ISSN 1330-3651 (Print), ISSN 1848-6339 (Online)
https://doi.org/10.17559/TV-20240127001297
Original scientific paper

IAGA: A New Method of Automatic Layout for Dimensioning Engineering Drawings

Yunlei SUN*, Zhaotong SHAO, Bingyi YAN

Abstract: This paper presents an automatic layout method for engineering drawing dimensioning based on an improved adaptive genetic algorithm (IAGA). The key novelty lies in transforming the complex dimensional layout optimization into a combinatorial directed acyclic graph problem enabling efficient evolutionary solving. Comparative analyses on multi-sized test cases validate the superiority of IAGA in dimensioning quality and performance over standard GA and adaptive GA approaches. Additional case studies on practical heat exchanger and vessel diagrams further demonstrate the technique's effectiveness in handling real-world industrial applications. This pioneering dimensional engineering automation solution promises to facilitate precise computer-aided manufacturability with enhanced efficiency.

Keywords: adaptive genetic algorithm; combinatorial optimization; computer-aided design; dimension annotations layout; directed acyclic graph

1 INTRODUCTION

Computer-aided design (CAD) is a technology that utilizes computer technology to assist in design and drawing [1]. Since its inception, it has been widely employed in architectural design, engineering design, mechanical design, etc., promoting continuous change and innovation in traditional product design methods and production modes, thus generating significant economic and social benefits. Engineering drawings serve as the programmatic documents of production and manufacturing [2], playing a crucial role in all stages of product production. Initially, engineering drawings were predominantly hand-drawn, requiring substantial human and material resources due to their intricate dimensioning. However, with the ongoing development of CAD technology, some mainstream CAD software has realized automatic generation of dimension marking functions. Nonetheless, when dealing with complex engineering drawings, automatically generated dimensioning layouts often exhibit various quality issues, as illustrated in Fig. 1. These issues include, but are not limited to: (1) potential interference between dimensioning and views; (2) overlapping dimension lines; and (3) an excessive number of dimensioning layout layers, leading to interference with other elements in the drawings. Addressing these problems typically necessitates manual adjustments by designers, resulting in reduced design efficiency. Consequently, numerous scholars have studied these issues, although most research focuses on dimensioning specific to particular industries or geometric features of the layout space, which limits its expansiveness and generalizability. In this paper, assuming that the location of the marking point in the dimensioning meets engineering requirements, we investigate the rationalization of dimension layout with horizontal and vertical dimensioning as the research focus. Initially, the relationships between dimensioning are inductively defined, and a Directed Acyclic Graph (DAG) is introduced to formally represent these relationships. Subsequently, the dimensioning layout problem is transformed into a graph-based combinatorial optimization problem, and a dimensioning layout algorithm based on an improved adaptive genetic algorithm is devised. Finally, validation and analysis of three sets of dimensional data of varying scales, as well as the dimensional annotation of heat exchangers and vessels in petrochemical static equipment, demonstrate that the dimensional layout generated by the proposed method meets manufacturing requirements. This approach effectively addresses the quality and efficiency issues associated with the interactive dimensional layout generation of current mainstream CAD software, significantly enhancing design efficiency and quality.

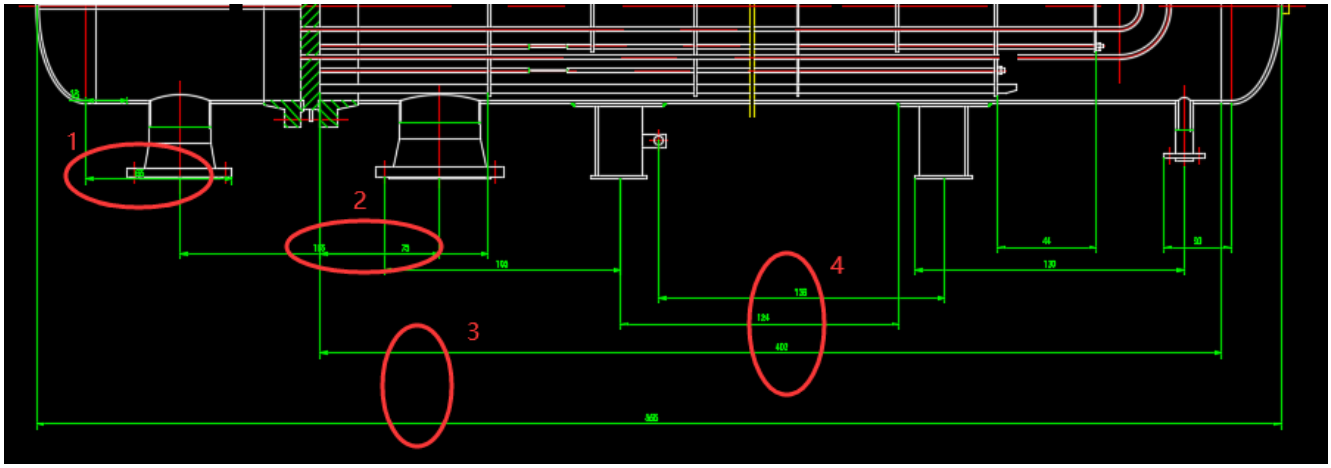

Figure 1 Dimensional layout frequently asked questions

2 RELATED WORK

2.1 Current Status of Research on Automatic Layout of Dimensioning

Currently, most mainstream CAD software packages have incorporated automatic dimension marking functions. In AutoCAD, the automatic marking function is activated through interactive commands. Initially, one selects the object for marking, after which the software automatically generates the marking using the DIM command and displays it on the drawing. Subsequently, the position of the dimension marking can be adjusted by dragging it or utilizing command options. In SolidWorks, upon creating the drawing view, users select the dimension or feature to be labeled within the drawing view. SolidWorks then invokes relevant functions to generate the dimension label, attaching it to the selected feature. Following this, SolidWorks' automatic arrangement function is employed to facilitate the automatic layout of dimensions, albeit often requiring additional adjustment via mouse dragging. For CATIA, NX, Pro/E, Inventor, and other mainstream CAD software, users first select the dimension type and features to be labeled manually. Subsequently, the software's associated tools automatically generate the dimension labels within the drawing. Finally, the layout is completed through mouse dragging or utilizing labeling editing tools. For the current mainstream CAD software interactive labelling operation is a relatively cumbersome problem, and a large number of scholars carry out research on it; the current research is mainly divided into two categories.

