# An Integrated Algorithm Method to Optimize Resource Allocation with a Case Study of Production Line

Chan-Wei Jao, Ya-Yu Dong, Yu-Sheng Lee, Tzu-Mei Liao, Ming-Chuan Chiu
*National Tsing Hua University*, *Taiwan*

**Abstract.** There is a continuous flow performed one product unit that is routed from process to process in a production line. It is an important issue to figure out how to allocate the optimal resource and buffers in a production line for increasing efficiency and reducing costs. However, using Simplified Swarm Optimization (SSO) to solve this problem is easy to be inflected by initial solutions. This study tries to find the optimal configuration in a production line. This study utilizes the integrated method, combing SSO with Simulated Annealing (SA) to find the best scheme efficiently. Then, compare the results by using SSO and SA to the results by using SSO only. The estimate result is the integrated method is better than using SSO only. The results will be verified by simulation in the end. The production line is consisting of four workstations, three finite-size buffers (queues), and an infinite supply of blank parts. This study wants to find the best scheme of numbers of machines and buffers for maximizing the profits. According to the analysis results, this study has shown that integrated method combining SSO with SA is better than SSO with this case study. It is expected that it could find the optimal solution more efficiently.

**Keywords.** Simulated Annealing (SA), Simplified Swarm Optimization (SSO)

## 1. Introduction

There is a continuous flow performed one product unit that is routed from process to process in a production line. Continuous flow means the materials being processed are continuously in motion, undergoing chemical reactions or subject to mechanical or heat treatment in a production line, and it can operate without a shutdown. Contrasted to batch production, continuous flow reduces waste problem in a traditional production line. However, how to the optimize resources allocation appropriately is still an important issue for us to discuss. As a result, this study is aim to figure out the optimal resources allocation and buffers in a production line for increasing efficiency and reducing costs.

Therefore, to optimize resources allocation in a production line, the aim of this study is to find the optimal configuration in a production line. That is, our target is to figure out how to allocate machines and buffers in a production line for increasing profits. This study is conducted as following steps. This study utilizes the integrated method, combining Simplified Swarm Optimization (SSO) which was originally proposed by Yeh [1] in 2009 with Simulated Annealing (SA) which was proposed by Metropolis et al [18] to find the better scheme for maximizing the profits. Then, compare the effect of the results by using the integrated method to the results by using SSO only. After analysis, the results will be verified by simulation in the end. This study expects to find the optimal solution more efficiently.

The paper is organized as follows. In chapter 2, this study discusses the literature review. Chapter 3 illustrates the methodology and the framework of this study. The case study of production line is discussed in chapter 4. Analysis and discussion are described in chapter 5. Conclusions and potential research issues for future study are given in chapter 6.

## 2. Literature Review

### *2.1 Simplified Swarm Optimization*

SSO is also called the discrete PSO (DPSO), an emerging, evolutionary, population-based, and stochastic optimization algorithm method utilized to compensate for the drawbacks of Particle Swarm Optimization (PSO) in solving discrete problems. SSO is exploited to improve the process of individual update with results of classifying the breast cancer data more efficiently and effectively. The SSO scheme improves the update mechanism, which is the core of soft computing based methods, and revises the self-adaptive parameter control procedure [2].

Empirical results have revealed that SSO has better convergence rate and higher quality solution than PSO by the simulation results [3,4]. Recently, SSO has attracted considerable attention and has been applied to different problems and fields widely. SSO was extended to general problems including continuous problems [5], breast cancer pattern [1], gene selection [6], no-wait flow-shop scheduling problem [7], and uncapacitated facility location problem [8]. To improve the performance and overcome the deficiencies of PSO, the Exchange Local Search (ELS) strategy is utilized with SSO to find a better solution from the neighborhood of the current solution which is produced by SSO [9]. Compared to PSO, SSO is suitable for solving discrete problems. The update mechanism of SSO is simple. However, this algorithm is easily reflected by initial solution.

