*applied sciences*

MDPI

*Article*

# A Fault Section Location Method for Distribution Networks Based on Divide-and-Conquer

**Qiao Zhao [1,*], Zengping Wang [1], Guomin Li [2], Xuanjun Liu [1] and Yuxuan Wang [1]**

[1] State Key Laboratory of Alternate Electrical Power System with Renewable Energy Sources, North China Electric Power University, Beijing 102206, China
[2] State Grid Beijing Electric Power Company, Beijing 100031, China
* Correspondence: 1172101060@ncepu.edu.cn; Tel.: +86-138-1050-8367

**Abstract:** In this paper, a fault location method based on divide-and-conquer (DAC) is proposed to solve the inadequacy problem that arises when using the traditional fault section location method based on the optimization model of logic operation. The problem is that it is difficult to balance speed and accuracy after the scale of the distribution network is expanded. First, the causal link between fault information and the faulty device was described using the road vector, the equivalent transformation of the logical operations in the traditional model was implemented with the properties of the road vector, and the numerical computational model of the fault location was constructed. Based on this, the optimization-seeking variable "approximation gain" was introduced to prove that the proposed model conforms to the recursive structure of DAC, and the method of applying DAC to locate faults is proposed. The method applies the "Divide-Conquer-Combine" recursive mode to locate faults, and each level of recursion contains only linear-time "approximation gain" operations and constant-time decomposition and combination operations. The efficiency analysis and simulation results show that the proposed method has linear-time complexity and can accurately locate faults in milliseconds, providing a reference for solving the fault location problem in large distribution networks.

**Keywords:** distribution network; fault location; divide-and-conquer; road vector

**Citation:** Zhao, Q.; Wang, Z.; Li, G.; Liu, X.; Wang, Y. A Fault Section Location Method for Distribution Networks Based on Divide-and-Conquer. *Appl. Sci.* **2023**, *13*, 5974. https://doi.org/10.3390/app13105974

Academic Editor: Ki-Yong Oh

Received: 26 February 2023
Revised: 7 May 2023
Accepted: 11 May 2023
Published: 12 May 2023

## 1. Introduction

Distribution network fault location is important in improving power system supply reliability as a prerequisite for fault isolation and power supply restoration [1,2]. Modern power distribution systems mostly adopt multi-sectioned and multi-linked network structures so that faults can be limited to a smaller area to meet the increasingly demanding power supply service needs of electricity consumers [3]. However, the increase in the number of sections reduces the supply radius, resulting in insignificant differences in the magnitude of short-circuit currents in adjacent lines, which makes the fault location method based on the traditional three-stage current protection principle no longer applicable [4]. In the meantime, the problems of network scaling, distortion of fault information, and multiple faults in extreme cases also put higher requirements and challenges on the accuracy and speed of fault location [5].

Along with the construction of distribution automation and the development of communication technology, feeder terminal units (FTUs) and other intelligent monitoring devices have become popular [6]. The FTU installed at the intelligent switch can be equipped with fault current detection elements and adaptively adjust the action threshold of the fault current detection elements according to the operation mode of the system [7]. After a fault in the distribution network, the FTU compares the current flowing at the switch with the preset threshold to determine whether a short-circuit current is flowing at the switch, and it reports the fault detection results to the fault location decision host, which applies the fault location algorithm to identify the fault location [8,9]. The basic working principle of the fault location method for distribution networks based on FTU fault information is shown in Figure 1. This method has the advantages of small data communication and convenient implementation, and it has gradually become the main means of fault location in medium-voltage distribution networks [10,11].

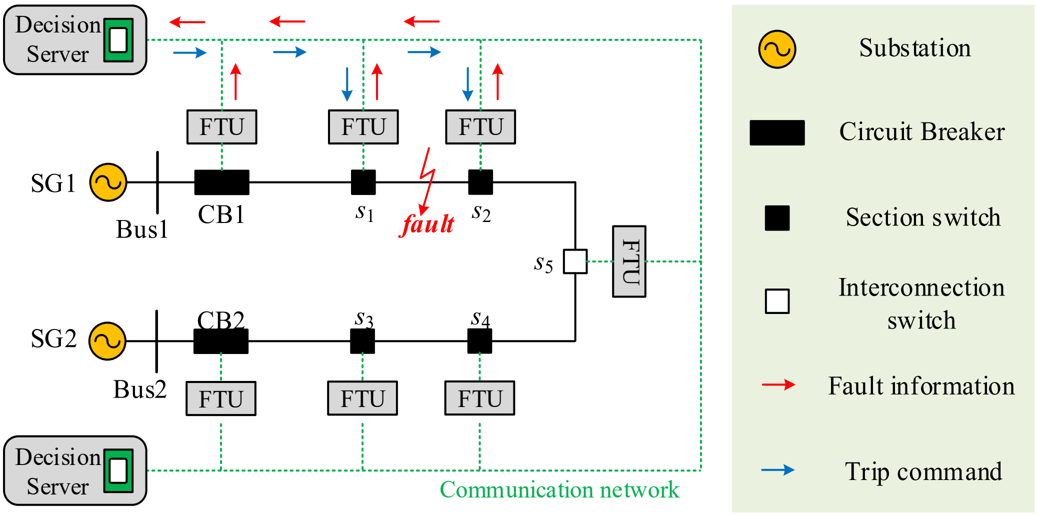


**Figure 1.** The basic working principle of the fault location method based on FTUs.

Distribution network fault location methods based on FTU fault information mainly include four types: the first type is based on matrix algorithm fault location methods, the second type is based on intelligent optimization algorithm fault location methods, the third type is based on linear integer programming model fault location methods, and the fourth type is fault location methods based on hierarchical optimization models.

(1) Matrix algorithm-based fault location methods for distribution networks. The basic principle of this method is to construct a fault discrimination matrix based on the topology of the distribution network and the fault information reported by FTUs and locate the faulty equipment via comprehensive analysis of the fault discrimination matrix [12,13]. The matrix algorithm has the advantages of simple principles and fast calculation speed. However, in actual engineering, FTUs are usually installed outdoors and work in harsh environments, resulting in possible misreporting or omission of the reported information. When the fault information is distorted, the matrix constructed by the algorithm cannot correctly reflect the real fault situation, thereby causing the algorithm function to fail [14]. Reference [15] proposes using the adjacency of FTUs measurement points to correct the fault information. Reference [16] proposes introducing telemetry data to check the localization results. However, these methods are only valid for partial cases of fault information distortion [17], and the fault tolerance process of the algorithm additionally increases fault location time. With the expansion of the network scale, the number of FTU measurement points increases, and the probability of fault information distortion increases; thus, the method is difficult to apply to the fault location problem of large distribution networks.

(2) Fault location methods based on the intelligent optimization algorithm. Reference [18] proposes applying logical operations to express the causal relationship between FTU fault information and faulty equipment and proposes a mathematical model of fault location based on the state approximation principle, which transforms fault location into a combinatorial optimization problem. Since then, scholars have successively used the genetic algorithm (GA) [19], binary particle swarm optimization algorithm (BPSO) [20], ant colony algorithm (AC) [21], harmony search algorithm (HS)

[22], and other intelligent algorithms to implement the solution of this model. The intelligent optimization algorithm has a strong information fault tolerance capability. However, the method requires iterative search operations in a high-dimensional solution space, which affects the speed of fault location [23]. Moreover, its search process has a certain degree of randomness, which leads to a certain probability of local convergence. Therefore, this method has the inherent defect of insufficient convergence stability [24].

