (of lifter j) $= \frac{1_{\{b_j>0\}} \times w_j}{\rho}, j \in \{1, \dots, L\}$.

Decision. When a TR arrives, the system must determine which lifter should transport the lot. The number of lots that can be queued in the lifter's buffer is limited. When it exceeds the buffer capacity of the lifter, an OHT carrying a wafer lot cannot load the lot into the buffer, and it circles around or waits on the road until there is room. This not only increases the travel time of the OHT, but also creates congestion around the lifter and adversely affects the operation of other OHTs. Therefore, an upper bound is set in advance on the sum of the number of TRs moving to the lifter and the number of TRs waiting in the buffer of the lifter. When the sum of the number of TRs proceeding to lifter j and the number of TRs waiting in the buffer of lifter j $(TR_j + b_j)$ reach the predetermined upper limit, the lifter j is not selected until the value falls.

Cost function. Two cost function types are used. When a TR is generated and a lifter is selected for the destination, the cost is defined as the average travel time from the source machine to the lifter. However, when the TR arrives at the lifter, the cost is the waiting time, which is calculated by multiplying the cycle time of the lifter by the

number of jobs waiting in the buffer. Formally, the cost C in state s with decision $a = j^*$ is

$$C(s, a = j^*) =$$

$$\begin{cases} \bar{\tau}_{ij^*}, & \text{if TR is generated from source machine } i \text{ and lifter } j^* \text{ is selected,} \\ \overline{CT}_{j^*} \times b_{j^*}, & \text{if the TR arrives at lifter } j^*, \\ 0, & \text{otherwise,} \end{cases}$$

where $\bar{\tau}_{ij^*}$ is the average travel time to source i from lifter j^* and $\overline{CT}_{j^*}$ is the average cycle time of lifter j^*.

Objective function. The goal of the decision problem is to minimize the expected delivery time, which is the sum of the travel time from the source machine to the lifter and the waiting time in the buffer. The objective function is accordingly formulated using Bellman's equation:

$$V(s) = \min_{j^* \in \{1,\dots,L\}} \sum_{i=1}^{M} \frac{\lambda_i}{\rho} \times \left[C(s, j^*) + \gamma V\left(s + e_{j^*}\right)\right] + \sum_{j=1}^{L} \frac{TR_j \times \mu_j}{\rho} \times \left[C(s, j) + \gamma V\left(s - e_j + e_{M+j}\right)\right] + \sum_{j=1}^{L} \frac{1_{\{b_j > 0\}} \times w_j}{\rho} \times \gamma V(s - e_{M+j}).$$

We define e_k as a $1 \times |2M|$ unit vector, for which the element corresponds to k^{th} position.

4. Solution Method for Lifter Assignment in an Actual Fab

Although the MDP model in Section 3 guarantees the provisioning of an optimal policy for decisions on a lifter assignment, there exists a critical computational issue from a practical standpoint. An MDP model is generally solved by using dynamic programming (DP). However, when the problem size approaches the size typically found in real-world applications, the MDP model requires tremendous computing

resources and time to handle the large number of variables. Because of this "*curse of dimensionality*," solutions of realistic problems usually cannot be obtained in a reasonable amount of time. Considering the size of an actual FAB and its complex operating environment, the size of the state space must be very large. To tackle this difficulty, we propose a solution method to efficiently solve the given problem.

4.1 Solution approach

The main idea of the proposed solution approach is to reduce the dimension of the original problem while keeping the quality of its solution. The dimension of the problem is determined by the number of source machines and destination lifters. In our solution approach, we employ the concepts of clustering and divide-and-conquer. First, the dimension of the source machines is reduced by grouping numerous machines into similar clusters. Second, we create several sub-problems by partitioning the destination lifters into sub-groups of manageable size. Then, the optimal policies are obtained for the sub-problems and all possible scenarios derived during the tournament. Finally, in the execution phase, the best lifter for each TR is determined by the tournament among the sub-problems.

Modelling each machine individually increases the complexity of the MDP model significantly because a typical FAB usually has hundreds to thousands of machines. Thus, we cluster the machines based on similar travel times from the machines to the lifters. For example, 40 machines are grouped into 7 clusters as shown by different colours in Figure 4. The machines within a cluster have similar distributions of travel time to destination lifter.

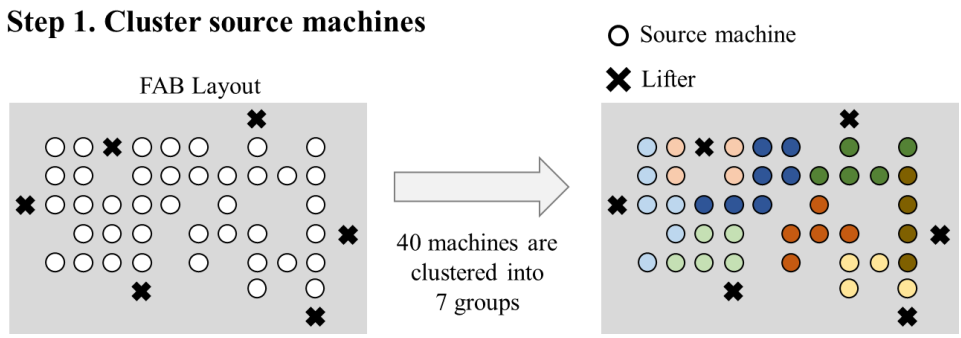

Figure 4: Example of step 1 of the solution approach: cluster source machines

Next, by partitioning the lifters into different groups we divide the problem into several sub-problems that can be solved optimally within a reasonable computational time. Thus, the number of lifters in each sub-problem is limited to a maximum of *N* lifters. For instance, as in Figure 5, the original problem is divided into 2 sub-problems coloured in red and blue, with a maximum of three lifters each. This means that the dimensionality of the original problem is cut by the number of sub-problems.

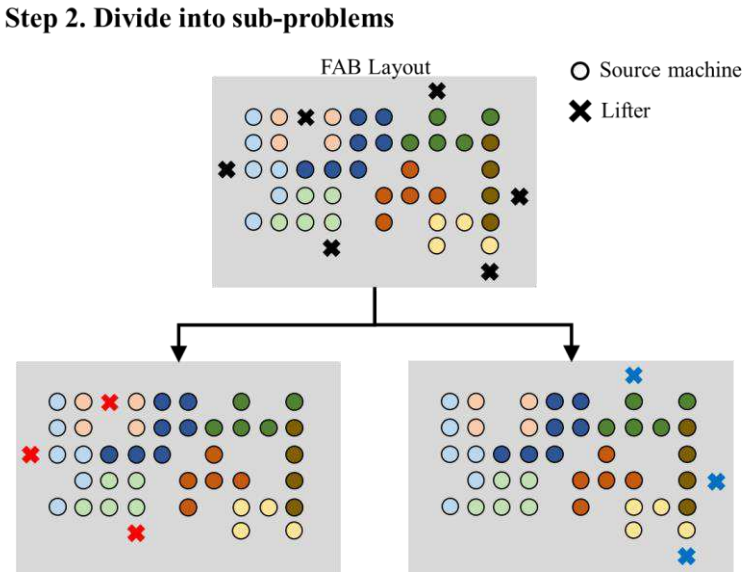

Figure 5: Example of step 2 of the solution approach: division into sub-problems

The next step is to obtain the optimal policies for each sub-problem. In order to obtain the final best lifter, it is also necessary to solve new sub-problems consisting of

the optimal lifters selected from the sub-problems. That is, all optimal policies for all possible problems have to be prepared *a priori*. In Figure 6, the nine scenarios that are possible based on a combination of two sub-problems that have three results, respectively, are shown. It is worth noting that if the number of groups is greater than *N*, the selected lifters are regrouped in the manner described above and the tournament selection method is applied.

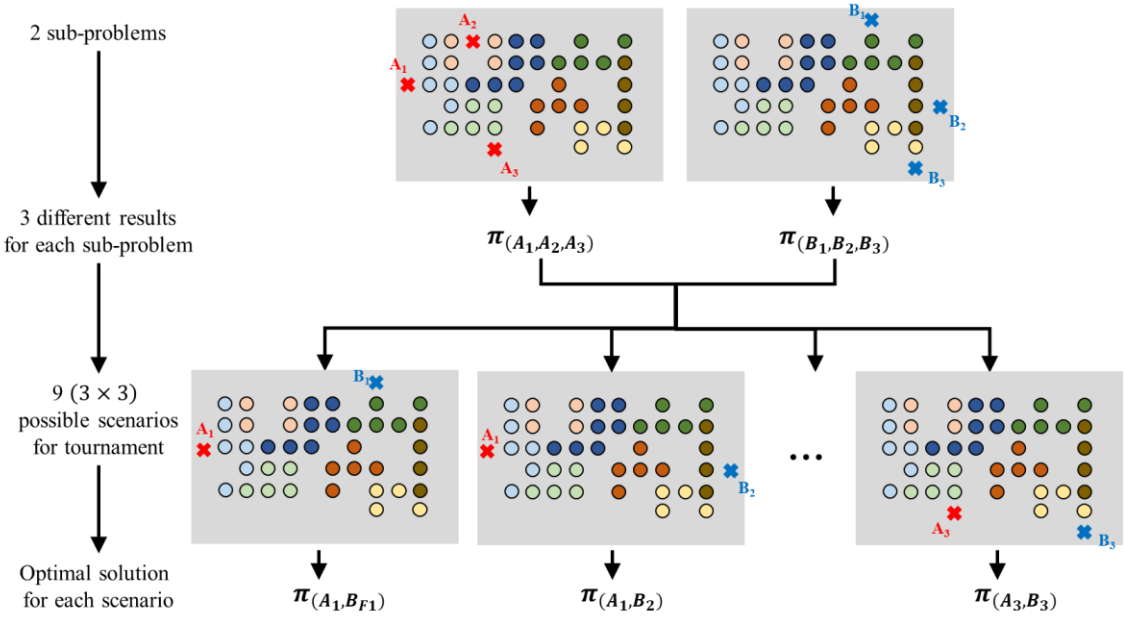

Figure 6: Example of step 3 of the solution approach: preparation of the optimal policies for all possible scenarios *a priori*

The final step is to select the best lifter for each TR in the execution phase within a short time, almost real-time. When a new TR is issued, the best lifter is derived by the tournament selection that only selects the optimal policies, which have already been solved in the previous step 3 (Figure 7).