### *2.2 Simulated Annealing*

The method for solving combinatorial optimization problems was first demonstrated by Kirkpatrick et al [19], and Cˇerný [20], independently. SA is a method for combinatorial optimization problems, such as minimizing functions of a lot of variables. SA describes a group of heuristic optimization techniques based on iterative improvement, and it is motivated by an analogy to the statistical mechanics of annealing in solids. This algorithm accepts all solutions which improve the objective function generally, while those which do not result in improvements may be accepted with non-zero probabilities [19].

In condensed matter physics, the process of SA consists of the following two steps. The first one is to increase the temperature of the heat bath to a maximum value at which the solid melts. The second one is to decrease carefully the temperature of the heat bath until the particles arrange themselves in the ground state of the solid.

When SA is utilized to solve problems, it refers to the search process to find each one can be Line solution (Feasible Solution). This study exploits Boltzman’s Function to decide whether to accept solutions. Recently, SA has many successful applications in many fields, including travelling salesman problem [20], global wiring [21], cluster problem [22], Job Shop Scheduling [23], and X-ray crystallography and solution NMR [24]. However, even SA is relatively easy to code and it can find an optimal solution statistically, but this algorithm cannot identify whether it has found an optimal solution and it requires other complimentary methods (e.g. branch and bound) to compensate for it.

### *2.3 Comparison of SSO and SA*

| | Advantage | Disadvantage |
|---|---|---|
| **SSO** | (1) It is suitable for solving discrete problems.<br>(2) Update mechanism is simple. | (1) It is easily influenced by initial solution. |
| **SA** | (1) It is relatively easy to code, even for complex problems.<br>(2) It can deal with arbitrary systems and cost functions.<br>(3) The implement time is short. | (1) The method cannot tell whether it has found an optimal solution. Some other complimentary method (e.g. branch and bound) is required to do this. |

Table1. Comparison of SSO and SA

Each of SSO and SA has its own advantages and disadvantages. It is expected that integrated these two algorithms to optimize resource allocation and maximize the profits. The estimate result with integrated method is better than the result with SSO.

## 3. Methodology

Because the uncertainty exists in the problem and many local optimal solutions exist, some simulation optimization addresses the limitation of not simulating all feasible solutions. This study develops a simulation optimization SSO in the first section, and integrates SA into SSO in the second section. The mathematical model is developed in the third section. The results indicate that the fusion of SA and SSO performs better than SSO with this case study.

### 3.1 *Simulation Optimization of SSO*

The simulation optimization of SSO can be used on the problem of discrete stochastic resource allocation. In SSO procedures, the population size, the terminating condition, and three pre-specified parameters need to be determined initially. Then, the particle's position will be kept or be updated by its pbest value or be updated by the gbest value or be replaced by new random value in every generation, according to this procedure.

In this equation, $i = 1, 2, \ldots, m$, where m is the population size. $Xi = (xi1, xi2,\ldots,xiD)$, where $xiD$ is the position value of the i-th particle with respect to the D-th dimension of the feature space. $Cw$, $Cp$ and $Cg$ are three predetermined positive constants with $Cw < Cp < Cg$. $Pi$, pbest, denotes the best solution achieved so far by itself. $Gi$, gbest, is the best solution achieved so far by the whole particle. x represents the new value for the particle in every dimension which are generated randomly.

### 3.2 I*ntegrating SA into SSO*

With the design of the SA combination with SSO is to select the global best among the set of positions more efficiently. The SA combination with SSO algorithm is stated as follows, and the procedure is shown in Fig1.

*Step 1:* Parameter setting for SA, SSO. For SA, the parameters are initial temperature (*T0*), the cooling rate (α), an integer temperature length *(K)*, and the frozen condition (the terminal condition for SA).
*Step 2:* Initiate population and evaluate the performance or fitness with SA.
*Step 3:* Update the best solution in initial solution.
*Step 4:* Pick a random neighbor x' of x, where x represents a scheme.
*Step 5:* Compute δ, where δ = f(x')- f(x).
*Step 6:* If $\delta > 0$, then x is updated by x'. If $\delta < 0$, then calculate PR(A). PR(A) = min $\{1, e^{(-\delta/T_k)}\}$, where $T_k$ is the current temperature. If PR(A) ≥ R, where R is generated randomly between 0 and 1, then we accept the scheme. Otherwise, we reject the scheme.
*Step 7:* Update $T_k$ by the formulation: $T_{k+1} = \alpha T_k$.
*Step 8:* If it reached the frozen condition, then we go to Step 9; otherwise, we go back to step 4.
*Step 9:* Determine the initial particle and generate population for SSO.
*Step 10:* Evaluate the performance or fitness of the population.
*Step 11:* Record Pbest and Gbest.
*Step 12:* Update particles position according to Eqs.(1).
*Step 13:* Evaluate the performance.
*Step 14:* Update Pbest and Gbest.
*Step 15:* If it reached the terminal condition, then we go to Step 16. Otherwise, we go back to step 12.
*Step 16:* The algorithm is stopped.

