Research Article

Solution to Shortest Path Problem Using a Connective Probe Machine

Jiuyun Sun, Huanhe Dong, Yuan Kong, and Yong Fang

College of Mathematics and Systems Science, Shandong University of Science and Technology, Qingdao 266590, China

Correspondence should be addressed to Yuan Kong; kongyuan@sdust.edu.cn and Yong Fang; fangyong@sdust.edu.cn

Received 23 June 2019; Revised 16 September 2019; Accepted 22 October 2019; Published 7 November 2019

Copyright © 2019 Jiuyun Sun et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

With the continuous urban scale expansion, traffic networks have become extremely complex. Finding an optimal route in the shortest time has become a difficult and important issue in traffic engineering study. In this study, a novel computing model, namely, probe machine, is used to solve this problem. Similar to previous studies, urban transport networks can be abstracted into maps, in which points representing places of origin, destinations, and other buildings constitute the data library and edges representing the road make up the probe library. The true solution can be obtained after one probe operation on the computing platform. And by comparing the solving process with Dijkstra’s and Floyd’s algorithms, the computing efficiency of the probe machine is clearly superior, although all three methods can solve the shortest path problem and obtain the same solution.

1. Introduction

Traffic engineering is the basic theory of the research and development of traffic engineering discipline. Its main purpose is to seek the transportation system planning, construction and management scheme with the largest travel efficiency, the least traffic accidents, the fastest traffic speed, the least transportation cost, and the lowest energy consumption. The rapid development of urban traffic and increase in the number of vehicles has made travelling more convenient for people in recent years. However, urban traffic is not smooth due to the numerous traffic network nodes and complex sections; some cities suffer from traffic congestion [1, 2]. Finding a suitable path for drivers is not only one of the solutions to promote the development of urban transportation [3–5], but also one of the main purposes in traffic engineering research. Therefore, we must address this issue by determining the shortest path. In traffic engineering, “shortest path” does not necessarily mean the shortest distance, but also the shortest time or the lowest cost.

The shortest path problem is a classical problem in graph theory, which has been applied in many fields [6]. Finding the path with the shortest distance is the most basic application of the shortest path problem, which is also a very practical problem. The basic objective of the shortest path problem is to find the path, with the lowest weight, between two points where every edge in the graph has its own weight value. Many algorithms to solve the shortest path problem have been proposed in previous studies, such as Dijkstra’s algorithm [7], Bellman–Ford algorithm [8], and Floyd’s algorithm [9]. The most classical algorithm for solving such a problem is Dijkstra’s algorithm. It can solve the shortest path problem from a given point to any point; however, it fails to solve the shortest path problem with a negative weight. Therefore, Floyd’s algorithm and other algorithms were proposed to solve the above problem.

In 2016, Xu proposed a new computing model, i.e., the probe machine [10]. Probe machine is a completely parallel computing model that can simultaneously process multiple pairs of data. Parallel computing can speed up operations, expand the scale of processing problems, and facilitate the ability of algorithms to handle problems [11, 12]. Compared with turing machines, probe machines can solve NP-complete problems after a probe operation, such as the Hamiltonian cycle problem [10], vertex problem [10], travelling salesman problem [13], working operation problem [14], and so on. The probes of a probe machine can be divided into connective probes and transitive probes. A connective probe can connect
data fibers from different data, while a transitive probe can transmit information between data fibers from different data. The probe machine is a form of bionic computing. It achieves complete parallelism by imitating the process of information transmission of neurons and reduces computation complexity. Bionic computing refers to the use of biological models to deal with practical problems, and it can often yield unexpected results in some fields [15, 16]. In recent studies, more computational models are used to solve all kinds of NP problems, such as SAT problem [17–21], vehicle routing problem [22–24], graph coloring problem [10, 25, 26], partner selection problem [27], and the other problems [28–33].

In this study, we first introduce a probe machine. Then, a shortest path problem, which is a problem of finding the path with the minimum distance, is solved by using Dijkstra’s algorithm, Floyd’s algorithm, and probe machine; the process using the three methods is then compared.

2. Probe Machine

The probe machine was first proposed by Professor Xu in 2016. It consists of nine parts including data library (X), probe library (Y), data controller (σ), probe controller (σ'), operation (τ), computing platform (λ), detector (η), true solution storage (Q), and residue controller (C).

