Heuristics for Selecting Machine Types and Determining Buffer Capacities in Assembly/Disassembly Systems

Keun-Chae Jeong and **Yeong-Dae Kim**

Department of Industrial Engineering
KAIST, Yusong-gu, Daejon 305-701, Korea

Abstract

We deal with a design problem of assembly/disassembly (AD) systems with finite buffer capacities where the times between failures, the times to repair, and the processing times are exponentially distributed with different parameter values. We present a solution procedure for finding the minimum cost configuration which achieves a desired throughput rate for an AD system. The configuration is defined by the types of machines to be used and capacities of buffers in the AD system. Results of computational experiments on randomly generated test problems show that the proposed heuristics give relatively good configurations in a reasonable amount of time.

1. Introduction

We consider a design problem for assembly/disassembly (AD) systems. The design problem considered in this paper is the problem of selecting machine types and determining buffer capacities so that a given desired throughput rate for the AD system can be achieved with the minimum cost.

Most of previous research on the design of AD systems deals with the problems of buffer allocation assuming that the decisions for machines are already made. Such research 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 (Park 1993). On the other hand, several algorithms have been devised for problems in which decisions for buffers and machines and/or pallets are considered at the same time. Solution of such design problems requires a tool for performance evaluation. Some use queuing theories for the analysis of the performance of AD systems, others propose algorithms based on decomposition methods. See Dallery and Gershwin (1992) and Gershwin (1994) for comprehensive reviews.

2. The design problem of AD systems

AD systems consist of machines in which assembly and/or disassembly operations take place and buffers that connect the machines. This paper focuses on AD systems in which sizes of buffers are finite, and the times between failures and the times to repair for the machines and the processing times of the operations are exponentially distributed.

We consider tree-structured AD systems, in which any two machines are connected by exactly one sequence of machines and buffers. It is assumed that input machines (machines with no upstream buffers) are never starved and output machines (machines with no downstream buffers) are never blocked.

The goal of the design of the AD system is to find the optimal system configuration. A *configuration* is defined by the types of machines to be used and the capacities of the buffers. The former defines a *machine configuration* and the latter defines a *buffer configuration*.

We first give notation used in this paper.

- N — number of machines
- $E_{\min}$ — desired system throughput rate
- $\mathbf{a}$ — vector $(a_1, a_2, \ldots, a_N)$, where a_i denotes type of machine 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 vectors $\mathbf{a}$ and $\mathbf{b}$
- n_i — number of alternative types for machine 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
- C_B — buffer cost per unit time
- $E(\mathbf{a}, \mathbf{b})$ — throughput rate associated with $(\mathbf{a}, \mathbf{b})$
- $Z(\mathbf{a}, \mathbf{b})$ — cost associated with $(\mathbf{a}, \mathbf{b})$

Using the above notation, the design problem can be mathematically stated as follows.

$$\text{Minimize } Z(\mathbf{a}, \mathbf{b}) = \sum_{i=1}^{N} C_M(a_i) + C_B \sum_{j=1}^{N-1} b_j$$

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

$1 \leq a_i \leq n_i$ and integer, for $i = 1, 2, \ldots, N$,

$l_j \leq b_j \leq u_j$ and integer, for $j = 1, 2, \ldots, N-1$.

The machine and buffer costs include both acquisition and operation costs. Here, the acquisition costs can be set to the equivalent uniform cash flow amounts of the costs discounted for the economic life of the machines or the buffers. Because of space restrictions or other economic and technological constraints, the capacities of buffers are assumed to be bounded by given lower and upper limits (**l** and **u**).

