Heuristics for selecting machines and determining buffer capacities in assembly systems

K.-C. Jeong\textsuperscript{a}, Y.-D. Kim\textsuperscript{b,\textasteriskcentered}

\textsuperscript{a}Department of Management Information, Cheju National University, Cheju-City, Cheju-Do 690-756, South Korea
\textsuperscript{b}Department of Industrial Engineering, Korea Advanced Institute of Science and Technology, Yusong-gu, Daejon 305-701, South Korea

Accepted 15 May 2000

Abstract

This paper considers a design problem of assembly systems consisting of stations and buffers that connect the stations. We present a solution procedure for finding the minimum cost configuration which gives a desired throughput rate for an assembly system. The configuration is defined by the machines to be used in stations (machine configuration) and capacities of buffers. Three heuristics are proposed, which simultaneously select the machines to be used in the stations and determine the capacities of the buffers. These heuristics start from an initial configuration such as a lower configuration, which consists of less efficient machines and large size buffers, or an upper configuration, which consists of more efficient machines and small size buffers. Then, the heuristics search for a near optimal solution by repeatedly generating promising machine configurations and determining (near) optimal buffer sizes for the machine configurations. Results of computational experiments show that the proposed heuristics give relatively good configurations in a reasonable amount of time. © 2000 Elsevier Science Ltd. All rights reserved.

Keywords: Heuristics; Assembly systems; Design

1. Introduction

We consider a design problem for assembly systems. The assembly system is defined as a manufacturing system in which some stations perform assembly operations, by which two or more parts or subassemblies are brought together to form a single unit. The design problem considered in this paper is the problem of selecting machines to be used in the stations from a set of candidate machines for the stations and determining capacities of buffers in such a way that a desired throughput rate of the assembly system can be achieved with the minimum cost.
Most of previous research on design of assembly systems deal with the problems of buffer allocation (determining capacities of buffers) assuming that decisions for selecting machines are already made, that is, parameters to characterize the machines are given. Such research on the buffer allocation problems can be classified into the following three categories according to the objective used: maximization of throughput rate, maximization of profits, and minimization of buffer capacity. Ho, Eyler, and Chien (1979), Jafari and Shanthikumar (1989), and Seong, Chang, and Hong (1995) propose algorithms based on a gradient search, dynamic programming, and multi-dimensional nonlinear search, respectively, for determining capacities of individual buffers to obtain the maximum throughput rate for a given total available capacity of buffers. For the same objective, Soyster et al. (1979) established upper and lower bounds for the throughput rate and formulate concave separable programs to determine buffer capacities. For balanced transfer lines, Hillier, So, and Boling (1993) and Powell (1994) describe the storage-bowl phenomenon, i.e. capacities of interior buffers are greater than or equal to those of end buffers in an optimal buffer allocation. For buffer allocation problems with the objective of maximizing the total revenue (considering selling price, production cost, and inventory holding cost), Altiok and Stidham (1983) and Smith and Daskalaki (1988) use the Hook and Jeeves method and the Powell method, respectively, which are search methods used for unconstrained optimization problems. On the other hand, for the objective of minimizing the total capacity of buffers required for a given desired throughput rate, Park (1993) proposes a beam search algorithm based on buffer utilization.

In other research, several algorithms have been devised for problems in which decisions for buffers and machines and/or pallets are considered at the same time, such as the algorithms of Liu and Sanders (1986), Dallery and Frein (1988), Johri (1993), and Martin (1994). Liu and Sanders deal with the problem of determining the number of pallets and capacities of buffers for flexible assembly systems, Dallery and Frein consider the problem of determining the number of identical machines in each station and the number of pallets in a flexible manufacturing system, and Johri solves the problem of determining the number of identical machines and buffer capacities in stations in a flexible assembly line, while Martin considers the problem of determining the length of the production line and capacities of buffers for the objective of maximizing total profit. On the other hand, Daganzo and Blumenfeld (1994) deal with a design problem of determining the number of parallel stations in assembly systems in which certain workstations are arranged in parallel.

Solution of such design problems requires a tool for performance evaluation. That is, to solve the design problems, the performance (throughput rate or average inventory levels) of the assembly systems should be estimated to see whether a certain configuration shows a desired performance or not. Various methods have been proposed for performance evaluation of assembly systems. Duda and Czachorski (1987), Baccelli, Massey, and Towsley (1989), and Liu and Perros (1991) use queuing theories for the analysis of the performance of assembly systems. These articles focus on computer systems with reliable processors (machines) and infinite buffer capacities. On the other hand, Gershwin (1991), Mascolo, David, and Dallery (1991), and Jeong and Kim (1998) propose computational algorithms based on decomposition methods assuming that machines are unreliable and buffers have finite capacities. They make different assumptions on the times between failures, the times to repair, and the processing times. Gershwin assumes that the up-time and down-time follow geometric distributions while Mascolo et al. assume that they are exponentially distributed. The algorithm proposed by Gershwin can deal with homogeneous systems in which processing times are equal and constant for all machines. On the
contrary, Mascolo et al. propose an algorithm that can deal with a non-homogeneous system with different but constant processing times by transforming the system into a homogeneous system in such a way that the performances of the two systems are close to each other. Unlike Gershwin and Mascolo et al., Jeong and Kim assume that the processing times as well as the times between failures and the times to repair are exponentially distributed and present an algorithm for evaluating the performance of such a non-homogeneous system. See the works of Dallery and Gershwin (1992) and Gershwin (1994) for comprehensive reviews of various methods for estimating the performance of flow line systems.

In this paper, we present heuristics for finding a minimum cost configuration of machines and buffers that gives a desired system throughput using the method of Jeong and Kim (1998) for performance evaluation of a configuration. For each station in an assembly system, there may be several machines that can perform the required operations but have different performance characteristics such as processing rates, failure rates, and repair rates. Since throughput rate of the assembly system is affected by the capacities of buffers as well as the machines to be used in the stations, a desired throughput can be obtained from several different configurations. Here, a configuration is defined by the machines to be used in the stations and the capacities of buffers. For example, a desired throughput can be achieved either by configurations that consist of more efficient machines and small size buffers or by configurations that consist of less efficient machines and large size buffers.

Unlike previous studies on the design of assembly systems, in which only numbers of machines and/or buffers are determined, this paper considers the case in which different types of machines are available for each station and gives a method to select a machine type as well as the numbers of machines for the selected machine types. Note that such a case cannot be handled by existing methods. Therefore, the method given in this paper can be considered more general and more practical than those suggested in the previous studies.

This paper is organized as follows. First, we describe the assembly system to be considered in this paper and define the design problem for the assembly system in the next section. Section 3 presents efficient heuristics for the design problem. To show the performance of the heuristics, computational tests are done on a number of assembly systems with various structures and the test results are reported in Section 4, which is followed by conclusions in Section 5.