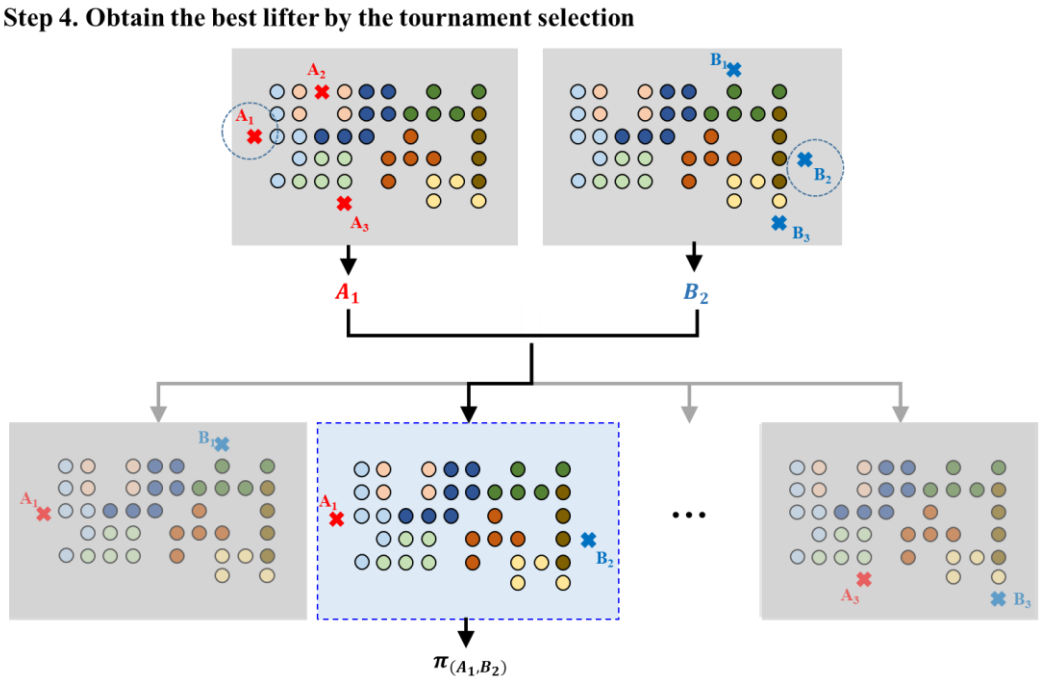

Figure 7: Example of step 4: Obtaining the best lifter by the tournament selection

4.2 Algorithms for lifter assignment

The detailed procedure is presented as pseudo-code in Tables 1a and 1b. Considering the real operation of FABs, a lifter assignment algorithm using MDP is not necessarily updated in a real-time manner owing to the fact that the optimal policies can be prepared by considering all possible situations *a priori*. However, a lifter assignment step should be activated in real time whenever a TR for inter-line transportation arrives. By considering this circumstance, we devised two algorithms for the actual FAB operations. One is the *Lifter Assignment Policy Preparation Module,* which constructs MDP models and derives lifter assignment policies using historical data periodically; the other is the *Lifter Assignment Module,* which assigns lifters in real time using policies derived in the *Lifter Assignment Policy Preparation Module*.

Table 1a: Pseudo-code for the operation of a real FAB – Lifter Assignment Policy Preparation Module

Input:	$\boldsymbol{M} = \{i \in \{1, \dots, M\}\}$: machine ID set
	$\boldsymbol{L} = \{j \in \{1, \dots, L\}\}$: lifter ID set
	τ_{ij} : the travel time from source machine i to lifter j, $\forall i \in \boldsymbol{M},\ \forall j \in \boldsymbol{L}$
	CT_j : the cycle time of lifter $j, \forall j \in \boldsymbol{L}$
	IT_i : the inter-arrival time of TRs in machine $i, \forall i \in \boldsymbol{M}$
	$(x, y)_i$: x, y coordinates of machine $i, \forall i \in \boldsymbol{M}$
	$(x, y)_j$: x, y coordinates of lifter $j, \forall j \in \boldsymbol{L}$
Step 1	**Cluster source machines**
Step 1-1	Make a dataset for clustering, $\boldsymbol{D} = \{\forall i \in \boldsymbol{M} \mid \tau_{ij} > 0, \forall j \in \boldsymbol{L}\}$ (machines which sent TRs to all lifters)
Step 1-2	Get clustering result, $\{(i,\ CID_i), \forall i \in \boldsymbol{D}\} \leftarrow ClusteringAlgorithm(\boldsymbol{D})$ (CID_i : cluster ID of machine i)
Step 1-3	Create a classifier which assigns a cluster ID for a machine, $CID \leftarrow f(x, y)$, by learning $\{((x, y)_i,\ CID_i), \forall i \in \boldsymbol{D}\}$
Step 1-4	Calculate $\bar{\tau}_{kj}, \overline{IT}_k,\ \overline{CT}_j\ \forall k \in \{1, \dots, N_{CID}\}, \forall j \in \boldsymbol{L}$
Step 2	**Divide into sub-problems**
Step 2-1	Make initial lifter groups $\boldsymbol{IG} = \left\{IG_1, \dots, IG_{\left\lfloor \frac{L}{N} \right\rfloor}\right\}$ using $(x, y)_j, \forall j \in \boldsymbol{L}$ $\left(2 \le n(IG_g) \le N, \forall g \in \left\{1, \dots, \left\lfloor \frac{L}{N} \right\rfloor\right\}\right)$
Step 2-2	Make additional groups, $\boldsymbol{AG}$, for all combinations that can be made when one lifter is picked from each group
Step 3	**Prepare the optimal policies for all possible scenarios a priori**
Step 3-1	Derive the optimal policy, π_{IG_g}, which determine one among the lifters in $IG_g, \forall g \in \left\{1, \dots, \left\lfloor \frac{L}{N} \right\rfloor\right\}$
Step 3-2	Derive the optimal policy, π_{ag}, which determine one among the lifters in each additional group $(ag), \forall ag \in \boldsymbol{AG}$
Output:	$\pi_{IG_g}, \forall g \in \left\{1, \dots, \left\lfloor \frac{L}{N} \right\rfloor\right\}$: the optimal policy for each initial lifter group
	$\pi_{ag}, \forall ag \in \boldsymbol{AG}$: the optimal policy for each additional lifter group
	$f(x, y)$: a classifier which assigns a cluster ID for a machine using x, y coordinates of a machine

Table 1a presents the pseudo-code for the *Lifter Assignment Policy Preparation Module*. In step 1, using the historical data of which machines sent TRs to all lifters, the source machines with similar travel time distributions for each lifter are grouped together. Next, in order to assign cluster IDs for machines that did not send TRs to all lifters, we create a classifier, $f(x, y)$, by learning the *x* and *y* coordinates and cluster IDs. By using this classifier, all machines can be grouped into clusters. After allocating a cluster to all machines, the average travel time from each cluster to each lifter ($\bar{\tau}_{kj}$)

and the inter-arrival time of the TR in each cluster ($\overline{IT}_k$) are obtained using data of the machines in each cluster.

In a similar way, in step 2, we group the lifters using their physical location information, $\boldsymbol{IG} = \left\{IG_1, \dots, IG_{\lfloor \frac{L}{N} \rfloor}\right\}$. That is, by using the coordinates *x* and *y* of all lifters, we cluster up to N lifters as one group. N can be set according to the computing environment in which the algorithm is used to manage the computational load. Then, we form additional groups, $\boldsymbol{AG}$, for all combinations that can be created when one lifter is picked from each group. For example, we create initial lifter groups for seven lifters with *N* set to three using the coordinates *x* and *y* of all lifters, $\boldsymbol{IG}$={(1,2,3), (4,5), (6,7)}. Based on these initial groups, the following additional groups are created: $\boldsymbol{AG}$={(1,4,6), (1,4,7), (1,5,6), (1,5,7), (2,4,6), (2,4,7), (2,5,6), (2,5,7), (2,4,6), (2,4,7), (2,5,6), (2,5,7)}.

In step 3, we derive the optimal policy for all lifter groups formed in step 2 ($\pi_{IG_g}, \forall g \in \left\{1, \dots, \left\lfloor \frac{L}{N} \right\rfloor\right\}$ and $\pi_{ag}, \forall ag \in \boldsymbol{AG}$). In other words, given the current state and source machine, a policy that determines the best lifter among lifters in the group is obtained. Optimal policies of additional lifter groups, $\pi_{ag}, \forall ag \in \boldsymbol{AG}$, are used when selecting the final optimal lifter through the tournament method in the *Lifter Assignment Module*. The best lifter in each initial group is determined by the current system state and cluster ID of the source machine. Since we do not derive the optimal policy from the model constructed in real time, we must consider all combinations of lifters that can be selected from each initial group. Therefore, by selecting one lifter from each group, we create additional groups for all cases that can be made and obtain the optimal policy of these groups.

Table 2b: Pseudo code for the operation of a real FAB – Lifter Assignment Module

Input:	i : Source machine ID
Step 4	**Obtain the best lifter by the tournament selection**

Step4-1 Get the current state, $S = (TR_1, \ldots, TR_j, \ldots, TR_L, b_1, \ldots, b_j, \ldots, b_L)$

Step 4-2 Get the cluster ID of i, $CID_i \leftarrow f((x,y)_i)$

Step 4-3 Get the best lifter among lifters in each initial lifter group given S and CID_i

$j^*_{IG_g} \leftarrow \pi_{IG_g}(S, CID_i), \forall g \in \left\{1, \ldots, \left\lfloor \frac{L}{N} \right\rfloor\right\}$

Step 4-4 Get the optimal lifter from the optimal policy of ag consisting of $j^*_{IG_g}, \forall g \in \left\{1, \ldots, \left\lfloor \frac{L}{N} \right\rfloor\right\}$ given S and CID_i

$j^* \leftarrow \pi_{ag}(S, CID_i),\ ag = \left\{j^*_{IG_1}, \ldots, j^*_{IG_{\lfloor \frac{L}{3} \rfloor}}\right\}$

Output: j^* : the optimal lifter to be sent TR

Table 1b shows the pseudo-code for the *Lifter Assignment Module*. When a TR is generated from source machine i, the *Lifter Assignment Module* checks the current state (S) and obtains the cluster ID of the source machine (CID_i) using the classifier (Step 4-2 in the *Lifter Assignment Module*). The module obtains the best lifters $\left(j^*_{IG_g}, \forall g \in \left\{1, \ldots, \left\lfloor \frac{L}{N} \right\rfloor\right\}\right)$ from the optimal policy of each initial lifter group $\left(\pi_{IG_g}(S, CID_i)\right)$ by using the machine's cluster ID and current state as inputs (Step 4-3 in the *Lifter Assignment Module*). Finally, in Step 4-4, the module finds a lifter group $\left(ag = \left\{j^*_{IG_1}, \ldots, j^*_{IG_{\lfloor \frac{L}{N} \rfloor}}\right\}\right)$ consisting of lifters selected from each initial group and selects the optimal lifter through the optimal policy of the group $\left(\pi_{ag}(S, CID_i)\right)$. Let us consider the previous example again. If lifter 1, lifter 4, and lifter 7 are selected as the optimal lifters in each initial lifter group, ***IG***={(1,2,3), (4,5), (6,7)}, we use the optimal policy of the group (1, 4, 7), $\pi_{(1,4,7)}$.