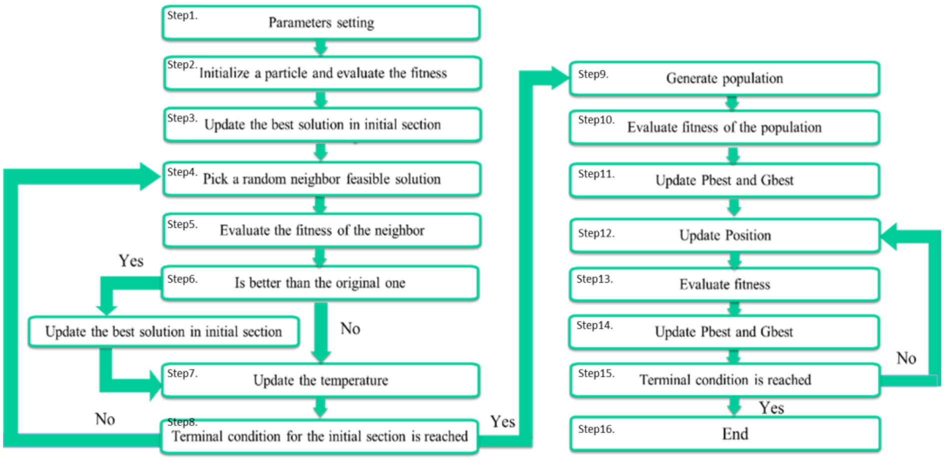


Fig1. The flowchart of SA combination with SSO algorithm.

### *3.3 The Mathematical Model*

In the study, the problem can be described as four parts and three buffers, selecting the amount of the stations and the buffers so as to maximize the profit:
Notations

$W_i$ number of machine in workstation *i*, $1 \le i \le m$.
$B_j$ size of capacity in buffer *j*, $1 \le j \le n$ .

$TH$ throughput of the production line
$M_c$ cost of operating for each machine.
$B_c$ cost of buffer for each capacity.
$P$ price of each product

Objective function:

$$Max\ Profit = P*E[TH]-M_c \sum_{i=1}^{m} W_i - B_c \sum_{j=1}^{n} B_j$$

Subject to：

$$1\leq W_i \leq 3, \quad \forall i$$
$$1\leq B_j \leq 10, \quad \forall j$$

## 4. Case Study

This study is discussing the production line is consisting of four parts, three finite-size buffers (queues), and an infinite supply of blank parts. This concept was originally proposed by Papadopoulos [17], the paper proposes a $K$-station line consists of $K$ machines in series, labeled…, and $K$1 locations for buffers, labeled …, and it is applied to production lines. This study utilizes the concept to the production line, and it is shown as Fig 2. This study wants to find the best scheme of numbers of machines and buffers for maximizing the profits.

We assume that the cost of adding a buffer capacity is $5,000, including venue cost and management cost. The cost of adding a machine is $25,000, including maintenance cost and labor cost. Revenue of selling a product is $100. Materials arrival follow exponential distribution which $\lambda$ = 5min, and the process time of workstation 1 follow exponential distribution which $\lambda$ = 20min, the process time of workstation 2 follow exponential distribution which $\lambda$ = 30min, the process time of workstation 3 follow exponential distribution which $\lambda$ = 12min, and the process time of workstation 4 follow exponential distribution which $\lambda$ = 15min.