2.1.1 Research on Particular Industry

Yi Shengyao et al. [3] proposed a feature-based dimension annotation method for mechanical drawings, achieving accurate and comprehensive annotation of dimensions in mechanical drawings. Wang Yanhong [4] proposed an automatic dimension annotation method for mechanical drawings based on specific business scenarios in the mechanical industry. Guo Huachen et al. [5] conducted secondary development on the PRO/E platform to achieve reasonable annotation of tool dimensions. Yanmei Wang [6] developed a dimension annotation system tailored for the packaging carton industry, addressing the structural characteristics of packaging cartons to achieve automatic dimensioning. Huang Liqiang et al. [7] addressed the structure of irregular folding cartons and achieved automatic dimension annotation for irregular carton sizes in the OpenGL development environment. Zhu Yinfan [8] proposed a dimension annotation method for planar porous components based on specific parts. Wang Hui et al. [9] designed an automatic annotation expert system tailored for steel structure plan and component construction drawings, achieving automatic generation of dimension annotations. Feng Mingyang et al. [10] proposed an adaptive dimension annotation method to address the cumbersome and non-standard procedures in the manual annotation process of engineering drawings for automobile tire tread patterns. Zibin Zhao et al. [11] used functions provided by Pro/E to extract dimensional information from the part information, employing layering to avoid overlapping dimensions and a hash function to determine the location of each layer. Wenjun Xia [12] realized automatic exclusion and dynamic adjustment of overlapping annotation based on a 3D BIM model of a railroad box girder using an FR force guidance improvement algorithm. Li Xuemei et al. [13], and others, addressed the unreasonable positioning of variant size labels during part variant accuracy adjustments, proposing a variant design-based algorithm for adjusting the positioning of size labels. Li et al. [14] studied the automatic layout of data annotation in the process of injection mold design, proposing a dynamic planning method to determine the optimal layout position of data annotations. The research on automatic generation of dimension annotations for specific industries has achieved a certain degree of automation. However, it has the following limitations:

- It can only be applied to specific industries and lacks general applicability.
- It generally applies only to dimension annotations for a small number of individual parts, making it difficult to scale up to large-scale problems.
- The generated dimension annotation layouts often require manual adjustments, indicating a limited level of automation.

2.2.2 Research on Geometric Features

Lu Guodong et al. [15] proposed a dimension annotation strategy based on the branch ideology. This method addresses the integrity issue of three-dimensional annotation and the correctness and clarity issues of two-dimensional annotation by employing a branching strategy for the layout space of dimensions. Liu Fuyun et al. [16], addressing interference and unreasonable spacing in two-dimensional engineering drawings, proposed a method for adjusting dimensioning based on the coordinates of the endpoints of dimensioning arrows and a method for judging dimensioning interference based on vector product analysis. Chen Le [17] proposed an automatic dimension adjustment method to tackle the issue of unreasonable dimension annotation layout. This was achieved through an analysis of the structure and layout characteristics of dimension annotation types. The dimensional intelligent layout algorithm proposed by Tao Wang [18] focuses on horizontal and vertical dimensions. By classifying their subsets and sorting inclusive relations with reference to their characteristics, an incremental approach to dimensional annotation is adopted to facilitate automatic layout. Xueliang Huang [19] and others introduced a meshing method to grid the layout space, using a matrix to express layout space states. This approach reduces the labeled layout problem to a small piecewise matrix problem, which is solved by searching for certain conditions in the state matrix. Yuan Bo [20] proposed the concept of subset partitioning for horizontal and vertical dimensions' characteristics and presented a dimension auto-layout algorithm based on subset partitioning. This algorithm converts dimension auto-layout optimization to dimension subset partitioning optimization, incorporating the simulated annealing algorithm for optimization. While research on geometric features has improved the quality of dimension annotation layouts to some extent, this approach has the following shortcomings:

- It requires a significant amount of effort to handle dimension annotation interference and determine and process drawing boundaries.
- It can only address the layout of dimension annotations on a small scale. When the scale of the problem increases, manual adjustments are still necessary.

2.2 Genetic Algorithms and their Improvement

GA (Genetic Algorithm), proposed by Holland [21] in 1973, is an optimization algorithm based on the theory of biological evolution for the problem of finding an optimal or near-optimal solution. It simulates the genetic and evolutionary processes in nature, searching the solution space of a problem by iteration and selection, and gradually optimizing the quality of the solution. GA algorithms have been widely used in the field of automated design since its proposal, including electronic circuit design, aerospace design, architectural design, chip design, and many other fields, which have significantly improved the efficiency and quality of automated design. As the complexity of the problem increases, the GA algorithm is easy to fall into local optimization and slow solution speed in practical applications. Traditional GA algorithms use fixed crossover probability and mutation probability. Too high crossover probability and mutation probability may lead to premature convergence or insufficient diversity of the population [22], while too low probability may lead to slower convergence. To address the above problems, Srinivas [23] proposed an adaptive genetic algorithm (AGA), which dynamically adjusts the crossover and mutation probabilities by the magnitude of individual fitness. The formulae are shown in Eq. (1) and Eq. (2).

$$P_c = \begin{cases} \dfrac{k_1\left(f_{\max} - f'\right)}{f_{\max} - f_{avg}}, f' \geq f_{avg} \\ k_2, f' < f_{avg} \end{cases} \tag{1}$$

$$P_m = \begin{cases} \dfrac{k_3\left(f_{\max} - f\right)}{f_{\max} - f_{avg}}, f' \geq f_{avg} \\ k_4, f' < f_{avg} \end{cases} \tag{2}$$

where P_c and P_m are the crossover probability and mutation probability respectively, f' is the larger fitness among the two individuals to be crossover, f_{avg} is the average value of the population fitness, f is the fitness of the mutated individuals, and $k_1 \sim k_4$ are the adaptive control parameters. However, this adaptive GA algorithm only considers the relationship between the general individuals and the optimal individuals in the population, and it is still possible to fall into the local optimum. Therefore, the adaptive adjustment for the genetic algorithm process should take into account the overall evolutionary degree of the population and the fitness of individuals, which is also a direction of algorithm improvement in this paper.

3 RELATIONSHIP BETWEEN DIMENSIONING

In engineering drawings, dimensioning is a method of identifying the dimensions of various parts of a part or assembly and is one of the most important means of ensuring that manufactured parts meet design requirements. This chapter will introduce the detailed structure of dimensioning in engineering drawings as well as the relationship and representation between dimensioning, and then propose a method for classifying the relationship of dimensioning in engineering drawings to facilitate the construction of mathematical models for this problem.

3.1 Definition of Dimensioning

As shown in Fig. 2, the dimensioning generally contains the following elements. (1) Dimension Lead Points: Dimensioned lead points are marked points or marked line segments used to indicate the location of a dimensioned reference point, usually used to indicate the beginning and end of a dimension. (2) Dimension Lead Direction: Dimension has four lead directions relative to the drawing, respectively, for the drawing above, below, left, right, and and so on. (3) Dimension Boundaries: Dimensional boundaries usually consist of a set of parallel lines that identify the extent and area of dimensional information. (4) Dimension line: A dimension line is a line marking dimensional information in a drawing, usually connecting and perpendicular to two-dimensional boundaries. (5) Dimension text: Dimension text is the text content of the labelled dimensional information.

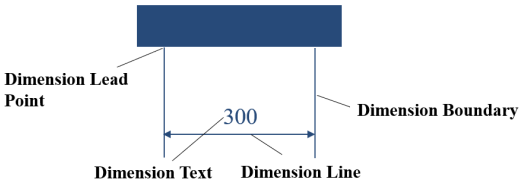

Figure 2 Dimensioning structure

3.2 Definition of Dimensioning Relationships

The dimensioning in engineering drawings is numerous and the relationship is complex. With the largest number of drawings, the most complex relationship between the horizontal and vertical dimension marking is the object of study; through the analysis of a large number of engineering drawings, the relationship between the horizontal and vertical dimension marking is summarized and generalized. Assuming that the horizontal coordinates of the two lead-in points for dimensioning are d_p1_x and d_p2_x, and taking dimensions $d1$ and $d2$ as an example, the following categorization of dimensioning relationships can be made. The non-containing type indicates that the labeling lines of the two dimensions do not intersect or intersect only at the endpoints of the two labeling lines. Regardless of the layout, this type of relationship between two dimension annotations will not result in overlapping dimension lines. In a two-dimensional coordinate system, the coordinates of the dimension extension line endpoints satisfy:

$$d1\_p2\_x \leq d2\_p1\_x \tag{3}$$

Their positional relationships are shown in Fig. 3.