2. The design problem of assembly systems

Assembly systems consist of stations and buffers that connect the stations. A station consists of a single machine that performs operations assigned to the station. This paper focuses on tree-structured assembly systems in which machines in the stations are unreliable and sizes of the buffers are finite. In tree-structured assembly systems, each buffer connects exactly two stations (an upstream station and a downstream station) and any two stations are connected by exactly one sequence of stations and buffers. Note that there are $N - 1$ buffers in a tree-structured assembly system with $N$ stations, as illustrated in Fig. 1. A machine in a station with two or more upstream buffers performs assembly operations. To start processing an operation, it takes one part from each of its upstream buffers.

A station (or more exactly, a machine in a station) is blocked if a downstream buffer of the station is full when it finishes processing a unit. Similarly, a station is starved if any one of its upstream buffers is empty when it discharges a processed unit. A station is blocked or starved mainly due to failures of its downstream or upstream stations, but it may also be so even when machines are operational, due to the
variability of processing times at downstream or upstream stations. In-process inventories may smooth and balance work flow in the assembly system by reducing these blocking and starvation phenomena. It is assumed that input stations (stations with no upstream buffers) are never starved and output stations (stations with no downstream buffers) are never blocked. Also, it is assumed that a machine does not fail when it is blocked or starved, but it can fail only when it is processing a unit. In most cases, one cannot be practically aware of machine breakdowns when the machine is idle but breakdowns can be identified only when the machine is being operated or when processing a unit. Since repair can be started only when breakdowns are identified, this assumption can be considered reasonable in effect. When a machine fails, the unit being processed remains at that machine until the machine is repaired, then the machine resumes processing the same unit.

The goal considered in this study for the design of the assembly system is to find an optimal system configuration. A configuration is defined by the machines to be used in the stations and the capacities of the buffers. The former defines a machine configuration and the latter defines a buffer configuration. Each station is characterized by the machine to be used in the station and each buffer is characterized by the storage capacity. Therefore, the cost and performance of the assembly system is a function of the machines to be used in stations and the capacities of buffers in the system. The optimal configuration is the one that requires the minimum cost for machines and buffers among all configurations, which give a desired throughput rate.

To describe the design problem more clearly, we first give the notations used in this paper.

\( N \): the number of stations (the number of buffers is \( N - 1 \)).

\( E_{\text{min}} \): desired system throughput rate.

\( \mathbf{a} \): vector \((a_1, a_2, \ldots, a_N)\), where \( a_i \) denotes (the index of) the machine to be used in station \( i \).

\( \mathbf{b} \): vector \((b_1, b_2, \ldots, b_{N-1})\), where \( b_j \) denotes capacity of buffer \( j \).

\((\mathbf{a}, \mathbf{b})\): configuration specified by machine configuration \( \mathbf{a} \) and buffer configuration \( \mathbf{b} \).

\( n_i \): the number of candidate machines which can be used in station \( i \).

\( \mathbf{l} \): vector \((l_1, l_2, \ldots, l_{N-1})\), where \( l_j \) denotes a lower bound on the capacity of buffer \( j \).

\( \mathbf{u} \): vector \((u_1, u_2, \ldots, u_{N-1})\), where \( u_j \) denotes an upper bound on the capacity of buffer \( j \).

\( C_M(\cdot) \): machine cost per unit time including acquisition and operation costs.

\( C_B \): buffer cost per unit time including acquisition and operation costs.

\( E(\mathbf{a}, \mathbf{b}) \): throughput rate of the assembly system associated with configuration \((\mathbf{a}, \mathbf{b})\).

\( Z(\mathbf{a}, \mathbf{b}) \): total cost (purchase cost and operation cost for machines and buffers) associated with configuration \((\mathbf{a}, \mathbf{b})\).
Using the above notation, the design problem can be mathematically stated as follows.

\[
\text{minimize } Z(a, b) = \sum_{i=1}^{N} C_M(a_i) + C_B \sum_{j=1}^{N-1} b_j
\]

subject to \( E(a, b) \geq E_{\text{min}} \),

\[ 1 \leq a_i \leq n_i \text{ and integer, for } i = 1, 2, \ldots, N, \]

\[ l_j \leq b_j \leq u_j \text{ and integer, for } j = 1, 2, \ldots, N - 1. \]

In the above formulation, decision variables are the machines to be used in the stations and the capacities of the buffers, i.e. configuration \((a, b)\). For example, for an assembly system that consists of two stations, configuration \((3, 2, 4)\) means a system in which the third machine is selected among candidate machines for station 1, the second machine is selected among candidate machines for station 2, and capacity of buffer 1 is 4. The machine and buffer costs include both acquisition and operation costs. (Other cost terms that do not have a direct impact on the selection of machines and buffers are not considered in this study.) Here, the acquisition costs can be set to the equivalent uniform cash flow amounts of the costs discounted over the economic life of the machines or the buffers. Therefore, the total cost can be considered as the configuration-specific cost per period. Because of space restrictions or other economic and technological constraints, capacities of buffers are assumed to be bounded by given lower and upper limits \((l, u)\). For example, if the floor space between two machines may not be large enough to hold more than two items, the upper bound on the capacity of the buffer between the two machines is set to 2.

A machine to be used in a station characterizes the efficiency of the station. The efficiency is defined as

\[
mr = \frac{m_1 \mu_1}{\mu_1 + \lambda_1},
\]

where \(\mu, \lambda, \text{and} \mu\) denote the mean processing rate, the mean failure rate, and the mean repair rate of a machine, respectively. In this paper, it is assumed that the following relationships hold for candidate machines for each station,

\[
C_M(1) \leq C_M(2) \leq \cdots \leq C_M(n_i),
\]

\[
\frac{\mu_1 \rho_1}{\mu_1 + \lambda_1} \leq \frac{\mu_2 \rho_2}{\mu_2 + \lambda_2} \leq \cdots \leq \frac{\mu_{n_i} \rho_{n_i}}{\mu_{n_i} + \lambda_{n_i}},
\]

where \(\rho_k, \lambda_k, \text{and} \mu_k\) denote the mean processing rate, the mean failure rate, and the mean repair rate of the \(k\)th candidate machine for a station, respectively. That is, candidate machines for a station are indexed in an increasing order of the costs and a more expensive candidate machine is more efficient than a cheaper one.

It is also assumed that the throughput rate of the assembly system is concave on the capacities of buffers. The concavity of the throughput is proved for simpler manufacturing systems. For example, Meester and Shanthikumar (1990) and Anantharam and Tsoucas (1990) prove the stochastic concavity of the throughput in tandem queuing systems without considering machine failures. In this research, an empirical study was done to see whether the throughput rate of the assembly system is concave or not. That is, throughput rates were obtained from simulation on an assembly system with four stations and three buffers and the results are illustrated in Fig. 2. In this study, the total available capacity of the
buffers is set to 12, i.e. \( b_1 + b_2 + b_3 = 12 \). In this case, \( b_3 \) is expressed as \( b_3 = 12 - b_1 - b_2 \), which means sizes of only two buffers are independent variables in the four-station assembly system. In this example, the throughput rate appears to be concave. Note that the throughput rate may decrease in certain ranges of \( b_1 \) and \( b_2 \) even though these increase. The concavity property is needed in this research since the methods suggested here use a greedy type search procedure to find an optimal buffer configuration.