4.3 A framework for system implementation

The proposed solution approach and algorithms theoretically enable effective assignment of lifters in actual FABs. The remaining question is how to apply these algorithms to real FAB operation. Figure 8 shows the system framework for lifter

assignment in real FABs. The two modules described above are aligned to the central operating system of AMHS; however, they work independently.

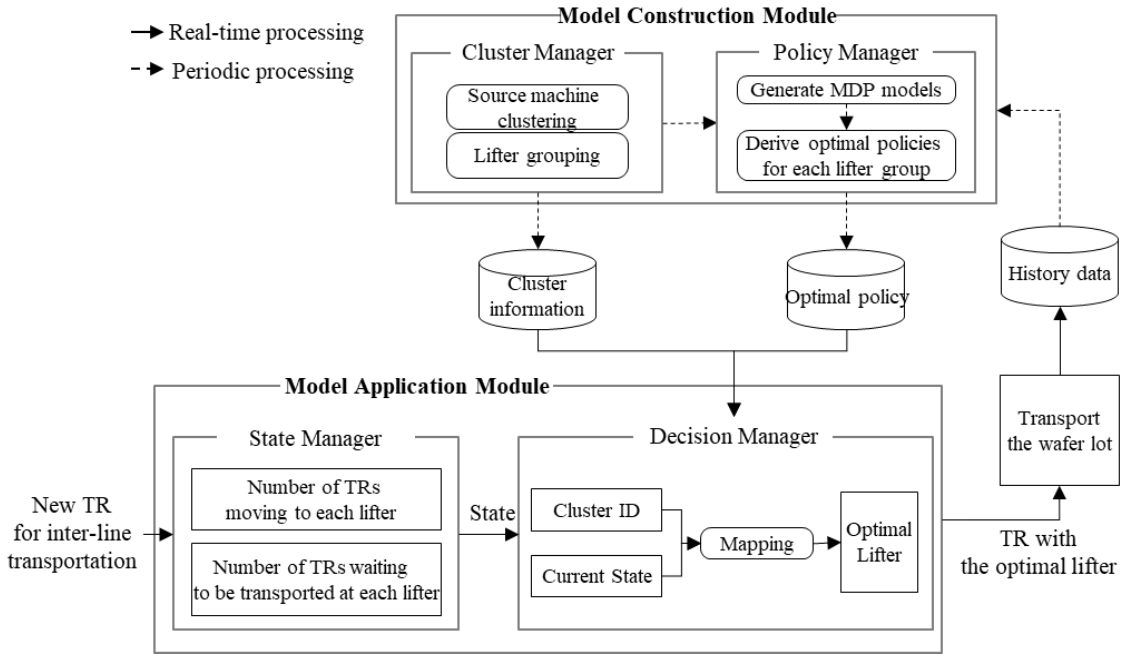

Figure 8: System implementation framework for lifter assignment in real FABs

The *Lifter Assignment Policy Preparation Module* produces optimal policies for lifter groups and cluster information for source machines by periodically using historical data. All these outputs are stored in the central system of AMHS in table format. The table of the optimal policy contains the system state, cluster ID, lifter group ID, and the optimal lifter ID in that group as columns. Using this table, given the current state and cluster ID, the best lifter in each group can be selected easily. The cluster information table stores the results from mapping the machine ID to the cluster ID.

In the *Lifter Assignment Module*, when a TR for inter-line transportation is generated, the state manager checks the current system state (the number of TRs moving to each lifter and the number of TRs waiting to be transported at each lifter). Then, the decision manager uses as input the name of the source machine that sent the TR and the current state. By automatically processing the information, it obtains the x

and *y* coordinates of the source machine and reads its cluster ID from the cluster information table. Next, using the current state and cluster ID, the decision manager identifies the best lifter for each initial group and determines the best lifter by applying the tournament technique. Thereafter, the wafer lot is transported by the lifter determined in the *Lifter Assignment Module*, and the transport records are stored in the history data table. These stored data are periodically used by the *Lifter Assignment Policy Preparation Module*.

5. Experimental Results

To verify the effectiveness of the proposed approach, we conduct both a simple analysis in a virtual environment and a simulation study that mimics a real-world FAB using an AMHS-oriented simulation model (emulator) offered by that company. Since it describes the actual FAB operating environment and is guaranteed to be a high-fidelity simulation, it is an appropriate testbed to examine various approaches to designing effective operational strategies for FABs.

In Section 5.1, we present the details of the experiments. Section 5.2 describes the results of the experiments in a simple environment. In this section, the merits of using our approach are presented. In Section 5.3, the experimental results obtained through the realistic FAB simulation are presented and discussed.

5.1 Experimental settings

We first conduct virtual experiments by assuming a small-sized environment. There are three lifters and eight machines. The travel times for all combinations of lifters and machines are shown in Table 2. The inter-arrival times for each machine, assumed to be random, are also shown. The values in the table are the mean values of the exponential distribution. The cells coloured grey indicate the shortest travel times from each source

machine to the lifter.

Table 3: Experimental settings for the simple (virtual) experiment

Travel time (Second)		Lifter ID			Inter-arrival time
		1	2	3	
Source machine ID	1	50	70	90	40
	2	60	90	120	45
	3	55	70	100	30
	4	100	110	140	35
	5	125	105	80	50
	6	100	70	80	40
	7	85	115	45	30
	8	55	120	80	45
Cycle time		35	40	30	

Next, we construct a simulation study based on historical data gathered from Samsung Semiconductor. The FAB we used for our simulation study consists of more than 800 source machines, more than 150 OHTs, and 7 lifters. We use data from March 2020; more detailed information about the data cannot be disclosed because of security issues. Since the goal of this study is to examine the effectiveness of the proposed approach for lifter assignment, we use two bench-marking assignment rules (described earlier) that are used in the actual FAB: the round-robin rule and the shortest-transportation-time rule. The round-robin rule distributes TRs evenly to all lifters. It primarily aims to balance utilization of all lifters. Since the distance from the source machine to the lifter is not considered, it is not expected to reduce delivery times. The shortest-transportation-time rule aims to transfer the TR to another floor as quickly as possible by selecting a lifter for which the weighted sum of the travel time between the two machines and the number of TRs being transferred is the smallest. In practice, the weight may be adjusted by line managers depending on changes in the operating environment. In our experiment, the weight is set from the historical data.

As explained in Section 4, the proposed algorithm uses clustering and classification techniques. We cluster source machines by using the well-known

DBSCAN clustering algorithm, which can provide robust clustering outcomes regardless of the distribution of data points and does not require the number of clusters *a priori*. DBSCAN assigns a cluster ID if there are at least $minPts$ data points within the distance ϵ from one data point p. For more detail on DBSCAN, refer to Ester et al. [35]. The algorithm parameters ϵ and $minPts$ are set to 0.2 and 3, respectively, and the Euclidean distance is used for the distance measure. To train faster and reduce the likelihood of falling into local optimal states, we standardize the average travel time from the machine to each lifter used as the data point. In addition, we individually assign cluster IDs to source machines that are judged as noise points because they do not belong to any cluster. To create the classifier, the *k*-nearest neighbour (*k*-NN) algorithm is used. We set the parameter k of the *k*-NN algorithm to 1 so that the cluster closest to the input machine is assigned. The Euclidean distance is also used as a distance measure in the *k*-NN algorithm. Finally, we derive the optimal solution of the MDP model by using the value iteration algorithm. The discount factor γ of the MDP model is set to 0.99 and the stopping criterion required for the value iteration algorithm is set to 0.001.

The proposed algorithm is implemented in the Java programming language. The computing environment used in the experiment is as follows: Intel Xeon 6146 (3.20GHz), 16 GB RAM, Windows 10 Enterprise. We limit the maximum allowable number of lifters per group to three to make the computational loads manageable.

5.2 Results of the simple experiment

Before testing the proposed approach on an emulator, we analyse its performance with a small-sized testbed. Under the conditions shown in Table 2, an optimal policy for assigning lifters is derived by utilizing the MDP model proposed in Section 3. Table 3

shows two notable cases from the optimal policy. First, since the objective of the MDP model is to minimize the total of the travel and waiting times, the lifter with the smallest value of $TR_j + b_j \times weight$ is selected. For instance, in the case of source machine 4, lifter 2 is selected by comparing the values of $TR_j + b_j \times weight$ for each lifter. (Unlike the shortest-transportation-time rule, the weight is optimally obtained from the model by considering the given system environment variable and future state.)

Table 4: Optimal policy of the simple experiment.

Case 1								
System state			TR_1	TR_2	TR_3	b_1	b_2	b_3
			0	1	2	3	0	3
Source machine ID	1	2	3	4	5	6	7	8
Optimal lifter ID	2	2	2	2	3	2	3	1
Case 2								
System state			TR_1	TR_2	tr_3	b_1	b_2	b_3
			0	0	0	0	0	0
Source machine ID	1	2	3	4	5	6	7	8
Optimal lifter ID	1	1	1	2	3	2	3	1

Secondly, case 2 reveals that the proposed approach can consider future system states when deriving an optimal policy. Assuming that all values comprising the state are zero (i.e., there are no TRs), source machines 1, 2, 3, 4, and 8 should all select lifter 1 because the travel time to it is the shortest. However, source machine 4 selects lifter 2 instead. This is because lifter 1 is frequently called by other source machines (1, 2, 3, and 8). This may generate severe congestion by making the lifter-assignment policy sub-optimal. To avoid such inefficiency, the proposed approach selects another lifter, thereby distributing the traffic volume and balancing the load. Thus, the lifter-assignment policy derived from the proposed approach can consider not only the immediate travel time but also the subsequent scenarios.