(3) Fault location methods based on the linear integer programming model. The basic principle of this method is to transform the mathematical model of fault location into a linear integer programming model (LIP) using algebraic operation substitution [25], ascending dimension [26], etc. Then, classical LIP methods (such as the branch and bound method, cut plane method, and implicit enumeration method) are applied to solve the model to locate the fault. The "optimality checking" step in the LIP solution method enables it to gradually approximate and eventually converge to the global optimal solution through iterations and thus has good convergence stability. However, the solution of the LIP model involves the computation of its continuous relaxed linear programming model (LP) several times, resulting in high time complexity for the algorithm [27]. Tests have shown that the fault location time of small and medium-sized distribution networks is in the range of seconds [26], which does not meet the requirements of rapidity.

(4) Fault location methods based on the hierarchical optimization model. Reference [28] proposes that the branch of the network is externally equivalent to a two-port based on the equivalence principle and constructs a hierarchical optimization model of fault location. The basic idea is to solve the mathematical model of fault location in a reduced dimension: first, locate the branch where the fault is located, then determine the faulty device from the faulty branch. Since then, scholars have successively used the multiverses optimization algorithm (MVO) [29], quantum computing and immune optimization algorithm(QIOA) [30], bald eagle search algorithm (BES) [31], and other intelligent algorithms to implement the solution of hierarchical optimization models. This method converts effectively reduces the size of solution space and improves the speed and accuracy of fault location. However, the method requires high soundness of FTU fault information at the branch port, which may produce misjudgment when the port information is distorted, and it cannot completely eradicate the defect of insufficient convergence stability of the intelligent algorithm.

The characteristics of the aforementioned various distribution network fault location methods are shown in Table 1.

**Table 1.** Characteristics of fault location methods.

| Method | Characteristics |
|---|---|
| Matrix algorithm | It has fast location speed but poor information fault tolerance. |
| Intelligent optimization algorithm | It has high fault information tolerance, but its computational efficiency and convergence stability are poor. |
| Linear integer programming algorithm | It has high fault information tolerance and strong convergence stability but poor computational efficiency. |
| Hierarchical fault location method | Its convergence reliability and computational efficiency are better than the intelligent optimization algorithm, but it requires higher information reliability in port nodes. |

In summary, a large number of results have been achieved in distribution network fault location methods, but the following problems still exist: on one hand, the traditional fault location model contains logical operations; thus, it is difficult to obtain a high efficient solution to the model through strict mathematical derivation; on the other hand, the current fault location methods are still difficult to use simultaneously. Furthermore, current fault location methods are not yet able to meet the requirements of accuracy and speed simultaneously.

In view of these problems, this paper proposes a fault section location method for a distribution network based on divide-and-conquer (DAC). The following three aspects are the main features of this work:

- The application of road vectors to establish causal links between fault information and faulty equipment, the equivalent transformation of logical operations in the mathematical model of traditional fault location, and the construction of a numerical computational model of fault location.
- Defines "compatible sets" and "approximation gains" and validates its properties, and then proves that the mathematical model constructed in this paper conforms to the recursive structure of DAC.
- Proposes a fault location method based on DAC, which uses the recursive model of "Divide-Conquer-Combine" in each level of recursion. The linear time "approximation gain" computations and constant time divide and combine operations are included in each layer of recursion to improve the computational efficiency while ensuring the accuracy of fault location.

The rest of this paper is organized as follows. In Section 2, the numerical calculation model of fault location is proposed. Section 3 demonstrates the recursive structure of the proposed model. In Section 4, the DAC-based fault location method is proposed, and its computational efficiency is analyzed. In Section 5, the accuracy and computational efficiency of the method in this paper are verified by simulation tests. Section 6 summarizes the paper.

## 2. Numerical Computational Model for Distribution Grid Fault Location

### *2.1. Basic Principle of Distribution Network Fault Location*

When a fault occurs in the distribution network, the FTU installed at the smart switch (including circuit breakers and section switches) detects the current crossing and uploads the fault detection results to the fault location decision servers, which determines the fault device based on the fault location algorithm. In this paper, we refer to the smart switch equipped with FTU as a node and the distribution line enclosed by the nodes as a section. The task of the fault location is to accurately locate the section where the fault occurs.

With the help of the fault hypothesis theory, the fault location problem can be transformed into a combinatorial optimization problem. The basic principle is as follows: assuming that all possible fault scenarios to construct a solution space, find the optimal solution in the solution space, and this optimal solution can give the most reasonable explanation to the fault information reported by the FTU. This section describes the coding rules of fault information, proposes the corresponding relationship between fault scenarios and fault information, and describes how the combinatorial optimization model for fault location was constructed.

#### 2.1.1. Fault Information Code

The network structure of a typical distribution network is shown in Figure 2, where SG indicates the main power provided by the substation, and the black squares indicate the nodes. This article specifies that the direction from the main power to the section is the positive direction of the section. For the convenience of the description, the nodes are numbered consecutively according to their distance from the main power, and the section number is taken as the smaller of its associated node number (i.e., the section number is taken as its originating point number). The distribution network shown in Figure 2 is numbered according to these rules, where $s_1$~$s_6$ denotes the node number, and $x_1$~$x_6$ denotes the section number. In this paper, sections associated with more than two nodes are called T-type sections, e.g., section $x_2$. In contrast, $x_1$, $x_3$~$x_6$ are called ordinary sections. In addition, in this paper, we applied sets $\boldsymbol{V}$ and $\boldsymbol{E}$ to represent the sets of nodes and sections of the distribution network, respectively, and the distribution network is represented as $\boldsymbol{G} = (\boldsymbol{V}, \boldsymbol{E})$.

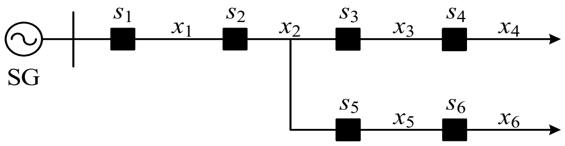


**Figure 2.** Typical active distribution network.

The distribution network has a radial network structure, and when a fault occurs, the fault current flows from the main power to the faulty equipment. For a distribution network containing $m$ nodes, the fault information vector $\boldsymbol{I} = [I_1, I_2, \ldots, I_m]$ indicates the fault information reported by FTUs, where the variables $I_i$ are defined as shown in Equation (1).

$$I_i = \begin{cases} 1 & \text{Node } s_i \text{ has flowed a short - circuit current} \\ 0 & \text{Others} \end{cases} \tag{1}$$

For a distribution network with a number of sections $n$, apply the variable $y_k$ ($k = 1, 2, \ldots, n$) to indicate whether section $x_k$ fault, $y_k$, is defined, as shown in Equation (2).

$$y_k = \begin{cases} 1 & \text{Section } x_k \text{ failure} \\ 0 & \text{No failure in section } x_k \end{cases} \tag{2}$$

On this basis, our assumed fault scenarios can be described by the set of fault sections $\boldsymbol{X}$, which is the set consisting of the sections where faults occur, defined as shown in Equation (3).

$$\boldsymbol{X} = \{x_k \in \boldsymbol{E} \mid y_k = 1\} \tag{3}$$

### 2.1.2. Switching Function

To measure the rationality of the fault information interpretation of the assumed fault scenario, it is necessary to construct the corresponding relationship between the set of fault sections $\boldsymbol{X}$ and the fault information. The fault information of node $s_i$ corresponding to $\boldsymbol{X}$ can be represented by the switch function $J_i(\boldsymbol{X})$. $J_i(\boldsymbol{X})$ takes $\boldsymbol{X}$ as the independent variable, which means that when the fault scenario of the distribution network is $\boldsymbol{X}$, the value of the fault information of node $s_i$ is taken when the information reported by node $s_i$ is not misreported or omitted.

