symmetry

MDPI

Article

Optimizing the Tolerance for the Products with Multi-Dimensional Chains via Simulated Annealing

Chen-Kun Tsung

Department of Computer Science and Information Engineering, National Chinyi University of Technology, No.57, Sec. 2, Zhongshan Rd., Taiping Dist., Taichung 411030, Taiwan; ckt@ncut.edu.tw

Abstract: The assembly is the last process of controlling the product quality during manufacturing. The installation guidance should provide the appropriate assembly information, e.g., to specify the components in each product. The installation guidance with low quality results in rework or the resource waste from the failure products. This article extends the dimensional chain assembly problem proposed by Tsung et al. to consider the multiple dimensional chains in the product. Since there are multiple dimensional chains in a product, the installation guidance should consider inseparability and acceptability as computing the installation guidance. The inseparability means that the qualities of all dimensional chains in the part should be evaluated together without separation, while the acceptability stands for that the size of each product should be satisfied with the specification. The simulated annealing (SA) algorithm is applied to design the assembly guidance optimizer named as AGO_{MDC} to compute the assembly guidance in the dimensional chain assembly problem with multiple dimensional chains. Since SA has high performance in searching neighbor solutions, the proposed approach could converge rapidly. Thus, proposed AGO_{MDC} could be applied in real-world application for the implementation consideration. The simulations consist of two parts: the feasibility evaluation and the algorithm configuration discussion. The first part is to verify the inseparability and acceptability that are the hard constraints of the assembly problem for the proposed AGO_{MDC}, and the second one is to analyze the algorithm configurations to calculate the assembly guidance with 80% quality. The simulation results show that the inseparability and acceptability are achieved, while the proposed AGO_{MDC} only requires more than two seconds to derive the results. Moreover, the recommended algorithm configurations are derived for evaluate the required running time and product quality. The configurations with product quality 80% are that the temperature descent rate is 0.9, the initial temperature is larger than 1000, and the iteration recommended function is derived based on the problem scale. The proposed AGO_{MDC} not only helps the company to save the time of rework and prevent the resource waste of the failure products, but is also valuable for the automatic assembly in scheduling the assembly processes.

Keywords: dimensional chain analysis; tolerance optimization; simulated annealing

Citation: Tsung, C.-K. Optimizing the Tolerance for the Products with Multi-Dimensional Chains via Simulated Annealing. *Symmetry* **2021**, *13*, 1780. https://doi.org/10.3390/sym13101780

Academic Editors: Natalie Baddour and Sergei D. Odintsov

Received: 6 August 2021
Accepted: 17 September 2021
Published: 25 September 2021

1. Introduction

Owing to information and communications technologies (ICT), Industry 4.0 becomes the popular topic in manufacturing, information science, and industrial management [1,2]. The major technologies that are applied to realize Industry 4.0 include Internet of Things to collect data [3–5], cloud in storing and preparing data [6–8], and artificial intelligence [9,10] and data analysis [11,12] for extracting the knowledge from data. So, the producing managers could use the derived knowledge to enhance manufacturing efficiency.

Assembly is the last step of manufacturing, and it is also the last process of controlling the production quality [13]. After receiving some components and the installation guidance, the assemblers pick up the specific components and assemble the selected components together to the products or modules. The real size of each component would be different with that of the design size, so the tolerance is allowed in manufacturing [14]. However,

the tolerance determines the size of the products, and that is the product quality. We use the bearing shown in Figure 1 as the example to explain the product quality. The bearing should be installed in the bottom bracket of a bike, as shown in Figure 1a, and the composition diagram is illustrated in Figure 1b. Each bearing consists of one outer race, some balls, one retainer, and one inner race. Since the balls and retainer should be pre-installed, we treat the balls and retainer are assembled as one module. So, the assembler would pick up one outer race, one retainer, and one inner race to assemble a bearing. The bearings have two surfaces to touch bottom bracket of a bike and the crank axles. The bottom bracket touches the outer race, while the crank axles touch the inner race. The quality of bearings is determined by the sizes of outer race and inner race, and they are assembled with the retainer module. The selection of the components determines the final sizes of the products. Therefore, the assembly process would be the last step in controlling the product quality.

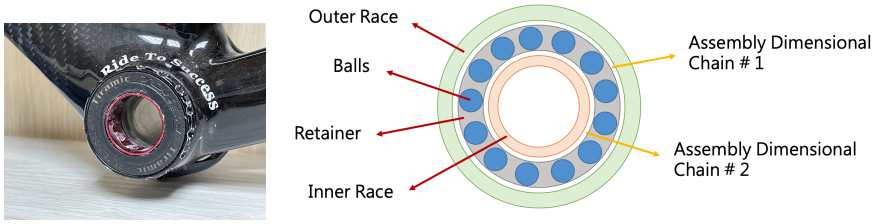

Figure 1. The example of the bearing for bike. (**a**) The bearing that is installed in the bike. (**b**) The composition diagram of a bearing.

If the assemblers arbitrarily pick up the components and assemble them together, the products may be out of the specification. Thus, the defect products require rework to fix the defect, and the products with poor quality will be abandoned. Inappropriate assembly process will increase the manufacturing cost. To increase the product quality and decrease the manufacturing cost, analyzing the dimensional chain to guarantee the size of each product [15,16] and computing the installation guidance to assemble appropriate components together [16,17] are feasible solutions. The dimensional chain is analyzed before manufacturing process and the installation guidance is calculated after manufacturing. Therefore, calculating the installation guidance is more appropriate than analyzing the dimensional chain from the perspective of the implementation.

From the above example, the product quality is determined by the components selected by assemblers. Quality control officers have to measure the component sizes one by one in the traditional manufacturing, and this manner requires huge cost. The ICT based services help measurers to collect the component sizes automatically [18,19], and the cost is reduced. Therefore, collecting all sizes of components becomes available and requires less cost than in traditional manufacturing.

Tsung et al. formulated the calculation of installation guidance as the dimensional chain assembly (DCA) problem, and proposed the assembly guidance optimizer (AGO) to calculate the installation guidance [17]. The DCA problem proposed by Tsung et al. considers the product that consists of 11 parts with one-dimensional chain. However, in the real-world instance, most products consist of some dimensional chains, but this property is not considered in the DCA. Considering that the bearing as shown in Figure 1, there are two dimensional chains, and they cross on the part retainer. Replacing the retainer to another one would results in different quality of both dimensional chains. Therefore, we modify the DCA problem to the dimensional chain assembly, with multiple dimensional chains problem named by DCA_{MDC} for involving the concept of multiple dimensional chains in this article. The major difference between single dimensional chain and multiple dimensional chains is to consider the inseparability of dimensional chains. Multiple dimensional chains may cross on one part, so replacing the part results in the different

quality of all corresponding dimensional chains. Therefore, the inseparability of multiple dimensional chains should be taken into account when evaluating the solution quality.

There is exactly one-dimensional chain in the DCA problem, so the solution quality is defined by the size of the dimensional chain. We extend this idea from the DCA problem to the DCA_{MDC} problem, but it fails. The solution quality comes from multiple dimensional chains in the DCA_{MDC} problem, so we have to normalize the sizes of all dimensional chains to the solution quality. According to the min-max theorem [20,21], minimizing the worst quality increases the solution quality for the whole solution. Thus, we consider the worse quality from all dimensional chains as the solution quality, so the inseparability could be satisfied.

Except for the inseparability of multiple dimensional chains, the solution feasibility is another critical issue. All product sizes are specified in the product design phase, and the designer would list the size information for components and sizes in the specification. The feasibility indicates that the quality of each dimensional chain should fit the requirements listed in the specification. To achieve the solution feasibility, we extend the AGO algorithm proposed by Tsung et al. [17] to the Assembly Guidance Optimizer with Multiple Dimensional Chains denoted by AGO_{MDC} to solve the DCA_{MDC} problem. The AGO_{MDC} algorithm is the application of the simulated annealing (SA) [22,23]. We proposed a new solution format, the local search algorithm, and the equation of calculating the maximum iteration to realize the solution with the approximation ratio 0.8.

In the simulation, we receive a batch of component sizes for assembling bearings from the assembly line, and the mission is to calculate the assembly guidance that the inseparability and the feasibility are satisfied. To increase the diversity of the dataset, we analyze the received size data to design the case generator, and we can generate synthetic data to cover more instance properties. From the first simulation results, the solution derived from the proposed AGO_{MDC} reaches both the inseparability and the feasibility. Moreover, AGO_{MDC} consume 562.6 milliseconds to derive the solution, so the running time is acceptable. The first simulation indicates that computing the feasible solutions by the proposed AGO_{MDC} is not difficult, and we organize the second simulation to analyze the algorithm configuration for ensuring the solutions with 80% quality. Therefore, the assembly manager could rapidly determine the algorithm configuration based on the problem properties, and the product quality is also guaranteed. From the simulation results, the configurations with product quality 80% are that the temperature descent rate is 0.9, the initial temperature is larger than 1000, and the iteration recommended function is derived based on the problem scale.

The major contributions of this article include: (1) formulating the DCA_{MDC} as the optimization problem for complex product assembly, (2) applying the SA to compute the installation guidance, and (3) proposing the algorithm configuration for the assembly manager for adjusting the proposed algorithm to compute the installation guidance with the guarantee of 80% product quality.

In the following article, some related works are discussed in Section 2. The DCA_{MDC} problem is defined in Section 3, and the example of assembling bearings would be given to explain the DCA_{MDC} problem. In Section 4, the main algorithm of AGO_{MDC} is present, and the details of the solution format and the solution quality are stated simultaneously. The simulation results and the application of AGO_{MDC} algorithm is discussed in Section 5, and the conclusion is in Section 6.

2. Related Works

The tolerance calculation helps to increase the product quality. Increasing purchasing provides high motivation in statistical tolerance calculations in Europe [24,25]. Moreover, in Europe, feature factories target at the term "Zero Defect Manufacturing" and set it as the milestone in 2020 roadmap [26]. We all know the importance of the tolerance calculation, but however, only a few companies apply the tolerance calculation in manufacturing in German from the analysis of Walter et al. [27]. In the past decade, collect all sizes of

each component is difficult, but it is realized easier currently due to the publication of the ICT applications, such as Internet of Things [28] and cloud computing [29]. So, more companies would like to consider the tolerance calculation, since the difficulty of data collection is eased.

In the assembly line, the assemblers receive an installation guidance, and they follow the instructions listed in the installation guidance to pick up the specify components and assemble the selected components together. The goal of the assembly problem is to determine the installation guidance to specify the components in a product such that the product quality fits the requirements of the specification. There are two manners to determine the installation guidance, and they are discussed as follows.

- The dimension chain analysis in the product design phase [15,16]: the properties of components are designed, including the ideal component size, the tolerance, and the size of touch surface. Then, the dimensional chain analysis is applied to estimate the product size. If the product size is unacceptable, the designer should modify the components until the product size is acceptable.
- The installation guidance analysis [17]: the installation guidance analysis analyzes the component sizes and determines the installation guidance to specific the components in each product. The major difference between the above manners is the analysis timing.