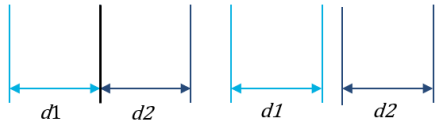

Figure 3 Non-inclusive

The All-inclusive type signifies that there are two intersections between the dimension lines of two dimension annotations. It is imperative to avoid placing dimension annotations of this relationship type on the same layer, as it will result in overlapping dimension lines. Following the principle of minimal intersections, the common layout approach is to have the dimension with the larger value encompass the dimension with the smaller value. In a two-dimensional coordinate system, the coordinates of the dimension extension line endpoints satisfy:

$$\begin{cases} d1\_p1\_x \geq d2\_p1\_x \\ d1\_p2\_x \leq d2\_p2\_x \end{cases} \tag{4}$$

Their positional relationships are shown in Fig. 4.

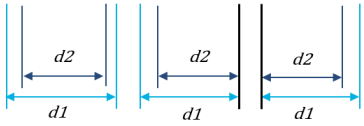

Figure 4 All-inclusive

The Semi-inclusive type denotes that the dimension lines of two dimension annotations intersect at a non-endpoint. It is also necessary to avoid placing dimension annotations of this relationship type on the same layer. While the layout is relatively flexible, different layout approaches will ultimately affect the total number of layers. In a two-dimensional coordinate system, the coordinates of the dimension extension line endpoints satisfy:

$$\begin{cases} d1\_p2\_x > d2\_p1\_x \\ d1\_p1\_x < d2\_p1\_x \\ d1\_p2\_x < d2\_p2\_x \end{cases} \tag{5}$$

Their positional relationships are shown in Fig. 5.

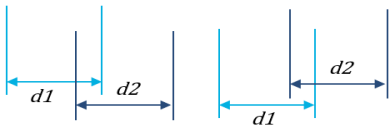

Figure 5 Semi-inclusive

4 DIRECTED ACYCLIC GRAPH MODEL OF DIMENSIONING RELATIONSHIPS

This chapter establishes a directed acyclic graph model for dimension annotation problems based on the summary and induction of dimension annotation relationships discussed earlier. Additionally, the problem of automatic dimension annotation layout is transformed into a combinatorial optimization problem based on the directed acyclic graph, and its mathematical formulation is provided.

4.1 Representation of Dimensioning Relationships

As shown in Fig. 6, based on the definition of dimensioning relationships proposed in 3.2, a representation of these relationships is given in the figure, where nodes in the graph denote dimensions and edges denote inclusion relationships between dimensioning annotations.

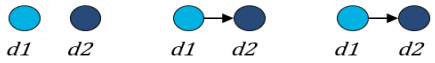

Figure 6 Dimension Relationship Representation

4.2 Directed Acyclic Graph Model of Dimensioning Relationships

A directed acyclic graph is a graph structure consisting of nodes and directed edges [24]. As an important concept in graph theory, it has been widely applied across various fields, including task scheduling, computer vision, data flow processing, compiler optimization, and more. In the context of the dimensioned layout problem, this paper employs a directed acyclic graph to model it, offering several key advantages:

- Directed acyclic diagrams do not contain any loops, which avoids loop inclusion between dimensioning.
- For the dimensioning of the full inclusion relationship, set the direction of the edges to be the node represented by the size with a large value pointing to the node represented by the size with a small value, which ensures that the dimensioning under the full inclusion relationship has the least number of intersections.
- Layering the dimension annotations through the path relationship between the nodes can effectively solve the problem of overlapping interference between the dimension annotations, avoiding the judgment of a large number of dimension annotations interfering with the overlap, and only needing to set an appropriate layer spacing to generate the dimension annotation layouts that are aligned in each layer and have no overlapping interference.

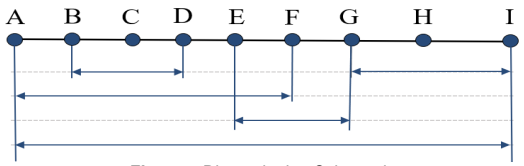

Figure 7 Dimensioning Schematic

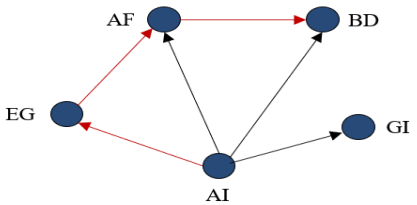

Figure 8 DAG model for dimensioning

Based on the classification and representation of dimension relationships in the previous text, as well as the

annotation specifications for dimension labeling in engineering drawings, a directed acyclic graph is constructed to model dimension annotation relationships. Taking the dimension annotation structure shown in Fig. 7 as an example, a directed acyclic graph model similar to that shown in Fig. 8 can be constructed.

4.3 Transformation and Formulation of Combinatorial Optimization Problems

As can be seen from 4.2, this paper introduces the directed acyclic graph model. According to the path relationship in the graph, it avoids the judgment of interference conflict between dimension annotations and also simplifies the objective function of the following optimization model, which is more conducive to the solution of the problem. According to the principle of least intersection, for the dimension annotations with full inclusion relationship, the direction of the edges between them is determined, i.e., the node represented by the dimension annotation with large value points to the node represented by the dimension annotation with small value. However, for dimensioning with a semi-inclusive relationship, their up-down relationship will not affect the number of intersection points but will affect the total number of layers after the dimensioning layout. As shown in Fig. 8, the path of the red arrow in the figure is the longest path of the directed acyclic diagram. It is easy to know that the total number of layers of the dimensioned layout and the longest path of all the paths in the corresponding directed acyclic diagram have the following relationship: Assuming that S is the length of the longest path of the directed acyclic diagram, and N is the total number of layers of the dimensioned layout of the engineering drawings, and assigning the same weights to each edge, which is 1 by default, then the relationship as shown in Eq. (6) is satisfied.

$$S = N - 1 \tag{6}$$

Based on the above analysis, a formulation of the dimensioned layout problem can be given: in a directed acyclic graph G, there are m edges with certain directions and n edges with uncertain directions. Determine the directions of these n edges so that the longest path S of the directed acyclic graph is the shortest. Obviously, the complexity of the problem is of exponential level and belongs to the NP-Hard problem, which is also a combinatorial optimization problem.

5 AUTOMATIC LAYOUT ALGORITHM FOR DIMENSIONING BASED ON IAGA (IMPROVED ADAPTIVE GENETIC ALGORITHM)

This chapter focuses on establishing an optimization model for solving the above problem and proposing an IAGA-based layout algorithm for dimensioning.