The switching function can be constructed based on the causal relationship between nodes and sections. We can take node $s_2$ in the distribution network shown in Figure 2 as an example. If a fault signal is reported at $s_2$, it is known that the fault signal at $s_2$ may be generated by sections $x_2$~$x_6$ as caused by the occurrence of a short-circuit fault. Therefore, the operation status of sections $x_2$~$x_6$ is related to the fault information of node $s_2$, and in this paper, sections $x_2$~$x_6$ are the causal devices of $s_2$. Similarly, the causal devices of other nodes in Figure 2 can be determined, as shown in Table 2.

**Table 2.** Causal devices corresponding to nodes in the distribution network.

| Node | Causal Device (Section) | Status Information of the Causal Device | Total Number |
|---|---|---|---|
| $s_1$ | $x_1$~$x_6$ | $y_1$~$y_6$ | 6 |
| $s_2$ | $x_2$~$x_6$ | $y_2$~$y_6$ | 5 |
| $s_3$ | $x_3$, $x_4$ | $y_3$, $y_4$ | 2 |
| $s_4$ | $x_4$ | $y_4$ | 1 |
| $s_5$ | $x_5$, $x_6$ | $y_5$, $y_6$ | 2 |
| $s_6$ | $x_6$ | $y_6$ | 1 |

Based on this analysis, the logical "OR" operation was introduced to express the causal connection between nodes and sections, and the switching function $J_i(\boldsymbol{X})$ of node $s_i$ was constructed according to Equation (4):

$$J_i(\boldsymbol{X})=\prod y_k \quad (4)$$

where $y_k$ denotes the causal device of node $s_i$ and " $\prod$ " denotes the superposition of logical "OR" operations.

To enhance understanding, the distribution network shown in Figure 2 is taken as an example to illustrate the calculation process of the switching function. Assume the fault scenario $\boldsymbol{X} = \{x_3, x_6\}$. The calculation process and results of the switching function of each node are shown in Equation (5).

$$\begin{cases} J_1(\boldsymbol{X}) = y_1 \vee y_2 \vee y_3 \vee y_4 \vee y_5 \vee y_6 = 0 \vee 0 \vee 1 \vee 0 \vee 0 \vee 1 = 1 \\ J_2(\boldsymbol{X}) = y_2 \vee y_3 \vee y_4 \vee y_5 \vee y_6 = 0 \vee 1 \vee 0 \vee 0 \vee 1 = 1 \\ J_3(\boldsymbol{X}) = y_3 \vee y_4 = 1 \vee 0 = 1 \\ J_4(\boldsymbol{X}) = y_4 = 0 \\ J_5(\boldsymbol{X}) = y_5 \vee y_6 = 0 \vee 1 = 1 \\ J_6(\boldsymbol{X}) = y_6 = 1 \end{cases} \quad (5)$$

where " $\vee$ " means the logical "OR" operation.

### 2.1.3. Objective Function

According to the state approximation theory, the essence of fault location is to find the most likely fault scenario $\boldsymbol{X}$, where the section state can provide the most reasonable explanation for the fault information reported by FTU. In other words, the smaller the difference between the switching function $J_i(\boldsymbol{X})$ of the fault sections set $\boldsymbol{X}$ and the fault information $I_i$ actually reported by FTU, $i$, the more reasonable the interpretation is, and the closer $\boldsymbol{X}$ is to the real fault situation. Therefore, the objective function of fault location is shown in Equation (6) [26]:

$$\min f(\boldsymbol{X}) = \sum_{i=1}^{m}[I_i \oplus J_i(\boldsymbol{X})] + \omega \sum_{x_k \in X} y_k \quad (6)$$

where $m$ indicates the total number of nodes in the distribution network, and " $\oplus$ " denotes the logical "XOR" operation. When $I_i = J_i(\boldsymbol{X})$, $I_i \oplus J_i(X) = 0$ ; otherwise, $I_i \oplus J_i(X) = 1$. $\omega$ is the error-proof factor, which takes values between 0 and 1.

Take the typical distribution network shown in Figure 1 as an example to illustrate the basic principle of fault location for distribution network. Suppose section $x_4$ fails, the fault information reported by FTU is $\boldsymbol{I} = [I_1$~$I_6] = [1\ 1\ 1\ 1\ 1\ 0\ 0]$, and the fault location algorithm is activated. According to Equation (2), each section has two operating states: "normal" and "fault". As the network contains six sections, there are $2^6$ possible fault scenarios. Each fault scenario can be described by a set of fault sections $\boldsymbol{X}$, and the $2^6$ $\boldsymbol{X}$ form the solution space, from which the task of fault location is to find the optimal solution. The switch functions and objective functions for some hypothetical fault scenarios are shown in Table 3.

**Table 3.** Vector ***X*** and its switching function and objective function.

| Number | Fault Scenario | $X$ | $[J_1(X)\sim J_6(X)]$ | $f(X)$ |
|---|---|---|---|---|
| 1 | $x_1$ failure | $\{x_1\}$ | [1 0 0 0 0 0] | $3+\omega$ |
| 2 | $x_2$ failure | $\{x_2\}$ | [1 1 0 0 0 0] | $2+\omega$ |
| 3 | $x_3$ failure | $\{x_3\}$ | [1 1 1 0 0 0] | $1+\omega$ |
| 4 | $x_4$ failure | $\{x_4\}$ | [1 1 1 1 0 0] | $\omega$ |
| 5 | $x_5$ failure | $\{x_5\}$ | [1 1 0 0 1 0] | $3+\omega$ |
| 6 | $x_6$ failure | $\{x_6\}$ | [1 1 0 0 1 1] | $4+\omega$ |
| 7 | $x_1$ and $x_2$ failure | $\{x_1, x_2\}$ | [1 1 0 0 0 0] | $2+2\omega$ |
| 8 | $x_1$ and $x_3$ failure | $\{x_1, x_3\}$ | [1 1 1 0 0 0] | $1+2\omega$ |
| … | … | … | … | … |
| $2^6$ | $x_1\sim x_6$ failure | $\{x_1, x_2, x_3, x_4, x_5, x_6\}$ | [1 1 1 1 1 1] | $2+6\omega$ |

By comparing the objective functions of $2^6$ assumed failure scenarios, it can be seen that the objective function of number 4 is the smallest; thus, $\boldsymbol{X} = \{x_4\}$ is taken as the optimal solution. Therefore, section $x_4$ is determined to be the fault section. The fault location operation is finished, and the result of the fault location is output.

## *2.2. Construction of the Numerical Computation Model*

In this example, we can see that the size of the solution space of the fault location problem is exponentially related to the number of sections in the distribution network. Assuming and comparing all possible fault scenarios to locate the fault section will seriously reduce computational efficiency. Meanwhile, the fault location model contains logical relation operations (such as "OR" operations in Equation (4) and "XOR" operations in Equation (6)), thereby making it impossible to apply the classical optimization algorithm with superior performance in achieving the solution. Although locating faults by means of intelligent algorithms with stochastic search characteristics can improve computational efficiency to a certain extent, they have the inherent defect of insufficient convergence stability, which makes it difficult to guarantee the accuracy of fault location.