The data placement mode of the data library is nonlinear. The data library is divided into n data pools X₁, X₂, ..., Xₙ. Data pool Xᵢ includes one type of data xᵢ, and data xᵢ comprises a body and pᵢ data fibers, as shown in Figure 1.

There are two types of probes in a probe machine, namely, connective probe and transitive probe. A connective probe can connect different data fibers from different data, and a transitive probe can transfer information from the source fiber to a destination fiber. In this study, the connective probe is used to solve the shortest path problem. The set of all probes between two data pools constitutes the probe pool; the probe library is a set of all probe pools, as shown in Figure 2. The data controller and probe controller add the corresponding data and probes into the computing platform, respectively. After the probe operation, the resulting aggregation is detected by a detector. The resulting true solution is put into the true solution storage. The residue is put into the residue collection, which decomposes it into basic data and returns it to the data pool, as shown in Figure 2.

3. Solving the Shortest Path Problem

In this study, an example of a directed graph is considered, as shown in Figure 3. The points on the graph are represented by ν₁, ν₂, ν₃, ν₄, ν₅, ν₆, ν₇, ν₈, ν₉; the distance from νᵢ to νⱼ is represented by wᵢⱼ. The shortest path from ν₁ to ν₉ is obtained. We will use Dijkstra’s algorithm, Floyd’s algorithm, and probe machine to solve the shortest path problem.

3.1. Solving the Shortest Path Problem Using Dijkstra’s Algorithm. P represents the distance from the starting point to point νᵢ, and T refers to the previous point in the shortest path. Sᵢ represents the set of points that have p mark in step i.

1. i = 0, S₀ = {ν₁}, P(ν₁) = 0, λ(ν₁) = 0, T(ν₁) = +∞, λ(νⱼ) = M, (j = 2, 3, ..., 9)
   Because ν₂ ∉ S₀, P(ν₂) + w₁₂ < T(ν₂), let T(ν₂) = P(ν₁) + w₁₂ = 6, λ(ν₂) = 1; Similarly, T(ν₃) = P(ν₁) + w₁₃ = 3, λ(ν₃) = 1; T(ν₄) = P(ν₁) + w₁₄ = 1, λ(ν₄) = 1.
   T(νᵢ) = min{P(νᵢ), T(νⱼ)}, so, let P(ν₁) = 1, S₁ = {ν₁, ν₂}, k = 4.

2. i = 1, let T(ν₂) = P(ν₂) + w₁₂ = 11, λ(ν₂) = 4; T(ν₃) = min{T(ν₂), T(ν₁)} = T(ν₃), so, let P(ν₂) = 3, S₂ = {ν₁, ν₂, ν₃}, k = 3.

3. i = 2, let T(ν₃) = P(ν₃) + w₃₈ = 11, λ(ν₃) = 3; T(ν₄) = min{T(ν₃), T(ν₁)} = T(ν₄), so, let P(ν₃) = 5, S₃ = {ν₁, ν₂, ν₃, ν₄}, k = 2.

4. i = 3, let T(ν₄) = P(ν₄) + w₃₈ = 6, λ(ν₄) = 2; T(ν₅) = min{T(ν₄), T(ν₁)} = T(ν₅), so, let P(ν₅) = 5, S₄ = {ν₁, ν₂, ν₃, ν₄, ν₅}, k = 5.

5. i = 4, let T(ν₅) = P(ν₅) + w₅₈ = 10, λ(ν₅) = 5; T(ν₆) = P(ν₅) + w₅₈ = 9, λ(ν₆) = 5; T(ν₇) = P(ν₅) + w₅₇ = 12, λ(ν₇) = 5; T(ν₈) = P(ν₅) + w₅₈ = 8, λ(ν₈) = 5; T(ν₉) = min{T(ν₅), T(ν₁)} = T(ν₉), T(ν₈), T(ν₇), T(ν₆), so, let P(ν₅) = 8, S₅ = {ν₁, ν₂, ν₃, ν₄, ν₅, ν₆, ν₇, ν₈, ν₉}, k = 9.

6. i = 5, let T(ν₆) = P(ν₆) + w₆₈ = 12, λ(ν₆) = 5; T(ν₇) = min{T(ν₆), T(ν₁)} = T(ν₇), so, let P(ν₆) = 7, S₆ = {ν₁, ν₂, ν₃, ν₄, ν₅, ν₆, ν₇, ν₈, ν₉}, k = 7.