5.1 Optimization of Model Construction

5.1.1 Objective Function

As shown in Eq. (7), the longest path of the directed acyclic graph is chosen as the objective function.

$$Min\ S \tag{7}$$

5.1.2 Decision Variables

In the problem of this paper, it is necessary to determine the direction of n edges with indeterminate directions, and the direction of each edge with indeterminate direction can be used as a decision variable. Suppose the set of edges is E. For each edge $(u, v) \epsilon E$, let its direction variable be X_{uv}. The decision variable is shown in Eq. (8).

$$\begin{cases} X_{uv} = 0,\ u \to v \\ X_{uv} = 1,\ v \to u \end{cases} \tag{8}$$

5.1.3 Constraints

(1) Each randomized solution generated cannot have loops, i.e., each topological sequence corresponding to a DAG has to contain all the nodes of the graph. Assuming that the set of nodes of the graph is V and the topological sequence is T, their relationship satisfies Eq. (9).

$$V = T \tag{9}$$

(2) For the edges whose directions have been determined, the corresponding values of the variables are 0 or 1, i.e., for each edge $(u, v) \epsilon E$, if the direction of the edge has been determined, the values of the variables satisfy Eq. (10).

$$X_{uv} = 0 \text{ or } X_{uv} = 1 \tag{10}$$

5.2 Algorithm Design

The general flow of the IAGA-based algorithm for solving this problem is shown in Fig. 9.

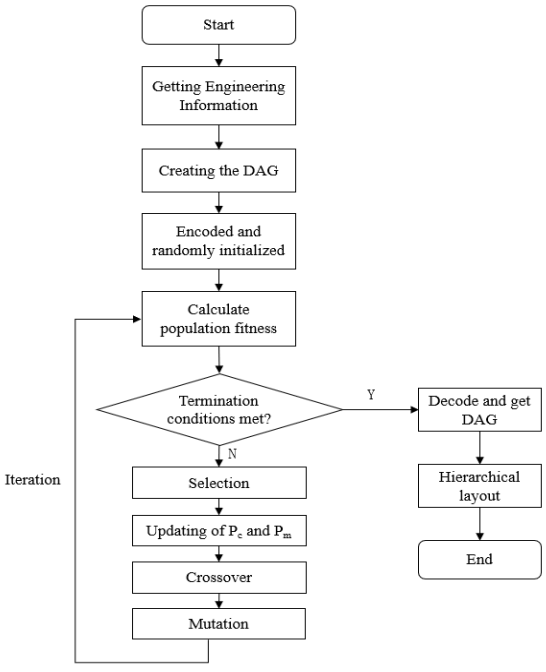

Figure 9 General flow of the algorithm

5.2.1 Directed Acyclic Graph Processing

In this paper, we store the directed acyclic graph as an adjacency matrix and use a Boolean two-dimensional array to mark the state of each edge in the graph and do the following with the storage matrix and the state array.

- An edge whose direction is determined is assigned a value of 0 or 1 to indicate the direction of the edge, and the state of the edge in the state array is set to True.
- An edge with uncertain direction is temporarily set to null and the status of the edge in the status array is set to False.

5.2.2 Coding Method

In the context of the dimension annotation automatic layout problem addressed in this article, variables represent the direction of edges in a directed acyclic graph (DAG), where each edge has two possible directions corresponding to binary encoding. Therefore, employing binary encoding provides a more convenient representation of the solution to this problem and reduces the additional workload incurred during decoding. The specific encoding is illustrated in Fig. 10. Initially, based on the containment relationships of dimension annotations, a corresponding DAG is constructed. In this DAG, black paths represent edges corresponding to full containment relationships, with their directions predetermined, while red paths represent edges corresponding to partial containment relationships with undetermined directions. Subsequently, an edge state matrix is constructed, where edges with determined directions are denoted by 0, and those with undetermined directions are denoted by 1. Finally, based on the edge state matrix, the edges with undetermined directions are transformed into one-dimensional chromosomes for subsequent fitness function calculation and genetic operations of the algorithm. Each binary digit represents the direction of an edge.

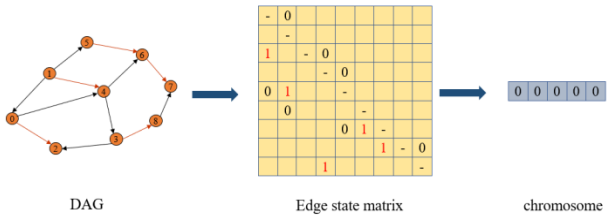

Figure 10 Example of encoding method

5.2.3 Fitness Function

In this paper, we design a dynamic programming algorithm based on depth-first traversal (DFS) to solve the length of the longest path of a directed acyclic graph as a fitness function of the genetic algorithm.

Step 1. Define an integer array *dp* to hold the longest path length for each node.

Step 2. Define a boolean array *visited* to mark whether individual nodes have been visited.

Step 3. For each node *i*, if it has not been visited, the depth-first traversal algorithm *DFS* is invoked to search it.

Step 4. In the *DFS* algorithm, the current node is first marked as visited, and then all neighbor nodes of the current node are traversed. For each neighbor node *j*, if it has not been visited yet, the *DFS* function is called recursively to search. At the same time, the longest path length *dp*[*i*] of the current node *i* is updated so that it is equal to the larger path length of the current node and the longer path lengths reached through all neighboring nodes.

Step5. Iterate through the *dp* array and find the maximum value which is the longest path length of the directed acyclic graph.

5.2.4 Improvements in Adaptive Crossover and Mutation Probabilities

The crossover probability P_c and the mutation probability P_m are important parameters affecting the convergence speed and searchability of genetic algorithms. The adaptive adjustment formulae of crossover probability and mutation probability in the AGA algorithm lack the consideration of the evolutionary degree of the whole population. Dispersion coefficient, as a statistical concept, is a statistical indicator of variability or volatility in a data set, which can be used to assess the dispersion of the whole population fitness, and its formula is shown in Eq. (11).

$$CV_n = \frac{\sqrt{N^{-1}\sum_{k=1}^{N}\left(f_k - \bar{f}\right)^2}}{N^{-1}\sum_{k=1}^{N} f_k} \tag{11}$$

where n is the number of generations of the current population evolution, N denotes the population size, f_k denotes the fitness of the individuals of the current population and $\bar{f}$ is the average of the fitness of all individuals of the current population. As the number of genetic generations of the algorithm increases, the quality of the population becomes higher and higher, and the degree of discretization becomes smaller and smaller, that is, CV_n becomes smaller and smaller, and gradually tends to zero. Therefore, according to the change rule of discrete coefficients and combined with Eq. (1) and Eq. (2) in the AGA algorithm to give the improved adaptive adjustment formula as shown in Eq. (12) and Eq. (13).

$$\begin{cases} P_c = P_c + a\dfrac{f_{\max} - f'}{f_{\max} - f_{\min}}, & \dfrac{CV_n}{CV_0} > b \\ P_c, & else \end{cases} \tag{12}$$

$$\begin{cases} P_m = P_m + c\dfrac{f_{\max} - f}{f_{\max} - f_{\min}}, & \dfrac{CV_n}{CV_0} > d \\ P_m, & else \end{cases} \tag{13}$$

where P_c and P_m are the crossover probability and mutation probability respectively, $f_{\max}$ and $f_{\min}$ are the maximum and minimum values of the fitness in the current population respectively, f' is the larger fitness in the two bodies to be crossed, a, b, c, d are the adaptive control parameters, and CV_0 is the initial coefficient of discretization, and it is easy to know that $CV_n / CV_0 \in [0, 1]$, and with the increase in the number of evolution generations, it gradually converges to 0. When CV_n / CV_0 is smaller than a specific set value, the

crossover and mutation probabilities are gradually increased, where the larger the values of a and c, the more obvious the change in probability, thus compensating for the lack of consideration of the overall evolutionary level of the population in the AGA algorithm.