In view of these problems, this section proposes to apply the road vector to describe the causal relationship between nodes and sections, to equivalently transform the logical operations in the model by the properties of the road vector, to construct the numerical computation model of fault location, and to lay the theoretical foundation for proposing a faster and more accurate fault location method.

### 2.2.1. Road Vector

The road that defines the section is the set of nodes and sections on the path from the section to the main power source. For the distribution network containing $m$ nodes, the road vector of section $x_k$ is $\boldsymbol{P}_k = [\boldsymbol{P}_k(1), \boldsymbol{P}_k(2), ..., \boldsymbol{P}_k(i), ..., \boldsymbol{P}_k(m)]$, where the element $\boldsymbol{P}_k(i)$ is defined as follows.

$$\boldsymbol{P}_k(i) = \begin{cases} 1 & \text{Node } s_i \text{ is on the road of } x_k \\ 0 & \text{Others} \end{cases} \tag{7}$$

Taking the distribution network shown in Figure 2 as an example, nodes $s_1\sim s_4$ are on the road of section $x_4$; thus, the road vector $\boldsymbol{P}_4$ of $x_4$ is shown in Equation (8).

$$\boldsymbol{P}_4 = [1\ \ 1\ \ 1\ \ 1\ \ 0\ \ 0] \tag{8}$$

Similarly, the road vectors of the other sections in Figure 2 can be determined: $\boldsymbol{P}_1 = [1\,0\,0\,0\,0\,0]$, $\boldsymbol{P}_2 = [1\,1\,0\,0\,0\,0]$, $\boldsymbol{P}_3 = [1\,1\,1\,0\,0\,0]$, $\boldsymbol{P}_5 = [1\,1\,0\,0\,1\,0]$, and $\boldsymbol{P}_6 = [1\,1\,0\,0\,1\,1]$.

When section $x_4$ has a fault, nodes $s_1$~$s_4$ are on the main power to section $x_4$ path; thus, nodes $s_1$~$s_4$ will flow the fault current and report the fault information. It can be seen that the road vector can describe the causal relationship between nodes and sections. For a hypothetical fault scenario $\boldsymbol{X}$, the switching function $J_i(\boldsymbol{X})$ of node $i$ can be rewritten in Equation (4) as

$$J_i(\boldsymbol{X}) = \prod_{x_k \in \boldsymbol{X}} \boldsymbol{P}_k(i) \tag{9}$$

Applying the vector $\boldsymbol{J}(\boldsymbol{X})$ represents the switching function corresponding to the assumed fault scenario $\boldsymbol{X}$, the vector $\boldsymbol{J}(\boldsymbol{X}) = [J_1(\boldsymbol{X}), J_2(\boldsymbol{X}), \ldots, J_m(\boldsymbol{X})]$. The combination of Equation (9) in matrix form and the vector $\boldsymbol{J}(\boldsymbol{X})$ is shown in Equation (10):

$$\boldsymbol{J}(\boldsymbol{X}) = \prod_{x_k \in \boldsymbol{X}} \boldsymbol{P}_k \tag{10}$$

where "$\prod$" means the superposition operation of "OR" is completed for multiple same-dimension vectors consecutively.

The distribution network shown in Figure 2 is used as an example to verify the effectiveness of the road vector in describing the causal relationship. Assuming the fault scenario $\boldsymbol{X} = \{x_3, x_6\}$, according to Equation (10), the vector $\boldsymbol{J}(\boldsymbol{X})$ is shown in Equation (11):

$$\boldsymbol{J}(\boldsymbol{X}) = \prod_{x_k \in \boldsymbol{X}} \boldsymbol{P}_k = \boldsymbol{P}_3 \vee \boldsymbol{P}_6 = [1\,1\,1\,0\,0\,0] \vee [1\,1\,0\,0\,1\,1] = [1\,1\,1\,0\,1\,1] \tag{11}$$

where "$\vee$" means that the corresponding elements of two same-dimension vectors undergo logical "OR" operations.

Equation (11) has the same result as Equation (5), which shows that the road vector can effectively describe the causal relationship between nodes and sections.

### 2.2.2. Properties of the Road Vector

The properties of the road vector are the basis for constructing and solving numerical computational models of fault location. These properties are stated first, and their mathematical proofs are given in Appendix A.

**Theorem 1.** *If the distribution network nodes and sections are numbered using the numbering rules in this paper, the following conclusions are equivalent.*

(1) $\boldsymbol{P}_j(i) = 1$;

(2) *Section $x_i$ is on the road of section $x_j$;*

(3) $\boldsymbol{P}_i = \boldsymbol{P}_i * \boldsymbol{P}_j$;

*where "$*$" means that the corresponding elements of two same-dimension vectors are multiplied ("$*$" satisfies the commutative law and the associative law).*

**Theorem 2.** *Let the set of sections in the distribution network be $\boldsymbol{E}$. For sections $x_i, x_j, x_r \in \boldsymbol{E}$, at least one of the following expressions holds:*

$$\boldsymbol{P}_i * \boldsymbol{P}_j * (\boldsymbol{e} - \boldsymbol{P}_r) = \boldsymbol{0} \tag{12}$$

$$\boldsymbol{P}_i * \boldsymbol{P}_r * (\boldsymbol{e} - \boldsymbol{P}_j) = \boldsymbol{0} \tag{13}$$

*where* **e** *denotes a vector of the same dimension as the road vector, and the element values are all 1.*

**Theorem 3.** *Let the set of sections in the distribution network be* $\boldsymbol{E}$ *, let* $\boldsymbol{X} \subseteq \boldsymbol{E}$ *be a subset of sections, and suppose section* $x_i \in \boldsymbol{E} - \boldsymbol{X}$ *. If* $\boldsymbol{P}_i * \prod\limits_{x_k \in \boldsymbol{X}}^{*} (\boldsymbol{e} - \boldsymbol{P}_k) = \boldsymbol{0}$ *, then* $\exists x_k \in \boldsymbol{X}$ *such that* $\boldsymbol{P}_i = \boldsymbol{P}_i * \boldsymbol{P}_k$ *.*

*where "* $\overset{*}{\prod}$ *" means that the superposition operation of "* $*$ *" is performed for multiple same-dimension vectors consecutively.*

### 2.2.3. Numerical Computational Expressions for the Switching Function

This section equivalent transforms the logic "OR" operation in Formula (10) of the switching function based on the properties of the road vector and deduces the numerical calculation expression of the switching function.

We claim that the following numerical computational relationship exists between the hypothetical fault scenario ***X*** and the switching function vector ***J***(***X***):

$$\boldsymbol{e} - \boldsymbol{J}(\boldsymbol{X}) = \prod_{x_k \in \boldsymbol{X}}^{*} (\boldsymbol{e} - \boldsymbol{P}_k) \tag{14}$$

The mathematical induction method is applied for the proof. Use the data structure "Queue" to represent ***X***, ***X*** = {$x_{k1}$, $x_{k2}$, …, $x_p$, $x_{p+1}$, …}. Let $\boldsymbol{X}_u$ be the queue consisting of the first *u* elements of ***X***.

$$\boldsymbol{X}_u = \{x_{kz} \in \boldsymbol{X} \mid 1 \le kz \le u\} \tag{15}$$

(1) When *u* = 1, by Equation (10), $\boldsymbol{J}(\boldsymbol{X}_1) = \boldsymbol{P}_{k1}$. Obviously, $\boldsymbol{e} - \boldsymbol{J}(\boldsymbol{X}_1) = \boldsymbol{e} - \boldsymbol{P}_{k1}$.