### 3. Solution methods

To solve the design problem defined in this paper, throughput rate of the assembly system should be estimated to check whether a configuration gives a prespecified desired throughput rate or not. Assembly systems with a small number of stations can be analyzed exactly using Markov chain models. However, there may be a large number of stations in most real assembly systems, and therefore they cannot be analyzed exactly because there exist an extremely large number of system states. Simulation or decomposition methods are often used for an approximate analysis of the assembly systems. In general, simulation needs much longer computation time than computational algorithms based on decomposition methods, although simulation can analyze more complex (realistic) systems.

Even with an efficient tool for performance evaluation of the assembly system, however, it takes very long time to evaluate all possible configurations since there are a large number of configurations to be considered. Therefore, we propose heuristics that can give good configurations in a reasonable amount of time. First, we suggest a method for determining the buffer capacities by which a desired throughput can be obtained with the minimum buffer cost for a given machine configuration.
3.1. Finding the best buffer configuration for a given machine configuration

Even for systems with the same set of machines for the stations, throughput rate may vary according to
the capacities of the buffers. Therefore, for a given machine configuration, it is necessary to find a buffer
configuration which requires the minimum capacity of buffers among those that give a desired through-
put rate. In this paper, an algorithm is developed for buffer allocation based on a gradient search and an
improvement procedure. The gradient search is used for finding a good solution (or alternative) for
buffer configuration and then this solution is improved by searching neighborhood solutions.

The buffer allocation algorithm first checks whether there exist a feasible buffer configuration. If
feasible buffer configurations do not exist, the algorithm stops. If the desired throughput rate is achieved
when buffer capacities are set to their lower bounds, the best buffer configuration is
\[
l_1; l_2; \ldots; l_N
\]

In the gradient search, the lower bound is used for an initial solution. Then, capacity of a buffer is increased
by one in such a way that such increase maximizes increase in throughput rate. This step is continued
until the desired throughput rate is obtained. After the gradient search is done, the algorithm improves
the solution by searching for a configuration that gives the desired throughput rate using a smaller total
buffer capacity than the current configuration. For the purpose, it first generates a solution with one less
total buffer capacity and then searches neighborhood solutions of the one generated. Neighborhood
solutions are generated by increasing the capacity of a buffer by one and decreasing capacity of another
buffer by one. If a new configuration (corresponding to a neighborhood solution) gives the desired
throughput rate, the current solution is replaced with the new solution and the improvement procedure
is repeated. Otherwise, the current solution is the minimum cost buffer configuration, since no other
buffer configuration has a smaller total capacity of buffers than the current one.

The buffer allocation algorithm can be summarized as follows. Let \( a \) denote the given machine
configuration and \( e_i \) denote the \( N-1 \) dimensional unit vector with \( e_{ii} = 1 \) and \( e_{ij} = 0 \) for all \( j \neq i \),
where \( e_{ij} \) is the \( j \)th component of the vector.

Algorithm BA.

**Step 1.** If \( E(a, u) < E_{\text{min}} \), stop. Feasible buffer configurations do not exist. If \( E(a, l) \geq E_{\text{min}} \), stop. The
best buffer configuration is \( l \). Otherwise, let \( b = l \).

**Step 2.** Let \( b = b + e_{j_{\text{max}}} \), where \( j_{\text{max}} = \text{argmax}\{E(a, b + e_j) - E(a, b)|b_j < u_j\} \).

**Step 3.** If \( E(a, b) < E_{\text{min}} \), go to step 2. Otherwise, go to step 4.

**Step 4.** Let \( b = b - e_{j_{\text{min}}} \), where \( j_{\text{min}} = \text{argmin}\{E(a, b) - E(a, b - e_j)|b_j > l_j\} \).

**Step 5.** Let \( b^* = b \). For all \( i \) and \( j \) such that \( b_i < u_i, b_j > l_j, \) and \( j \neq i \), do:
   - If \( E(a, b + e_i - e_j) > E(a, b^*) \), let \( b^* = b + e_i - e_j \).

**Step 6.** If \( E(a, b^*) > E(a, b) \), let \( b = b^* \) and go to step 5. Otherwise, go to step 7.

**Step 7.** If \( E(a, b^*) \geq E_{\text{min}} \), let \( b = b^* \) and go to step 4. Otherwise, stop. The best buffer configuration is
\( b + e_{j_{\text{min}}} \).

3.2. Lower and upper configurations

In the heuristics suggested here for the design problem, lower and upper configurations are used as an
initial configuration. The lower configuration is a configuration that gives the desired throughput rate
with the minimum machine cost. On the other hand, the upper configuration is a configuration that gives
the desired throughput rate with the minimum buffer cost. That is, less expensive machines and larger
size buffers are used in the lower configuration, whereas more expensive machines and smaller size
buffers are used in the upper configuration. Starting from the lower configuration, the heuristics attempt
to reduce the total cost by changing machines to more efficient (and more expensive) ones and reducing
the buffer capacities, which results in increase in the machine cost but decrease in the buffer cost that can
overcome such increase. Similarly, starting from the upper configuration, the heuristics attempt to
reduce the total cost by changing machines to cheaper (and less efficient) ones and increasing the buffer
capacities, which results in increase in the buffer cost but decrease in the machine cost.

The heuristics suggested for the design problem use lower and upper configurations obtained by the
following heuristic procedures, since it is difficult to obtain the exact lower and upper configurations.
Currently, there exists no known (efficient) method to obtain exact lower and upper configuration except
for the full enumeration method, which requires excessive computation time. We first define \( \Delta^+_i(a, b) \) and \( \Delta^-_i(a, b) \) which denote the positive and negative gradients of the throughput per unit cost,
respectively, as follows.

\[
\Delta^+_i(a, b) = \begin{cases} 
\frac{E(a_1, a_2, \ldots, a_i + 1, \ldots, a_N, b) - E(a, b)}{C_M(a_i + 1) - C_M(a_i)} & \text{for } a_i < n_i \\
0 & \text{for } a_i = n_i 
\end{cases}
\]

\[
\Delta^-_i(a, b) = \begin{cases} 
\frac{E(a, b) - E(a_1, a_2, \ldots, a_i - 1, \ldots, a_N, b)}{C_M(a_i) - C_M(a_i - 1)} & \text{for } a_i > 1 \\
\infty & \text{for } a_i = 1 
\end{cases}
\]

Then, a lower configuration \((a^L, b^L)\) and an upper configuration \((a^U, b^U)\) can be obtained using
Algorithms LC and UC, respectively.