Above approaches could provide installation guidance, but the dimensional chain analysis takes place in the design phase, while the installation guidance analysis is executed after the manufacturing processes. The dimensional chain analysis provides precise prediction in product quality only if the quality of manufacturing processes is guaranteed. The installation guidance analysis considers real size information, and thus, the installation guidance analysis provides the installation guidance based on the real sizes of the components. Therefore, this article considers the installation guidance analysis.

The installation guidance analysis could be formulated as the optimization problem, and some heuristic algorithms could be applied to determine the installation guidance. Kim et al. used taboo search and SA to determine the swap of the assembly process to minimize mean tardiness [30]. Ge et al. proposed the homogeneous Markov chain of reassembly process for the assembly process of remanufactured crankshaft [31], and the term "reassembly" has been applied to enhance the product quality. Rausch et al. focused on easing the assembly rework to enhance the assembly efficiency, and the authors applied Monte Carlo simulation to identify potential misalignments [32]. Homri et al. consider the parts' form defects to the over-constrained assemblies [33]. Morse et al. mentioned that the variability should be allowed and controlled to maintain the product function [34]. Tsung et al. consider the SA to compute the installation guidance to maximize the quality for all products [17].

The DCA_{MDC} problem differs from the DCA problem that is proposed by Tsung et al. [17] in terms of the number of dimensional chains. The DCA problem consists of a one-dimensional chain, and the DCA_{MDC} problem has multiple dimensional chains. In the DCA_{MDC} problem, the multi-objective decision analysis [35–37] should be considered to transform multiple objectives to single objective. There are some techniques of deriving the single objective, and we consider the score normalization in the DCA_{MDC} problem. For the perspective of the feasibility, the size of each dimensional chain should be satisfied with the specification. Therefore, the importance of each dimensional chain is equal, and the score normalization is feasible in the DCA_{MDC} problem.

3. The Dimensional Chain Assembly with Multiple Dimensional Chains Problem

Before formulating the problem, some preliminaries are necessary, including the target of assembly, the installation guidance, the dimensional chains, and the symbolic system. Thus, the preliminaries are given in Section 3.1, the problem definition is in Section 3.2, and an example is provided to explain the problem and the goal in Section 3.3.

3.1. Preliminary

- Background of Assembly: Suppose the target product consists of $|P|$ parts. As the example shown in Figure 1 demonstrates, the bearings have three parts, i.e., $|P| = 3$, and they are outer race, retainer module, and inner race. The assembler picks up $|P|$ components where one per part and assembles $|P|$ selected components together to produce one product. In Figure 1, the assembler picks up three components—one outer race, one retainer module, and one inner race and then assembles them together. Given $|P|$ parts and that each part has $|T|$ components, the number of components that are delivered to the entrance of the assemble line is $|T| \times |P|$. So, the assembler selects $|P|$ components to produce one product and repeats $|T|$ times to finish $|T|$ products in total.
- Component Sizes: The label $com_{j,y}$ indicates y-th component of j-th part, where $0 \leq y < |T|$ and $0 \leq j < |P|$. The component $com_{j,y}$ has several component sizes, and $s_{j,y,z}$ is applied to represent the z-th component size of $com_{j,y}$. For example, the outer race in Figure 1b has two component sizes, and they are the outside diameter and inside diameter.
- Multiple dimensional chains: The major difference between DCA proposed by Tsung et al. [17] and DCA_{MDC} is the number of dimensional chains in a product. In the DCA, the product has exactly one-dimensional chain, e.g., crankshafts, but there are several dimensional chains in a product for DCA_{MDC}, e.g., bearings. Suppose each product consists of $|C|$ dimensional chains in DCA_{MDC}; it means that each product has $|C|$ product sizes. Thus, $|C| = 2$ in Figure 1b. Chain number one consists of outer race and retainer module, while chain number two consists of retainer module and inner race. Moreover, one part may belong to several chains. The retainer module shown in Figure 1 belongs to both dimensional chains.
- Product tolerance: Since each product has $|C|$ dimensional chains in DCA_{MDC}, there are $|C|$ product sizes in each product. According to the assembly position in the dimensional chain k, each component $com_{j,y}$ has three statuses: (1) $com_{j,y}$ contains other components, (2) $com_{j,y}$ is in other components, and (3) $com_{j,y}$ does not belong to the dimensional chain k. For the chain number one in Figure 1b, the retainer module is in the outer race, and the inner race is not in the chain number one. After assembling a product, $|C|$ product sizes are derived, and the label ps_k^i represents the product size for k-th chain in i-th product. Thus, the tolerance for the dimensional chain is the gap between ps_k^i and the ideal product size.
- Product specification: The product specification includes three values—the lower bound of tolerance, the central value, and the upper bound of tolerance for each dimensional chain. The central value is the ideal product size, while the product specification specifies the ranges of the acceptable products for the dimensional chain. Since the cutting process may result in the tolerance, the product designer provides the CAD size for each part with a specific range. Here, we assume that the sizes of all components are satisfied with the range requirement. The out-of-range components could be fixed by the rework, so the assumption is rational.
- The installation guidance: The installation guidance specifies the components of each product, so the assemblers could follow the guidance to assemble products. Since each product consists of $|P|$ components, a two-dimensional structure ($|T|$ by $|P|$) is used to represent the installation guidance for assembling $|T|$ products, as illustrated in Figure 2. The element ge_{rs} in the installation guidance represents the component index for r-th product with s-th part, so each row indicates the components of the product. The details of the installation guidance will be provided when introducing the proposed solution in Section 4.2.

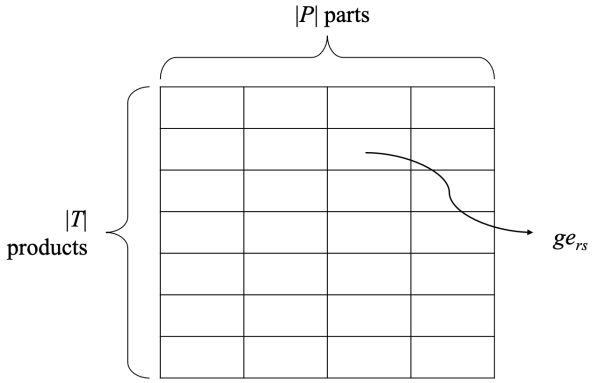

Figure 2. The structure of the installation guidance.

3.2. Definition of DCA_{MDC}

An instance of DCA_{MDC} consists of $|T| \times |P|$ components and $|C|$ dimensional chains, and $|T|$ products will be generated eventually. The y-th component in j-th part $com_{j,y}$ has several component sizes, and z-th component size is $s_{j,y,z}$. Each product has $|C|$ product sizes, and each product size ps_k^i matches to the corresponding dimensional chain. To calculate ps_k^i, the decision variable $x_{j,y,z}$ as shown in Equation (1) is applied to formulate the relationship between components in each dimensional chain. Consider that two components $com_{j,y}$ and $com_{j',y}$ where $j \neq j'$ with component sizes $s_{j,y,z}$ and $s_{j',y,z}$ for the same dimensional chain, respectively. If $com_{j',y}$ contains $com_{j,y}$ which means $com_{j,y}$ is in $com_{j',y}$, the size of the assemble surface between $com_{j,y}$ and $com_{j',y}$ is $(s_{j',y,z} - s_{j,y,z})$ that is identical to $(s_{j',y,z} \times 1) + (s_{j,y,z} \times -1)$. The decision variables of $com_{j,y}$ and $com_{j',y}$ are 1 and -1, respectively.

$$x_{j,y,z} = \begin{cases} -1 & \text{if } com_{j,y} \text{ is assembled in other part} \\ 0 & \text{if } com_{j,y} \text{ does not infolve in dimensional chain} \\ 1 & \text{if } com_{j,y} \text{ contains other components.} \end{cases} \quad (1)$$

Therefore, the product size ps_k^i for chain k could be derived in Equation (2). The idea of calculation is to sum up the product of component sizes and the corresponding decision variables for all possible permutations.

$$ps_k^i = \sum_{\forall j} \sum_{\forall y} \sum_{\forall_z} s_{j,y,z} \times x_{j,y,z}. \quad (2)$$

Given an instance of DCA_{MDC} and the product specification with the lower bound lb_k, the central value cv_k, and the upper bound ub_k for chain k, where $lb_k \leq 0$, $cv_k \in R$, and $0 \leq ub_k$, the goal of DCA_{MDC} is to calculate the installation guidance, so that each ps_k^i is within the lower bound of tolerance and the upper bound of tolerance from the specification for all dimensional chains. A feasible solution of DCA_{MDC} should be satisfied with the inseparability and acceptability:

- Inseparability: The multi-dimensional chains mean that the product consists of several dimensional chains, and some dimensional chains may cross on one part. Thus, the part that is crossed by some dimensional chains cannot be separated as evaluating the solution quality.
- Acceptability: The acceptability means that the tolerance of each dimensional chain must be satisfied with the specification. From the product specification, the range of acceptable tolerance is illustrated in Figure 3. The red line indicates the feasible tolerance range that the product sizes should fall into. The optimal solution indicates that ps_k^i is identical to cv_k for each i.

Figure 3. The relationship between the central value and the feasible ranges.

Computing the solution with inseparability and acceptability is the goal of solving DCA_{MDC}. However, this is insufficient for the perspective of implementation, because the assembly managers have no idea to extend the solution to other DCA_{MDC} problems. Therefore, the other objective of this article is to provide the approach to compute the solution with 80% quality.

3.3. A Case Study of Bearing Assembly

Given the bearing assembly line, the composition diagram is shown in Figure 4. Suppose that the balls are preinstalled in the retainer, and the balls and the retainer could be treated as the retainer module. The assembly process includes three parts, i.e., $|P| = 3$, the outer race, the retainer module, and the inner race with labels $j = 1, 2, 3$, respectively. Each part has two sizes in this example where the outside size is marked as number one, while inside size is number two. Therefore, the outer race has outside diameter $s_{1,y,1}$ and inside diameter $s_{1,y,2}$; the retainer module has outside diameter $s_{2,y,1}$ and inside diameter $s_{2,y,2}$; the inner race has outside diameter $s_{3,y,1}$ and inside diameter $s_{3,y,2}$ for y-th componment. The outer race and retainer module are involved in the chain number one while chain number two has the retainer module and inner race. Equation (3) is derived by expanding Equation (2). Since only component sizes $s_{1,y,2}$ and $s_{2,y,1}$ are in dimensional chain number one, e.g., $k = 1$, the values of $x_{1,y,2}$ and $x_{2,y,1}$ are non zero. Thus, the product sizes of i-th product is calculated by Equations (4) and (5) for chain number two.