7. i = 6, let T(ν₇) = min{T(ν₆), T(ν₁)} = T(ν₇), so, let P(ν₆) = 9, S₇ = {ν₁, ν₂, ν₃, ν₄, ν₅, ν₆, ν₇, ν₈, ν₉}, k = 6.

8. i = 7, let P(ν₈) = 11, S₈ = {ν₁, ν₂, ν₃, ν₄, ν₅, ν₆, ν₇, ν₈, ν₉}, k = 8.

So, the shortest path from ν₁ to ν₉ is ν₁ → ν₃ → ν₄ → ν₅ → ν₆, as shown in Figure 4.

3.2. Solving the Shortest Path Problem Using Floyd’s Algorithm. According to the graph, the adjacency matrix of the graph is as follows:

\[
\begin{pmatrix}
0 & 6 & 3 & 1 & \infty & \infty & \infty & \infty & \infty \\
6 & 0 & \infty & \infty & \infty & \infty & \infty & \infty & \infty \\
3 & \infty & 0 & 2 & \infty & \infty & \infty & \infty & \infty \\
1 & \infty & 2 & 0 & \infty & \infty & \infty & \infty & \infty \\
\infty & \infty & \infty & 0 & 10 & \infty & \infty & \infty & \infty \\
\infty & \infty & \infty & \infty & \infty & 0 & 2 & \infty & \infty \\
\infty & \infty & \infty & \infty & \infty & 0 & \infty & 4 & \infty \\
\infty & \infty & \infty & \infty & \infty & \infty & 0 & \infty & \infty \\
\infty & \infty & \infty & \infty & \infty & \infty & \infty & 0 & \infty \\
\end{pmatrix}
\] (1)
Figure 1: Data library [10].

Figure 2: Connective probe machine [10].
Step 1: taking $v_2$ as the intermediary point and updating the matrix,

$$A = \begin{pmatrix}
0 & 6 & 3 & 1 & 7 & \infty & \infty & \infty & \infty \\
\infty & 0 & \infty & \infty & 1 & \infty & \infty & \infty & \infty \\
\infty & 2 & 0 & 2 & 3 & \infty & \infty & \infty & \infty \\
\infty & \infty & \infty & 0 & \infty & 10 & \infty & \infty & \infty \\
\infty & \infty & \infty & 6 & 0 & 4 & 3 & 6 & 2 \\
\infty & \infty & \infty & \infty & \infty & 0 & 2 & \infty & \infty \\
\infty & \infty & \infty & \infty & \infty & \infty & 0 & 4 & \infty \\
\infty & \infty & \infty & \infty & \infty & \infty & \infty & 0 & \infty \\
\infty & \infty & \infty & \infty & \infty & \infty & \infty & 3 & 0 \\
\end{pmatrix} \quad (2)$$

Step 2: taking $v_3$ as the intermediary point and updating the matrix,

$$A = \begin{pmatrix}
0 & 5 & 3 & 1 & 7 & \infty & \infty & \infty & \infty \\
\infty & 0 & \infty & \infty & 1 & \infty & \infty & \infty & \infty \\
\infty & 2 & 0 & 2 & 3 & \infty & \infty & \infty & \infty \\
\infty & \infty & \infty & 0 & \infty & 10 & \infty & \infty & \infty \\
\infty & \infty & \infty & 6 & 0 & 4 & 3 & 6 & 2 \\
\infty & \infty & \infty & \infty & \infty & 0 & 2 & \infty & \infty \\
\infty & \infty & \infty & \infty & \infty & \infty & 0 & 4 & \infty \\
\infty & \infty & \infty & \infty & \infty & \infty & \infty & 0 & \infty \\
\infty & \infty & \infty & \infty & \infty & \infty & \infty & 3 & 0 \\
\end{pmatrix} \quad (3)$$

Step 3: taking $v_4$ as the intermediary point and updating the matrix,

$$A = \begin{pmatrix}
0 & 5 & 3 & 1 & 7 & 11 & \infty & \infty & \infty \\
\infty & 0 & \infty & \infty & 1 & \infty & \infty & \infty & \infty \\
\infty & 2 & 0 & 2 & 3 & 12 & \infty & \infty & \infty \\
\infty & \infty & \infty & 0 & \infty & 10 & \infty & \infty & \infty \\
\infty & \infty & \infty & 6 & 0 & 4 & 3 & 6 & 2 \\
\infty & \infty & \infty & \infty & \infty & 0 & 2 & \infty & \infty \\
\infty & \infty & \infty & \infty & \infty & \infty & 0 & 4 & \infty \\
\infty & \infty & \infty & \infty & \infty & \infty & \infty & 0 & \infty \\
\infty & \infty & \infty & \infty & \infty & \infty & \infty & 3 & 0 \\
\end{pmatrix} \quad (4)$$