(2) Suppose that $\boldsymbol{e} - \boldsymbol{J}(\boldsymbol{X}_p) = \prod\limits_{x_k \in \boldsymbol{X}_p}^{*} (\boldsymbol{e} - \boldsymbol{P}_k)$ when *u* = *p*. When *u* = *p* + 1, we apply Equation (10) to expand $\boldsymbol{J}(\boldsymbol{X}_{p+1})$:

$$\boldsymbol{e} - \boldsymbol{J}(\boldsymbol{X}_{p+1}) = \boldsymbol{e} - (\boldsymbol{P}_{k1} \vee \boldsymbol{P}_{k2} \vee \ldots \vee \boldsymbol{P}_{kp} \vee \boldsymbol{P}_{kp+1}) = \boldsymbol{e} - \left[\boldsymbol{J}(\boldsymbol{X}_p) \vee \boldsymbol{P}_{kp+1}\right] \tag{16}$$

Because for any 0–1 vector ***a***, ***b***, we have $\boldsymbol{a} \vee \boldsymbol{b} = \boldsymbol{a} + \boldsymbol{b} - \boldsymbol{a} * \boldsymbol{b}$, $\boldsymbol{a} - \boldsymbol{a} * \boldsymbol{b} = \boldsymbol{a} * (\boldsymbol{e} - \boldsymbol{b})$,

$$\begin{aligned} \boldsymbol{e} - \boldsymbol{J}(\boldsymbol{X}_{p+1}) &= \boldsymbol{e} - \left[\boldsymbol{J}(\boldsymbol{X}_p) + \boldsymbol{P}_{kp+1} - \boldsymbol{J}(\boldsymbol{X}_p) * \boldsymbol{P}_{kp+1}\right] \\ &= \boldsymbol{e} - \left\{\boldsymbol{J}(\boldsymbol{X}_p) + \left[\boldsymbol{e} - \boldsymbol{J}(\boldsymbol{X}_p)\right] * \boldsymbol{P}_{kp+1}\right\} = \left[\boldsymbol{e} - \boldsymbol{J}(\boldsymbol{X}_p)\right] * \left(\boldsymbol{e} - \boldsymbol{P}_{kp+1}\right) \end{aligned} \tag{17}$$

Substituting the assumptions $\boldsymbol{e} - \boldsymbol{J}(\boldsymbol{X}_p) = \prod\limits_{x_k \in \boldsymbol{X}_p}^{*} (\boldsymbol{e} - \boldsymbol{P}_k)$ into Equation (17), we obtain

$$\boldsymbol{e} - \boldsymbol{J}(\boldsymbol{X}_{p+1}) = \left(\boldsymbol{e} - \boldsymbol{P}_{kp+1}\right) * \prod_{x_k \in \boldsymbol{X}_p}^{*} (\boldsymbol{e} - \boldsymbol{P}_k) = \prod_{x_k \in \boldsymbol{X}_{p+1}}^{*} (\boldsymbol{e} - \boldsymbol{P}_k) \tag{18}$$

This proves that Equation (14) holds. By shifting the terms of Equation (14), the numerical expression of the switching function is obtained as follows.

$$\boldsymbol{J}(\boldsymbol{X}) = \boldsymbol{e} - \prod_{x_k \in \boldsymbol{X}}^{*} (\boldsymbol{e} - \boldsymbol{P}_k) \tag{19}$$

### 2.2.4. Numerical Computational Expression for the Minimum Fault Set Constraint

In Equation (6) of the objective function calculation for fault location, we introduced a second term to the right side of the equation. This is due to the possibility that the minimum value of the first term of Equation (6) corresponds to multiple assumed fault scenarios. Taking the distribution network shown in Figure 2 as an example, assuming section $x_3$ faults, the fault information reported by the FTUs is $\boldsymbol{I} = [I_1\text{\textasciitilde}I_6] = [1\ 1\ 1\ 0\ 0\ 0]$. For the assumed fault scenarios $\boldsymbol{X}_1 = \{x_3\}$, $\boldsymbol{X}_2 = \{x_1,x_3\}$, $\boldsymbol{X}_3 = \{x_2,x_3\}$, $\boldsymbol{X}_4 = \{x_1,x_2,x_3\}$, according to Equation (19) of the switching function, we can obtain $\boldsymbol{J}(\boldsymbol{X}_1) = \boldsymbol{J}(\boldsymbol{X}_2) = \boldsymbol{J}(\boldsymbol{X}_3) = \boldsymbol{J}(\boldsymbol{X}_4) = [1\ 1\ 1\ 0\ 0\ 0] = \boldsymbol{I}$ (the same results can be obtained by applying the logical expression (10) of the switching function because the two are equivalent), substituting the first term of Equation (6): $\sum_{i=1}^{n}\left[I_i \oplus J_i(\boldsymbol{X})\right] = 0$. That is, the assumed fault scenarios $\boldsymbol{X}_1$~$\boldsymbol{X}_4$ all lead to the minimum value of the first term of Equation (6).

This phenomenon has been discussed in more depth in [18], and the theory of "minimum fault set" has been proposed to solve this problem. The meaning of the "minimum fault set" theory is that the fault scenario with the lowest number of fault sections has the highest probability of occurrence. The mathematical description is as follows: for the set of fault sections $\boldsymbol{X}$, if there exists a proper subset $\boldsymbol{X}'$ of $\boldsymbol{X}$ such that $\boldsymbol{J}(\boldsymbol{X}) = \boldsymbol{J}(\boldsymbol{X}')$, then $\boldsymbol{X}$ is determined to be an infeasible solution that does not satisfy the "minimum fault set" constraint. Taking this fault case as an example, $\boldsymbol{X}_1$ is a proper subset of $\boldsymbol{X}_2$~$\boldsymbol{X}_4$ with $\boldsymbol{J}(\boldsymbol{X}_1) = \boldsymbol{J}(\boldsymbol{X}_2) = \boldsymbol{J}(\boldsymbol{X}_3) = \boldsymbol{J}(\boldsymbol{X}_4)$; thus, $\boldsymbol{X}_2$~$\boldsymbol{X}_4$ is an infeasible solution that does not satisfy the "minimum fault set" constraint; thus, the unique optimal solution $\boldsymbol{X}_1$ can be determined.

For the application of the "minimum fault set" constraint, it is common to add a penalty function to modify the objective function (e.g., the second term in Equation (6)) to ensure that the result of the fault location satisfies the "minimum fault set" requirement. However, this will expand the number of feasible solutions, which is not conducive to improving computational efficiency.

In this paper, we propose applying the numerical computational expression of the road vector to express the "minimum fault set" constraint. Considering that in these fault scenarios, sections $x_1$ and $x_2$ are on the road of $x_3$, the following inference was made according to Theorem 1.