5.2.5 Selection Operations

For the dimension annotation automatic layout problem, a selection strategy combining a tournament selection operator with an elitism selection operator is designed. The tournament selection operator is a simple and efficient selection strategy that simulates tournaments. In each round of the tournament, a certain number of individuals (referred to as the tournament size) are randomly selected from the current best population. Among them, the individual with the best (or worst) fitness is chosen as part of the parent population. The tournament format introduces randomness, ensuring the diversity and exploratory nature of the algorithm. The elitism selection operator is a strategy that preserves the individuals with the best fitness in the current population and directly passes them to the next generation. Therefore, the combination of the tournament selection operator and the elitism selection operator effectively balances the requirements of local and global search, thereby improving the performance and adaptability of the algorithm. The detailed steps are as follows:
Step1: Evaluate the performance of each individual in the population in the solution space using the fitness function to determine the fitness value of each individual.
Step 2: Select several individuals with the highest fitness values from the current population, referred to as elite individuals, and directly transfer them to the next generation population.
Step 3: Perform multiple tournament selections for the remaining vacancies in the population. In each tournament selection, randomly select a certain number of individuals from the population, and then choose the individual with the highest fitness as the winner. Repeat this process until the population size reaches the preset value.

5.2.6 Crossover Operations

As illustrated in Fig. 11, for the dimension annotation layout problem, a uniform crossover operator is employed for crossover operations. Initially, two parent individuals are selected from the population, and a set of crossover mask arrays, $p[]$, of the same length as the individual's genes, is randomly generated, with each value in the array ranging from [0, 1]. Subsequently, each gene position is compared with the crossover probability, Pc, in sequence. If $p < Pc$, crossover occurs at that position; otherwise, no crossover occurs. Finally, after completing the crossover, newly generated individuals are obtained.

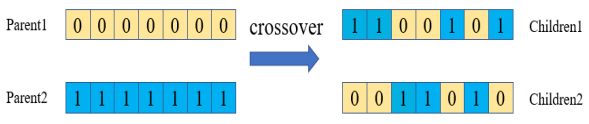

Figure 11 Example of a crossover operator

5.2.7 Improvement of Mutation Operations

In order to better control the impact of mutations, this paper proposes a mutation operator based on the characteristics of paths in directed acyclic graphs (DAGs), combined with the adaptive mutation probability proposed earlier. This design allows the algorithm to balance the diversity of the population and its search capability during the iterative process, thereby enhancing the overall performance of the algorithm. For a node in a directed acyclic graph, its indegree refers to the number of directed edges pointing to that node, while its outdegree refers to the number of edges originating from that node and pointing to other nodes. For two nodes corresponding to two partially contained dimension annotations (connected by an edge with an undetermined direction), when the direction of the edge points to the node with the smaller indegree, it is more likely to increase the length of the path where the edge belongs, thus resulting in an increase in the length of the longest path in the entire graph. Similarly, when the direction of the edge points to the node with the larger indegree, the probability of decreasing the length of the longest path in the graph increases. The operation of the mutation operator is illustrated in Fig. 12, and its specific steps are as follows:
Step 1: Select the individual to be mutated and determine the gene positions to be mutated based on the mutation probability.
Step 2: Locate the two nodes, i and j, connected by the edge corresponding to the gene position in the DAG.
Step 3: Retrieve the indegrees In[i] and In[j] of the two nodes and the edge Eij between them from the indegree array.
Step 3.1: If $In[i] < In[j]$, change the direction of edge Eij to point from i to j.
Step 3.2: If $In[i] > In[j]$, change the direction of edge Eij to point from j to i.
Step 4: Obtain the mutated individual and add it to the population.

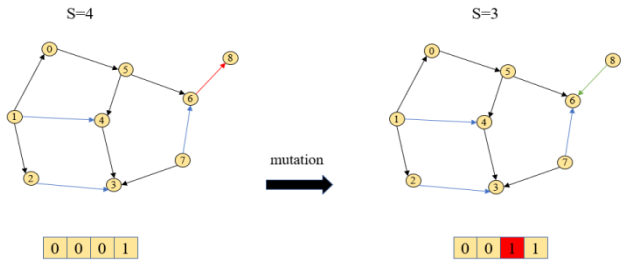

Figure 12 Principle of operation of mutation

5.2.8 Dimensional Hierarchy Algorithm

Finally, the directed acyclic graph is obtained after the optimization of the algorithm. In this paper, we design a topology sorting-based dimensions hierarchical layout algorithm with the following steps:
Input: directed acyclic graph G.
Output: dimensional layering result set *layers*.
Step 1: Initialize a two-tier collection *layers* and a queue Q.
Step 2: Calculate the in-degree of each node in the graph and add the nodes with in-reading of 0 to the queue Q.

Step 3: When queue *Q* is not empty, perform the following.

Step 3.1: Initialize an empty collection *layer* to store the nodes of the current layer.

Step 3.2: Iterate through all the nodes in the queue and for each node *u* perform the following.

Step 3.2.1: Add node *u* to the list of *layer*.

Step 3.2.2: For all successor nodes *v* of node *u*, reduce their in-degree by one.

Step 3.3.3: If the in-degree of node *v* becomes 0, add it to queue *Q*.

Step 3.3: Add the current *layer* to the list of *layers*.

Step 4: Returns *layers* as the result.

6 EXPERIMENTAL ANALYSIS

6.1 Validation of the Effectiveness of the IAGA Algorithm

In this paper, the algorithm code is written on Visual Studio 2022 integrated development environment using C# programming language, different numbers of dimension labeling coordinates are randomly generated, experiments are conducted under the same set of coordinate data, and comparative experiments are conducted with GA and AGA algorithms under the same algorithm parameters. The parameter settings of the algorithm are shown in Tab. 1. Five sets of data with the number of labeled points of 30, 50, and 100, and 800 were randomly generated for the experiment, and the details of the data are shown in Tab. 2.

Table 1 Algorithm Parameter Setting

Number of dimensions	parameter setting
30	Population size: 50 Maximum number of evolutionary generations: 200 Fixed evolutionary parameter: 0.7 0.05 Adaptive evolutionary parameters: [0.5, 0.9] [0.01, 0.1]
50	Population size: 50 Maximum number of evolutionary generations: 400 Fixed evolutionary parameter: 0.7 0.05 Adaptive evolutionary parameters: [0.5, 0.9] [0.01, 0.1]
100	Population size: 50 Maximum number of evolutionary generations: 700 Fixed evolutionary parameter: 0.7 0.05 Adaptive evolutionary parameters: [0.5, 0.9] [0.01, 0.1]
800	Population size: 50 Maximum number of evolutionary generations: 1200 Fixed evolutionary parameter: 0.7 0.05 Adaptive evolutionary parameters:[0.5, 0.9] [0.01, 0.1]

Table 2 Experimental data information

Number of dimensioning	Number of all-inclusive and non-inclusive relationships	Number of semi-inclusive relationships
30	217	52
50	679	146
100	2661	510
800	9423	2437

As can be seen from Fig. 13, with a data volume of 30, all three algorithms, GA, AGA and IAGA, converge to the same value after the algorithms evolve for 200 generations, but there are differences in the performance of the algorithms, the GA and AGA algorithms converge at about 160 and 150 generations, respectively, but the IAGA algorithm, which is improved by this paper, converges at about 130 generations, and it can be verified that the IAGA algorithm outperforms the two algorithms, GA and AGA, at a data volume of 30. As can be seen from Fig. 14, with a data amount of 50, there is no obvious gap between the convergence images of the three algorithms before 50 generations, but after 100 generations of evolution, the gap between the three algorithms is gradually obvious, and finally the GA algorithm converges at about 300 generations, the AGA algorithm converges at about 270 generations and the convergence value is improved by 4 compared to the GA algorithm, while the IAGA algorithm converges at about 220 generations and the the convergence value is improved by 4 and 2 over the GA and AGA algorithms, respectively, indicating that the improved algorithm in this paper outperforms the other two algorithms in terms of performance and solvability at a data size of 50.