Step 4: taking $v_5$ as the intermediary point and updating the matrix,

$$A = \begin{pmatrix}
0 & 5 & 3 & 1 & 7 & 11 & 10 & 13 & 9 \\
\infty & 0 & \infty & 7 & 1 & 5 & 4 & 7 & 3 \\
\infty & 2 & 0 & 2 & 3 & 7 & 6 & 9 & 5 \\
\infty & \infty & \infty & 0 & \infty & 10 & \infty & \infty & \infty \\
\infty & \infty & \infty & 6 & 0 & 4 & 3 & 6 & 2 \\
\infty & \infty & \infty & \infty & \infty & 0 & 2 & \infty & \infty \\
\infty & \infty & \infty & \infty & \infty & \infty & 0 & 4 & \infty \\
\infty & \infty & \infty & \infty & \infty & \infty & \infty & 0 & \infty \\
\infty & \infty & \infty & \infty & \infty & \infty & \infty & 3 & 0 \\
\end{pmatrix} \quad (5)$$

Figure 3: Shortest path problem.

Figure 4: The shortest path.
Step 5: taking \( v_6 \) as the intermediary point and updating the matrix,

\[
A = \begin{pmatrix}
0 & 5 & 3 & 1 & 7 & 11 & 10 & 13 & 9 \\
\infty & 0 & \infty & 7 & 1 & 5 & 4 & 7 & 3 \\
\infty & 2 & 0 & 2 & 3 & 7 & 6 & 9 & 5 \\
\infty & \infty & \infty & \infty & 0 & \infty & 10 & 12 & \infty \\
\infty & \infty & \infty & 6 & 0 & 4 & 3 & 6 & 2 \\
\infty & \infty & \infty & \infty & \infty & 0 & 2 & \infty & \infty \\
\infty & \infty & \infty & \infty & \infty & 0 & 4 & \infty & \infty \\
\infty & \infty & \infty & \infty & \infty & 0 & \infty & 0 & \infty \\
\infty & \infty & \infty & \infty & \infty & 3 & 0 & \infty & \infty \\
\end{pmatrix}
\]  
(6)

Step 6: taking \( v_7 \) as the intermediary point and updating the matrix,

\[
A = \begin{pmatrix}
0 & 5 & 3 & 1 & 7 & 11 & 10 & 13 & 9 \\
\infty & 0 & \infty & 7 & 1 & 5 & 4 & 7 & 3 \\
\infty & 2 & 0 & 2 & 3 & 7 & 6 & 9 & 5 \\
\infty & \infty & \infty & \infty & 0 & \infty & 10 & 12 & \infty \\
\infty & \infty & \infty & 6 & 0 & 4 & 3 & 6 & 2 \\
\infty & \infty & \infty & \infty & \infty & 0 & 2 & \infty & \infty \\
\infty & \infty & \infty & \infty & \infty & 0 & 4 & \infty & \infty \\
\infty & \infty & \infty & \infty & \infty & 0 & \infty & 0 & \infty \\
\infty & \infty & \infty & \infty & \infty & 3 & 0 & \infty & \infty \\
\end{pmatrix}
\]  
(7)

Step 7: taking \( v_9 \) as the intermediary point and updating the matrix,

\[
A = \begin{pmatrix}
0 & 5 & 3 & 1 & 7 & 11 & 10 & 12 & 9 \\
\infty & 0 & \infty & 7 & 1 & 5 & 4 & 6 & 3 \\
\infty & 2 & 0 & 2 & 3 & 7 & 6 & 8 & 5 \\
\infty & \infty & \infty & \infty & 0 & \infty & 10 & 12 & 16 \infty \\
\infty & \infty & \infty & 6 & 0 & 4 & 3 & 5 & 2 \\
\infty & \infty & \infty & \infty & \infty & 0 & 2 & \infty & \infty \\
\infty & \infty & \infty & \infty & \infty & 0 & 4 & \infty & \infty \\
\infty & \infty & \infty & \infty & \infty & 0 & \infty & 0 & \infty \\
\infty & \infty & \infty & \infty & \infty & 3 & 0 & \infty & \infty \\
\end{pmatrix}
\]  
(8)

Thus, the shortest path from \( v_1 \) to \( v_8 \) is

\[v_1 \rightarrow v_3 \rightarrow v_2 \rightarrow v_5 \rightarrow v_9 \rightarrow v_8.\]

3.3. Solving the Shortest Path Problem Using the Probe Machine. The probe machine solves the shortest path problem as follows.