Algorithm LC.

\textbf{Step 0.} Let \( a = (1, 1, \ldots, 1) \).

\textbf{Step 1.} If \( E(a, u) \geq E_{\min} \), find the best buffer configuration \( b^* \) for \( a \) using Algorithm BA and stop. The lower configuration is \((a, b^*)\). Otherwise, go to step 2.

\textbf{Step 2.} Let \( a = (a_1, a_2, \ldots, a_i - 1, \ldots, a_N) \), where \( i^+ = \arg \max \{\Delta^+_i(a, u)\} \). Go to step 1.

Algorithm UC.

\textbf{Step 1.} Let \( a = (n_1, n_2, \ldots, n_N) \). If \( E(a, l) \leq E_{\min} \), find the best buffer configuration \( b^* \) for \( a \) using Algorithm BA and stop. The upper configuration is \((a, b^*)\). Otherwise, go to step 2.

\textbf{Step 2.} Let \( a = (a_1, a_2, \ldots, a_i - 1, \ldots, a_N) \), where \( i^- = \arg \min \{\Delta^-_i(a, l)\} \).

\textbf{Step 3.} If \( E(a, l) < E_{\min} \), stop. The upper configuration is \((a_1, a_2, \ldots, a_i - 1, \ldots, a_N, l)\). Otherwise, go to step 2.

In the following sections, we present the three heuristics for selecting the machines to be used in
stations and determining the buffer capacities at the same time. The heuristics use algorithm BA, the positive and negative gradients $\Delta^+_i(a, b)$ and $\Delta^-_i(a, b)$, and the lower and upper configurations $(a^L, b^L)$ and $(a^U, b^U)$.

### 3.3. Heuristic 1

First, it is checked whether the cheapest (least efficient) configuration, i.e. $(1, 1, \ldots, 1, 1)$, gives the desired throughput rate or not. If it does, it is obviously an optimal configuration. Otherwise, it is checked whether there exist feasible configurations through evaluation of the configuration with the most efficient (most expensive) machines. If feasible configurations exist, starting from the lower configuration, the machine to be used in the station (say, station $i^*$) with the maximum positive gradient is replaced with a much more efficient (but expensive) one, i.e. $a_{i^*}$ is changed to $a_{i^*} + 1$. Then, the best buffer configuration is found by algorithm BA for the changed machine configuration $(a_1, a_2, \ldots, a_{i^*} + 1, \ldots, a_N)$. This procedure is repeated until it is evident that such changes do not reduce cost any longer, that is, the lower bound $l$ becomes the best buffer configuration for a machine configuration. Finally, the minimum cost configuration is selected among all configurations generated in this procedure. The heuristic is summarized in the following algorithm.

**Algorithm H1.**

1. **Step 1.** If $E(1, 1, \ldots, 1) \geq E_{\text{min}}$, stop. The best configuration is $(1, 1, \ldots, 1, 1)$. If $E(n_1, n_2, \ldots, n_N, u) < E_{\text{min}}$, stop. Feasible configurations do not exist. Otherwise, let $k = 1$, $(a^L, b^L) = (a^L, b^L)$, and $Z^L = Z(a^L, b^L)$.

2. **Step 2.** Let $k = k + 1$ and $a^k = (a_1, a_2, \ldots, a_{i^*} + 1, \ldots, a_N)$, where $i^* = \arg\max\{\Delta^+_i(a_{i^*} - 1, b_{i^*} - 1)\}$. Find the best buffer configuration $b^k$ for $a^k$ using Algorithm BA and let $Z^k = Z(a^k, b^k)$.

3. **Step 3.** If $b^k > l$, go to step 2. Otherwise, stop. The solution is $(a^k, b^k)$, where $k = \arg\min(Z^k)$.

### 3.4. Heuristic 2

This heuristic also starts from the lower configuration but does not use the concept of gradients. For each of all stations in which more efficient machines can be used (for each of $i$ such that $a_i < n_i$), the machine is replaced with a much more efficient one, while leaving the machines to be used in other stations unchanged. (As a result of this change, the machine cost increases, but the buffer cost may decrease since the desired throughput rate can be obtained with smaller size buffers.) Then, the best buffer configuration is found for the changed machine configuration. Among all of newly generated configurations, the one that requires the minimum cost is selected. If the selected configuration requires less cost than the incumbent solution, the incumbent solution is replaced with the selected one. This procedure is repeated until such changes cannot reduce the cost any longer. The heuristic is summarized on the following page.
Algorithm H2.

Step 1. If $E(1, 1, \ldots, 1, 1) \geq E_{\text{min}}$, stop. The best configuration is $(1, 1, \ldots, 1, 1)$. If $E(n_1, n_2, \ldots, n_N, \mathbf{u}) < E_{\text{min}}$, stop. Feasible configurations do not exist. Otherwise, let $(\mathbf{a}, \mathbf{b}) = (\mathbf{a}^L, \mathbf{b}^L)$ and $Z^* = Z(\mathbf{a}, \mathbf{b})$.

Step 2. For all $i$ such that $a_i < n_i$ do:

Let $\mathbf{a}_i = (a_1, a_2, \ldots, a_i + 1, \ldots, a_N)$, find the best buffer configuration $\mathbf{b}_i$ for $\mathbf{a}_i$ using Algorithm BA and let $Z_i = Z(\mathbf{a}_i, \mathbf{b}_i)$.

Step 3. Find $\hat{i} = \arg\min\{Z_i\}$. If $Z_{\hat{i}} < Z^*$, let $(\mathbf{a}, \mathbf{b}) = (\mathbf{a}_{\hat{i}}, \mathbf{b}_{\hat{i}})$ and $Z^* = Z_{\hat{i}}$ and go to step 2. Otherwise, stop. The solution is $(\mathbf{a}, \mathbf{b})$.

3.5. Heuristic 3

Unlike Heuristics 1 and 2, this heuristic starts from the upper configuration. However, the basic idea of this heuristic is similar to that of Heuristic 2. For each of all stations in which less efficient (but cheaper) machines can be used (for each of $i$ such that $a_i > 1$), the machine is replaced with a less efficient one. (As a result of this change, the buffer cost increases but the machine cost decreases.) Then, the best buffer configuration is obtained for the changed machine configuration. Among newly generated configurations, the one that requires the minimum cost is selected. If the selected configuration requires less cost than the incumbent solution, the incumbent solution is replaced with the new solution. This procedure is repeated until such changes cannot reduce the cost any longer. A procedure for the heuristic is given in the following algorithm.

Algorithm H3.