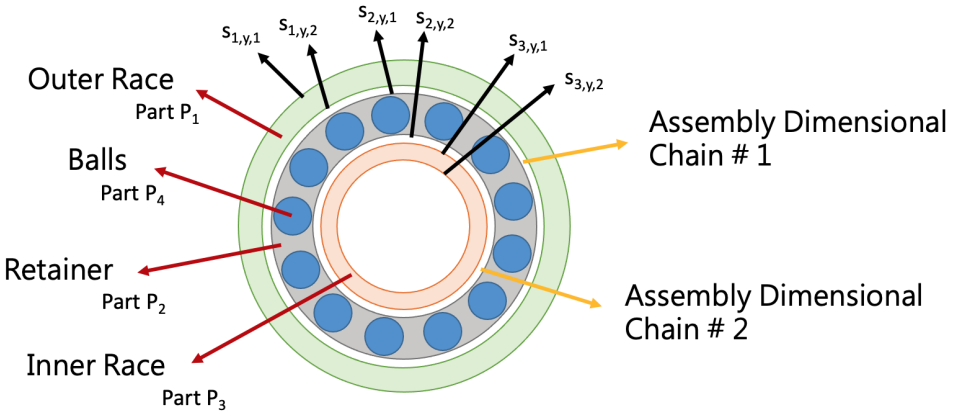

Figure 4. The assembly information of bearing.

$$ps_k^i = s_{1,y,1} \times x_{1,y,1} + s_{1,y,2} \times x_{1,y,2} + s_{2,y,1} \times x_{2,y,1} + s_{2,y,2} \times x_{2,y,2} + s_{3,y,1} \times x_{3,y,1} + s_{3,y,2} \times x_{3,y,2} \quad (3)$$

$$ps_1^i = s_{1,y,2} \times x_{1,y,2} + s_{2,y,1} \times x_{2,y,1} \quad (4)$$

$$ps_2^i = s_{2,y,2} \times x_{2,y,2} + s_{3,y,1} \times x_{3,y,1} \quad (5)$$

Considering that the assembly line receives 12 components that four for part number one, four for part number 2, and four for part number 3, as shown in Table 1, the label wp_{jx} indicates the x-th workpiece for j-th part. The objective is to assemble four bearings with the specification listed in Table 2. In other words, the sizes of the acceptable bearing should be within 0.15 and 0.4 for chain number one, and 0.1 and 0.43 for chain number two.

Table 1. The size information for each component.

Workpiece	wp_{j1}	wp_{j2}	wp_{j3}	wp_{j4}
$(s_{1,y,1}, s_{1,y,2})$	(12.5, 10.08)	(12.5, 10.2)	(12.5, 10.14)	(12.5, 10.1)
$(s_{2,y,1}, s_{2,y,2})$	(9.97, 7.17)	(9.86, 7.2)	(9.9, 7.18)	(9.9, 7.15)
$(s_{3,y,1}, s_{3,y,2})$	(7, 5)	(6.8, 5)	(6.87, 5)	(6.9, 5)

Table 2. The specification of target bearings of the example listed in Table 1.

	Chain Number One	Chain Number Two
lb_k	-0.05	-0.15
cv_k	0.2	0.25
ub_k	0.2	0.18

Suppose that the installation guidance with four bearings is (wp_{11}, wp_{21}, wp_{31}), (wp_{12}, wp_{22}, wp_{32}), (wp_{13}, wp_{23}, wp_{33}), and (wp_{14}, wp_{24}, wp_{34}). The guidance indicates that the first product consists of three components, and they are wp_{11}, wp_{21}, and wp_{31} from part 1, 2, and 3 respectively. The gaps between ps_k^i and cv_k are $(0.11, 0.17)$, $(0.34, 0.4)$, $(0.24, 0.31)$, and $(0.2, 0.25)$ for (chain number one, chain number two), respectively. For convenience, we use Δ_k^i to represent the gap that is the tolerance between ps_k^i and cv_k. The first bearing is unacceptable because the sizes are 0.11 and 0.17 for chain number one and chain number two, but however the acceptable specification should be within $(0.15, 0.4)$ and $(0.1, 0.43)$. The size of chain number two of the first bearing is smaller than the lower bound of the specification, so the first bearing is unacceptable. Eventually, the solution is infeasible.

4. Assembly Guidance Optimizer with Multiple Dimensional Chains

The DCA_{MDC} is a permutation problem. For the perspective of permutation, the goal is to find out a permutation that the installation guidance is satisfied with the inseparability and acceptability, as shown in Section 3.2. The exhaustive search [38] could obtain the optimal solution with minimum ps_k^i, but the time complexity is increased by the number of products, i.e., $|T|$, and the assembly complexity, i.e., $|P|$. Computing the optimal would pay more computation cost, so the near-optimal solutions are more appropriate than the optimal solution.

Several heuristic algorithms are proposed to compute the near optimal solution in large-scale search space; for example, genetic algorithm (GA) [39], Tabu search [40], ant colony optimization [41], SA [42,43], and flower pollination algorithm [44]. The heuristic algorithms could be classified into single solution-based (SS) approach, e.g., Tabu search and SA, and multi-solution-based (MS) approach, e.g., GA, ant colony optimization, and flower pollination algorithm. The major difference is that the MS approaches estimate several solutions in an iteration and only one solution is processed in the SS solution. In an iteration, MS approaches require a longer processing time than that of SS approaches, and MS approaches provide wider search than SS approaches. Theoretically, the MS approaches output the solutions with better quality than SS approaches, because of the search width. The MS approaches provide higher solution quality than that of the SS approaches, but the quality of output solution is not so stable as that of the SS approaches. The MS approaches parallelly search solutions but reaching better solution is based on the search probability. Thus, the SS approaches deeply search from single solution, so the solutions obtained by the SS approaches are more stable than that of the MS approaches. The candidates of SS approaches are SA and Tabu search algorithm. Both of SA and Tabu search finds neighbors from the solutions iteratively. Tabu search considers the Tabu list to prevent previously visited solutions, while SA allows the evaluation of visited solutions. Tabu list is helpful in the convergence in the small-scale problem but the probability of visiting previous solutions is low. Tabu search requires the computation resource to maintain Tabu list, but extra computation resource is unnecessary for SA. Therefore, SA is more appropriate than Tabu search algorithm for solving large optimization problems.

4.1. Proposed Algorithm

We extend the algorithm AGO proposed by Tsung et al. [17] to design the Assembly Guidance Optimizer with Multiple Dimensional Chains (AGO_{MDC}) to compute the installation guidance for DCA_{MDC}. The proposed AGO_{MDC} is shown in Algorithm 1. Given the DCA_{MDC} problem instance D and the algorithm configuration $(T, r, iter, v)$, the goal of AGO_{MDC} is to calculate the feasible solution that the installation guidance is satisfied with the constraints of then inseparability and the acceptability.

Algorithm 1: The algorithm of the proposed AGO_{MDC} for the DCA_{MDC} problem.

input : DCA_{MDC} instance D and the algorithm configuration $(T, r, iter, v)$ where T is the initial temperature, r is the temperature descent rate, $iter$ is the maximum iteration, and v is the variation degree.
output: The assembly guidance sol

```
1 sol ← solGenerator() ;
2 while not meet the stop condition do
3     sol_nb ← nbFinder(sol, v);
4     if ΔE > 0 then
5         sol ← sol_nb;
6     else if diverse(sol_nb) = True then
7         sol ← sol_nb;
8     T ← T × r;
9 return sol ;
```

In line 1, the initial solution sol is generated by the function $solGenerator()$. The manager could use various approaches to implement $solGenerator()$ based on the problem consideration. We apply the random generator in $solGenerator()$ to maximize the solution diversity. The goal of the installation guidance is to assemble $|T|$ products, so the product index of y-th component $com_{j,y}$ in part j-th part is determined randomly.

In line 2, the stop condition is used to control the solution quality and the algorithm running time, and the candidates of the stop conditions are maximum CPU time, the maximum number of iterations, the number of iteration that the solution quality is not improved, etc. Longer computation time theoretically results in better solution quality, but the marginal utility will be decreased for long running time. The setting of the stop condition will be discussed in the simulation, and we will provide the recommended configurations for managers.

In line 3, a local search algorithm is implemented in $nbFinder(sol, v)$, and the algorithm of $nbFinder(sol, v)$ is shown in Algorithm 2. Randomly changing the components of the same part from two products to generate a new neighborhood is the major idea of $nbFinder(sol, v)$. Therefore, in line 2 and 3, the indexes of two products and the target part are selected, and the corresponding components are exchanged in the neighborhood sol_{nb}. The variation degree v is the input parameter. Larger setting of v indicates more components are swapped in $nbFinder(sol, v)$, and it means that sol_{nb} is much different from sol.

Algorithm 2: The local search algorithm of $nbFinder(sol, v)$.

input : sol the solution for DCA_{MDC}, the variation degree v.
output: The neighborhood sol_{nb} for sol

1 **for** $z = 0$ **to** v **do**
2 pick up two target product indexes ix_1^t and ix_2^t from $(1, |T|)$ without replacement;
3 pick up one target part index ix^p from $(1, |P|)$;
4 $sol_{nb} \leftarrow swap(com_{ix^p, ix_1^t}, com_{ix^p, ix_2^t})$;
5 **return** sol_{nb} ;

After generating the neighborhood, we have sol and sol_{nb}, and the mission in each iteration is to determine the solution that will survive in the next iteration. The survival solution is determined in the area from line 4 to line 7. We use the label $\Delta E = q(sol) - q(sol_{nb})$ to represent the gap of the solution quality between sol and sol_{nb}, where $q(sol)$ returns the solution quality of sol, and the details of $q(sol)$ will be described in Section 4.3. The quality of sol is better than sol_{nb} when $\Delta E > 0$, and sol will survive in the next iteration, as shown in Line 5. Otherwise, we use the function $diverse(sol_{nb})$ to determine the survival of sol_{nb}. The function $diverse(sol_{nb})$ considers the acceptance threshold as shown in Equation (6), to accept sol_{nb} or not.

$$th = exp(-\Delta E/T) \tag{6}$$

If th is larger than the random number from $(0, 1)$, sol_{nb} is accepted, and rejected otherwise. The sol_{nb} with worse quality is accepted in the early stage, and the probability of accepting sol_{nb} becomes smaller as decreasing the temperatures. Therefore, we can use $diverse(sol_{nb})$ to increase the solution diversity to discover more candidate solutions.

4.2. Solution Format

Assembling $|T| \times |P|$ components to $|T|$ products is the goal of DCA_{MDC}, and there are $|C|$ dimensional chains in each product. We use two data structure to represent the information of a solution, and they are the installation guidance and the sizes of all dimensional chains as illustrated in Figures 2 and 5. According to Equation (2), we use a $|T|$ by $|P|$ two-dimensional array, that is named by guidance element set $GE = \{ge_{rs}\}$, where $1 \leq r \leq |T|$ and $1 \leq s \leq |P|$, to represent the installation guidance, as shown in Figure 2. Each row in GE represents the components of a product, and each element ge_{rs} saves the component index for part s. So, the assembler picks up the specific components of the installation guidance to assemble the product. The second data structure is the chain size, and a $|T|$ by $|C|$ two-dimensional array, that is named by chain size element set $CSE = \{cse_{vw}\}$, where $1 \leq v \leq |T|$ and $1 \leq w \leq |C|$, is apply to save the chain size information. We pick up some components of a row from GE, calculate all chain sizes by using Equation (2), and save the chains sizes to the corresponding row in CSE.