3.3.1. Data Library Construction

\[
E^2(v_i) = \bigcup_{j=1}^{9} E^2(v_j),
\]
(9)

\( E^2(v_i) \) is the data pool of \( v_i \), and \( E^2(v_j) \) is a set of two long paths centered around \( v_j \). The two long paths are denoted as \( v_{ij} \) and \( i, l, j \) are different from each other. At the same time, because this problem is a directed graph, each path has one data fiber. The data library is as follows:

\[
E^2(v_i) = \bigcup_{j=1}^{9} E^2(v_j),
\]
(10)

The data fibers are as follows:

\[
\begin{align*}
\mathcal{E}(v_{215}) &= v_{515}, \\
\mathcal{E}(v_{235}) &= v_{4235}, \\
\mathcal{E}(v_{2312}) &= v_{x_{312}}, \\
\mathcal{E}(v_{314}) &= v_{x_{314}}, \\
\mathcal{E}(v_{416}) &= v_{x_{416}}, \\
\mathcal{E}(v_{436}) &= v_{x_{436}}, \\
\mathcal{E}(v_{456}) &= v_{x_{456}}, \\
\mathcal{E}(v_{524}) &= v_{x_{524}}, \\
\mathcal{E}(v_{526}) &= v_{x_{526}}, \\
\mathcal{E}(v_{527}) &= v_{x_{527}}, \\
\mathcal{E}(v_{528}) &= v_{x_{528}}, \\
\mathcal{E}(v_{529}) &= v_{x_{529}}, \\
\mathcal{E}(v_{647}) &= v_{x_{647}}, \\
\mathcal{E}(v_{657}) &= v_{x_{657}}, \\
\mathcal{E}(v_{768}) &= v_{x_{768}}, \\
\mathcal{E}(v_{758}) &= v_{x_{758}}, \\
\mathcal{E}(v_{758}) &= v_{x_{758}}, \\
\mathcal{E}(v_{758}) &= v_{x_{758}}, \\
\end{align*}
\]  
(11)

3.3.2. Probe Library Construction. In order to avoid arbitrary data aggregation, if there is a probe between data \( v_{ij} \) and \( v_{ij} \), one of the following two conditions need to be met:
(1) \(|i, j| \cap |t, a, b| = |l, j| \cap |a, b| = 1\),
(2) \(t \in [i, j], i \in |a, b|, |l, j| \cap |a, b| = 0\) \hspace{1cm} (12)

At the same time, without loss of generality, because this is a directed graph, the direction of the probe needs to be same as the direction of the edge. The probe library established is as follows:

\[
Y = \bigcup_{i=0, l=0}^{i=t \leq 2} Y_{it}, \quad (i < t),
\]
\[
Y_{23} = \{x^2_{215}x^2_{312}\},
\]
\[
Y_{24} = \{x^2_{215}x^2_{456}, x^2_{215}x^6_{456}\},
\]
\[
Y_{25} = \{x^2_{215}x^2_{224}, x^2_{215}x^2_{526}, x^2_{215}x^2_{527}, x^2_{215}x^2_{528}, x^2_{215}x^2_{529}, x^2_{235}x^2_{524}, x^2_{235}x^2_{526}, x^2_{235}x^2_{527}, x^2_{235}x^2_{528}, x^2_{235}x^2_{529}\},
\]
\[
Y_{26} = \{x^2_{215}x^2_{657}, x^2_{235}x^2_{657}\},
\]
\[
Y_{27} = \{x^2_{215}x^2_{758}, x^2_{235}x^2_{758}\},
\]
\[
Y_{29} = \{x^2_{215}x^2_{958}, x^2_{235}x^2_{958}\},
\]
\[
Y_{34} = \{x^6_{314}x^6_{436}\},
\]
\[
Y_{35} = \{x^2_{312}x^2_{224}, x^2_{312}x^2_{526}, x^2_{312}x^2_{527}, x^2_{312}x^2_{528}, x^2_{312}x^2_{529}\},
\]
\[
Y_{36} = \{x^6_{314}x^6_{657}\},
\]
\[
Y_{45} = \{x^6_{456}x^6_{224}\},
\]
\[
Y_{46} = \{x^6_{416}x^6_{647}, x^6_{436}x^6_{647}, x^6_{456}x^6_{647}\},
\]
\[
Y_{47} = \{x^6_{416}x^6_{768}x^6_{436}x^6_{768}x^6_{456}x^6_{768}\},
\]
\[
Y_{56} = \{x^2_{526}x^2_{647}, x^2_{526}x^2_{657}\},
\]
\[
Y_{57} = \{x^6_{526}x^6_{768}, x^6_{527}x^6_{758}\},
\]
\[
Y_{59} = \{x^2_{529}x^2_{558}\},
\]
\[
Y_{67} = \{x^6_{647}x^6_{768}, x^6_{657}x^6_{768}\}.
\]