Step 1. If $E(1, 1, \ldots, 1, 1) \geq E_{\text{min}}$, stop. The best configuration is $(1, 1, \ldots, 1, 1)$. If $E(n_1, n_2, \ldots, n_N, \mathbf{u}) < E_{\text{min}}$, stop. Feasible configurations do not exist. Otherwise, let $(\mathbf{a}, \mathbf{b}) = (\mathbf{a}^U, \mathbf{b}^U)$ and $Z^* = Z(\mathbf{a}, \mathbf{b})$.

Step 2. For all $i$ such that $a_i > 1$, do:

Let $\mathbf{a}_i = (a_1, a_2, \ldots, a_i - 1, \ldots, a_N)$, find the best buffer configuration $\mathbf{b}_i$ for $\mathbf{a}_i$ using Algorithm BA and let $Z_i = Z(\mathbf{a}_i, \mathbf{b}_i)$.

Step 3. Find $\hat{i} = \arg\min\{Z_i\}$. If $Z_{\hat{i}} < Z^*$, let $(\mathbf{a}, \mathbf{b}) = (\mathbf{a}_{\hat{i}}, \mathbf{b}_{\hat{i}})$ and $Z^* = Z_{\hat{i}}$ and go to step 2. Otherwise, stop. The solution is $(\mathbf{a}, \mathbf{b})$.

Computation times of the heuristics primarily depend on the number of times the buffer allocation problem is solved. In the worst case, Algorithm BA is executed $\sum_{i=1}^N (n_i - 1)$ times in Heuristic 1, whereas it is executed $N \sum_{i=1}^N (n_i - 1)$ times in Heuristics 2 and 3. Therefore, Heuristic 1 may require shorter time than Heuristics 2 and 3.

3.6. Improving a solution

Solutions obtained from the above three heuristics may be improved if they are slightly changed. Therefore, we suggest an improvement procedure, in which neighborhood solutions are generated from
the current solution and one of them is selected as a new solution. In the suggested procedure, three alternative machine configurations are generated from the current configuration: (1) by replacing the machine to be used in a station with a more efficient one; (2) by replacing the machine to be used in a station with a less efficient one; and (3) by replacing the machine to be used in one station with a more efficient one and the machine to be used in another station with a less efficient one. The stations in which the machines are to be replaced are selected using the positive and negative gradients, that is, the machines to be used in the stations that have the maximum and the minimum gradients are replaced with a more efficient and a less efficient one, respectively. Then, the best buffer configuration is obtained for each of the three alternatives and the current incumbent configuration is replaced with the best alternative if it reduces the total cost. This procedure is repeated until the cost cannot be reduced any longer by such replacements. The following summarizes the procedure.

Algorithm IM.

Step 0. Let \((a, b)\) be an initial solution (obtained from one of the three heuristics) and \(Z = Z(a, b)\).

Step 1. Let \(a^+ = (a_1, a_2, \ldots, a_i^+ + 1, \ldots, a_N)\), where \(i^+ = \arg\max\{\Delta^+(a, b)\}\). Find the best buffer configuration \(b^+\) for \(a^+\) using Algorithm BA, and let \(Z^+ = Z(a^+, b^+)\).

Step 2. Let \(a^- = (a_1, a_2, \ldots, a_i^- - 1, \ldots, a_N)\), where \(i^- = \arg\min\{\Delta^-(a, b)\}\). Find the best buffer configuration \(b^-\) for \(a^-\) using Algorithm BA, and let \(Z^- = Z(a^-, b^-)\).

Step 3. Let \(a^\pm = (a_1, a_2, \ldots, a_i^+ + 1, \ldots, a_i^- - 1, \ldots, a_N)\). Find the best buffer configuration \(b^\pm\) for \(a^\pm\) using Algorithm BA, and let \(Z^\pm = Z(a^\pm, b^\pm)\).

Step 4. If \(Z^* = \min\{Z^+, Z^-, Z^\pm\} < Z\), let \(Z = Z^*\) and \((a, b) = (a^*, b^*)\), where \((a^*, b^*)\) is the configuration corresponding to \(Z^*\), and go to step 1. Otherwise, stop. The solution is \((a, b)\).

The overall procedure suggested in this study is depicted as a flowchart in Fig. 3. In the procedure, Algorithms LC and UC find initial solutions, i.e. lower and upper configurations. Then, starting from the initial solutions, Algorithms H1–H3 searches for solutions that give better objective function values. Finally, Algorithm IM improves the solutions resulting from Algorithms H1–H3. In Algorithms LC, UC, H1, H2, H3, and IM, Algorithm BA is used to find the best buffer configuration for a given machine configuration.

We give an example to show how the heuristics work. The assembly system in the example consists of seven stations and six buffers as shown in Fig. 4(b). In this example, up and down times of machines and processing times are assumed to be exponentially distributed. There are three candidate machines for each of the stations, and the machine costs and failure \((\lambda)\), repair \((\mu)\), and processing \((\rho)\) rates are given in Table 1. The buffer cost is 1.0206 and the lower and upper bounds on the buffer capacities are 5 and 15, respectively. Assume the desired throughput rate of the assembly system is 0.7632.

As shown in Table 2, starting from a lower configuration \{\(1, 1, 1, 1, 1, 1\), \((11, 9, 15, 15, 15, 13)\)\}, Heuristic 1 generated 14 configurations and selected configuration \{\(1, 1, 2, 2, 1, 3, 2\)\}, \((9, 8, 12, 9, 9, 5)\)\}. Starting from the lower configuration as well, Heuristic 2 found the same configuration after 6 iterations. Heuristic 3 also found the same configuration after 10 iterations, starting from an upper configuration \{\(3, 3, 3, 3, 3, 3\)\}, \((5, 5, 6, 6, 6, 5)\)\}. In the heuristics, throughput rate is calculated with a decomposition algorithm of Jeong and Kim (1998). After a full enumeration, we found that the solution obtained by the
heuristics was the best solution. In Heuristic 1, Algorithm BA was executed once per iteration, hence it was executed 14 times. On the other hand, in Heuristics 2 and 3, Algorithm BA can be executed up to seven times per iteration. In Heuristic 2, Algorithm BA was executed 41 times for six iterations, while it was executed 62 times for ten iterations in Heuristics 3.

Fig. 3. A flowchart for the overall solution procedure.

(a) 3-station structure

(b) 7-station structure

(c) 11-station structure

Fig. 4. Structures of the system included in the test.
4. Computational experiments

Performance of the suggested heuristics was tested through computational experiments on assembly systems in which the times between failures and the times to repair for the machines and the processing times of the operations are exponentially distributed. Such system characteristics can be found in unpaced (asynchronous) assembly systems in which processing times at different machines are different. In such systems, the processing times may be different because it is difficult to balance workloads perfectly in the systems. Moreover, if multiple products or multiple models are produced in a system, processing times on the same machine may differ for different items. In addition to this, the stochasticity of processing times may be caused by minor disturbances in the manufacturing system such as tool breakage, shortage of lubricating oil, and variability of human workers. Simulation and the decomposition method of Jeong and Kim (1998) are used for an approximate analysis of the assembly systems. In the experiments, both simulation and the decomposition method are used for small assembly systems but only the decomposition method is used for large systems.