5.3 Empirical experimental results

5.3.1 Clustering results

It commonly becomes more difficult to obtain an optimal policy as the problem size increases when using MDP. Therefore, we propose an efficient algorithm to place machines into a manageable number of clusters. Figure 9 shows the clustering results for the actual FAB. The symbol X coloured in black represents the location of the lifter; a circle indicates each source machine's location. Circles of the same colour refer to the same cluster. Forty clusters are formed in the FAB used in the experiment.

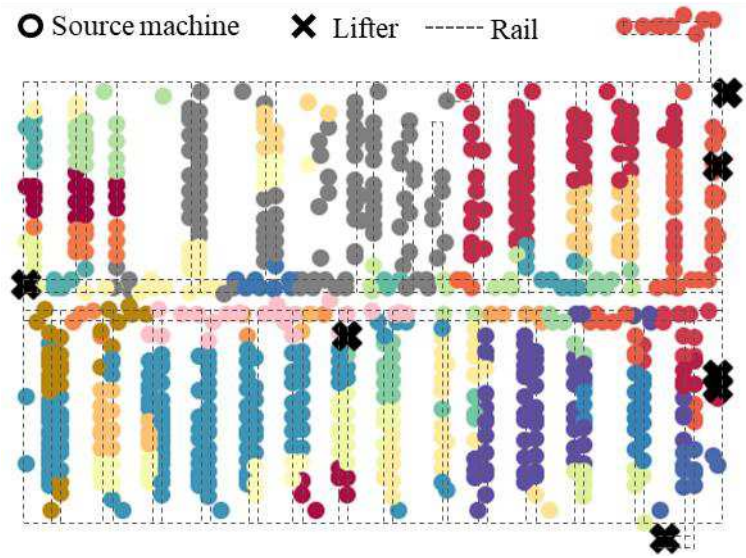

Figure 9: Aggregation result of source machines

The clustering outcomes shown in Figure 9 appear generally reasonable: machines located near a lifter are grouped together because of their relatively short travel times, whereas machines that are located far from the lifters and have long travel times are similarly clustered.

5.3.2 Performance evaluation with emulator

In order to evaluate the performance of the three lifter assignment rules (including the two bench-marking rules), six performance indicators are measured. For TRs sent to lifters, the average travel time, the average waiting time, and the number of completed TRs are recorded. For all TRs, we also measure the average travel time, the number of completed TRs, and the number of TRs with a travel time over three minutes. Recall that the travel time refers to the elapsed time from the unloading of the wafer lots from the source machine to their loading into the buffer port of the lifter. Waiting time is the time from the moment of loading the lot into the buffer port of the lifter to the moment the rack-master picks up a wafer lot (a warm-up period of 30 min is set to reach the steady state of the model).

Table 4 and Figure 10 show the results from the three assignment rules. One unit on the *x*-axis in Figure 10 is 10 min. As shown in both the table and figure, the proposed algorithm outperforms the bench-marking rules in all six measures. First, for TRs sent to lifters, when the proposed algorithm is implemented, the average travel time is reduced by 19.7% (165 s→133 s) over that of the round-robin rule and 7% (143 s →133 s) over that of the shortest-transportation-time rule. The average waiting time of the proposed algorithm is longer than that of the shortest-transportation-time rule, but only very slightly. The sum of the average travel time and the average waiting time is the shortest when the proposed algorithm is applied. It is worth pointing out that the proposed algorithm aims to find an optimal lifter-assignment policy that minimizes the sum of the average travel time and the waiting time, which is why it is superior to the bench-marking rules. As it considers both the immediate cost and the cost of the future system states, it provides solid policies that more effectively disperse traffic congestion.

Another notable finding, which may be observed in the graphs in Figure 10, is that the proposed algorithm also yields the shortest travel time while covering the most TRs in the time window we analysed. We thereby infer that the proposed algorithm performs stably compared to the bench-marking rules. This may have been on account of the fact that it can consider future system states and thus avoids undesirable situations *a priori*.

For all TRs, the proposed algorithm is superior to the bench-marking rules. The average travel time from the source machine to the destination is reduced by 2.3% (137.7 s →134.5 s) compared to the round-robin and 0.7% (135.6 s →134.5 s) compared to the shortest-transportation-time rule. This is empirically important in that the decision on TRs sent to the lifters does not affect other TRs negatively. It is also worth noting that the proposed algorithm has fewer TRs with a travel time over 3 min, even though it handles slightly more TRs than did the other rules. These results indicate that the proposed algorithm prevents delivery delays and uses OHTs effectively.

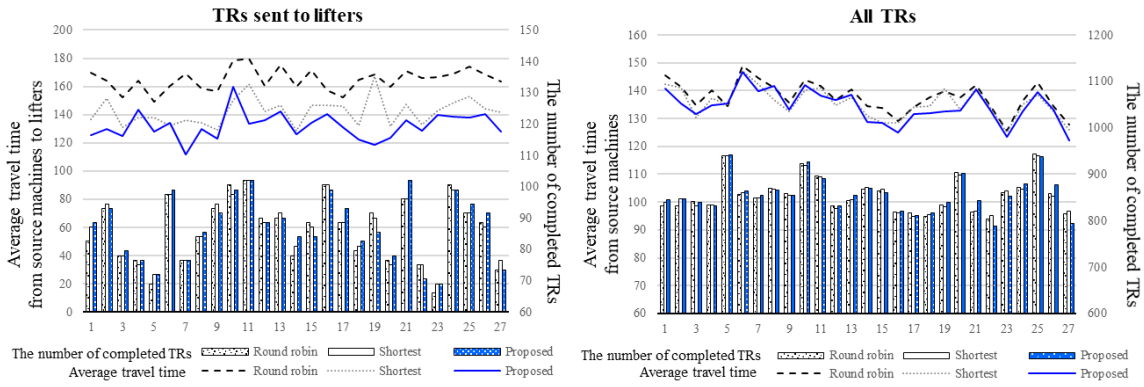

Figure 10: Results from the three assignment rules by time window

Table 5: Performance of the three lifter assignment rules

TRs sent to lifters			
Lifter assignment rule	Round robin	Shortest transportation time	Proposed algorithm
Number of completed TRs	2,327	2,336	2,348
Average travel time (sec)	165.2	142.7	132.6

Average waiting time (sec)	15.6	29.6	18.5
All TRs			
Lifter assignment rule	Round robin	Shortest transportation time	Proposed algorithm
Number of completed TRs	23,093	23,090	23,129
Average travel time (sec)	137.7	135.6	134.5
Number of TRs with travel time over three minutes	6,482	6,113	6,044

The economic implications of the above results are as follows. When the proposed algorithm is implemented in an actual FAB, the travel time of OHTs will be decreased. Therefore, fewer OHTs will be required than in the previous scenario. Since the load factor of an OHT is a function of its travel time, we can calculate the OHT-investment cost savings. Based on the simulation results that showed a 2.3% reduction in travel time to process all TRs, we estimate that approximately 2.5 million USD (2.76 billion KRW) would be saved if the proposed algorithms are implemented in actual FABs operating approximately 3,000 OHTs.

5.3.3 Sensitivity Analysis

To check the robustness of the proposed algorithm, we conduct a sensitivity analysis to help predict its performance under unusual situations in which the load factors of OHTs and lifters increase in accordance with TRs. To examine such situations, we arbitrarily increase the arrival rate of the next TR to be 1.05 times, 1.1 times, 1.15 times, or 1.2 times higher than the historical data.

The results of the sensitivity analysis are shown in Table 5. Of the six measures, we illustrate as representative the average travel time, which is regarded as the most important measure in practice. Table 5 shows that, if the arrival rate of the next TR increases, the travel time to lifters also increases regardless of the type of assignment rule. Nonetheless, the proposed algorithm has the shortest travel time, even for the highest arrival rates we consider. To confirm this statistically, we conduct a *t*-test to

examine the difference among the results. Based on the *t*-tests with a *p*-value <0.05, we conclude that the improvement in performance owing to the proposed algorithm is statistically significant.

Next, we test the performance of the proposed method by using out-of-sample data (specifically, a different dataset). As shown in Table 6, the travel times are increased for all test instances when we use a different dataset. This is expected because the dataset used for deriving the optimal policy is different from that used for testing the performance. Nevertheless, it is worth noting that there is no statistically significant difference in three out of the five test instances. This suggests that, even when the operating environment is altered by a small amount, the operating policy derived from the proposed approach will continue to perform adequately. This means that real-time updating of the lifter assignment policy, which requires tremendous computational resources, is not necessary. Hence, our approach of periodically updating the lifter assignment policy is valid in practice.

Table 6: Sensitivity analysis results for arrival rate of TRs (*p*-value in parenthesis)

Lifter assignment rule	Change of arrival rate				
	× 1.00	× 1.05	× 1.10	× 1.15	× 1.20
Proposed	132.5	136.6	139.4	143.1	146.1
Shortest transportation time	142.7 (<0.05)	147.5 (<0.05)	150.7 (<0.05)	158.7 (<0.05)	159.3 (<0.05)
Round robin	165.2 (<0.05)	168.2 (<0.05)	172.5 (<0.05)	174.7 (<0.05)	178.0 (<0.05)

Table 6: Sensitivity analysis results using out-of-sample data (p-value in parenthesis)

Lifter assignment rule	Change of arrival rate				
	× 1.00	× 1.05	× 1.10	× 1.15	× 1.20
Proposed	132.5	136.6	139.4	143.1	146.1
Proposed (out-of-sample data)	139.6 (<0.05)	138.8 (0.2)	144.2 (<0.05)	146.5 (0.1)	149.7 (0.1)

This series of experiments, both idealized and realistic, verified the effectiveness of the proposed approach. In particular, there is a significant improvement over the

round-robin rule, which has no mechanism to consider travel time when TRs need transportation to a lifter. Moreover, our approach outperforms the fastest transfer option that does not consider a long-term perspective, showing the importance of accounting for a system's future state when assigning TRs to lifters.