3.3.3. Probe Operation. The data controller and probe controller are used to extract the corresponding amount of data and probes from the data library and probe library and add them to the computing platform. After the probe operation, all possible solutions to the problem are obtained. Through the detector, the optimal solution is output as follows:

\[
v_1 \rightarrow v_3 \rightarrow v_2 \rightarrow v_5 \rightarrow v_9 \rightarrow v_8.
\] \hspace{1cm} (14)

4. Results and Discussion

Although all three methods can solve the shortest path problem and obtain the same solution, there are some obvious differences in the solving process and computing efficiency. Dijkstra’s algorithm is based on the breadth-first search algorithm, which calculates the shortest path from the starting point to all other points. The calculation step of Dijkstra’s algorithm is to expand from the starting point to the layer by layer. Floyd’s algorithm is based on dynamic programming algorithm to find the shortest path between any two points in the graph. The first step of Floyd’s algorithm is to find the adjacency matrix of the graph, and the second step is to update the adjacency matrix by taking each point as the intermediate point. And, the solution process of the probe machine is mainly divided into two steps: the first step is to build the data library and the probe library, and the second step is to get all possible solutions through the probe operation. It can be seen from the solution process that each step of Dijkstra’s algorithm and the Floyd algorithm is part of solving the shortest path problem, but the real solution process of the probe machine is only probe operation. The purpose of the remaining steps of the probe machine is to prepare for the probe operation or to screen the solution calculated by the probe operation. This phenomenon should be attributed to the way of storing data. When the Dijkstra algorithm and the Floyd algorithm are used, the linear data placement mode and the sequential data processing mode are taken, so the solution process needs to be performed step by step; however, in the probe machine, a nonlinear data storage method is adopted, so the probe machine can simultaneously process multiple pairs of data. In other words, the probe machine is a completely parallel
computational model, which is the biggest difference between the probe machine and other computing models.

Moreover, Dijkstra’s and Floyd’s algorithms are only used to deal with problems in graph theory. The probe machine is a completely parallel computing model, which can also get the true solution after one probe operation when solving NP-complete problems. Although we only use the probe machine to deal with the shortest path problem in this paper, we can see the powerful computing power of the probe machine through the solving process.

5. Conclusion

In urban traffic planning, the shortest path problem is a very basic problem. In this study, a probe machine was used to solve this problem. It can solve complex traffic problems after a single probe operation, which improves the efficiency and shortens the time required for solving traffic planning problems. When dealing with actual traffic problems, the speed at which a probe machine solves them is much higher than the speed of other models. A probe machine has high computing power; however, it is still a new concept. It has several shortcomings and needs to be improved further to handle specific problems. For example, when dealing with complex traffic problems, complex data and probe libraries need to be constructed. The process of constructing these libraries cannot be completely replaced by a probe machine or computer. The more complex the problem to be solved is, the more difficult it becomes to construct data library and probe library. In the future, we will continue to study the probe machine and attempt to improve it.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (grant nos. 61602188 and 11975143), Shandong Provincial Natural Science Foundation (grant no. ZR2019QD018), and Scientific Research Foundation of Shandong University of Science and Technology for Recruited Talents (grant nos. 2017RCJJ068 and 2017RCJJ069).

References