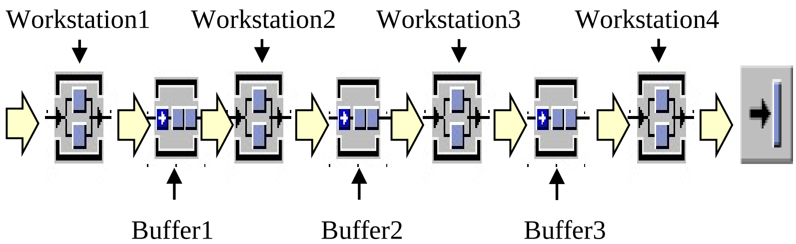


Fig 2. Production Line

*A. Objective Function*

$$Max\ Profit = P*E[TH]-M_c \sum_{i=1}^{m} W_i - B_c \sum_{j=1}^{n} B_j$$

*B. Constraints*

$$1\leq W_i \leq 3, \quad 1 \leq i \leq 4, \ i \in N .$$
$$1\leq B_j \leq 10, \quad 1 \leq j \leq 3, \ j \in N .$$

*C. Parameter*

$M_c$=25000
$B_c$=5000
$P$ =100

## 5. Analysis

SSO is a numerous particle meta-heuristic algorithm, so its searchable solution space is relatively larger. While SA is a single particle meta-heuristic algorithm, its searchable solution space is smaller compared to SSO. The variation of SA is also larger than SSO. Therefore, using the characteristics of Simulated Annealing Algorithm, the updating process is rapid, to find a particle with a better fitness function value with a small number of iterations. And then the particles become the initial solution of SSO, to ensure that in the iterative process, the particles can move more quickly to the optimal search space. As features of SSO search range is large, it can compensate for the shortcomings of simulated annealing, which can be easy to fall into the local optimal solution.

In this section the results of iteration begins to converge was compared by SA + SSO and SSO methods, as Fig 3 shown. We also compare the simulation time between the two-methods.

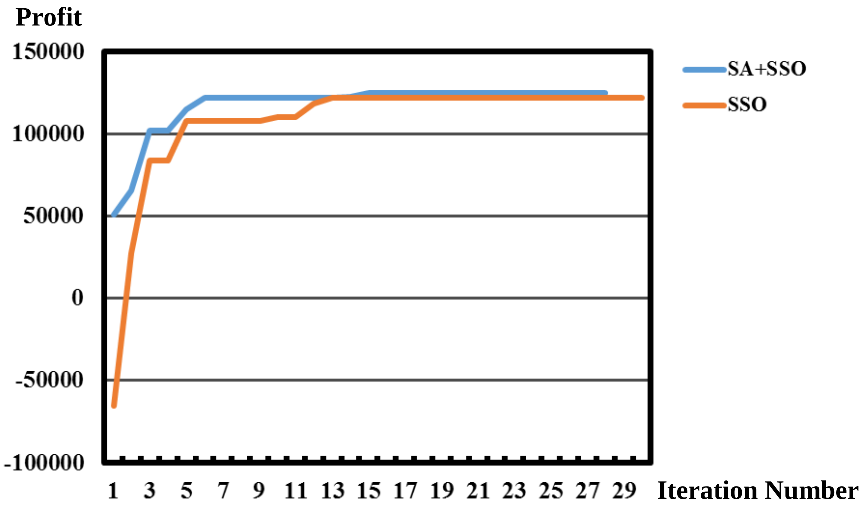


Fig 3. Optimal solution for each iteration

SA+SSO was converged at iteration at 7, and SSO was converged at 13. We can observe that SA+SSO is more efficient. In this issue, Applying SA to initialize solution is an effective method. This means that SA + SSO can find the same quality solution with fewer simulation resources.

Besides, because SA is a heuristic algorithm for single particles, the expected computing time is shorter than SSO. The comparison of simulation time of two results is shown in Table 2. In the same simulation number, the simulation time of SA + SSO is better than SSO.

| Method | SA+SSO | SSO |
|---|---|---|
| **Simulation time(sec)** | 8.142 | 10.83 |

Table 2. Simulation time of two methods

For generating the initial solution, the integrated method can search in feasible region more efficiently, and avoid generating initial solution randomly which will waste simulation resources to search in bad solution space.