In this paper, it is assumed that the following relationships in costs and efficiencies hold for each machine,

$$C_M(1) \leq C_M(2) \leq \ldots \leq C_M(n_i),$$

$$\frac{\mu_1\rho_1}{\mu_1+\lambda_1} \leq \frac{\mu_2\rho_2}{\mu_2+\lambda_2} \leq \ldots \leq \frac{\mu_{n_i}\rho_{n_i}}{\mu_{n_i}+\lambda_{n_i}},$$

where ρ_k, λ_k, and μ_k denote the processing, failure, and repair rates of the type-k machine, respectively. That is, machine types are numbered in increasing order of the costs and a more expensive machine has higher efficiency than a cheaper one. It is also assumed that the throughput rate of the AD system is concave.

3. Solution methods

To solve the design problem defined in this paper, the throughput rate of the AD system should be estimated to check whether a configuration gives the desired throughput rate or not. AD systems with a small number of stations can be analyzed exactly using Markov chain models, but most of real AD systems cannot be because there exist an extremely large number of system states. In this paper, throughput rates of the AD systems are computed by the algorithm of Jeong and Kim (1995), which is based on decomposition methods. Using the failure, repair, and processing rates of the machines and the capacities of the buffers in the system, the algorithm gives a good estimate for the throughput rate in short time.

It takes very long time to evaluate all possible configurations since there are many alternative configurations. Therefore, we propose heuristics which can give good configurations in a reasonable amount of time.

Finding the best buffer configuration for a given machine configuration

We present an algorithm for finding a buffer configuration which requires the minimum capacity of buffers among those that give a desired throughput rate. Let **a** denote the given machine configuration and $\mathbf{e}_i$ 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 $E(\mathbf{a}, \mathbf{u}) < E_{\min}$, stop. Feasible buffer configurations do not exist. If $E(\mathbf{a}, \mathbf{l}) \geq E_{\min}$, stop. The best buffer configuration is **l**. Otherwise, let $\mathbf{b} = \mathbf{l}$.

Step 2. Let $\mathbf{b} = \mathbf{b} + \mathbf{e}_{j_{\max}}$, where $j_{\max} = \text{argmax}\ \{E(\mathbf{a}, \mathbf{b}+\mathbf{e}_j) - E(\mathbf{a}, \mathbf{b}) \mid b_j < u_j\}$.

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

Step 4. Let $\mathbf{b} = \mathbf{b} - \mathbf{e}_{j_{\min}}$, where $j_{\min} = \text{argmin}\ \{E(\mathbf{a}, \mathbf{b}) - E(\mathbf{a}, \mathbf{b}-\mathbf{e}_j) \mid b_j > l_j\}$.

Step 5. Let $\mathbf{b}^* = \mathbf{b}$. For all i and j such that $b_i < u_i$, $b_j > l_j$, and $j \neq i$, do: If $E(\mathbf{a}, \mathbf{b} + \mathbf{e}_i - \mathbf{e}_j) > E(\mathbf{a}, \mathbf{b}^*)$, let $\mathbf{b}^* = \mathbf{b} + \mathbf{e}_i - \mathbf{e}_j$.

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

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

Lower and upper configurations

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. We define $\Delta_i^+(\mathbf{a}, \mathbf{b})$ and $\Delta_i^-(\mathbf{a}, \mathbf{b})$ which denote the positive and negative gradients, respectively, as follows.

$$\Delta_i^+(\mathbf{a}, \mathbf{b}) = \begin{cases} \dfrac{E(\mathbf{a}+\mathbf{e}_i, \mathbf{b}) - E(\mathbf{a}, \mathbf{b})}{C_M(a_i+1) - C_M(a_i)} & \text{for } a_i < n_i \\ 0 & \text{for } a_i = n_i \end{cases}$$

$$\Delta_i^-(\mathbf{a}, \mathbf{b}) = \begin{cases} \dfrac{E(\mathbf{a}, \mathbf{b}) - E(\mathbf{a}-\mathbf{e}_i, \mathbf{b})}{C_M(a_i) - C_M(a_i-1)} & \text{for } a_i > 1 \\ \infty & \text{for } a_i = 1 \end{cases}$$

We use a lower configuration $(\mathbf{a}^L, \mathbf{b}^L)$ and an upper configuration $(\mathbf{a}^U, \mathbf{b}^U)$ which are obtained using the following algorithms, LC and UC, respectively.