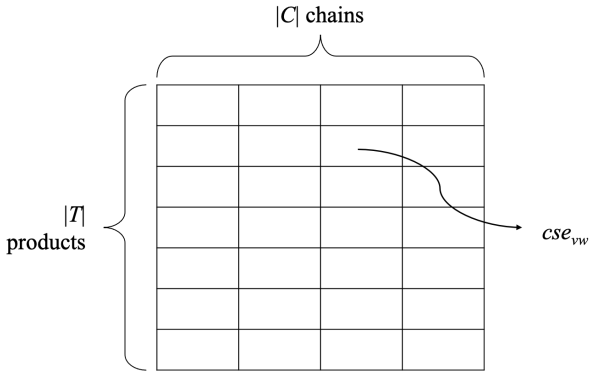

Figure 5. The structure of the chain sizes.

4.3. Solution Quality

The solution is feasible only if the installation guidance is satisfied with the constraints of inseparability and the acceptability, as defined in Section 3.2. We use Δ_k^i for each product i to evaluate the product quality. The ideal product takes place when the value of Δ_k^i is minimized, e.g., $ps_k^i = cv_k$, and minimizing Δ_k^i is the goal otherwise. However, there are two dimensional chains in the bearing assembly problem, so that we have two values to evaluate the quality of the given feasible solution. The arithmetic mean is the straightforward approach of normalizing several values to one. Unfortunately, the ranges between lb_k and ub_k of all dimensional chains may be different. For example, the range of chain number one is $0.05 + 0.2 = 0.25$ and $0.15 + 0.18 = 0.33$ for chain number two in Table 2. Considering that the (ps_1^i, ps_2^i) are (0.39, 0.25) and (0.2, 0.43) for the bearing i and $i + 1$, the values of Δ_k^i are (0.19, 0) and (0, 0.18), respectively, and the arithmetic mean values are 0.095 and 0.09. For the perspective of the arithmetic mean, the product $i + 1$ is better than the product i because of $0.095 > 0.09$. However, Δ_1^i does not meet ub_1, but Δ_2^{i+1} meets ub_2, so product i is better than the product $i + 1$ for the perspective of minimizing Δ_k^i. We have no ideas of ways to distinguish which one is better by the arithmetic mean. Thus, the arithmetic mean is inappropriate in measuring the solution quality in DCA_{MDC}.

To determine the solution quality, we have to consider following two issues in the DCA_{MDC} problem:

1. Asymmetry tolerance at_k^i: the lb_k and ub_k may be different.
2. Multiple dimensional chains: each product i has some Δ_k^i.

We have to normalize the above scores to one value to represent the solution quality of the give product. The asymmetry tolerance at_k^i is defined in Equation (7). Each Δ_k^i is normalized by the tolerance range, i.e., lb_k or ub_k, so the tolerance could be estimated in the same criteria.

$$at_k^i = \begin{cases} (ps_k^i - cv_k)/lb_k & if\, ps_k^i < cv_k, \\ (ps_k^i - cv_k)/ub_k & otherwise. \end{cases} \tag{7}$$

The relationship between the product size ps_k^i, cv_k, lb_k, and up_k is illustrated in Figure 6. We use three products to explain the calculation of the solution quality, where $ps_k^1 = 0.06$, $ps_k^2 = 0.2$, and $ps_k^3 = 0.39$. The product 2 is optimal because of $\Delta_k^2 = 0$. The quality of product 1 is $at_k^1 = (0.06 - 0.2)/-0.15 = -0.14/-0.15 = 0.93$ for k-th part, and the quality of product 3 for the part k is $at_k^3 = (0.39 - 0.2)/0.2 = 0.19/0.2 = 0.95$. The distance from ps_k^1 to $cv_k + lb_k$ and ps_k^3 to $cv_k + ub_k$ is the same, but $at_k^3 > at_k^1$ because of $ub_k > lb_k$. So, at_k^i can correctly describe the solution quality. Moreover, the product is feasible only when $at_k \leq 1$ for k-th part and infeasible otherwise.

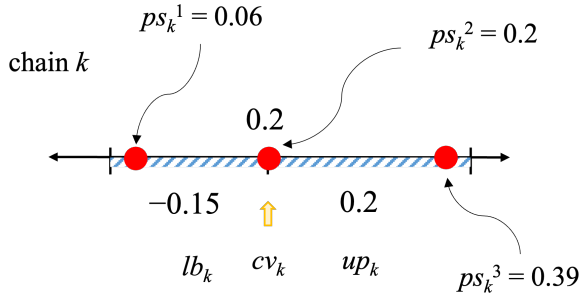

Figure 6. The relationship between the product sizes and the product specification.

The other consideration of multiple dimensional chains indicates that each product has $|P|$ values of at_k^i. We consider the maximum at_k^i to represent the quality of the product i as defined in Equation (8). The worst at_k^i could be treated as the quality bottleneck, and the product quality is improved when the worst at_k^i is decreased. Therefore, applying Equation (8) to be the solution quality is reasonable.

$$q_i = \max_{\forall k} at_k^i \tag{8}$$

Thus, the solution quality $q(sol)$ is derived as shown in Equation (9) by the same concept of Equation (8).

$$q(sol) = \max_{\forall i} q_i \tag{9}$$

4.4. Tracing AGO_{MDC}

In this section, the example listed in Table 1 is applied to trace the proposed AGO_{MDC} in finding the solution of DCA_{MDC}. The major mission is to demonstrate the processes of AGO_{MDC} in solving the problem. Therefore, the solutions and the algorithm status in each iteration are saved, and the solution might not be optimized.

In the beginning, AGO_{MDC} receives the composition diagram illustrated in Figure 1, the size information shown in Figure 4, and the product specification listed in Table 2, and the algorithm configuration. Initial temperature 50, temperature descent rate 0.98, and the maximum round 10 are considered. To focus on introducing the process of AGO_{MDC}, the entire information of solutions and algorithm status is listed in Appendix A.

From Table A1, the initial solution does not fit the acceptability because $q(sol) = 1.8$. In the first ten iterations, the neighbors with higher quality are accepted in iteration three, four, and nine, and the solutions with lower quality are accepted in iteration number eight. In other iterations, the neighbors with the same quality are accepted, since the random value is larger than the value of th that is defined in Equation (6). The details in terms of solutions and algorithm status are listed in Tables A1 and A2, respectively. However, in iteration seven, the neighbor has the same quality with that of the current solution, but the neighbor is rejected. The value of th is one which means that accepting the neighbor in high probability, but the random value is one. Therefore, the neighbor is rejected, and the current solution survives in the next iteration.

The results show the process of the proposed AGO_{PDC}, and the information in terms of accepting neighbors is displayed in Table A2. In the early stage, AGO_{MDC} accepts the neighbors with lower quality in high probability. As decreasing the temperatures, the value of th becomes lower and lower. The solution quality would be improved by considering more iterations.

5. Simulation

The real-world bearing assembly process is considered, and the CAD size and the assembly specification are illustrated in Figure 7, and they are calculated by the dimensional chain analysis during the product design phase. The real-world size data of 43×3

components, i.e., 43 bearings will be produced, are collected and the simulation instance generator is developed based on the collected size data. The simulated size data generated by the simulation instance generator is helpful in increasing the diversity of the simulation. The specification of the simulation platform is CPU i7-7700, RAM 16 GB, SSD 128 G, and using Visual Studio C++ to implement AGO_{MDC}. In the following simulations, we focus on analyzing the solution quality generated by the proposed AGO_{MDC} and provide the optimized algorithm configuration for the manager of the assembly line in the company.

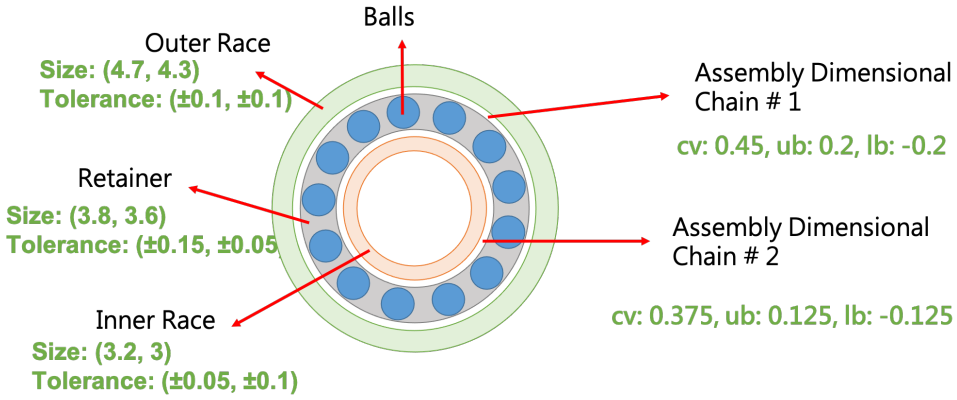

Figure 7. The bearing assembly case that is used to evaluate the performance of the proposed AGO_{MDC}.

5.1. Feasibility Analysis

The algorithm configuration $(T, r, iter)$ = (5000, 0.98, 10,000) is considered in this simulation to evaluate the effectiveness in resolving DCA_{MDC}. The setting is identical to the suggestion proposed in [17]. Moreover, $v = 1$ is applied to AGO_{MDC} to carefully trace the variance of neighborhood solutions. Each instance runs ten times, and the values are averaged for the final score. AGO_{MDC} consumes 562.6 ms to derive the solution with quality 0.51384 on average. The variation of the solution quality is illustrated in Figure 8. The processes of AGO_{MDC} include three phases: the exploration stage in the first 1000 iterations, the convergence stage from iteration 1000 to 4800, and the stable stage after iteration 4800. In the exploration stage, AGO_{MDC} tries its best to reach all kinds of solutions that also include the solutions with worse quality. In the convergence stage, the solution quality is improved slowly by accepting better solutions in high probability. Accepting a lower quality solution is possible, but the possibility is not high. Eventually, the solution quality is not improved, obviously because the temperature is low. The possibility of accepting the worse solution is based on the temperature and the quality of the neighborhood solution. Higher temperatures and higher solution quality result in higher possibility of accepting neighborhood solutions. The curve illustrated in Figure 8 demonstrates the relationship between the probability of accepting lower quality solution and the algorithm configurations. Therefore, the proposed AGO_{MDC} resolves DCA_{MDC} and the running time is rational for the implementation consideration.

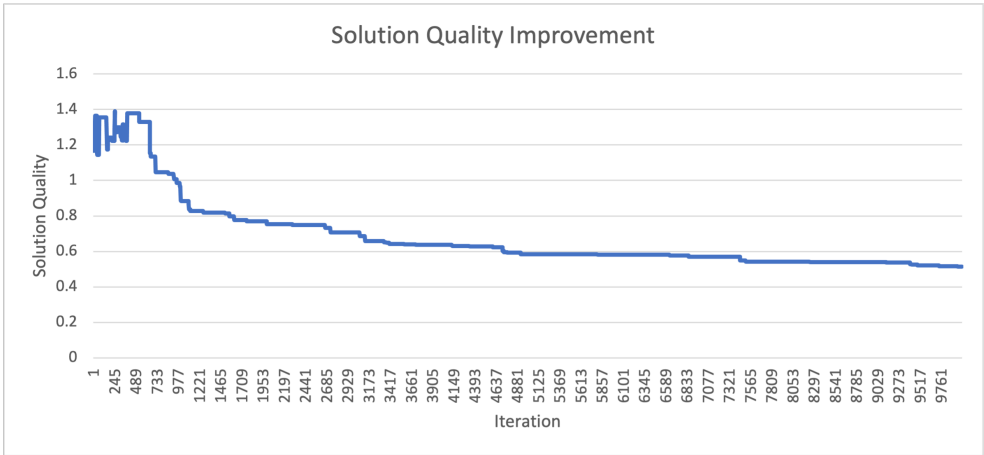

Figure 8. The improvement history of the solution quality for the configuration (5000, 0.98, 10,000) and $v = 1$.