6. Conclusion

This study addresses the problem of assigning TRs to lifters in semiconductor manufacturing. As the production process becomes increasingly complex, more capacity is required to cope with it. The use of multiple floor lines provides a means to increase capacity without construction of another FAB line; however, it requires meticulous control of OHTs to manage the traffic. In this study, we solve a lifter-assignment problem that allocates TRs to an appropriate lifter via MDP. Unlike the rules used in the actual FAB lines, our model can provide an optimal lifter-assignment policy by considering the system's future states. We also propose an algorithm using a well-known clustering method to efficiently reduce the problem size. To the best of our knowledge, neither such a future-oriented formulation nor such an efficient solution approach have been introduced in the previous literature.

The effectiveness of our approach relative to two bench-marking rules is demonstrated with a simulation incorporating data from the operation of an actual FAB. The proposed algorithm reduces the travel time significantly compared with the bench-marking rules. Sensitivity analysis also confirms the robustness of the proposed algorithm. In the context of semiconductor manufacturing, our approach is expected to provide major economic advantages over the status quo if implemented in an actual FAB. For example, our approach should reduce the travel time of all TRs by 2.3%. As a result, approximately 2.5 million USD (2.76 billion KRW) would be saved.

Moreover, our study elucidates the possibility of autonomous control in assigning lifters. Since our model is based on an optimization scheme that uses the system information as an input variable, it can easily consider the changes in FAB lines and automatically update the lifter assignment polices. Considering that the current rules require manual adjustments - varying weights on the basis of the system's status - our approach may reduce the workload of operators in semiconductor manufacturing.

Declarations

Funding

This research was partially supported by the National Research Foundation of Korea(NRF) grant funded by the Ministry of Science and ICT (No.NRF-2019R1F1A1063365)

Conflicts of interest/Competing interests

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Availability of data and material

Not applicable. (The authors cannot disclose data because of security issues.)

Code availability

Not applicable.

Ethical Approval

Not applicable.

Consent to participate

Not applicable.

Consent to publications

Not applicable.

References

[1] Na B, Woo JE, Lee J (2016) Lifter assignment problem for inter-line transfers in semiconductor manufacturing facilities. *International Journal of Advanced Manufacturing Technology* 86**:**1615-26. https://doi.org/10.1007/s00170-015-8327-0

[2] Kim J, Yu G, Jang YJ (2016) Semiconductor FAB layout design analysis with 300-mm FAB Data: "Is minimum distance-based layout design best for semiconductor FAB design?". *Computers and Industrial Engineering* 99:330-346. https://doi.org/10.1016/j.cie.2016.02.012

[3] Agrawal GK, Heragu SS (2006) A survey of automated material handling systems in 300-mm Semiconductor FABs. *IEEE Transactions on Semiconductor Manufacturing* 19:112-20. https://doi.org/10.1109/TSM.2005.863217

[4] Le-Anh T, De Koster MBM (2006) A review of design and control of automated guided vehicle systems. *European Journal of Operational Research* 171:1-23. https://doi.org/10.1016/j.ejor.2005.01.036

[5] Qiu L, Hsu W, Huang S, Wang H (2002) Scheduling and routing algorithms for AGVs: a survey. *International Journal of Production Research* 40:745-60. https://doi.org/10.1080/00207540110091712

[6] Xie C, Allen TT (2015) Simulation and experimental design methods for job shop scheduling with material handling: a survey. *International Journal of Advanced Manufacturing Technology* 80:233-43. https://doi.org/10.1007/s00170-015-6981-x

[7] Huang CJ, Chang KH, Lin JT (2012) Optimal vehicle allocation for an automated materials handling system using simulation optimization. *International Journal of Production Research* 50:5734-46. https://doi.org/10.1080/00207543.2011.622311

[8] Liao DY, Fu HS (2004) Speed delivery: Dynamic OHT allocation and dispatching in large-scale 300-mm AMHS management. *IEEE Robotics and Automation Magazine* 22–32. https://doi.org/10.1109/MRA.2004.1337824

[9] Lin JT, Wu CH, Huang CW (2013) Dynamic vehicle allocation control for automated material handling system in semiconductor manufacturing. *Computers and Operations Research* 40:2329-39. https://doi.org/10.1016/j.cor.2013.04.007

[10] Wang FK, Lin JT (2004) Performance evaluation of an automated material handling system for a wafer fab. *Robotics and Computer-Integrated Manufacturing* 20:91–100. https://doi.org/10.1016/j.rcim.2003.08.002

[11] Kim BI, Park J (2009) Idle vehicle circulation policies in a semiconductor FAB. *Journal of Intelligent Manufacturing* 20:709. https://doi.org/10.1007/s10845-008-0159-4

[12] Vahdani B (2014) Vehicle positioning in cell manufacturing systems via robust optimization. *Applied Soft Computing* 24:78-85. https://doi.org/10.1016/j.asoc.2014.07.001

[13] Lee S, Lim DE, Kang Y, Kim HJ (2021) Clustered multi-task sequence-to-sequence learning for autonomous vehicle repositioning. *IEEE Access* 9: 14504-15. https://doi.org/10.1109/ACCESS.2021.3051763

[14] Schmaler R, Schmidt T, Schoeps M, Luebke J, Hupfer R, Schlaus N (2017) Simulation based evaluation of different empty vehicle management strategies with considering future transport jobs. In *Proceedings of the 2017 Winter Simulation Conference* (WSC), 294. 1-12. https://doi.org/10.1109/WSC.2017.8248071

[15] de Koster RBM, Le-Ahn T, Van der Meer JR (2004) Testing and classifying vehicle dispatching rules in three real-world settings. *Journal of Operations Management* 22:369-86. https://doi.org/10.1016/j.jom.2004.05.006

[16] Egbelu PJ, Tanchoco JMA (1984) Characterization of automatic guided vehicle dispatching rules. *International Journal of Production Research* 22:359-74. https://doi.org/10.1080/00207548408942459

[17] Im K, Kim K, Lee S (2009) Effective vehicle dispatching method minimizing the blocking and delivery times in automatic material handling systems of 300 mm semiconductor fabrication. *International Journal of Production Research* 47:3997-4011. https://doi.org/10.1080/00207540801914934

[18] Kuo CH, Huang CS. 2006. Dispatching of overhead hoist vehicles in a fab intrabay using a multimission-oriented controller. *International Journal of Advanced Manufacturing Technology* 27:824-32. Dispatching of overhead hoist vehicles in a fab intrabay using a multimission-oriented controller

[19] Bozer YA, Yen C (1996) Intelligent dispatching rules for trip-based material handling systems. *Journal of Manufacturing Systems* 15(4):226-239. https://doi.org/10.1016/0278-6125(96)84549-3

[20] Kim BI, Oh S, Shin J, Jun M (2007) Effectiveness of vehicle reassignment in a large-scale overhead hoist transport system. *International Journal of Production Research* 45:789-802. https://doi.org/10.1080/00207540600675819

[21] Le-Ahn T, De Koster MBM (2007) On-line dispatching rules for vehicle-based internal transport systems. *International Journal of Production Research* 43:1711-28. https://doi.org/10.1080/00207540412331320481

[22] Chen HK, Hsueh CF, Chang MS (2006) The real-time time-dependent vehicle routing problem. *Transportation Research Part E: Logistics and Transportation* 42:383-408. https://doi.org/10.1016/j.tre.2005.01.003

[23] Kim CW, Tanchoco JMA (1991) Conflict-free shortest-time bidirectional AGV routing. *International Journal of Production Research* 29:2377-2391. https://doi.org/10.1080/00207549108948090

[24] Smolic-Rocak N, Bogdan S, Kovacic Z, Petrovic T(2010) Time Windows Based Dynamic Routing in Multi-AGV Systems. *IEEE Transactions on Automation Science* 7:151-155. https://doi.org/10.1109/TASE.2009.2016350

[25] Taghaboni-Dutta F, Tanchoco JMA (2007) Comparison of dynamic routeing techniques for automated guided vehicle system. *International Journal of Production Research* 33:2653-69. https://doi.org/10.1080/00207549508945352

[26] Bartlett K, Lee J, Ahmed S, Nemhauser G, Sokol J, Na B (2014) Congestion-aware dynamic routing in automated material handling systems. *Computers and Industrial Engineering* 70:176-182. https://doi.org/10.1016/j.cie.2014.02.002

[27] Lau H, Woo SO (2008) An agent-based dynamic routing strategy for automated material handling systems. *International Journal of Computer Integrated Manufacturing* 21:269-288. https://doi.org/10.1080/09511920701241624

[28] Kim H, Lim DE, Lee S (2020) Deep learning-based dynamic scheduling for semiconductor manufacturing with high uncertainty of automated material handling system capability. *IEEE Transactions on Semiconductor Manufacturing* 33:13-22. https://doi.org/10.1109/TSM.2020.2965293

[29] Siebert M, Bartlett K, Kim H, Ahmed S, Lee J, Nazzal D, Nemhauser G, Sokol J (2018) Lot targeting and lot dispatching decision policies for semiconductor manufacturing: optimization under uncertainty with simulation validation. *International Journal of Production Research* 56:629-41. https://doi.org/10.1080/00207543.2017.1387679

[30] Lee S, Kim Y, Kahng H, Lee SK, Chung S, Cheong T, Shin K, Park J, Kim SB (2020) Intelligent traffic control for autonomous vehicle systems based on machine learning. *Expert Systems with Applications* 144:113074. https://doi.org/10.1016/j.eswa.2019.113074

[31] Jimenez J, Kim B, Fowler J, Mackulak G, Choung YI (2002) Operational modeling and simulation of an inter-bay AMHS in semiconductor wafer fabrication. In *Proceedings of the Winter Simulation Conference*, 2, 1377-1382. https://doi.org/10.1109/WSC.2002.1166405

[32] Lee J, Kim S, Na B (2018) Optimization of lifter operations for inter-line transfers in semiconductor manufacturing facilities. *International Journal of Mechanical and Production Engineering Research and Development* 8:167-178. DOI: 10.24247/ijmperdaug201819

[33] Puterman ML (2014) Markov Decision Processes: discrete stochastic dynamic programming. John Wiley & Sons. DOI: 10.1002/9780470316887

[34] Ross SM (1996) Stochastic processes. Wiley, New York

[35] Ester M, Kriegel HP, Sander J, Xu X (1996) A density-based algorithm for discovering clusters in large spatial datasets with noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining* 94:226-231