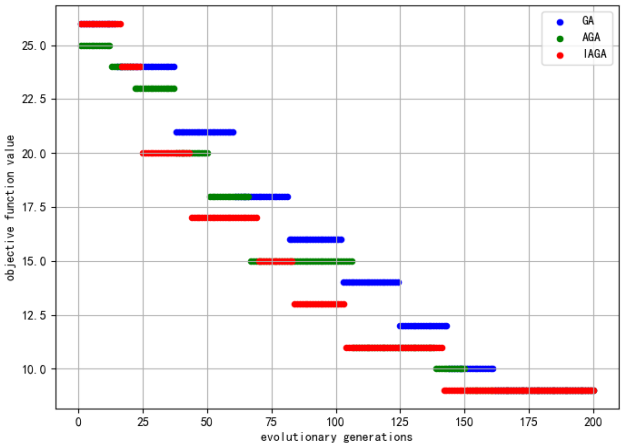

Figure 13 Convergence image for a data volume of 30

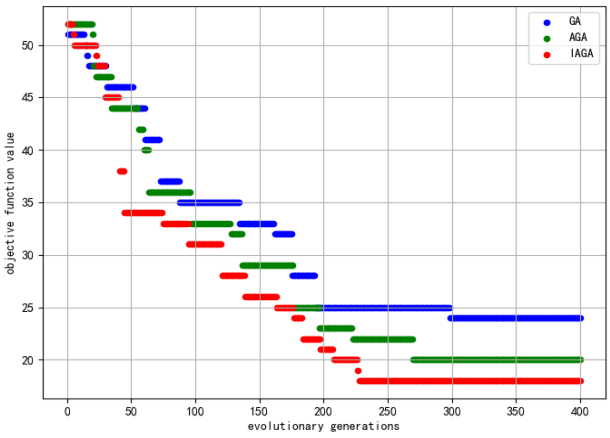

Figure 14 Convergence image for a data volume of 50

As can be seen in Fig. 15, at a data size of 100, the gap between the three algorithms begins to be apparent when the algorithms evolve for 200 generations, with the GA algorithm beginning to converge at about 530 generations, the AGA algorithm beginning to converge at about 460 generations, with an improvement in convergence value of 3 over the GA algorithm, and the IAGA algorithm beginning to converge at about 380 generations and with an improvement in convergence of 2 and 4 over the GA and AGA algorithms, respectively. From Fig. 16, it can be observed that when the data size is 800, the optimization

ability of the GA algorithm is relatively poor, as it did not converge after 1200 iterations. Compared to the GA algorithm, the AGA algorithm shows some improvement in solving capability, with an increase of 29 in the objective function value. Meanwhile, the IAGA algorithm exhibits excellent performance on larger-scale data, converging at around 780 iterations with an objective function value of 139. This represents an improvement of 74 and 45 compared to the GA and AGA algorithms, respectively.

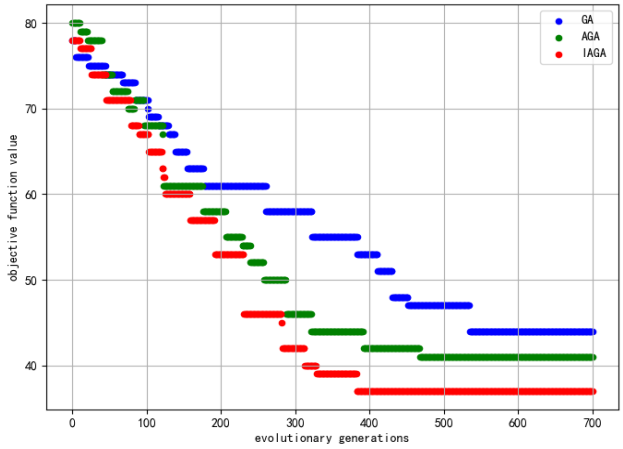

Figure 15 Convergence image for a data volume of 100

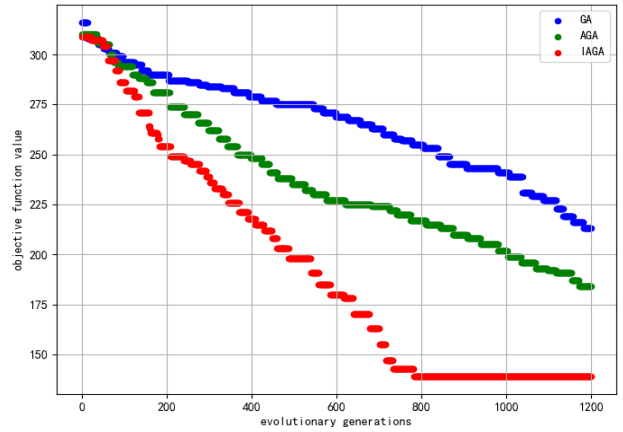

Figure 16 Convergence image for a data volume of 800

In summary, under four different data sizes, the improved IAGA algorithm proposed in this paper demonstrates superior performance and solving capability compared to the other two algorithms. Moreover, as the data scale increases, the performance improvement of our algorithm compared to the other two algorithms becomes more significant, indicating the effectiveness of the solution approach and improvement strategy proposed in this paper.

6.2 Practical Scenario Applications

In this paper, the effectiveness of the algorithms in this paper is verified on engineering drawings of real applications using C# language and Visual Studio 2022 integrated development environment and AutoCAD 2017 secondary development interface. Petrochemical static equipment refers to stationary equipment or equipment components [25] used in the petrochemical industry for storing, processing, or transferring crude oil, petroleum products, and chemical products without involving mechanical movement. Heat exchanger is a common type of equipment in static equipment, mainly used in the transfer of heat between different fluids; in static equipment, the main role of the container is to store, separate, cool, heat, and some chemical reactions. Comparative experiments were conducted on the assembly diagrams of two complex devices in static equipment: heat exchangers and containers. As depicted in Fig. 17 and Fig. 18, the dimension annotation layouts for the heat exchanger and container, respectively, were generated using AutoCAD parametric driving. In contrast, Fig. 19 and Fig. 20 display the dimension annotation layouts for the heat exchanger and container generated using the method proposed in this paper. Through comparative analysis, specific benefits of the IAGA method can be inferred as follows:

- Addressed the interference issue between most dimension lines and patterns, enhancing the readability of the drawings.
- Addressed the problem of dimension line overlap, resulting in clearer expression of dimension annotation information.
- Alleviated the issue of uneven spacing between dimension annotation layers, leading to a more reasonable layout.
- Mitigated the problem of excessive layers of dimension annotation, reducing wastage of drawing space.
- In addition, the method proposed in this paper also has the following limitations:
- It is unable to achieve automatic adjustment of dimension annotation text, resulting in potential interference between text and adjacent dimension line leaders.
- It cannot completely avoid interference between dimension lines and patterns in dimension annotations.