In the first 1000 iterations, the solution quality dramatically changes and becomes stable in the following iterations. The final solution quality is 0.51384 on average. The $nbFinder(sol)$ in AGO_{MDC} exchanges the components of two products for the same part. AGO_{MDC} follows Equation (2) to calculate the solution quality, so the inseparability is realized. The final solution quality is 0.51384 that is smaller than 1, so the acceptability is matched. Eventually, the solution derived by AGO_{MDC} is feasible.

Moreover, the solution quality is larger than 1 during the first 1000 iterations, and it implies that the solution derived by AGO_{MDC} is infeasible. In the early stage, the value of T is hight, so the value of th is closed to 1 according to Equation (6). So, AGO_{MDC} accepts the neighbors with worse solution quality in high probability. When T is decreased, the probability of accepting the solutions with worse quality is also decreased. In the early stage, AGO_{MDC} seeks the solutions widely, and the search strategy changes to seek the elite solution after some iterations. If AGO_{MDC} only accepts better solutions, the initial solution determines the quality of the output solution. Therefore, the solution quality that the algorithm only accepts elite solutions may be worse than the acceptance strategy applied in AGO_{MDC}.

5.2. Configuration Optimization

In this simulation, we use the cases that we received from the company to design the simulation case generator. The product specification as shown in Figure 7 is the input of the case generator, and then the case generator outputs the component sizes. So, we can evaluate the performance of AGO_{MDC} by variant problem instances to study the optimized configuration. We investigate two issues that are the number of products and the product precision and provide the algorithm configuration for the manager of the assembly line. We consider the following three steps to find out the suggested configuration: (1) find out the optimal setting of the temperature descent rate r by considering the adequate settings; (2) using the derived r to analyze the setting of the maximum iteration; (3) calculating the initial temperature T, and the details are as shown below.

- The setting of the temperature descent rate r. To calculate the optimal setting of r, we evaluate the performance of various settings of v with the other extreme settings, and they are $T = 5000$, $iter = 10,000$, and the number of products is 250. The results of the solution quality and the running time are illustrated in Figures 9 and 10, respectively. The setting of $r = 0.9$ provides higher solution quality than that of others, while the corresponding running time is 3350 (ms). The time required by AGO_{MDC} to calculate the solution is acceptable. Therefore, we consider $r = 0.9$ in the following simulations.

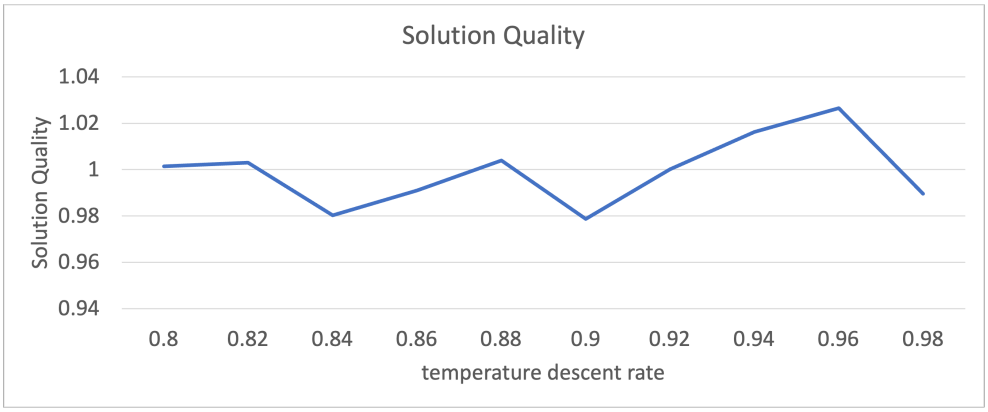

Figure 9. The solution quality variation for different temperature descent rate.

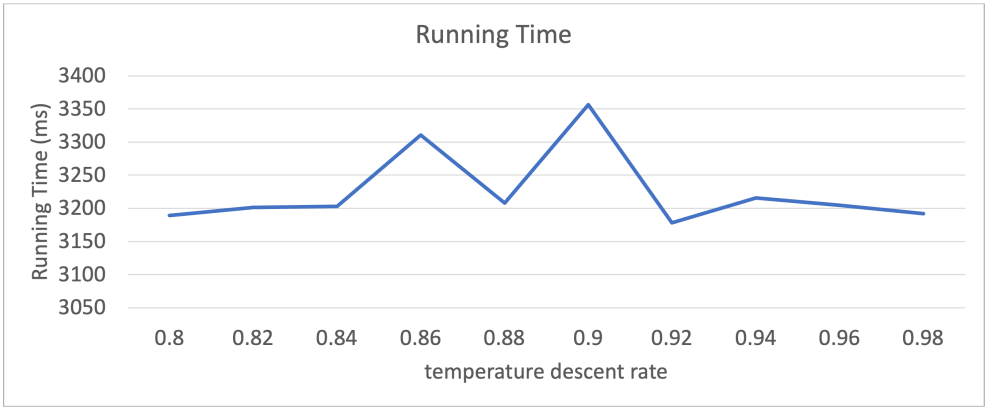

Figure 10. The running time variation for different temperature descent rate.

- The setting of the maximum iteration. Considering $r = 0.9$ is derived from the above simulations, we evaluate the performance of the proposed AGO_{MDC} under the settings of maximum iterations from 500 to 5000 with the gap 500. The simulation results of the solution quality and the running time are illustrated in Figures 11 and 12, respectively. The solutions in 50 and 100 products are acceptable after 1000 rounds, and the case with 150 products requires 3000 rounds. However, we should provide more iterations to calculate acceptable solutions in the cases with 200 and 250 products. The running time illustrated in Figure 12 is still acceptable in all instances, so we design the simulation with longer algorithm execution time to calculate the feasible solutions.

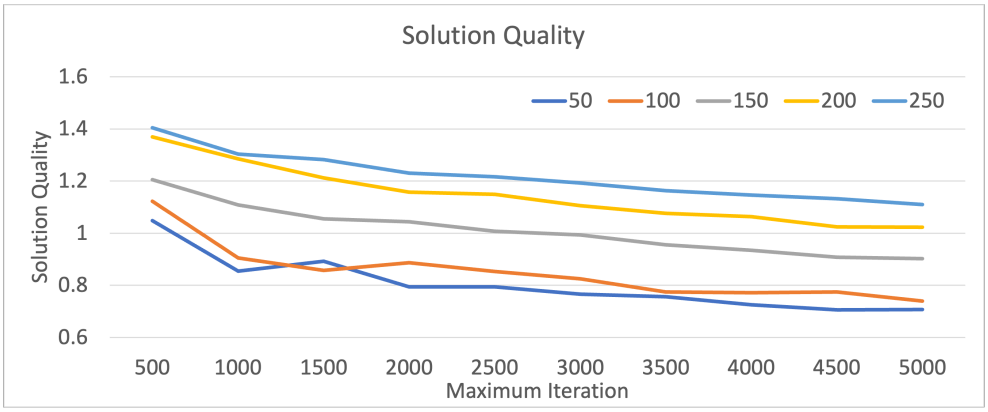

Figure 11. The solution quality variation captured from the instances with different maximum iterations and $r = 0.9$ in different problem scales.

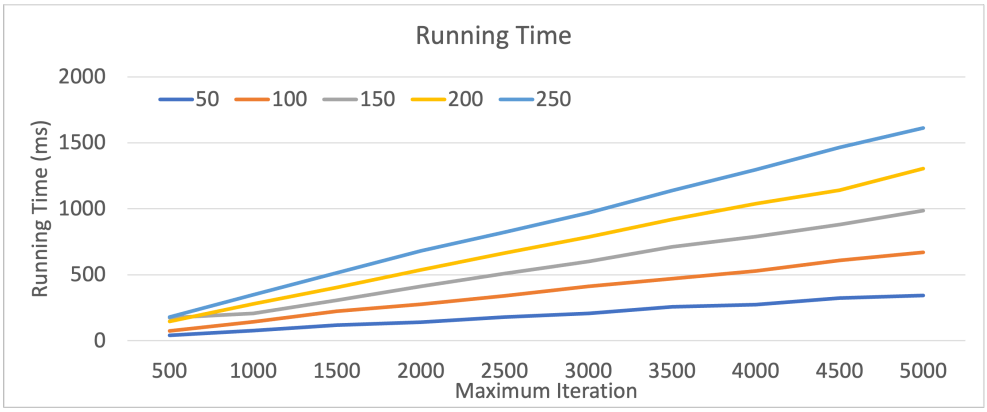

Figure 12. The running time variation captured from the instances with different maximum iterations and $r = 0.9$ in different problem scales.

It is difficult to find out the relationship between the problem scale and the minimum iterations for reaching the feasible solutions when two variables are considered: the initial temperature and the maximum iteration. Thus, we consider three levels of the initial temperatures: low level $T = 300$, medium level $T = 1300$, and high level $T = 2300$ to evaluate performance under various settings of the maximum iteration. Moreover, minimizing at_k^i is the objective in DCA_{MDC}, but the products with the quality $at_k^i \leq 1$ are feasible. From the perspective of the implementation, $at_k^i = 0$ is optimal, but $q(sol) = 0.8$ is acceptable. The products with $at_k^i = 0.9$ are also feasible, but if the products will be assembled with other parts, 20% tolerance is more appropriate than that of 10%. Therefore, the goal of this simulation is to capture the performance of various maximum iterations under three levels of the initial temperatures when the solution quality converges to $q(sol) \leq 0.8$.

The simulation results are illustrated in Figure 13 for the solution quality and Figures A1–A3 for the running time. We observed that three curves are overlapping in each sub-graph. The curve variance is slight in different levels of the initial temperatures, so various initial temperatures do not affect the convergence too much. Moreover, the solution quality meets the objective $q(sol) \leq 0.8$ in all instances eventually. We list the thresholds of the initial temperatures when reaching $q(sol) \leq 0.8$, and the results are shown in Table 3. The maximum threshold in each problem scale is marked in bold style. The increment of the temperature threshold is proportional to the problem scale linearly for the problems with more than 100 products, so we can use the linear regression analysis to find out the increment function. The instances with 50 products and 100 products require the same maximum iterations, so we consider the number of products from 100 to 250. Using the data in Table 3, we apply the linear regression analysis [45,46] to derive the recommended maximum iteration, as shown in Equation (9), where z is the number of products. Therefore, we can use Equation (10) to recommend the appropriate settings of the maximum iteration $f(z)$ ot derive expected solution quality 0.8.

$$f(z) = 200z - 16,000 \tag{10}$$

Table 3. The settings of maximum iterations that result in the solution quality $q(sol) \leq 0.8$. The maximum threshold in each problem scale is marked in bold style.