Figures

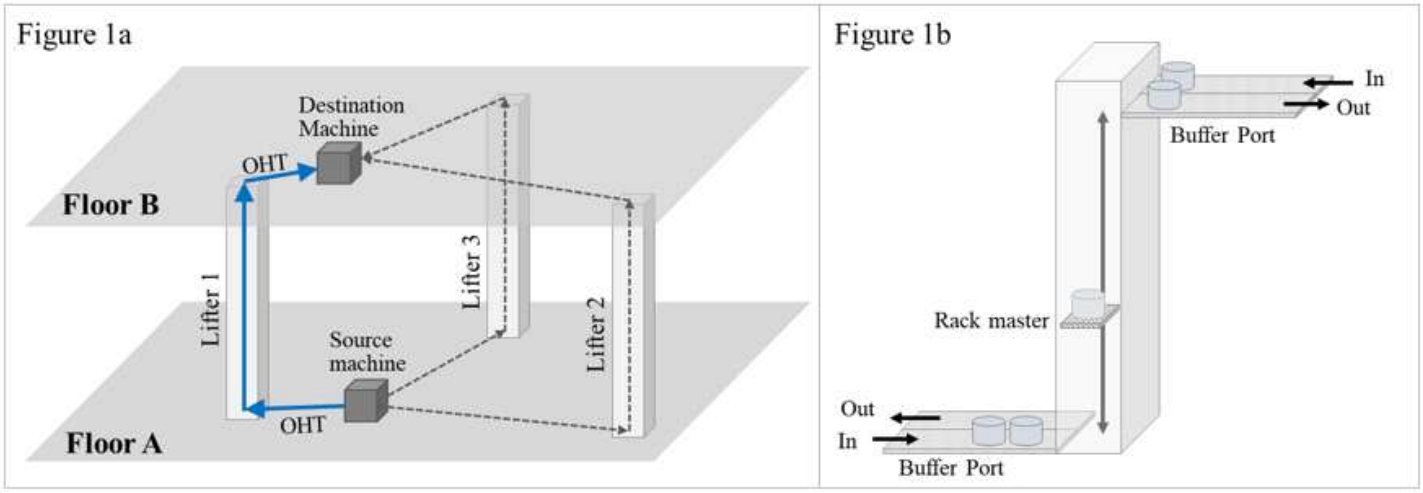

Figure 1

A typical process of the inter-line transportation in semiconductor FAB (Figure 1a); The physical structure of a lifter (Figure 1b)