The paper also compared the time performance of the IAGA method with the interactive method currently used for manual adjustment. For the dimension annotation layouts of the heat exchanger and the container, the IAGA method was run 30 times each, and the average time spent was recorded. Additionally, three proficient users of CAD software manually adjusted each layout 10 times, and the average adjustment time for each layout was calculated for each user. The comparative results are shown in Tab. 3, where the efficiency of the IAGA method, compared to the current interactive operation approach, improved by 93.6% and 94.3% for the heat exchanger and container dimension layouts, respectively.

Table 3 Time Performance Comparison

Equipment Name	Manual adjustment time	IAGA Runtime	Enhanced efficiency
heat exchangers	76 s	4.83 s	93.6%
vessel	69 s	3.92 s	94.3%

Experimental results demonstrate that our proposed method can be well applied to the dimension annotation layout of engineering drawings in the static equipment industry, further validating the correctness of our problem transformation and modeling. Our method exhibits good scalability and can be extended to various other industries, providing solutions for the development and optimization of current mainstream CAD software. However, due to industry barriers, certain modifications to the algorithm are

required during practical application to align with actual business requirements. Additionally, since heuristic algorithms inherently involve randomness, our method may lack stability under certain circumstances.

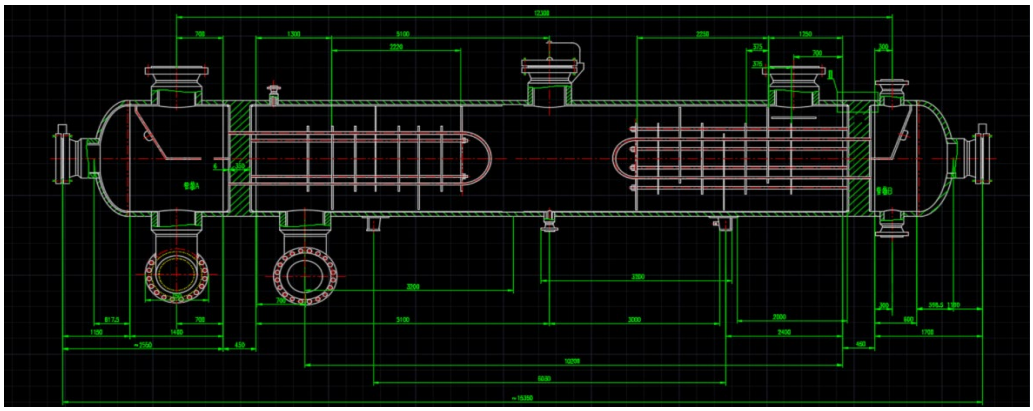

Figure 17 Heat exchanger dimensioning parameterization drive results

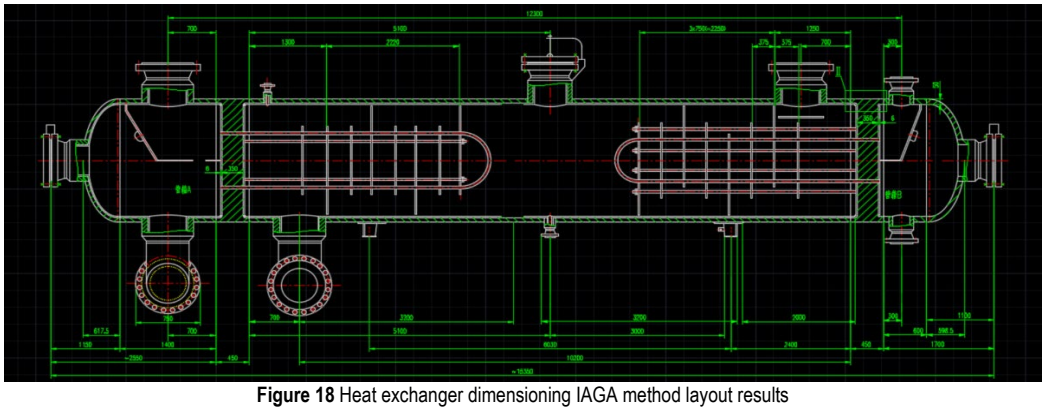

Figure 18 Heat exchanger dimensioning IAGA method layout results

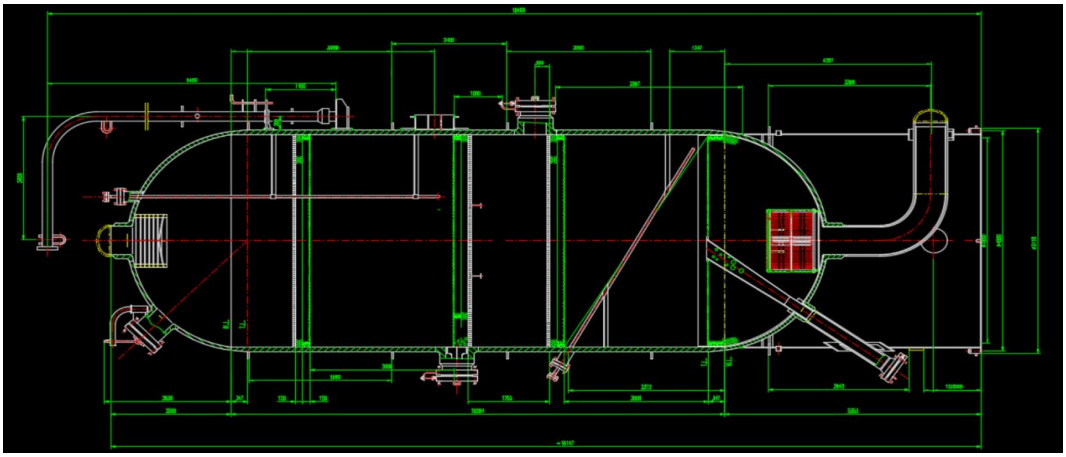

Figure 19 Vessel dimensioning parameterization drive results

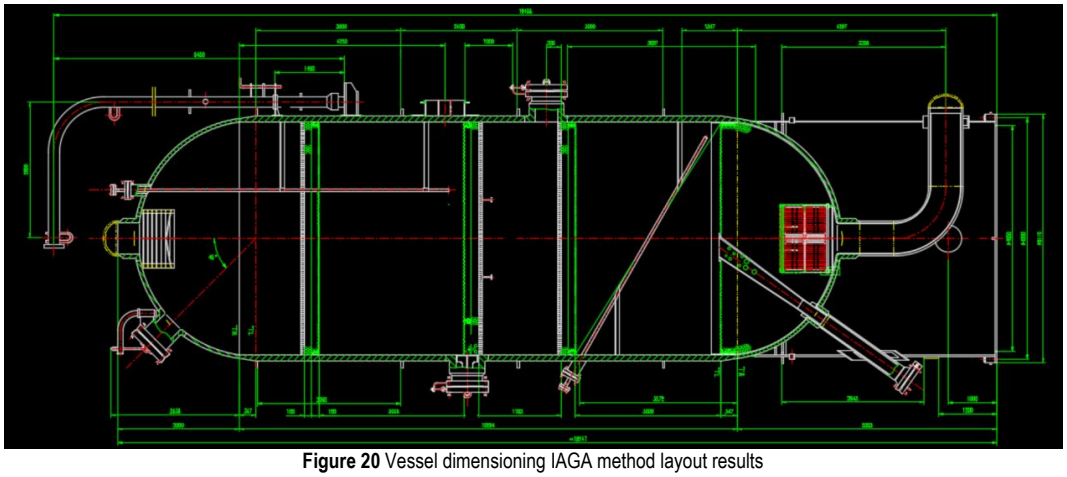

Figure 20 Vessel dimensioning IAGA method layout results

7 CONCLUSION

In conclusion, this paper develops an automatic dimensioning layout method for engineering drawings by modeling dimensional relationships as a directed acyclic graph optimization problem, solvable via an improved adaptive genetic algorithm. The core innovations encompass the formal dimensioning characterization, DAG-based problem transformation and specifically designed evolutionary operators. Extensive simulation experiments and practical case studies on heat exchanger and vessel dimensioning validate the method's efficacy in generating high-quality dimensional layouts completely automatically. This work successfully automates a previously manual dimensioning bottleneck in computer-aided design workflows, significantly boosting efficiency. With further extensions to cover supplementary dimension types and aesthetic criteria, the proposed technique shows immense promise for real-world engineering drawing applications.

8 REFERENCES

[1] RAO, S. (2000). *Research and Implementation of Automatic Engineering Drawing Generation Technology*. Graduate School of Chinese Academy of Sciences (Institute of Computing Technology).

[2] BAO, L. (2018). *Automatic Dimensioning of Engineering Drawings Based on Feature Recognition*. Dalian University of Technology.

[3] Yi, S. Y., Cao, Y. J., & Liang, B. (2021). Research on the Dimension Marking Method of Mechanical Parts Drawing Based on Feature Dimensions. *Mechaniacl Engineer,06*, 3.

[4] Wang, Y. H. (2022). Research on Dimensioning Techniques for Mechanical Drawings. *Papermaking Equipment & Materials, 51*(11),102-104.

[5] Guo, H. C., Luo, H. B., & Cheng, H. W. (2007). Application of Secondary Development of Pro/E on Cutting Tool Drawing Dimensioning. *Tool Engineering, 41*(10), 55.

[6] Wang, Y. M. (2006). *Dimensioning Research and Software Development for Packaging Carton CAD System*. Tianjin University of Science and Technology.

[7] Huang, L. Q., Wu, D. B., & Huang, Y. (2011). Research on Dimensioning Methods of CAD System for Shaped Folding Cartons. *Packaging Engineering, 32*(13), 35-38.