	Low Level	Medium Level	High Level
50	**4000**	**4000**	**4000**
100	**4000**	**4000**	**4000**
150	**13,000**	10,000	**13,000**
200	**22,000**	**22,000**	19,000
250	31,000	**34,000**	**34,000**

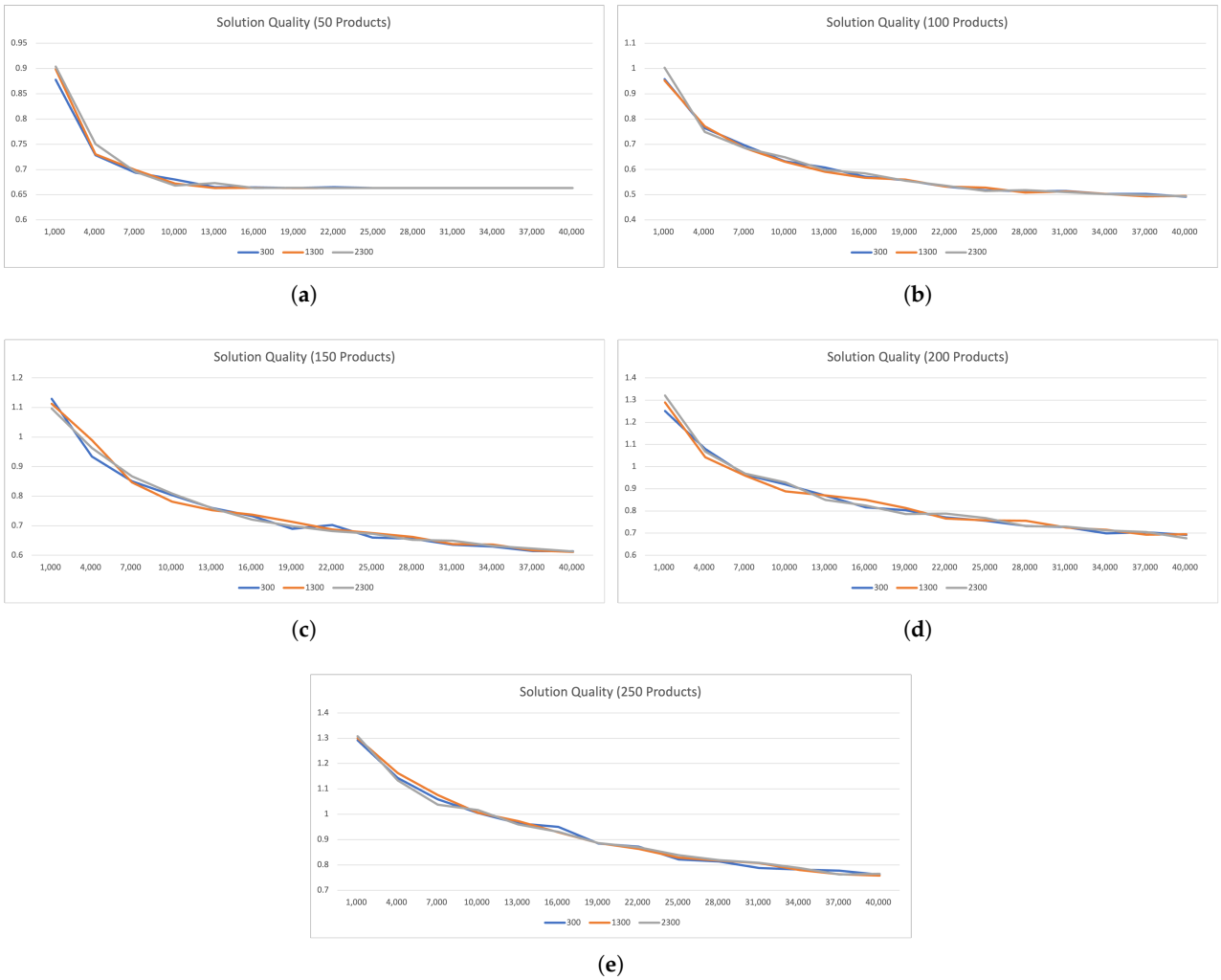

Figure 13. The convergence of the solution quality in different problem scales under the initial temperatures are low, medium, and high levels. (**a**) The convergence of the solution quality in 50 products. (**b**) The convergence of the solution quality in 100 products. (**c**) The convergence of the solution quality in 150 products. (**d**) The convergence of the solution quality in 200 products. (**e**) The convergence of the solution quality in 250 products.

- The setting of the initial temperature. We consider $r = 0.9$ and the maximum iterations determined by Equation (10) to find out the optimal settings of initial temperatures from 1000 to 40,000 with gap 3000. The results are illustrated in Figure 14. Since the running time is acceptable that is evaluated in the above simulations, we skip the discussion of the running time. From Figure 14, we have the following observations: (1) all results are bounded by $q(sol) = 0.8$; (2) various initial temperatures do not affect the solution quality too much; (3) the curves of solution quality are bounded within 0.7 and 0.8. The solution result means that the initial temperatures slightly affect the solution quality, but not too much. Therefore, the initial temperatures could be selected from 1000 to 40,000 arbitrarily, and the solution quality would be better than 0.8.

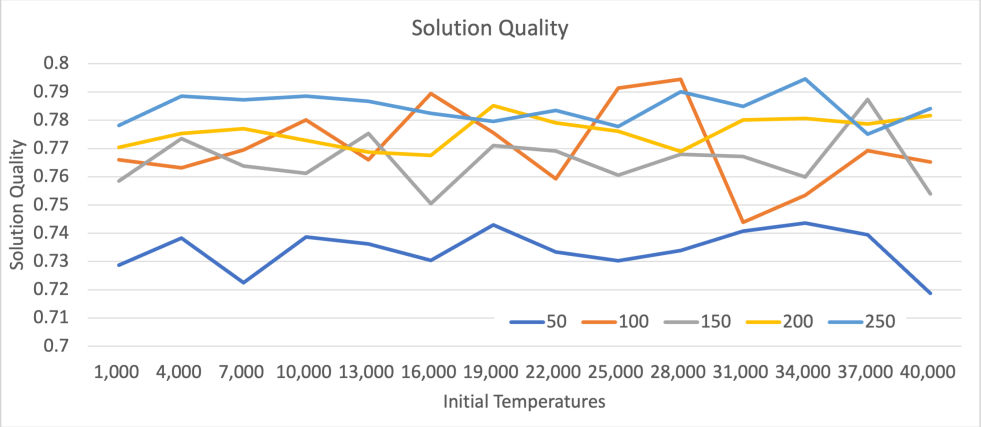

Figure 14. The solution quality variation for different settings of the initial temperatures, where the maximum iterations are considered in Table 3.

5.3. Discussion

According to the results from the above simulations, the recommended configuration of deriving the solution quality better than 0.8 is listed as follows:

- The initial temperature: the values should be larger than 1000.
- The temperature descent rate: 0.9 provides high probability in reaching the solutions with higher quality than others.
- The maximum iteration: using Equation (10) to determines the maximum iteration $f(z)$ for the number of products z.

We did not find the specific setting of initial temperature in the above simulations. The initial temperatures determine the decision on accepting the solutions with lower quality. According to the acceptance threshold defined in Equation (6), the probability of accepting the worse quality solution in higher temperatures is larger than that in lower temperatures. Thus, the initial temperatures determine the search breadth, and that also affects the solution quality indirectly. We organize the simulations to confirm the assumption. We consider $r = 0.9$ to observe the solution quality variation in the first 2000 iterations for assembling 150 products. The results captured from the initial temperatures 100, 500, and 3000 are illustrated in Figure 15. The degree of the curve variation becomes larger as the initial temperatures are increased, and the solutions with different properties can be reached with high probability. Moreover, higher temperatures not only increase the acceptance probability, but also determine the accepted solution quality. In Figure 15c, the curve suddenly raised in round 300, and the amount is larger than that in Figure 15a,b. So higher temperatures would help on effectively probing various properties of solutions than the lower settings.

Wider search results in higher probability of reaching the solutions with better quality theoretically. Therefore, the manager could increase the setting of the initial temperatures to improve the probability in reaching better solutions if the solution does not reach the expected goal.

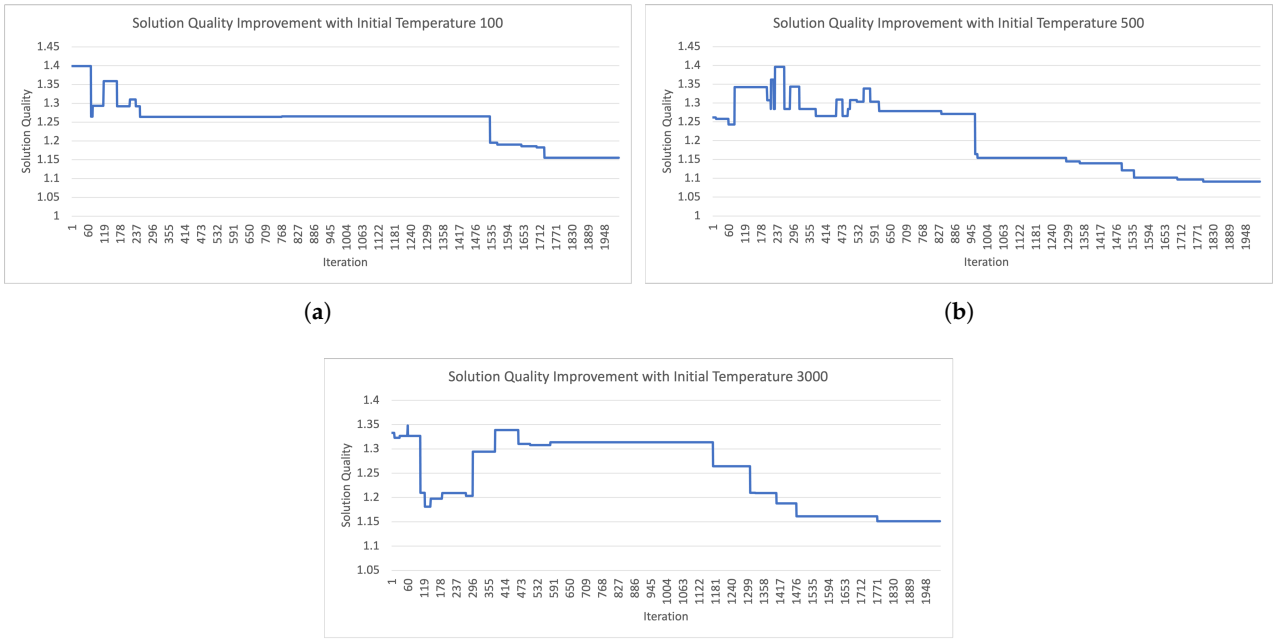

Figure 15. Comparing the solution quality for the configurations with initial temperatures 100, 500, and 3000, and $r = 0.9$ in first 2000 rounds. (**a**) The solution quality variance in the instance with initial temperature 100. (**b**) The solution quality variance in the instance with initial temperature 500. (**c**) The solution quality variance in the instance with initial temperature 3000.