Algorithm LC

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

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

Step 2. Let $\mathbf{a} = (a_1, a_2, ..., a_{i^+} + 1, ..., a_N)$, where $i^+ = \arg\max\{\Delta_i^+(\mathbf{a}, \mathbf{u})\}$. Go to step 1.

Algorithm UC

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

Step 2. Let $\mathbf{a} = (a_1, a_2, ..., a_{i^-} - 1, ..., a_N)$, where $i^- = \arg\min\{\Delta_i^-(\mathbf{a}, \mathbf{l})\}$.

Step 3. If $E(\mathbf{a}, \mathbf{l}) < E_{\min}$, stop. The upper configuration is $(a_1, a_2, ..., a_{i^-} + 1, ..., a_N, \mathbf{l})$. Otherwise, go to step 2.

<u>*Heuristics for the design problem*</u>

For solving the design problem, we present three heuristics and an improving heuristic.

Algorithm H1

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

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

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

Algorithm H2

Step 1. If $E(1, 1, ..., 1, \mathbf{l}) \ge E_{\min}$, stop. The best configuration is $(1, 1, ..., 1, \mathbf{l})$. If $E(n_1, n_2, ..., n_N, \mathbf{u}) < E_{\min}$, 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 $\mathbf{a}_i = (a_1, a_2, ..., a_i+1, ..., a_N)$, 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 $(\mathbf{a}, \mathbf{b})$.

Algorithm H3

Step 1. If $E(1, 1, ..., 1, \mathbf{l}) \ge E_{\min}$, stop. The best configuration is $(1, 1, ..., 1, \mathbf{l})$. If $E(n_1, n_2, ..., n_N, \mathbf{u}) < E_{\min}$, 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 $\mathbf{a}_i = (a_1, a_2, ..., a_i-1, ..., a_N)$, 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. The solutions from the above heuristics can be improved using the following algorithm.

Algorithm IM

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

Step 1. Let $\mathbf{a}^+ = (a_1, a_2, ..., a_{i^+} + 1, ..., a_N)$, where $i^+ = \arg\max\{\Delta_i^+(\mathbf{a}, \mathbf{b})\}$. Find the best buffer configuration $\mathbf{b}^+$ for $\mathbf{a}^+$ using algorithm BA, and let $Z^+ = Z(\mathbf{a}^+, \mathbf{b}^+)$.

Step 2. Let $\mathbf{a}^- = (a_1, a_2, ..., a_{i^-} - 1, ..., a_N)$, where $i^- = \arg\min\{\Delta_i^-(\mathbf{a}, \mathbf{b})\}$. Find the best buffer configuration $\mathbf{b}^-$ for $\mathbf{a}^-$ using algorithm BA, and let $Z^- = Z(\mathbf{a}^-, \mathbf{b}^-)$.

Step 3. Let $\mathbf{a}^\pm = (a_1, a_2, ..., a_{i^+} + 1, ..., a_{i^-} - 1, ..., a_N)$. Find the best buffer configuration $\mathbf{b}^\pm$ for $\mathbf{a}^\pm$ using algorithm BA, and let $Z^\pm = Z(\mathbf{a}^\pm, \mathbf{b}^\pm)$.

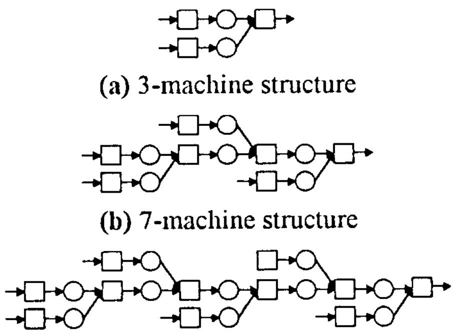
Step 4. If $Z^* = \min\{Z^+, Z^-, Z^\pm\} < Z$, let $Z = Z^*$ and $(\mathbf{a}, \mathbf{b}) = (\mathbf{a}^*, \mathbf{b}^*)$, where $(\mathbf{a}^*, \mathbf{b}^*)$ is the configuration corresponding to Z^*, and go to step 1. Otherwise, stop. The solution is $(\mathbf{a}, \mathbf{b})$.