In the experiments, three types of the structures of the assembly systems given in Fig. 4 were used for problem generation. For each type of the structures, ten test problems were randomly generated as follows. Here, $R \sim U(x, y)$ means that $R$ follows $U(x, y)$, the uniform distribution with range $x$ to $y$.

1. The number of candidate machines which can be used in a station is 3 for the problems with the 3- and 7-station structures and 2 for those with the 11-station structure.
2. The processing rate of the \( k \)th candidate machine for station \( i \) is generated as follows: 
\[
\rho_1 \sim \mathcal{U}(0.9, 1.1) \quad \text{and} \quad \rho_k \sim \mathcal{U}(\rho_{k-1}, 1.2 \times \rho_{k-1}), \quad \text{for} \quad k = 2, 3, \ldots, n_i, \quad \text{for all stations}.
\]

3. The repair rate of the \( k \)th candidate machine for station \( i \) is generated as follows: 
\[
\mu_1 \sim \mathcal{U}(0.09, 0.11) \quad \text{and} \quad \mu_k \sim \mathcal{U}(\mu_{k-1}, 1.2 \times \mu_{k-1}), \quad \text{for} \quad k = 2, 3, \ldots, n_i, \quad \text{for all stations}.
\]

4. The failure rate of the \( k \)th candidate machine for station \( i \) is generated as follows: 
\[
\lambda_1 \sim \mathcal{U}(0.0045, 0.0055) \quad \text{and} \quad \lambda_k \sim \mathcal{U}(0.8 \times \lambda_{k-1}, \lambda_{k-1}), \quad \text{for} \quad k_i = 2, 3, \ldots, n_i, \quad \text{for all stations}.
\]

5. The machine cost of the \( k \)th candidate machine for station \( i \) is generated as follows: 
\[
C_M(1) \sim \mathcal{U}(40 \times 0.9, 40 \times 1.1) \quad \text{and} \quad C_M(k) \sim \mathcal{U}(C_M(k - 1), 1.2 \times C_M(k - 1)), \quad \text{for} \quad k = 2, 3, \ldots, n_i, \quad \text{for all stations}.
\]
6. The buffer cost, $C_B$ is generated from $U(0.9, 1.1)$.
7. The lower and upper bounds on the buffer capacity are 5 and 15, respectively, for all buffers.
8. The desired throughput is set by $E_{\min} = (E(1,1,\ldots,1) + E(n_1,n_2,\ldots,n_N,\mathbf{u}))/2$.

Performance of the proposed heuristics may be affected by both the efficiency of the search method (for finding a good configuration) and the accuracy of the performance evaluation method (for evaluating the throughput rate of a given configuration). Since there is no existing algorithm for the design problem considered in this study, the heuristic algorithms suggested here could only be compared with each other and with the full enumeration algorithm. The following eight methods were included in the comparison:

Methods 1, 2, and 3: Heuristics 1, 2, and 3, respectively, using the decomposition method for performance evaluation.
Method 4: Full enumeration method (for selecting configurations) using the decomposition method for performance evaluation.
Methods 5, 6, and 7: Heuristics 1, 2, and 3, respectively, using simulation for performance evaluation.
Method 8: Full enumeration method using simulation for performance evaluation.

Since simulation cannot be used for performance evaluation in the problems of the 7- and 11-station structures due to excessive computation time, the above eight methods were tested on the problems of the 3-station structure only. In the simulation, ten replications were made for estimating performance of a configuration so as to obtain statistically reliable simulation results. In each replication, 31 000 time units were simulated and statistics during $[1000, 31 000]$ were gathered. The test was done using C language on a personal computer with a Pentium processor (120 MHz).

Results are shown in Table 3. In all problems tested, Methods 1 and 2 (H1 and H2) found the same solutions as those found by Method 4, and Method 3 (H3) found the same solutions except for two problems. Similar results can be seen from comparison of Methods 5, 6, and 7 with Method 8. These mean that the search methods used in the proposed heuristics, especially those used in Heuristics 1 and 2, seem to work very well (the same solutions were obtained by the search method and the full enumeration method). These also show that the lower configuration obtained by using the positive gradient works better as a starting configuration than the upper configuration obtained by using the negative gradient. Configurations obtained from the methods that use the decomposition algorithm for performance evaluation required larger buffer capacities than those obtained from the methods that use simulation, because the decomposition algorithm underestimated throughput rates of the assembly systems (in this case). The average errors (differences in the total costs) of configurations obtained from the methods using the decomposition method for performance evaluation were less than 1% compared with those obtained from Method 8. Most of these errors came from the error of the decomposition method.

Results of a comparison of the heuristics on large sized problems are shown in Tables 4 and 5. In many test problems, the proposed heuristics gave the same solution as those obtained from the full enumeration method. (Out of 20 test problems, Heuristics 1, 2, and 3 gave the best solutions in 18, 19, and 15 problems, respectively.) In these cases too, Heuristics 1 and 2 outperformed Heuristic 3 in solution quality. The average computation times for Heuristics 1, 2, and 3 were 7.2, 19.8, and 31.8 min, respectively, for the problems of the 7-station structure, and 15.6, 67.8, and 133.2, respectively, for those of the 11-station structure. The full enumeration method required much longer computation time (108.6 min
for problems with 7 machines and 800.4 min for those with 11 machines). Note that the computation
time required for the full enumeration method was not too long since many solutions were fathomed
during the enumeration by using a solution obtained from the heuristics as an upper bound.

The decomposition algorithm used for performance evaluation in the proposed heuristics is an
approximation method. Therefore, configurations obtained from the proposed heuristics may not actu-
ally give the desired throughput rate as shown in Tables 4 and 5. In fact, in the problems of the 7- and 11-
station structures, the decomposition algorithm overestimated throughput rates, and hence the config-
uration of the solution did not give the desired throughput rate. On the other hand, in the problems of the
3-station structure, the decomposition method underestimated throughput rates, and hence the desired
throughput can actually be satisfied by less efficient machines or smaller buffer capacities. Therefore, the
configurations obtained from the heuristics need to be adjusted.