The set of fault sections $\boldsymbol{X}$ satisfying the minimum fault set constraint is equivalent to the following numerical computational expression.

$$\forall x_i, x_j \in \boldsymbol{X}, \boldsymbol{P}_i \neq \boldsymbol{P}_i * \boldsymbol{P}_j \tag{20}$$

The meaning of Equation (20) is that the sections in the set of fault sections $\boldsymbol{X}$ are not on each other's road, which is a sufficient and necessary condition for $\boldsymbol{X}$ to satisfy the "minimum fault set" constraint. The proof is given below.

(1) Necessity. Suppose for the sake of contradiction that the set of fault sections $\boldsymbol{X}$ satisfies the "minimum fault set" constraint but $\exists x_i, x_j \in \boldsymbol{X}$ such that $\boldsymbol{P}_i = \boldsymbol{P}_i * \boldsymbol{P}_j$. Then: $\boldsymbol{P}_i - \boldsymbol{P}_i * \boldsymbol{P}_j = \boldsymbol{P}_i * (\boldsymbol{e} - \boldsymbol{P}_j) = \boldsymbol{0}$. Let $\boldsymbol{X}_a = \boldsymbol{X} - x_i$, $\boldsymbol{X}_b = \boldsymbol{X}_a - x_j$, according to Equation (19):

$$\begin{aligned} \boldsymbol{J}(\boldsymbol{X}) &= \boldsymbol{e} - \prod_{x_k \in \boldsymbol{X}}^{*} (\boldsymbol{e} - \boldsymbol{P}_k) = \boldsymbol{e} - \left[(\boldsymbol{e} - \boldsymbol{P}_i) * (\boldsymbol{e} - \boldsymbol{P}_j)\right] * \prod_{x_k \in \boldsymbol{X}_b}^{*} (\boldsymbol{e} - \boldsymbol{P}_k) \\ &= \boldsymbol{e} - \left[\boldsymbol{e} - \boldsymbol{P}_i * (\boldsymbol{e} - \boldsymbol{P}_j) - \boldsymbol{P}_j\right] * \prod_{x_k \in \boldsymbol{X}_b}^{*} (\boldsymbol{e} - \boldsymbol{P}_k) \\ &= \boldsymbol{e} - (\boldsymbol{e} - \boldsymbol{P}_j) * \prod_{x_k \in \boldsymbol{X}_b}^{*} (\boldsymbol{e} - \boldsymbol{P}_k) = \boldsymbol{J}(\boldsymbol{X}_a) \end{aligned} \tag{21}$$

$\boldsymbol{X}_a$ is a proper subset of $\boldsymbol{X}$, contradicting the assumption that $\boldsymbol{X}$ satisfies the "minimum fault set" constraint.

(2) Sufficiency. Suppose for the sake of contradiction that $\forall x_i, x_j \in \boldsymbol{X}$, $\boldsymbol{P}_i \neq \boldsymbol{P}_i * \boldsymbol{P}_j$, but the set of fault section $\boldsymbol{X}$ does not satisfy the "minimum fault set" constraint. Then,

there exists a proper subset $\boldsymbol{X}_a$ of $\boldsymbol{X}$ such that $\boldsymbol{J}(\boldsymbol{X}_a)=\boldsymbol{J}(\boldsymbol{X})$. Let $\boldsymbol{X}_b=\boldsymbol{X}-\boldsymbol{X}_a$, and $\boldsymbol{X}_b\neq\varnothing$. Then: $\boldsymbol{X}=\boldsymbol{X}_a\cup\boldsymbol{X}_b$. According to Equations (14) and (19):

$$\begin{aligned}\boldsymbol{J}(\boldsymbol{X}) &= \boldsymbol{e}-\overset{*}{\prod_{x_k\in\boldsymbol{X}}}(\boldsymbol{e}-\boldsymbol{P}_k)=\boldsymbol{e}-\overset{*}{\prod_{x_k\in\boldsymbol{X}_a}}(\boldsymbol{e}-\boldsymbol{P}_k)*\overset{*}{\prod_{x_k\in\boldsymbol{X}_b}}(\boldsymbol{e}-\boldsymbol{P}_k)\\ &= \boldsymbol{e}-\left[\boldsymbol{e}-\boldsymbol{J}(\boldsymbol{X}_a)\right]*\left[\boldsymbol{e}-\boldsymbol{J}(\boldsymbol{X}_b)\right]\\ &= \boldsymbol{J}(\boldsymbol{X}_a)+\boldsymbol{J}(\boldsymbol{X}_b)-\boldsymbol{J}(\boldsymbol{X}_a)*\boldsymbol{J}(\boldsymbol{X}_b)\end{aligned} \tag{22}$$

Because $\boldsymbol{J}(\boldsymbol{X}_a)=\boldsymbol{J}(\boldsymbol{X})$, $\boldsymbol{J}(\boldsymbol{X}_b)-\boldsymbol{J}(\boldsymbol{X}_a)*\boldsymbol{J}(\boldsymbol{X}_b)=\boldsymbol{0}$, i.e., $\boldsymbol{J}(\boldsymbol{X}_b)*\left[\boldsymbol{e}-\boldsymbol{J}(\boldsymbol{X}_a)\right]=\boldsymbol{0}$. Let $x_i\in\boldsymbol{X}_b$ and $\boldsymbol{X}_c=\boldsymbol{X}_b-x_i$. According to Equation (19):

$$\begin{aligned}\boldsymbol{J}(\boldsymbol{X}_b)*\left[\boldsymbol{e}-\boldsymbol{J}(\boldsymbol{X}_a)\right] &= \left[\boldsymbol{e}-\overset{*}{\prod_{x_k\in\boldsymbol{X}_b}}(\boldsymbol{e}-\boldsymbol{P}_k)\right]*\left[\boldsymbol{e}-\boldsymbol{J}(\boldsymbol{X}_a)\right]\\ &= \left[\boldsymbol{e}-(\boldsymbol{e}-\boldsymbol{P}_i)*\overset{*}{\prod_{x_k\in\boldsymbol{X}_c}}(\boldsymbol{e}-\boldsymbol{P}_k)\right]*\left[\boldsymbol{e}-\boldsymbol{J}(\boldsymbol{X}_a)\right]\\ &= \left[\boldsymbol{e}-\overset{*}{\prod_{x_k\in\boldsymbol{X}_c}}(\boldsymbol{e}-\boldsymbol{P}_k)\right]*\left[\boldsymbol{e}-\boldsymbol{J}(\boldsymbol{X}_a)\right]+\boldsymbol{P}_i*\left[\boldsymbol{e}-\boldsymbol{J}(\boldsymbol{X}_a)\right]*\overset{*}{\prod_{x_k\in\boldsymbol{X}_c}}(\boldsymbol{e}-\boldsymbol{P}_k)\\ &= \boldsymbol{0}\end{aligned} \tag{23}$$

The two preceding and following terms on the right-hand side of Equation (23) are both 0–1 vectors; thus, they are both **0** vectors by adding them to equal the **0** vector. According to Equation (14), $\left[\boldsymbol{e}-\boldsymbol{J}(\boldsymbol{X}_a)\right]=\overset{*}{\prod_{x_k\in\boldsymbol{X}_a}}\left(\boldsymbol{e}-\boldsymbol{P}_k\right)$, substitute the latter term on the right-hand side of Equation (23):

$$\begin{aligned}\boldsymbol{P}_i*\left[\boldsymbol{e}-\boldsymbol{J}(\boldsymbol{X}_a)\right]*\overset{*}{\prod_{x_k\in\boldsymbol{X}_c}}(\boldsymbol{e}-\boldsymbol{P}_k) &= \boldsymbol{P}_i*\overset{*}{\prod_{x_k\in\boldsymbol{X}_a}}\left(\boldsymbol{e}-\boldsymbol{P}_k\right)*\overset{*}{\prod_{x_k\in\boldsymbol{X}_c}}(\boldsymbol{e}-\boldsymbol{P}_k)\\ &= \boldsymbol{P}_i*\overset{*}{\prod_{x_k\in\boldsymbol{X}_a\cup\boldsymbol{X}_c}}(\boldsymbol{e}-\boldsymbol{P}_k)=\boldsymbol{0}\end{aligned} \tag{24}$$