6. Conclusions

Most products or modules are assembled by several parts, and the bearing discussed in this article consists of three parts and two surfaces. The major difficulty is that some dimensional chains would cross on the same part, so the gaps between different dimensional chains that crossed on the same part should be evaluated without separation. The proposed AGO_{MDC} considers the inseparability to compute the assembly guidance for assemblers. From the simulation results, the solutions derived by AGO_{MDC} are satisfied with the constraints of inseparability and acceptability. For the implementation, the appropriate algorithm configurations are suggested to find the solutions with acceptable product quality. Moreover, the running time of calculating the solution is not higher than 2 s for 250 products. Thus, the running time is low enough to be applied to move the modules from manufacturing lines to assembly lines. Therefore, the proposed approach provides high possibility of implementation.

Optimizing the algorithm configurations is the major objective of solving the optimization problems. However, the optimal solution is not the first concern in manufacturing because the real-world processes would consider more additional issues, e.g., labor cost and the rework time. Since the manufacturing cost comes from several issues, minimizing the cost without decreasing the product quality is the major concern of most companies. The producing managers prefer to find out the balance between the solution quality and the manufacturing cost rather than reaching the quality optimality. On the other hand, the scheduling becomes more important as the automated manufacturing is getting popular. Once the assembly guidance is prepared, the assembly lines could automatically assemble the products according to the assembly guidance. Therefore, the proposed AGO_{MDC} provides critical contributions on the integration about automated manufacturing.

Funding: This research was funded by Ministry of Science and Technology of the Republic of China grant number MOST 109-2221-E-167 -030 -MY3.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable

Acknowledgments: Author likes to thank anonymous reviewers and editors for their valuable comments and suggestions.

Conflicts of Interest: The author declares no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

ICT	Information and Communications Technologies
DCA	Dimensional Chain Assembly
AGO	Assembly Guidance Optimizer
SA	Simulated Annealing

Appendix A. Tracing AGO_{MDC} by the Example of Table 1

Table A1. The solution details in each iteration derived by AGO_{MDC} for the example listed in Table 1.

Iteration	i	wp_{jx}	ps_1^i	ps_2^i	at_1^i	at_2^i	q_i	$q(sol)$
1	1	(3, 2, 3)	0.2	0.28	0	0.17	0.17	1.8
	2	(0, 0, 0)	0.11	0.17	0.18	0.53	1.8	
	3	(1, 3, 1)	0.3	0.3	0.5	0.28	0.5	
	4	(2, 1, 2)	0.28	0.33	0.4	0.44	0.44	
2	1	(3, 2, 2)	0.2	0.31	0	0.33	0.33	1.8
	2	(0, 0, 0)	0.11	0.17	0.18	0.53	1.8	
	3	(1, 3, 1)	0.3	0.3	0.5	0.28	0.5	
	4	(2, 1, 3)	0.28	0.3	0.4	0.28	0.4	
3	1	(3, 2, 0)	0.2	0.18	0	0.47	0.47	1.8
	2	(0, 0, 2)	0.11	0.3	0.18	0.28	1.8	
	3	(1, 3, 1)	0.3	0.3	0.5	0.28	0.5	
	4	(2, 1, 3)	0.28	0.3	0.4	0.28	0.4	
4	1	(3, 2, 0)	0.2	0.18	0	0.47	0.47	0.6
	2	(2, 0, 2)	0.17	0.3	0.6	0.28	0.6	
	3	(1, 3, 1)	0.3	0.3	0.5	0.28	0.5	
	4	(0, 1, 3)	0.22	0.3	0.4	0.28	0.28	
5	1	(3, 2, 0)	0.2	0.18	0	0.47	0.47	0.5
	2	(1, 0, 2)	0.23	0.3	0.6	0.28	0.28	
	3	(2, 3, 1)	0.24	0.3	0.5	0.28	0.28	
	4	(0, 1, 3)	0.22	0.3	0.1	0.28	0.28	
6	1	(3, 2, 0)	0.2	0.18	0	0.47	0.47	0.5
	2	(1, 0, 2)	0.23	0.3	0.15	0.28	0.28	
	3	(0, 3, 1)	0.18	0.3	0.4	0.28	0.4	
	4	(2, 1, 3)	0.28	0.3	0.4	0.28	0.4	
7	1	(3, 2, 0)	0.2	0.18	0	0.47	0.47	0.5
	2	(1, 0, 2)	0.23	0.3	0.15	0.28	0.28	
	3	(2, 3, 1)	0.24	0.3	0.2	0.28	0.28	
	4	(0, 1, 3)	0.22	0.3	0.1	0.28	0.28	

Table A1. *Cont.*

Iteration	i	wp_{jx}	ps_1^i	ps_2^i	at_1^i	at_2^i	q_i	$q(sol)$
8	1	(3, 2, 0)	0.2	0.18	0	0.47	0.47	0.5
	2	(1, 0, 2)	0.23	0.3	0.15	0.28	0.28	
	3	(2, 3, 1)	0.24	0.3	0.2	0.28	0.28	
	4	(0, 1, 3)	0.22	0.3	0.1	0.28	0.28	
9	1	(0, 2, 0)	0.18	0.18	0.4	0.47	0.47	1
	2	(1, 0, 2)	0.23	0.3	0.15	0.28	0.28	
	3	(2, 3, 1)	0.24	0.3	0.2	0.28	0.28	
	4	(3, 1, 3)	0.15	0.3	0.1	0.28	1	
10	1	(1, 2, 0)	0.3	0.18	0.5	0.47	0.5	1.8
	2	(0, 0, 2)	0.11	0.3	1.8	0.28	1.8	
	3	(2, 3, 1)	0.24	0.3	0.2	0.28	0.28	
	4	(3, 1, 3)	0.15	0.3	1	0.28	1	

Table A2. The algorithm status in each iteration for the example listed in Table 1.

Iteration	T	$q(sol)$	$q(sol_{nb})$	ΔE	th	Random Value	Accept $q(sol_{nb})$
1	100	1.8	1.8	0	1	0.74	Yes
2	98	1.8	1.8	0	1	0.18	Yes
3	96.04	1.8	0.6	1.2	N/A	N/A	Yes
4	94.12	0.6	0.5	0.1	N/A	N/A	Yes
5	92.24	0.5	0.5	0	1	0.4	Yes
6	90.39	0.5	0.5	0	1	0.52	Yes
7	88.58	0.5	0.5	0	1	1	No
8	86.81	0.5	1	−0.5	1.01	0.59	Yes
9	85.08	1	1.8	−0.8	1.01	0.69	Yes
10	83.37	1.8	1.8	0	0.99	0.9	Yes

Appendix B. The Simulation Rsults of Running Time

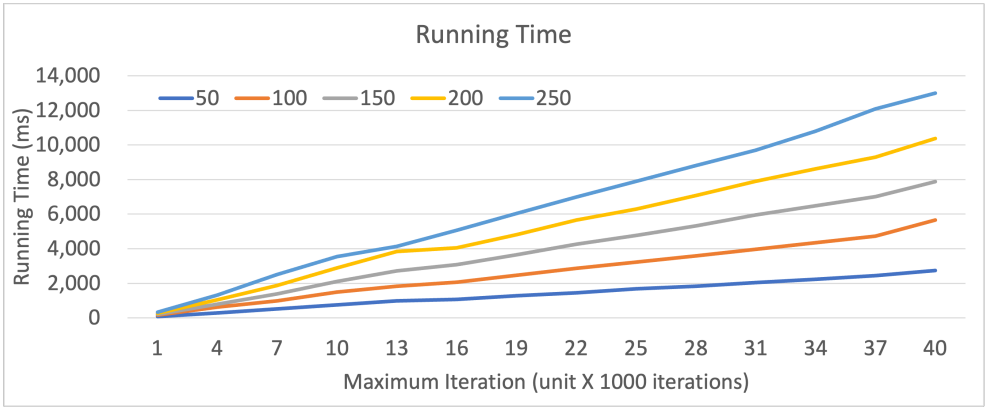

Figure A1. The comparison of the running time between various maximum iterations and problem scales under $r = 0.9$ and $T = 300$.

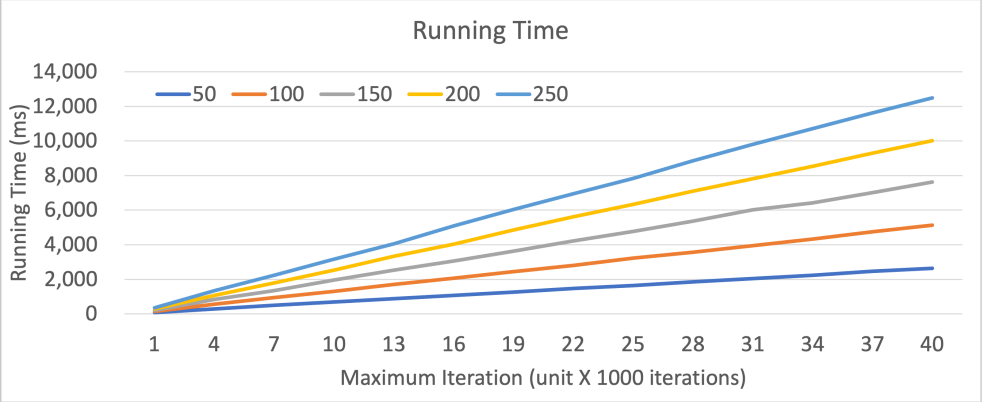

Figure A2. The comparison of the running time between various maximum iterations and problem scales under $r = 0.9$ and $T = 1300$.

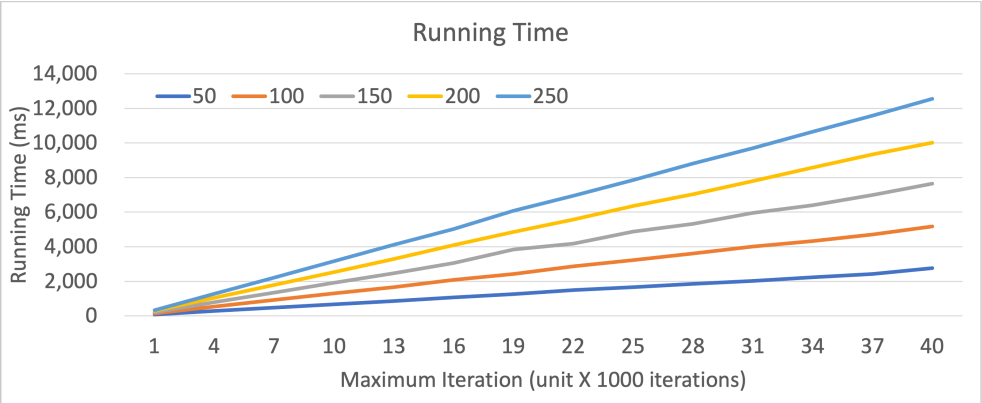

Figure A3. The comparison of the running time between various maximum iterations and problem scales under $r = 0.9$ and $T = 2300$.