We suggest an adjustment method based on the observation that configurations obtained from the
heuristics are slightly different in buffer capacities from those obtained from the full enumeration
method in which simulation is used for performance evaluation (see Table 3). In the adjustment method,
buffer capacities are reallocated so that the configurations can satisfy more exactly the desired throughput

Table 3
Results of experiments on problems of the 3-station structure

<table>
<thead>
<tr>
<th>Problem</th>
<th>Method</th>
<th>a</th>
<th>b</th>
<th>Z(a, b)</th>
<th>RDP^a</th>
<th>Method</th>
<th>a</th>
<th>b</th>
<th>Z(a, b)</th>
<th>RDP^a</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1, 2</td>
<td>2, 3, 1</td>
<td>8, 8</td>
<td>149</td>
<td>0.74</td>
<td>5, 6</td>
<td>2, 3, 1</td>
<td>7, 8</td>
<td>147.9</td>
<td>0.00</td>
</tr>
<tr>
<td>2</td>
<td>1, 2, 3</td>
<td>3, 1, 3</td>
<td>5, 6</td>
<td>141.8</td>
<td>0.78</td>
<td>5, 6, 7</td>
<td>3, 1, 3</td>
<td>5, 5</td>
<td>140.7</td>
<td>0.00</td>
</tr>
<tr>
<td>3</td>
<td>1, 2, 3</td>
<td>3, 2, 3</td>
<td>5, 7</td>
<td>137.8</td>
<td>0.73</td>
<td>5, 6, 7</td>
<td>3, 2, 3</td>
<td>5, 6</td>
<td>136.8</td>
<td>0.00</td>
</tr>
<tr>
<td>4</td>
<td>1, 2, 3</td>
<td>3, 2, 3</td>
<td>5, 7</td>
<td>137.8</td>
<td>0.73</td>
<td>5, 6, 7</td>
<td>3, 2, 3</td>
<td>5, 6</td>
<td>136.8</td>
<td>0.00</td>
</tr>
<tr>
<td>5</td>
<td>1, 2, 3</td>
<td>3, 2, 3</td>
<td>5, 7</td>
<td>137.8</td>
<td>0.73</td>
<td>5, 6, 7</td>
<td>3, 2, 3</td>
<td>5, 6</td>
<td>136.8</td>
<td>0.00</td>
</tr>
<tr>
<td>6</td>
<td>1, 2, 3</td>
<td>3, 2, 3</td>
<td>5, 7</td>
<td>137.8</td>
<td>0.73</td>
<td>5, 6, 7</td>
<td>3, 2, 3</td>
<td>5, 6</td>
<td>136.8</td>
<td>0.00</td>
</tr>
<tr>
<td>7</td>
<td>1, 2, 3</td>
<td>3, 2, 3</td>
<td>5, 7</td>
<td>137.8</td>
<td>0.73</td>
<td>5, 6, 7</td>
<td>3, 2, 3</td>
<td>5, 6</td>
<td>136.8</td>
<td>0.00</td>
</tr>
<tr>
<td>8</td>
<td>1, 2, 3</td>
<td>3, 2, 3</td>
<td>5, 7</td>
<td>137.8</td>
<td>0.73</td>
<td>5, 6, 7</td>
<td>3, 2, 3</td>
<td>5, 6</td>
<td>136.8</td>
<td>0.00</td>
</tr>
<tr>
<td>Average</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

^a Relative deviation percentage of the objective function value of each method from that of Method 8.
Table 4
Results of experiments on problems of the 7-station structure (Methods H1, H2, H3, and F denote Heuristics 1, 2, 3 and full enumeration, respectively)

<table>
<thead>
<tr>
<th>Problem</th>
<th>Method</th>
<th>Cost</th>
<th>$E_{\min}$</th>
<th>$E^a$</th>
<th>$E_s^b$</th>
<th>RDP$^c$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>H1, H2, H3, F</td>
<td>339.24</td>
<td>0.765</td>
<td>0.765</td>
<td>0.760</td>
<td>0.76</td>
</tr>
<tr>
<td>2</td>
<td>H1, H2, F</td>
<td>352.42</td>
<td>0.760</td>
<td>0.762</td>
<td>0.756</td>
<td>0.76</td>
</tr>
<tr>
<td>3</td>
<td>H3</td>
<td>355.11</td>
<td>0.760</td>
<td>0.757</td>
<td>0.50</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>H1, H2, F</td>
<td>352.23</td>
<td>0.825</td>
<td>0.825</td>
<td>0.822</td>
<td>0.47</td>
</tr>
<tr>
<td>5</td>
<td>H3</td>
<td>358.16</td>
<td>0.826</td>
<td>0.826</td>
<td>0.825</td>
<td>0.17</td>
</tr>
<tr>
<td>6</td>
<td>H1</td>
<td>326.08</td>
<td>0.777</td>
<td>0.777</td>
<td>0.777</td>
<td>0.06</td>
</tr>
<tr>
<td>7</td>
<td>H3</td>
<td>328.67</td>
<td>0.778</td>
<td>0.784</td>
<td>0.77</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>H3</td>
<td>327.76</td>
<td>0.777</td>
<td>0.778</td>
<td>0.772</td>
<td>0.82</td>
</tr>
<tr>
<td>9</td>
<td>H2, H3, F</td>
<td>350.85</td>
<td>0.777</td>
<td>0.777</td>
<td>0.770</td>
<td>0.89</td>
</tr>
<tr>
<td>10</td>
<td>H1, H2, F</td>
<td>351.05</td>
<td>0.751</td>
<td>0.752</td>
<td>0.737</td>
<td>2.05</td>
</tr>
<tr>
<td></td>
<td>H3</td>
<td>327.62</td>
<td>0.751</td>
<td>0.752</td>
<td>0.739</td>
<td>1.85</td>
</tr>
<tr>
<td></td>
<td>H1</td>
<td>350.73</td>
<td>0.751</td>
<td>0.752</td>
<td>0.746</td>
<td>0.85</td>
</tr>
<tr>
<td></td>
<td>H2</td>
<td>352.67</td>
<td>0.751</td>
<td>0.752</td>
<td>0.746</td>
<td>0.85</td>
</tr>
<tr>
<td></td>
<td>H1, H2, F</td>
<td>335.83</td>
<td>0.744</td>
<td>0.744</td>
<td>0.741</td>
<td>0.47</td>
</tr>
<tr>
<td></td>
<td>H3</td>
<td>337.43</td>
<td>0.747</td>
<td>0.744</td>
<td>0.744</td>
<td>0.37</td>
</tr>
<tr>
<td></td>
<td>H3, F</td>
<td>346.59</td>
<td>0.785</td>
<td>0.786</td>
<td>0.778</td>
<td>1.01</td>
</tr>
<tr>
<td></td>
<td>H1, H2, F</td>
<td>346.67</td>
<td>0.786</td>
<td>0.786</td>
<td>0.783</td>
<td>0.46</td>
</tr>
<tr>
<td></td>
<td>H3</td>
<td>343.9</td>
<td>0.709</td>
<td>0.710</td>
<td>0.692</td>
<td>2.50</td>
</tr>
<tr>
<td></td>
<td>H1, H2, F</td>
<td>355.9</td>
<td>0.708</td>
<td>0.708</td>
<td>0.696</td>
<td>1.78</td>
</tr>
<tr>
<td></td>
<td>H3</td>
<td>515.7</td>
<td>0.724</td>
<td>0.725</td>
<td>0.721</td>
<td>0.45</td>
</tr>
<tr>
<td></td>
<td>H1, H2, F</td>
<td>522.4</td>
<td>0.715</td>
<td>0.716</td>
<td>0.703</td>
<td>1.81</td>
</tr>
<tr>
<td>Average</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0.71</td>
</tr>
</tbody>
</table>

$^a$ Throughput rate estimated by the computational algorithm.
$^b$ Throughput rate estimated by simulation.
$^c$ Relative deviation percentage, i.e. $|E - E_s|/E_s \times 100(\%)$.