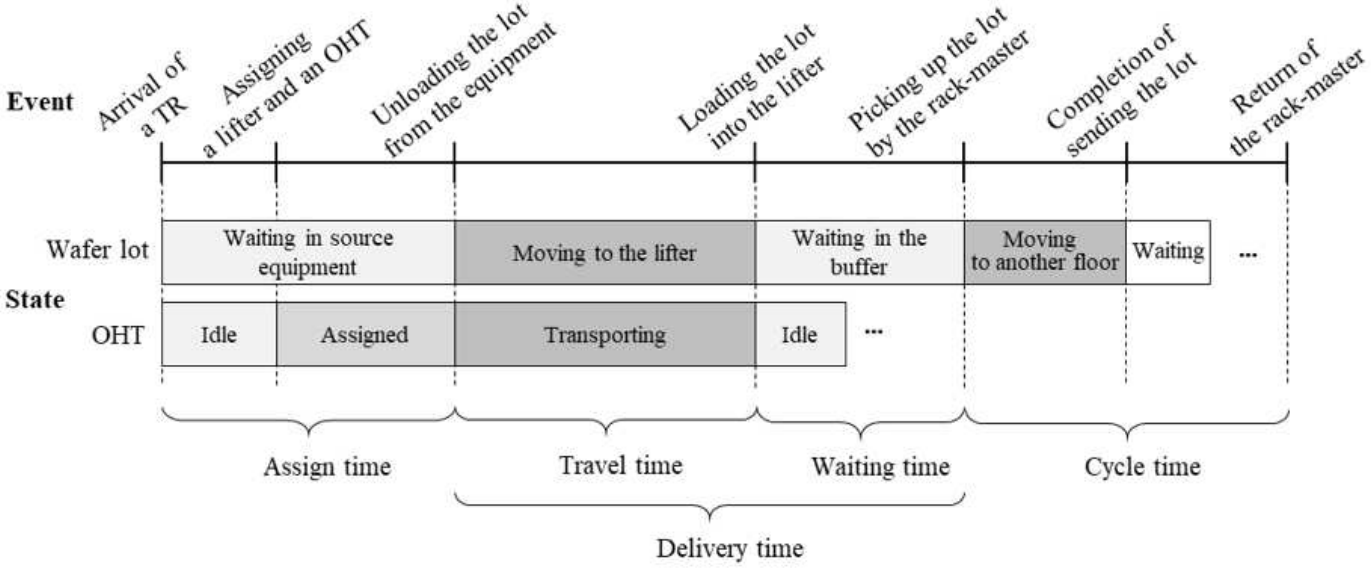

Figure 2

Event flow and time definition of inter-line transportation

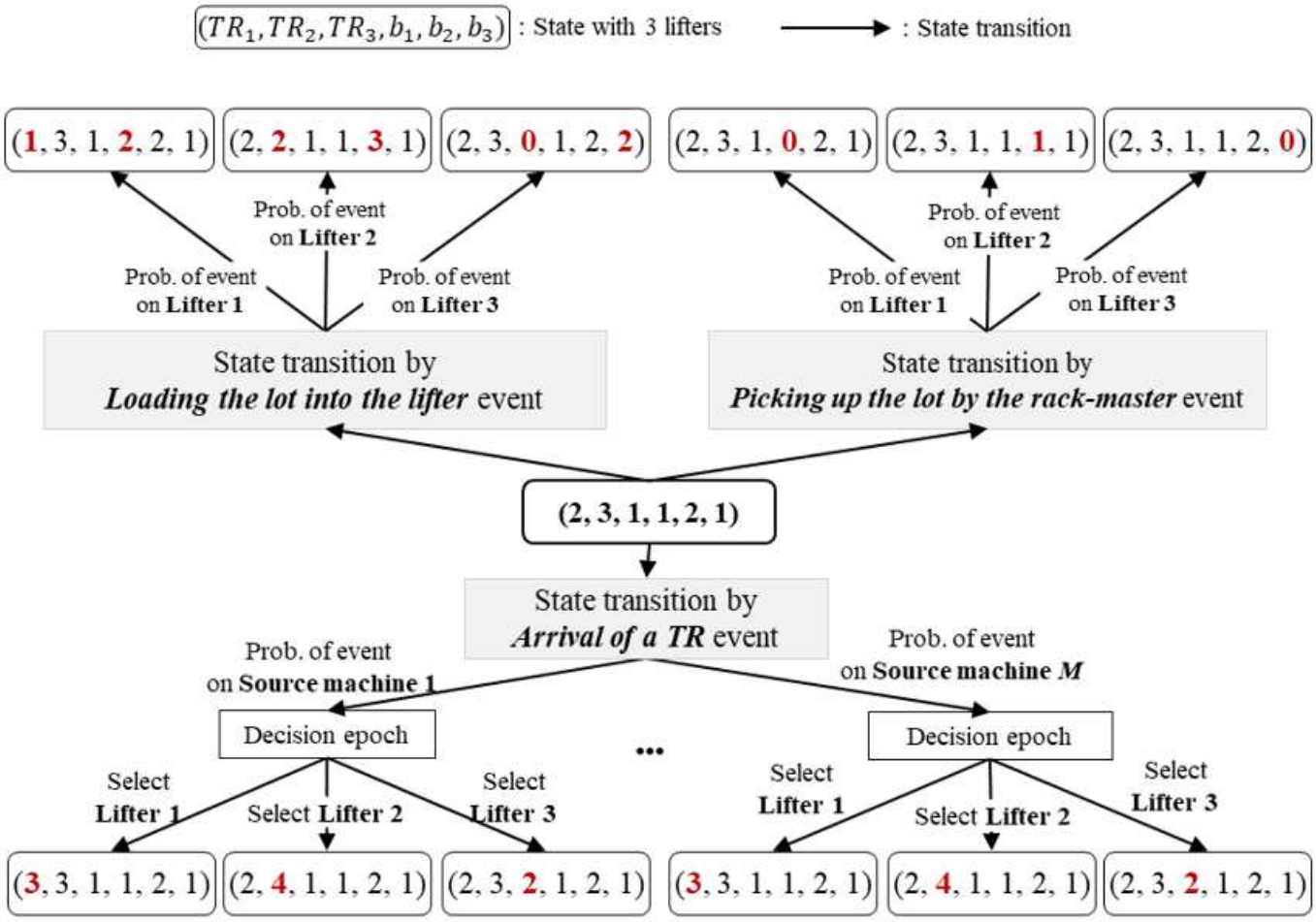

Figure 3

Example of state-transition of the system with three lifters

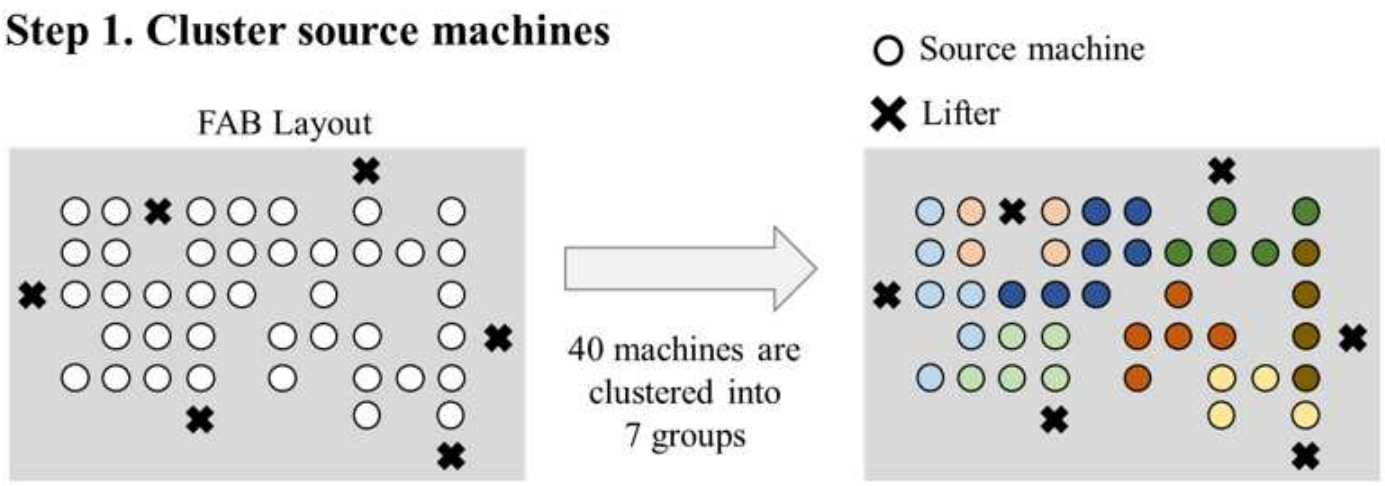

Figure 4

Example of step 1 of the solution approach: cluster source machines

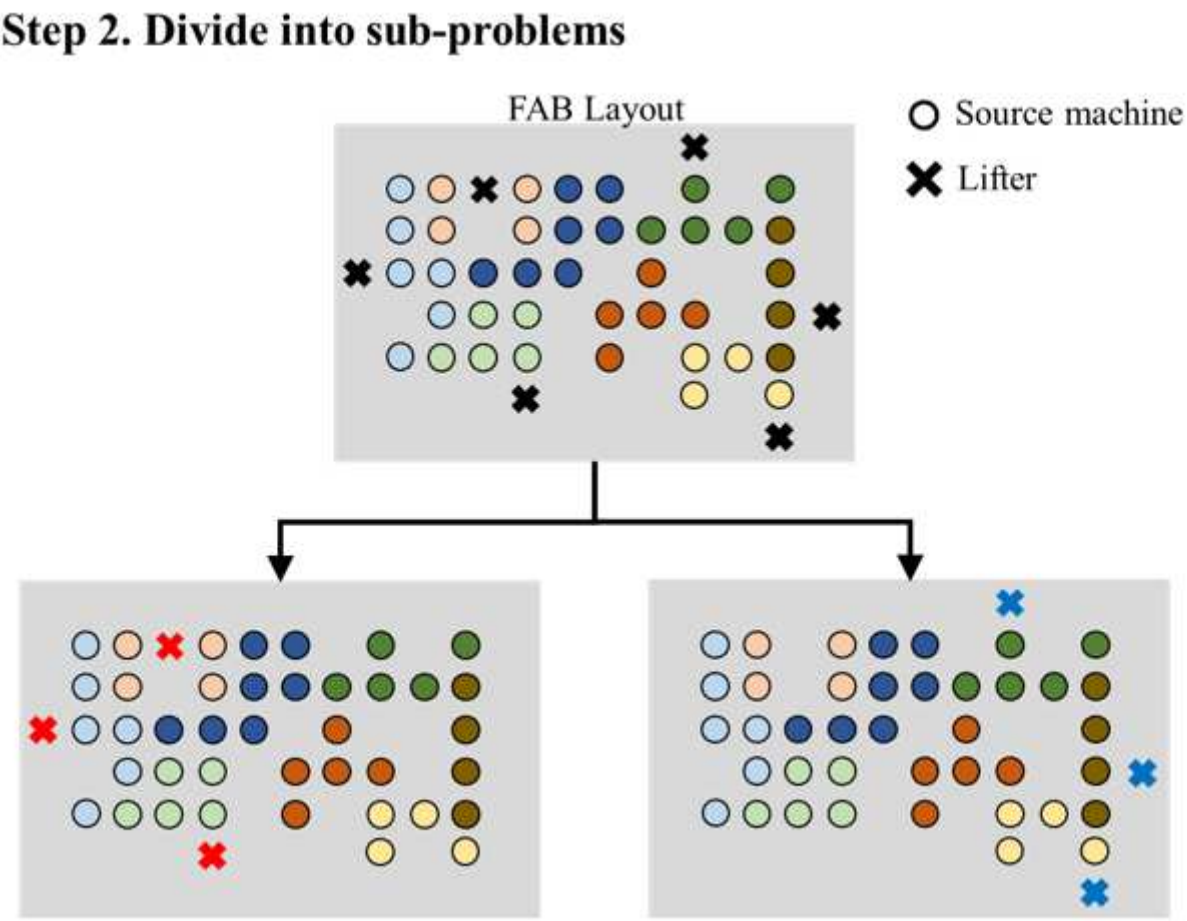

Figure 5

Example of step 2 of the solution approach: division into sub-problems

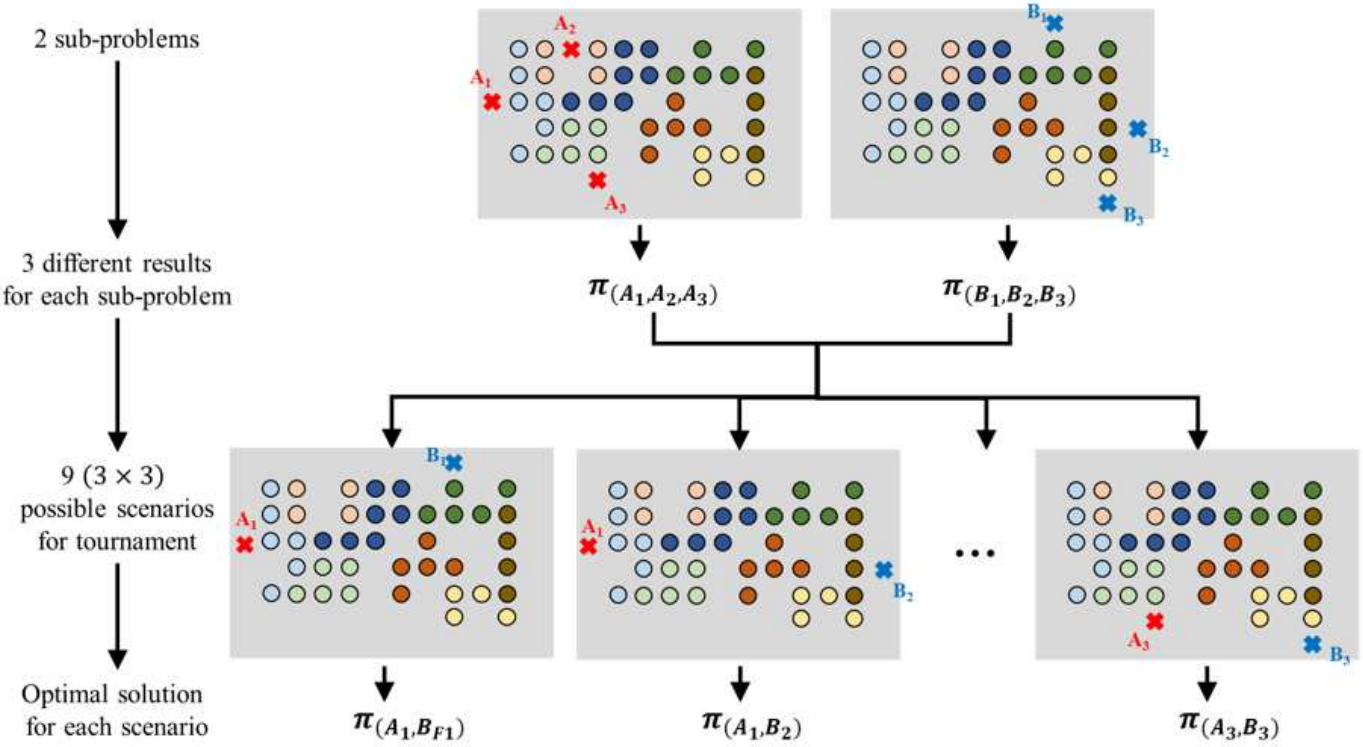

Figure 6

Example of step 3 of the solution approach: preparation of the optimal policies for all possible scenarios a priori