References

1. Kang, H.S.; Lee, J.Y.; Choi, S.; Kim, H.; Park, J.H.; Son, J.Y.; Kim, B.H.; Do Noh, S. Smart manufacturing: Past research, present findings, and future directions. *Int. J. Precis. Eng.-Manuf.-Green Technol.* **2016**, *3*, 111–128. [CrossRef]
2. Tao, F.; Qi, Q.; Liu, A.; Kusiak, A. Data-driven smart manufacturing. *J. Manuf. Syst.* **2018**, *48*, 157–169. [CrossRef]
3. Chan, Y.W.; Chien, F.T.; Chang, M.K.; Ho, W.C.; Hung, J.C. A coalitional graph game approach for minimum transmission broadcast in iot networks. *IEEE Access* **2020**, *8*, 24385–24396. [CrossRef]
4. Zhou, X.; Liang, W.; Kevin, I.; Wang, K.; Wang, H.; Yang, L.T.; Jin, Q. Deep-learning-enhanced human activity recognition for Internet of healthcare things. *IEEE Internet Things J.* **2020**, *7*, 6429–6438. [CrossRef]
5. Yang, C.T.; Chen, S.T.; Den, W.; Wang, Y.T.; Kristiani, E. Implementation of an intelligent indoor environmental monitoring and management system in cloud. *Future Gener. Comput. Syst.* **2019**, *96*, 731–749. [CrossRef]
6. Chang, C.H.; Yang, C.T.; Lee, J.Y.; Lai, C.L.; Kuo, C.C. On construction and performance evaluation of a virtual desktop infrastructure with GPU accelerated. *IEEE Access* **2020**, *8*, 170162–170173. [CrossRef]
7. Yang, C.T.; Liu, J.C.; Huang, K.L.; Jiang, F.C. A method for managing green power of a virtual machine cluster in cloud. *Future Gener. Comput. Syst.* **2014**, *37*, 26–36. [CrossRef]
8. Kristiani, E.; Yang, C.T.; Huang, C.Y.; Ko, P.C.; Fathoni, H. On construction of sensors, edge, and cloud (ISEC) framework for smart system integration and applications. *IEEE Internet Things J.* **2020**, *8*, 309–319. [CrossRef]
9. Zhou, X.; Xu, X.; Liang, W.; Zeng, Z.; Shimizu, S.; Yang, L.T.; Jin, Q. Intelligent small object detection based on digital twinning for smart manufacturing in industrial CPS. *IEEE Trans. Ind. Inform.* **2021**, in press.
10. Zhou, X.; Liang, W.; Shimizu, S.; Ma, J.; Jin, Q. Siamese neural network based few-shot learning for anomaly detection in industrial cyber-physical systems. *IEEE Trans. Ind. Informatics* **2020**, *17*, 5790–5798. [CrossRef]
11. Zhou, X.; Hu, Y.; Liang, W.; Ma, J.; Jin, Q. Variational LSTM enhanced anomaly detection for industrial big data. *IEEE Trans. Ind. Inform.* **2020**, *17*, 3469–3477. [CrossRef]
12. Zhang, X.; Ming, X.; Yin, D. Application of industrial big data for smart manufacturing in product service system based on system engineering using fuzzy DEMATEL. *J. Clean. Prod.* **2020**, *265*, 121863. [CrossRef]
13. Galetto, M.; Verna, E.; Genta, G. Accurate estimation of prediction models for operator-induced defects in assembly manufacturing processes. *Qual. Eng.* **2020**, *32*, 595–613. [CrossRef]

14. Haghighi, P.; Ramnath, S.; Chitale, A.; Davidson, J.K.; Shah, J.J.; Center, M. Automated tolerance analysis of Mechanical Assemblies from a CAD model with PMI. *Comput.-Aided Des. Appl.* **2020**, *17*, 249–273. [CrossRef]
15. Xing, Y.; Jiang, C.; Luo, L. A new approach of manufacturing tolerance allocation based on NSGA-III algorithm for the in-process monitoring of sheet metal parts. In *Advances in Asset Management and Condition Monitoring*; Springer: Cham, Switzerland, 2020; pp. 1233–1242.
16. Zhang, R.; Liu, X.; Yi, Y. Technology of Automatic Generation and Update of Assembly Dimension Chain for Assembly Process. In Proceedings of the 2020 IEEE International Conference on Mechatronics and Automation (ICMA), Beijing, China, 13–16 October 2020; pp. 472–477.
17. Tsung, C.K.; Ho, T.F.; Huang, H.Y.; Yang, S.H.; Tsou, P.N.; Tsai, M.C.; Huang, Y.P. Computing the Assembly Guidance for Maximizing Product Quality in the Virtual Assembly. *Sustainability* **2020**, *12*, 4690. [CrossRef]
18. Zhao, Y.J.; Xu, W.H.; Xi, C.Z.; Liang, D.T.; Li, H.N. Automatic and accurate measurement of microhardness profile based on image processing. *IEEE Trans. Instrum. Meas.* **2021**, *70*, 1–9.
19. Jiang, Z.; Zhang, L.; Qiu, T. Automatic high-precision measurement technology of special-shaped surface parts based on robot arms. *J. Phys. Conf. Ser.* **2020**, *1693*, 012214. [CrossRef]
20. Hsieh, Y.P.; Mertikopoulos, P.; Cevher, V. The limits of min-max optimization algorithms: Convergence to spurious non-critical sets. *PMLR* **2021**, *139*, 4337–4348.
21. Daskalakis, C.; Panageas, I. The limit points of (optimistic) gradient descent in min-max optimization. *arXiv* **2018**, arXiv:1807.03907.
22. Wang, Y.; Bu, G.; Wang, Y.; Zhao, T.; Zhang, Z.; Zhu, Z. Application of a simulated annealing algorithm to design and optimize a pressure-swing distillation process. *Comput. Chem. Eng.* **2016**, *95*, 97–107. [CrossRef]
23. Zhan, S.H.; Lin, J.; Zhang, Z.J.; Zhong, Y.W. List-based simulated annealing algorithm for traveling salesman problem. *Comput. Intell. Neurosci.* **2016**, *2016*, 1712630. [CrossRef]
24. Walter, M.S. Dimensional and geometrical tolerances in mechanical engineering—A historical review. *Mach. Des.* **2019**, *11*, 67–74.
25. Mansoor, E.M. The application of probability to tolerances used in engineering designs. *Proc. Inst. Mech. Eng.* **1963**, *178*, 29–39. [CrossRef]
26. Xing, Y.; Wang, Y. Minimizing assembly variation in selective assembly for auto-body parts based on IGAOT. *Int. J. Intell. Comput. Cybern.* **2018**, *11*, 254–268. [CrossRef]
27. Walter, M.S.; Klein, C.; Heling, B.; Wartzack, S. Statistical Tolerance Analysis—A Survey on Awareness, Use and Need in German Industry. *Appl. Sci.* **2021**, *11*, 2622. [CrossRef]
28. Tao, F.; Zuo, Y.; Da Xu, L.; Zhang, L. IoT-based intelligent perception and access of manufacturing resource toward cloud manufacturing. *IEEE Trans. Ind. Informatics* **2014**, *10*, 1547–1557.
29. Tsung, C.K.; Yen, C.T.; Wu, W.F. A software defined-based hybrid cloud for the design of smart micro-manufacturing system. *Microsyst. Technol.* **2018**, *24*, 4329–4340. [CrossRef]
30. Kim, Y.D.; Lim, H.G.; Park, M.W. Search heuristics for a flowshop scheduling problem in a printed circuit board assembly process. *Eur. J. Oper. Res.* **1996**, *91*, 124–143. [CrossRef]
31. Ge, M.; Hu, J.; Liu, M.; Zhang, Y. Reassembly classification selection method based on the Markov Chain. *Assem. Autom.* **2018**, *38*, 476-486. [CrossRef]
32. Rausch, C.; Nahangi, M.; Haas, C.; Liang, W. Monte Carlo simulation for tolerance analysis in prefabrication and offsite construction. *Autom. Constr.* **2019**, *103*, 300–314. [CrossRef]
33. Homri, L.; Goka, E.; Levasseur, G.; Dantan, J.Y. Tolerance analysis—Form defects modeling and simulation by modal decomposition and optimization. *Comput.-Aided Des.* **2017**, *91*, 46–59. [CrossRef]
34. Morse, E.; Dantan, J.Y.; Anwer, N.; Söderberg, R.; Moroni, G.; Qureshi, A.; Jiang, X.; Mathieu, L. Tolerancing: Managing uncertainty from conceptual design to final product. *Cirp Ann.* **2018**, *67*, 695–717. [CrossRef]
35. Scala, N.M.; Kutzner, R.; Buede, D.; Ciminera, C.; Bridges, A. Multi-objective decision analysis for workforce planning: A case study. In *Proceedings of the Industrial and Systems Engineering Research Conference*; Institute of Industrial Engineers: Orlando, FL, USA, 2012.
36. Deb, K. Multi-objective optimization. In *Search Methodologies*; Springer: Boston, MA, USA, 2014; pp. 403–449
37. Kumar, R.; Chandrawat, R.K.; Sarkar, B.; Joshi, V.; Majumder, A. An advanced optimization technique for smart production using α-cut based quadrilateral fuzzy number. *Int. J. Fuzzy Syst.* **2021**, *23*, 107–127. [CrossRef]
38. Chang, K.H.; Markov, I.L.; Bertacco, V. Post-placement rewiring and rebuffering by exhaustive search for functional symmetries. In Proceedings of the ICCAD-2005. IEEE/ACM International Conference on Computer-Aided Design, San Jose, CA, USA, 6–10 November 2005; pp. 56–63.
39. Whitley, D. A genetic algorithm tutorial. *Stat. Comput.* **1994**, *4*, 65–85. [CrossRef]
40. Glover, F.; Laguna, M. Tabu search. In *Handbook of Combinatorial Optimization*; Springer: Boston, MA, USA, 1998; pp. 2093–2229.
41. Dorigo, M.; Birattari, M.; Stutzle, T. Ant colony optimization. *IEEE Comput. Intell. Mag.* **2006**, *1*, 28–39. [CrossRef]
42. Van Laarhoven, P.J.; Aarts, E.H. Simulated annealing. In *Simulated Annealing: Theory and Applications*; Springer: Dordrecht, The Netherlands, 1987; pp. 7–15.
43. Khurshid, B.; Maqsood, S.; Omair, M.; Sarkar, B.; Ahmad, I.; Muhammad, K. An Improved Evolution Strategy Hybridization With Simulated Annealing for Permutation Flow Shop Scheduling Problems. *IEEE Access* **2021**, *9*, 94505–94522. [CrossRef]

44. Saha, S.; Chatterjee, D.; Sarkar, B. The ramification of dynamic investment on the promotion and preservation technology for inventory management through a modified flower pollination algorithm. *J. Retail. Consum. Serv.* **2021**, *58*, 102326. [CrossRef]
45. Kumari, K.; Yadav, S. Linear regression analysis study. *J. Pract. Cardiovasc. Sci.* **2018**, *4*, 33. [CrossRef]
46. Montgomery, D.C.; Peck, E.A.; Vining, G.G. *Introduction to Linear Regression Analysis*; John Wiley and Sons: Hoboken, NJ, USA, 2021.