Table 5
Results of experiments on problems of the 11-station structure (Methods H1, H2, H3, and F denote Heuristics 1, 2, 3 and full enumeration, respectively)

<table>
<thead>
<tr>
<th>Problem</th>
<th>Method</th>
<th>Cost</th>
<th>$E_{\min}$</th>
<th>$E^a$</th>
<th>$E_s^b$</th>
<th>RDP$^c$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>H1, H2, H3, F</td>
<td>532.2</td>
<td>0.701</td>
<td>0.702</td>
<td>0.699</td>
<td>0.30</td>
</tr>
<tr>
<td>2</td>
<td>H1, H2, H3, F</td>
<td>529.2</td>
<td>0.725</td>
<td>0.727</td>
<td>0.711</td>
<td>2.19</td>
</tr>
<tr>
<td>3</td>
<td>H1, H2, H3, F</td>
<td>526.0</td>
<td>0.719</td>
<td>0.721</td>
<td>0.704</td>
<td>2.31</td>
</tr>
<tr>
<td>4</td>
<td>H1, H2, H3, F</td>
<td>530.1</td>
<td>0.718</td>
<td>0.719</td>
<td>0.704</td>
<td>2.19</td>
</tr>
<tr>
<td>5</td>
<td>H1, H2, H3, F</td>
<td>525.6</td>
<td>0.693</td>
<td>0.694</td>
<td>0.682</td>
<td>1.84</td>
</tr>
<tr>
<td>6</td>
<td>H1, H2, H3, F</td>
<td>523.6</td>
<td>0.734</td>
<td>0.737</td>
<td>0.731</td>
<td>0.72</td>
</tr>
<tr>
<td>7</td>
<td>H1, H2, H3, F</td>
<td>543.9</td>
<td>0.709</td>
<td>0.710</td>
<td>0.692</td>
<td>2.50</td>
</tr>
<tr>
<td>8</td>
<td>H1, H2, H3, F</td>
<td>535.9</td>
<td>0.708</td>
<td>0.708</td>
<td>0.696</td>
<td>1.78</td>
</tr>
<tr>
<td>9</td>
<td>H1, H2, H3, F</td>
<td>515.7</td>
<td>0.724</td>
<td>0.725</td>
<td>0.721</td>
<td>0.45</td>
</tr>
<tr>
<td>10</td>
<td>H1, H2, H3, F</td>
<td>522.4</td>
<td>0.715</td>
<td>0.716</td>
<td>0.703</td>
<td>1.81</td>
</tr>
<tr>
<td>Average</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1.61</td>
</tr>
</tbody>
</table>

$^a$ Throughput rate estimated by the computational algorithm.
$^b$ Throughput rate estimated by simulation.
$^c$ Relative deviation percentage, i.e. $|E - E_s|/E_s \times 100(\%)$. 
rate (but the machine configuration is not changed). That is, if the throughput rate estimated from simulation is less (greater) than the desired throughput rate, the capacity of a buffer is increased (decreased) in such a way that such a change in the buffer maximizes (minimizes) change in throughput. Buffer capacities are changed with this method until no more changes are necessary.

The above adjustment procedure was applied to configurations obtained from the heuristics using the decomposition method for performance evaluation. Results are summarized in Table 6, which shows changes in the total buffer capacity and the total cost, and additional computation time needed for the adjustment. The changes are relatively small, but they seem to increase as the number of machines increases. This is because the difference of the throughput rates estimated by the decomposition algorithm and simulation increases as the number of stations increases. Although simulation is used for performance evaluation, additional computation time for this adjustment is relatively short since only a few configurations need to be evaluated. Using the procedure, we can obtain configurations which exactly give the desired throughput rate.

5. Concluding remarks

We considered the problem of finding a minimum cost configuration of an assembly system for a given desired throughput rate. Three heuristics were suggested for the design problem. Starting from a lower or upper configuration, these heuristics generate promising machine configurations and then find the best buffer configurations for the machine configurations. To test the performance of the proposed heuristics, computational experiments were done on a number of test problems. The test results showed that the proposed heuristics (especially, Heuristic 1) gave relatively good configurations within a reasonable computation time. The proposed heuristics can be applied not only to the assembly system but also to many other manufacturing systems in which machines to be used and buffer capacities should be determined, if appropriate performance evaluation tools are available.

Although the design problem was solved sequentially in this study, i.e. selecting machines first and then buffers or vice versa, one might devise a method that determines machines and buffers simultaneously. However, it is not expected that this method will outperform the suggested heuristics in terms of the solution quality or computation time. It may require excessively long computation time since the search space is much larger. Also, as can be seen from the results of comparison with alternatives

<table>
<thead>
<tr>
<th>Structure</th>
<th>Change in total buffer capacity (unit)(^a)</th>
<th>Change in cost (%)(^b)</th>
<th>Computation time (min)</th>
</tr>
</thead>
<tbody>
<tr>
<td>3-station structure</td>
<td>0.75</td>
<td>0.53</td>
<td>0.51</td>
</tr>
<tr>
<td>7-station structure</td>
<td>2.05</td>
<td>0.62</td>
<td>13.77</td>
</tr>
<tr>
<td>11-station structure</td>
<td>6.2</td>
<td>1.25</td>
<td>133.32</td>
</tr>
</tbody>
</table>

\(^a\) Difference between buffer capacities used in the adjusted configuration and the original one.

\(^b\) Percentage deviation of costs of the adjusted configuration from those of the original one.
obtained from the full enumeration, the alternatives obtained from the suggested sequential method can be considered good enough.

This research can be extended in several ways. First, one can develop an efficient implicit enumeration method to find the best configuration by using the configuration obtained from the heuristics as a starting configuration. If the heuristic solution is used as an upper bound, time required to search for the best solution can be reduced, because many solutions can be fathomed by the upper bound. Also, one might examine another type of design problems, of which the objective is to maximize the total profit by considering equipment costs, inventory holding costs, and selling price of the products, etc. This type of design problems can be solved by a solution method similar to the heuristics suggested in this paper.

Acknowledgements

This research was supported by the Korea Science and Engineering Foundation under grant 961-1012-095-1.

References