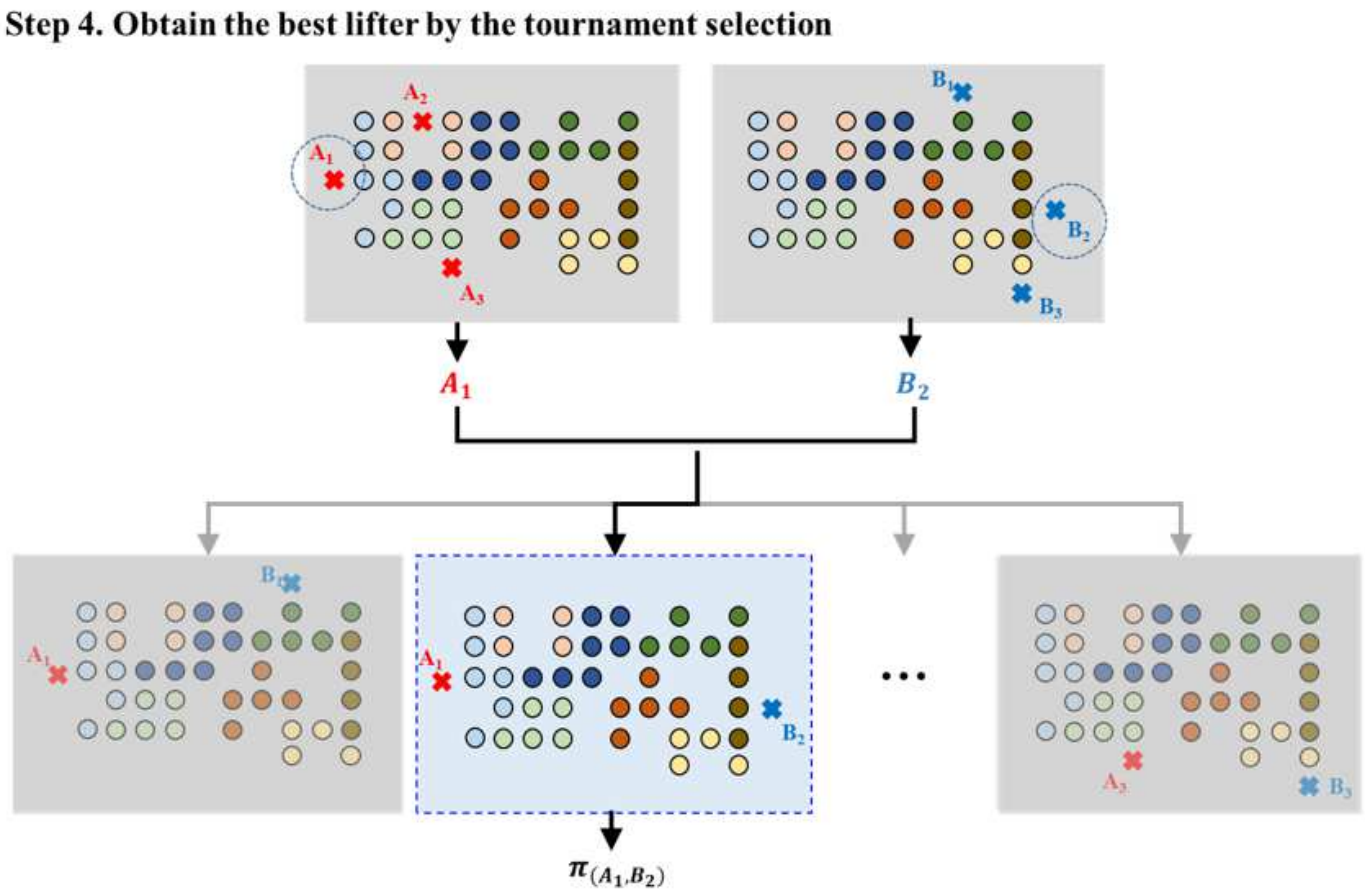

Figure 7

Example of step 4: Obtaining the best lifter by the tournament selection

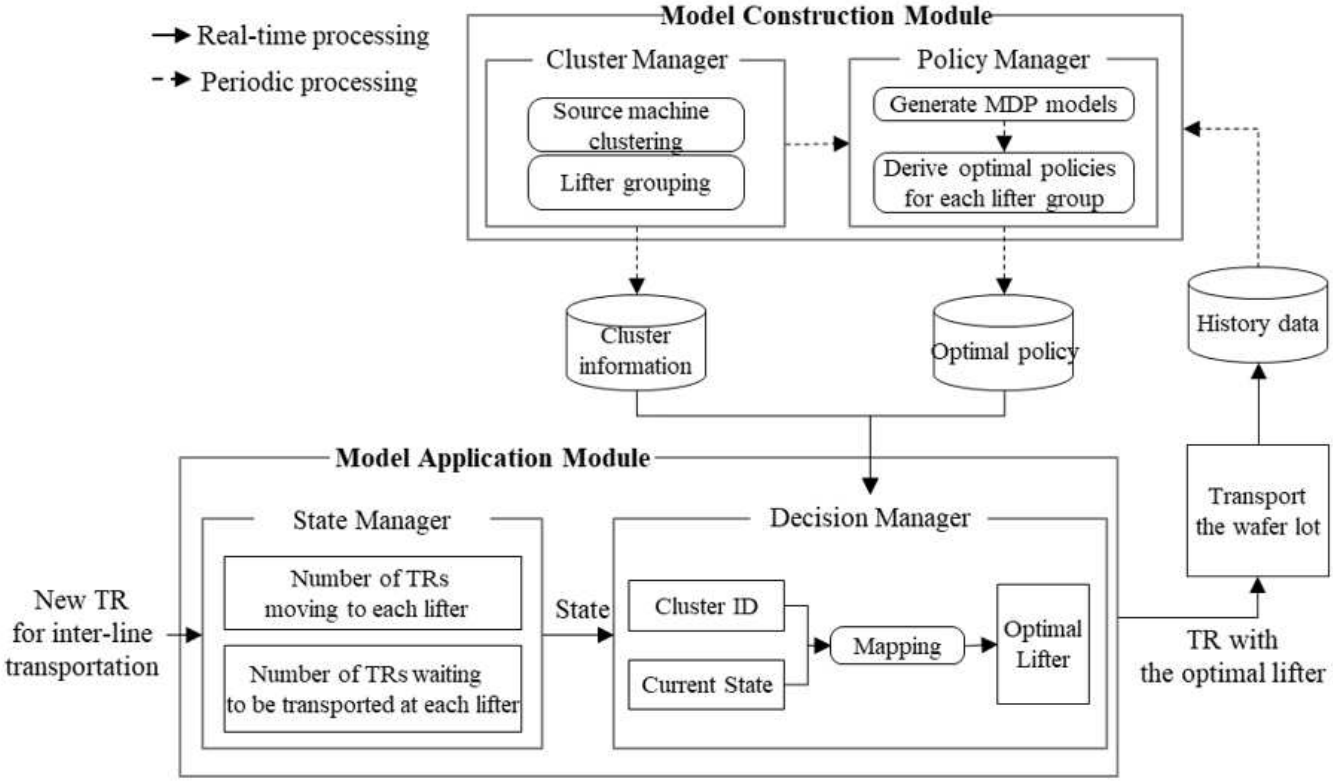

Figure 8

System implementation framework for lifter assignment in real FABs

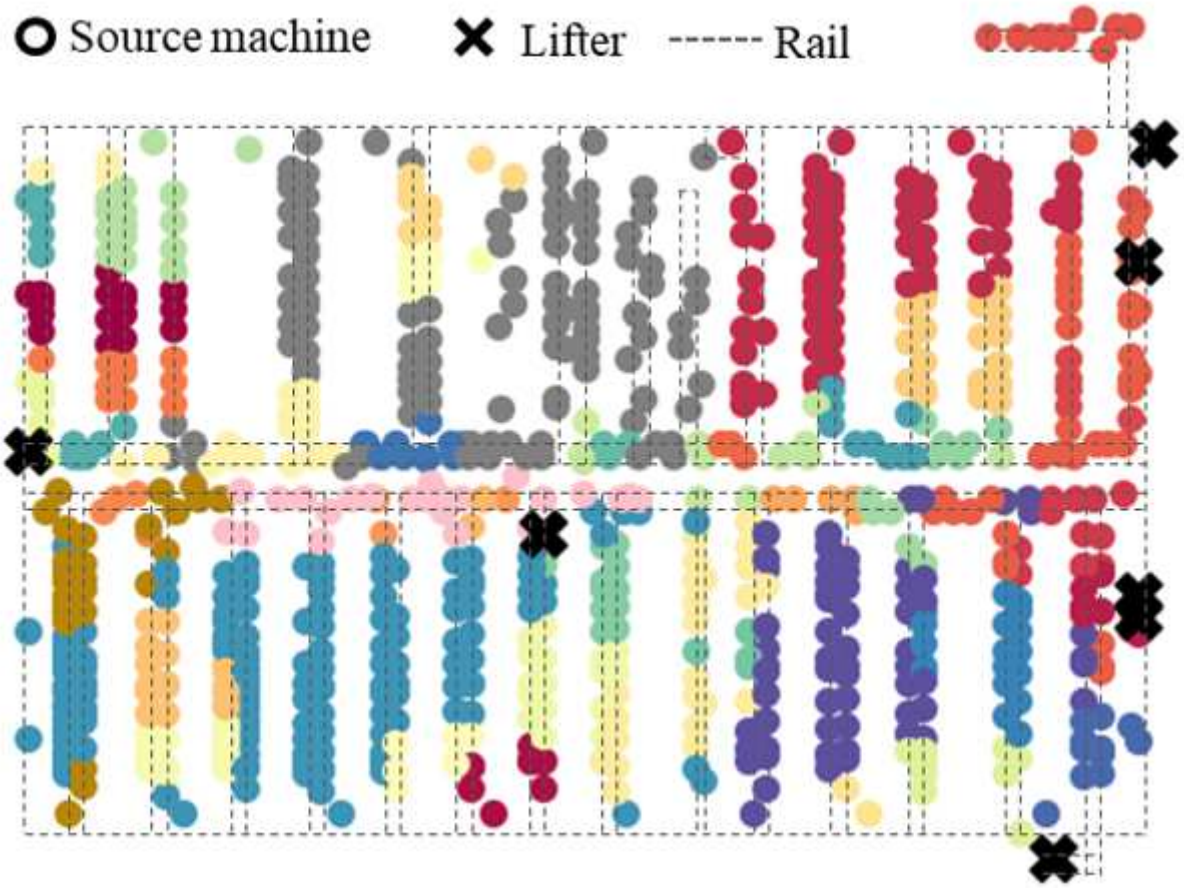

Figure 9

Aggregation result of source machines

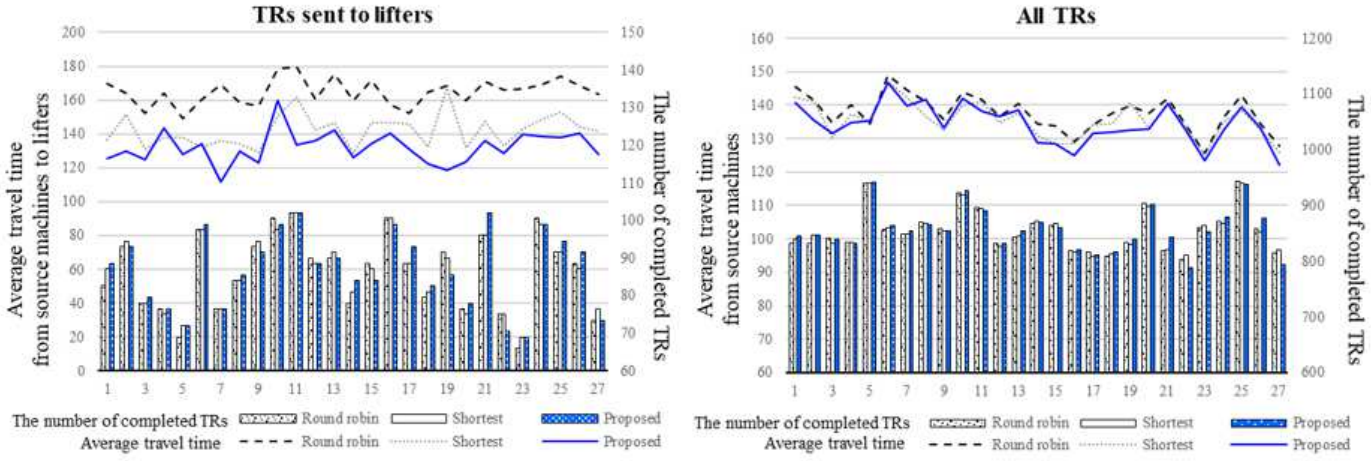

Figure 10

Results from the three assignment rules by time window