This study compares convergence generation and time by these two methods individually. According to the results, no matter the convergence generation or time, the integrated method has shown the better results. As expected at first, the integrated method is better than using SSO only.

Through the integrated method combing SSO with SA, the optimal allocation of number of machines and buffers are figured out as Table 3.

| Variable | $W_1$ | $W_2$ | $W_3$ | $W_4$ | $B_1$ | $B_2$ | $B_3$ |
|---|---|---|---|---|---|---|---|
| **Number(unit)** | 3 | 3 | 2 | 2 | 2 | 3 | 1 |

Table 3. Optimal solution for resource allocation

**5.1 Discussion**

This study conducts sensitivity analysis in the bottleneck workstation and its previous and following buffer. The optimal size of capacity for Buffer 1 is two. For Buffer 1, when we add one more buffer or reduce the size of capacity to one, the profit decreases, as Fig.4 shown. The optimal size of capacity for Workstation 2 is three. When the size of capacity decreases to two, the profit becomes worse. Add the size of capacity to four, and the profit becomes better. However, four machines for Workstation 2 is not in our feasible solution, so the optimal size of capacity for Workstation 2 in our problem is three, as Fig.5 shown. For Buffer 2, the optimal size of capacity is three. The size of capacity for Buffer 2 becomes more or fewer, and the profit is worse, as Fig.6 shown.

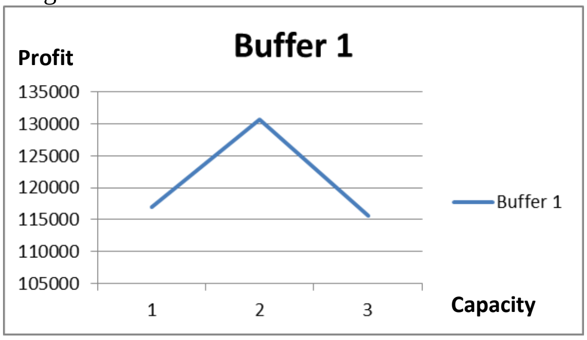


Fig.4. The size of capacity for Buffer 1.

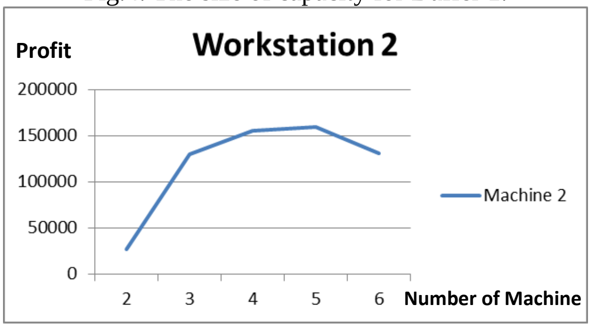


Fig.5. The size of capacity for Workstation 2.

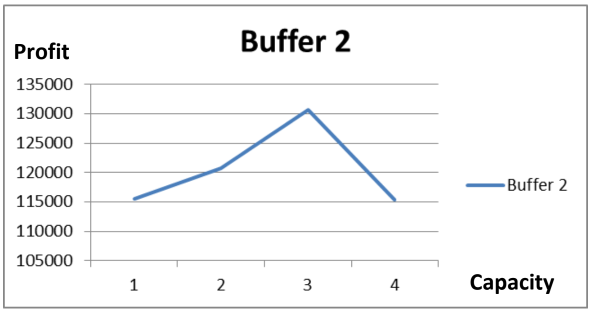


Fig.6. The size of capacity for Buffer 2.

## 6. Conclusion

This integrated method overcomes the drawbacks of SSO, which is easily influenced by the initial solution. This study generates the initial solution with SA, which is a fast searching algorithm. As a result, this study utilizes the integration method combing SSO and SA, proving that it could find the optimal solution more efficiently using lower simulation resources and time with this case study. Although the improvement range is not significant, the integrated method is expected to be used in the problems with larger feasible solution region, and obtain more significant effects with it.