[8] Zhu, Y. F. (2000). Size label of lacunosis apart. *Journal of Zhejiang University of Technology.*

[9] Wang, H. & Li, C. D. Automatic dimensioning of steel structure design expert systems. *Journal of Guangxi University of Science and Technology, 7*(4), 32-34.

[10] Feng, M. Y., Ying, L. H., Sun, G. J., Dong, Y. D., Zhang, F. L., & Liu, Y. C. (2018). Adaptive Processing of Dimensioning Tire Patterns in Engineering Drawings. *Chinese Journal of Automotive Engineering, 008*(004), 287-269.

[11] Zhao, Z. B. & Niu, Q. Z. (2014). The Research and Implementation of Auto-Dimension in Engineering Drawing Based on Pro/E. *Applied Mechanics and Materials, 615*, 615-619. https://doi.org/10.4028/www.scientific.net/AMM.610.615

[12] Xia, W. J. (2020). Research on Intelligent Layout Algorithm for Reinforcement Drawing of Railway Box Beam. *Railway Standard Design, 64*(06), 88-92.

[13] Li, X. M., Chen, Z. Y., & Chang, Q. Q. (2017). Study Design Variant Dimension Accuracy Position Adjustment Algorithm. *Modular Machine Tool & Automatic Manufacturing Technique, 10*, 30-33.

[14] Li, C. L., Lee, Y. H., & Yu, K. M. (2006). Automatic datum dimensioning for plastic injection mould design and manufacturing. *The International Journal of Advanced Manufacturing Technology, 28*, 370-378. https://doi.org/10.1007/s00170-004-2374-2

[15] Lu, G. D., Huang, C. L., & Peng, Q. S. (2001). Research on automatic dimensioning based on divide and conquer strategy. *Journal of Computer-Aided Design & Computer Graphics, 13*(6), 521-526.

[16] Liu, F. Y., Geng, L. D., Jiang, Y. Z., & Wu, Q. (2022). An Adaptive Adjustment Method for Dimension of Engineering Drawings. *Machiney Design & Manufacture, 10*, 186-190 + 195.

[17] Chen, L., Chen, L., Lang, P. C., & Ding, W. J. (2016). Research on the Dimension of Engineering Drawings' Automatic Adjusting Technology. *Machine Design & Research, 03*, 73-76.

[18] Wang, T., Mo, R., & WAN, N. (2010). Automatic Layout Algorithm and Realization of Engingeering Drawing Dimension. *Aeronautical Computing Technique, 40*(02), 73-76.

[19] Huang, X. L., Chen, G., Chen, L. P., & Wang, Q. F. (2008). A Grid Partitioning Method for Automatic Placement of Dimensions. *Journal of Computer-Aided Design & Computer Graphics, 20*(8), 1070-1077.

[20] Yuan, B., Huang, G., & Sun, J. G. (2000). Automatic dimension placement algorithm. *Journal of Tsinghua University (Science and Technology), 40*(01), 61-64.

[21] Holland, J. H. & John, H. (1973). Genetic Algorithms and the Optimal Allocation of Trials. *Siam Journal on Computing, 2*(2), 88-105. https://doi.org/10.1137/0202009

[22] Wu, Z. H., Yang, R. F., Guo, C. X., & Ge, S. C. (2020). Steering System Identification Based on Improved Adaptive Genetic Algorithm. *Science Technology and Engineering, 20*(11), 4436-4441.

[23] Srinivas, M. & Patnaik, L. M. (1994). Adaptive probabilities of crossover and mutation in genetic algorithms. *IEEE Trans on Systems Man and Cybernetics, 24*(4), 656-667. https://doi.org/10.1109/21.286385

[24] Walker, D. M., Corrêa, D. C., & Algar, S. D. (2023). Directed Acyclic Graph networks to characterize phase space evolution with application to musical composition and industrial maintenance. *Expert Systems with Applications, 211*, 118586. https://doi.org/10.1016/j.eswa.2022.118586

[25] Cong, G. P. (2013). *Study on Intelligent Decision-making of Inspection and maintenance Based on Risk and Condition for Petrochemical Equipment*. Dalian University Of Technology.

Contact information:

Yunlei SUN, Associate professor
(Corresponding author)
Qingdao Institute of Software,
College of Computer Science & Technology,
China University of Petroleum (East China),
Qingdao, 266580, China
E-mail: sunyunlei@upc.edu.cn

Zhaotong SHAO, Master
Qingdao Institute of Software,
College of Computer Science & Technology,
China University of Petroleum (East China),
Qingdao, 266580, China
E-mail: 1980177866@qq.com

Bingyi YAN, Master
Qingdao Institute of Software,
College of Computer Science & Technology,
China University of Petroleum (East China),
Qingdao, 266580, China
E-mail: 1010183933@qq.com