According to Theorem 3, $\exists x_j\in\boldsymbol{X}_a\cup\boldsymbol{X}_c$ such that $\boldsymbol{P}_i=\boldsymbol{P}_i*\boldsymbol{P}_j$, where $x_i\in\boldsymbol{X}_b\subseteq\boldsymbol{X}$, $x_j\in\boldsymbol{X}_a\cup\boldsymbol{X}_c\subseteq\boldsymbol{X}$, contradicting the assumption that $\forall x_i,x_j\in\boldsymbol{X}$, $\boldsymbol{P}_i\neq\boldsymbol{P}_i*\boldsymbol{P}_j$.

This proves that this inference holds.

### 2.2.5. Numerical Computational Expression for the Objective Function

When considering the minimum fault set constraint, the objective function shown in Equation (6) can be simplified and rewritten in matrix form, as shown in Equation (25):

$$\min f(\boldsymbol{X})=\boldsymbol{e}^T\left[\boldsymbol{I}\oplus\boldsymbol{J}(\boldsymbol{X})\right] \tag{25}$$

where " $\oplus$ " means that the corresponding elements of two same-dimension vectors undergo logical "XOR" operations.

The vectors $\boldsymbol{I}$ and $\boldsymbol{J}(\boldsymbol{X})$ are both 0–1 vectors. Because for any 0–1 vector ***a***, ***b***, we have $\boldsymbol{e}^T\left(\boldsymbol{a}\oplus\boldsymbol{b}\right)=\left(\boldsymbol{a}-\boldsymbol{b}\right)^T\left(\boldsymbol{a}-\boldsymbol{b}\right)$, we can translate Equation (25) into a numerical expression as shown in Equation (26).

$$\min f(\boldsymbol{X})=\left[\boldsymbol{I}-\boldsymbol{J}(\boldsymbol{X})\right]^T\left[\boldsymbol{I}-\boldsymbol{J}(\boldsymbol{X})\right] \tag{26}$$

In summary, we complete the equivalent transformation of the logical operations in the fault location model by using Equation (26) as the objective function, Equation (19) as the equality constraint, and Equation (20) as the inequality constraint to construct the numerical computational model for fault location, as shown in Equation (27).

$$\begin{cases} \min f(\boldsymbol{X}) = \boldsymbol{e}^T\left[\boldsymbol{I} \oplus \boldsymbol{J}(\boldsymbol{X})\right] \\ \boldsymbol{J}(\boldsymbol{X}) = \boldsymbol{e} - \prod\limits_{x_k \in \boldsymbol{X}}^{*}\left(\boldsymbol{e} - \boldsymbol{P}_k\right) \\ \forall x_i, x_j \in \boldsymbol{X}, \boldsymbol{P}_i \neq \boldsymbol{P}_i * \boldsymbol{P}_j \end{cases} \tag{27}$$

It can be seen in Equation (27) that the numerical computation model for fault location does not contain logical operations, which lays a theoretical foundation for a highly efficient solution for the model obtained through strict mathematical derivation.

## 3. Recursive Structure of Fault Location for the Distribution Network

Divide-and-conquer (DAC) is a classical algorithm design strategy whose core idea is to apply the recursive mode of "divide-conquer-combine" to solve the target problem in a reduced dimension, which is an effective means to improve the efficiency of the algorithm [32]. In this section, we introduce the concepts of "compatibility sets" and "approximation gains" to demonstrate the recursive structure of the fault location problem and provide theoretical support for applying DAC to locate faults in distribution networks quickly and accurately.

### *3.1. Theoretical Foundation*

#### 3.1.1. Compatible Sets

Let $\boldsymbol{E}$ be the set of sections in the distribution network, $\boldsymbol{X} \subseteq \boldsymbol{E}$ be the set of fault sections, and $x_i \in \boldsymbol{X}$. The meaning of the compatible set of section $x_i$ is the condition that the road vector $\boldsymbol{P}_k$ of the other sections $x_k \in \boldsymbol{X} - x_i$ in $\boldsymbol{X}$ should be satisfied for $\boldsymbol{X}$ to satisfy the inequality constraint described in Equation (20). We define the compatible set $\boldsymbol{C}_i$ of section $x_i$ according to the inequality constraint expression (20).

$$\boldsymbol{C}_i = \{x_k \in \boldsymbol{E} \mid \boldsymbol{P}_i \neq \boldsymbol{P}_i * \boldsymbol{P}_k \,\&\, \boldsymbol{P}_k \neq \boldsymbol{P}_i * \boldsymbol{P}_k\} \tag{28}$$

The properties of compatible sets are the basis of the argument for recursive structures. These properties are stated first, and their mathematical proofs are given in Appendix B.

**Theorem 4.** *Let the set of sections in the distribution network be* $\boldsymbol{E}$, *sets of fault sections be* $\boldsymbol{X}_a, \boldsymbol{X}_b \subseteq \boldsymbol{E}$, *and* $\boldsymbol{X}_a \cup \boldsymbol{X}_b = \varnothing$. *Then,* $\boldsymbol{X}_a \cup \boldsymbol{X}_b$ *satisfies the inequality constraint equivalent to the following conclusion:* $\boldsymbol{X}_a, \boldsymbol{X}_b$ *satisfies the inequality constraint and* $\forall x_i \in \boldsymbol{X}_a$, $\boldsymbol{X}_b \subseteq \boldsymbol{C}_i$.

**Theorem 5.** *If section* $x_i$ *is on the road of section* $x_k$, *then* $\boldsymbol{C}_i \subseteq \boldsymbol{C}_k$.

**Theorem 6.** *If section* $x_i$ *is on the road of section* $x_k$ *and the path from* $x_k$ *to* $x_i$ *does not contain a T-type section, then* $\boldsymbol{C}_i = \boldsymbol{C}_k$.