4. Computational experiments

Performance of the proposed heuristics was tested on randomly generated test problems. Three types of the structure of the AD systems were used in the problems (Figure 1). The following eight methods were included in the comparison.

Methods 1, 2, and 3: Heuristics 1, 2, and 3, respectively
Method 4: Full enumeration method (for selecting configurations) using the decomposition algorithm for performance evaluation
Methods 5, 6, and 7: These methods are the same as Heuristics 1, 2, and 3, respectively, except that simulation is used for performance evaluation
Method 8: Full enumeration method using simulation for performance evaluation

The above eight methods were tested on the problems of the 3-machine structure only. For the comparison, ten problems were randomly generated. The test was done on a personal computer with a Pentium processor.



(a) 3-machine structure

(b) 7-machine structure

(c) 11-machine structure

Figure 1. Structures of the system included in the test

The results are summarized in Table 1. In all problems tested, Methods 1 and 2 (H1 and H2) always 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 average errors (differences in the total costs) of configurations obtained from the heuristics were less than 1% compared with those obtained from Method 8. Most of these errors came from the error of the decomposition method.

Table 1. Results of the problems of the 3- machine structure

Method	1	2	3	4	5	6	7	8
RDP†	0.64	0.64	0.86	0.64	0	0	0.34	0

† RDP denotes the relative deviation percentage of the objective function value of each method from that of Method 8.

Ten problems are tested for each of the 7- and 11-machine structures. The results are summarized in Table 2. Out of 20 test problems, Heuristics 1, 2, and 3 gave the best solution in 18, 19, and 15 problems, respectively. The average computation times for Heuristics 1, 2, and 3 were 0.84, 2.34, and 3.82 hours, respectively, for the problems of the 7-machine structure, and 1.84, 8.06, and 15.89, respectively, for those of the 11-machine structure. The full enumeration method required much longer computation time (12.94 hours for problems with 7 machines and 95.52 hours for those with 11 machines).

Table 2. Results of the problems of the 7- and 11-machine structures

Structure	7-machine				11-machine			
Method	H1	H2	H3	F†	H1	H2	H3	F†
RDP‡	0.01	0.00	0.34	0	0	0	0	0

† F denotes the full enumeration method.
‡ RDP denotes the relative deviation percentage of the objective function value of each method from that of Method F.

5. Concluding remarks

We considered the problem of finding the minimum cost configuration of an AD system for a given desired throughput rate. Three heuristics were suggested for the problem. 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 AD system but also to many other manufacturing systems in which machine types and buffer capacities should be determined, if appropriate performance evaluation tools are available. This research can be extended to development of an efficient implicit enumeration method to find the best configuration by using the configuration obtained from the heuristics as a starting configuration.

References

Dallery, Y. and Gershwin, S. B., 1992, Manufacturing Flow Line Systems: a Review of Models and Analytical Results. *Queueing Systems*, **12**, 3-94.

Gershwin, S. B., 1994, *Manufacturing Systems Engineering* (Englewood Cliffs, NJ: Prentice-Hall).

Jeong, K-C. and Kim, Y-D., 1995, Performance Analysis of Assembly/Disassembly systems with Unreliable Machines and Random Processing Times. Technical Report, Department of Industrial Engineering, Korea Advanced Institute of Science and Technology, Yusong-gu, Daejon 305-701, Korea.

Park, T., 1993, A Two-Phase Heuristic Algorithm for Determining Buffer Sizes of Production Lines. *International Journal of Production Research*, **31**, 613-631.