## 7. Reference

[1] Yeh, W.-C., Chang, W.-W., & Chung, Y. Y. (2009). A new hybrid approach for mining breast cancer pattern using discrete particle swarm optimization and statistical method. *Expert Systems with Applications*, *36*(4), 8204–8211. doi:10.1016/j.eswa.2008.10.004

[2] Yeh, W.-C. (2012). Simplified swarm optimization in disassembly sequencing problems with learning effects. *Computers & Operations Research*, *39*(9), 2168–2177. doi:10.1016/j.cor.2011.10.027

[3] Kennedy, J., & Eberhart, R.-C. (1995). Particle swarm optimization. *Proceedings of IEEE International Conference on Neural Networks*. Piscataway, NJ. pp. 1942-1948.

[4] Kennedy, J., & Eberhart, R.-C. (1997). A discrete binary version of the particle swarm algorithm. *Proceedings of the World Multiconference on Systemics,Cybernetics and Informatics*. Piscataway, NJ. pp. 4104-4109

[5] Yeh, W.-C. (2009). A two-stage discrete particle swarm optimization for the problem of multiple multi-level redundancy allocation in series systems. *Expert Systems with Applications*, *36*(5), 9192–9200. doi:10.1016/j.eswa.2008.12.024

[6] Lai, C.-M., Yeh, W.-C., & Chang, C.-Y. (2016). Gene selection using information gain and improved simplified swarm optimization. *Neurocomputing*, *218*, 331–338. doi:10.1016/j.neucom.2016.08.089

[7] Pan, Q.-K., Fatih Tasgetiren, M., & Liang, Y.-C. (2008). A discrete particle swarm optimization algorithm for the no-wait flowshop scheduling problem. *Computers & Operations Research*, *35*(9), 2807–2839. doi:10.1016/j.cor.2006.12.030

[8] Guner, A.-R., & Sevkli, M. (2008). A discrete particle swarm optimization algorithm for Uncapacitated facility location problem. *Journal of Artificial Evolution and Applications*, *2008*, 1–9. doi:10.1155/2008/861512

[9] Bae, C., Yeh, W.-C., Wahid, N., Chung, Y.-Y., & Liu, Y. (2012). A new simplified swarm optimization (SSO) using exchange local search scheme.

*International Journal of Innovative Computing, Information and Control, 8*(6), 4391–4406.

[10] Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., & Teller, E. (1953). Equation of state calculations by fast computing machines. *The Journal of Chemical Physics, 21*(6), 1087. doi:10.1063/1.1699114

[11] Kirkpatrick, S., Gelatt, C.-D., & Vecchi, M.-P. (1983). Optimization by simulated Annealing. *Science, 220*(4598), 671–680. doi:10.1126/science.220.4598.671

[12] Černý, V. (1985). Thermodynamical approach to the traveling salesman problem: An efficient simulation algorithm. *Journal of Optimization Theory and Applications, 45*(1), 41–51. doi:10.1007/bf00940812

[13] Vecchi, M.-P., & Kirkpatrick, S. (1983). Global wiring by simulated Annealing. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 2*(4), 215–222. doi:10.1109/tcad.1983.1270039

[14] Selim, S.-Z., & Alsultan, K. (1991). A simulated annealing algorithm for the clustering problem. *Pattern Recognition, 24*(10), 1003–1008. doi:10.1016/0031-3203(91)90097-o

[15] van Laarhoven, P.-J.-M., Aarts, E.-H.-L., & Lenstra, J.-K. (1992). Job shop scheduling by simulated Annealing. *Operations Research, 40*(1), 113–125. doi:10.1287/opre.40.1.113

[16] Brünger, A.-T., Adams, P.-D., & Rice, L.-M. (1997). New applications of simulated annealing in x-ray crystallography and solution NMR. *Structure, 5*(3), 325–336. doi:10.1016/s0969-2126(97)00190-1

[17] Papadopoulos, H.-T., & Vidalis, M.-I. (2001). A heuristic algorithm for the buffer allocation in unreliable unbalanced production lines. *Computers & Industrial Engineering, 41*, 261-277.