#### 3.1.2. Approximation Gains

Let the set of sections in the distribution network be $\boldsymbol{E}$, $\boldsymbol{X} \subseteq \boldsymbol{E}$ be the set of fault sections, and section $x_i \notin \boldsymbol{X}$. The approximation gain of section $x_i$ to $\boldsymbol{X}$ describes the contribution to reducing the objective function $f(\boldsymbol{X})$ after section $x_i$ is added to $\boldsymbol{X}$ (that is, section $x_i$ is determined to be the fault section). The approximate gain of section $x_i$ to set $\boldsymbol{X}$ is shown in Equation (29).

$$\Delta_i^{\boldsymbol{X}} = f(\boldsymbol{X}) - f(\boldsymbol{X} \cup x_i) \tag{29}$$

By expanding $f(\boldsymbol{X})$ and $f(\boldsymbol{X} \cup x_i)$ with Equation (26) and substituting Equation (19) for $J(\boldsymbol{X})$, the equation for $\Delta_i^{\boldsymbol{X}}$ can be deduced:

$$\Delta_i^{\boldsymbol{X}} = \left[ \left(2\boldsymbol{I} - \boldsymbol{e}\right) * \prod_{x_k \in \boldsymbol{X}}^{*} \left(\boldsymbol{e} - \boldsymbol{P}_k\right) \right]^T \boldsymbol{P}_i \tag{30}$$

In particular, if $\boldsymbol{X}$ is the empty set, i.e., $\boldsymbol{X} = \varnothing$, then the superscript of $\Delta_i^{\varnothing}$ can be omitted and abbreviated as $\Delta_i$, which is referred to as the approximation gain of section $x_i$, as shown in Equation (31).

$$\Delta_i = f(\varnothing) - f(i) \tag{31}$$

Substitute $\prod_{x_k \in \varnothing}^{*} \left(\boldsymbol{e} - \boldsymbol{P}_k\right) = \boldsymbol{e}$ into Equation (30) to obtain the calculation formula of $\Delta_i$.

$$\Delta_i = \left(2\boldsymbol{I} - \boldsymbol{e}\right)^T \boldsymbol{P}_i \tag{32}$$

Extending the concept of section approximation gain, the approximation gain of the set of sections is defined. Let sets of sections $\boldsymbol{X}, \boldsymbol{X}_a \subseteq \boldsymbol{E}$ be such that $\boldsymbol{X} \cap \boldsymbol{X}_a = \varnothing$. The approximation gain of $\boldsymbol{X}$ to $\boldsymbol{X}_a$ describes the contribution to reducing the objective function $f(\boldsymbol{X}_a)$ after adding all sections in $\boldsymbol{X}$ to $\boldsymbol{X}_a$. The approximation gain of $\boldsymbol{X}$ to set $\boldsymbol{X}_a$ is shown in Equation (33).

$$\Delta_{\boldsymbol{X}}^{\boldsymbol{X}_a} = f(\boldsymbol{X}_a) - f(\boldsymbol{X}_a \cup \boldsymbol{X}) \tag{33}$$

In particular, if $\boldsymbol{X}_a$ is the empty set, i.e., $\boldsymbol{X}_a = \varnothing$, then the superscript of $\Delta_{\boldsymbol{X}}^{\varnothing}$ can be omitted and abbreviated as $\Delta_{\boldsymbol{X}}$, which is referred to as the approximation gain of the set of sections $\boldsymbol{X}$.

The property of the approximation gains is the basis of the argument for the recursive structure. These properties are stated first, and their mathematical proofs are given in Appendix C.

**Theorem 7.** *Let the set of sections in the distribution network be* $\boldsymbol{E}$. *For section* $x_i \in \boldsymbol{E}$, *set of sections* $\boldsymbol{X} \subseteq \{x_k \in \boldsymbol{E} \mid \boldsymbol{C}_i \subseteq \boldsymbol{C}_k\}$, $\boldsymbol{X} \neq \varnothing$, *and set* $\boldsymbol{R} \subseteq \boldsymbol{C}_i$, *we have* $\Delta_{\boldsymbol{R}}^{\boldsymbol{X}} = \Delta_{\boldsymbol{R}}^{i}$.

### *3.2. Recursive Structure*

The core idea of DAC is to apply the recursive structure to solve the target problem in a reduced dimension. Specifically, DAC follows a divide-conquer-combine approach in each level of recursion: it breaks the problem into several subproblems that are similar to the original problem but smaller in size, solves the subproblems recursively (if the subproblem satisfies the "bottoms out" condition, the recursion is stopped and the solution is solved directly), and then combines these solutions to create a solution to the original problem.

From this, we can deduce the prerequisites for applying DAC to solve the target problem; that is, the target problem should have the following recursive structure.

- The original problem and the subproblem should have the same descriptive form.
- When the "bottoms out" condition is satisfied, it enters the "base case" and can be solved directly.
- If the condition of "bottoms out" is not satisfied, it enters the "recursive case", and the solution of the original problem can be created by combining the solutions of the subproblem.

This section discusses the description of the subproblem, the base case, and the recursive case of the fault section location for the distribution network, and the construction of the recursive structure of the fault location problem is described.

### 3.2.1. Subproblem Description

The subproblem of distribution network fault section location is the problem of solving its subnetwork fault location result. The mathematical description of the subnetwork and its fault location results are presented here, followed by the formal consistency of the original problem with the subproblem is proved.

The mathematical description of the subnetwork is as follows. Let the set of sections in the distribution network be $\boldsymbol{E}$. Given section $x_i \in \boldsymbol{E}$, we define the set of sections $\boldsymbol{E}_i$ : $\boldsymbol{E}_i = \{x_k \in \boldsymbol{E} \mid \boldsymbol{P}_i = \boldsymbol{P}_i * \boldsymbol{P}_k\}$. The distribution network subgraph $\boldsymbol{G}_i = (\boldsymbol{V}_i, \boldsymbol{E}_i)$ is called a subnetwork starting from the section $x_i$. Among them, $\boldsymbol{V}_i$ is the set of nodes with the same section number as $\boldsymbol{E}_i$.

According to Theorem 1, the essence of the subnetwork starting from section $x_i$ is the set of all distribution network sections (containing $x_i$) that contain section $x_i$ on all the roads. Taking the multi-branch distribution network shown in Figure 3 as an example, section $x_3$ is on the road of sections $x_4 \sim x_{14}$. We construct the set $\boldsymbol{E}_3 = \{x_k \in \boldsymbol{E} \mid 3 \le k \le 14\}$, $\forall x_k \in \boldsymbol{E}_3$, according to Theorem 1, with $\boldsymbol{P}_3 = \boldsymbol{P}_3 * \boldsymbol{P}_k$. Then, we construct the set $\boldsymbol{V}_3 = \{s_k \in \boldsymbol{V} \mid 3 \le k \le 14\}$, and the subgraph $\boldsymbol{G}_3 = (\boldsymbol{V}_3, \boldsymbol{E}_3)$ is a subnetwork starting from section $x_3$.

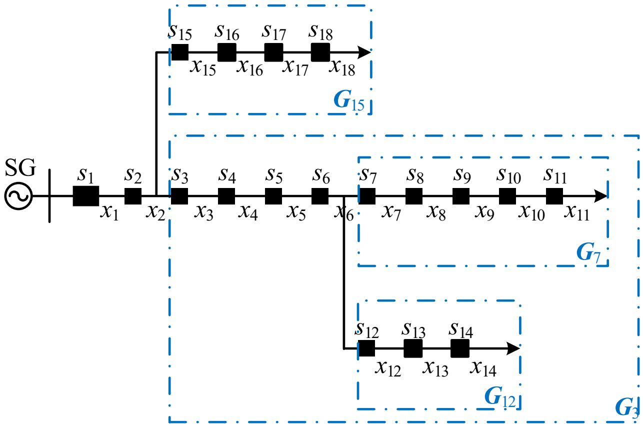

$s_{15}$ $s_{16}$ $s_{17}$ $s_{18}$
$x_{15}$ $x_{16}$ $x_{17}$ $x_{18}$
$\boldsymbol{G}_{15}$
SG
$s_1$ $s_2$ $s_3$ $s_4$ $s_5$ $s_6$ $s_7$ $s_8$ $s_9$ $s_{10}$ $s_{11}$
$x_1$ $x_2$ $x_3$ $x_4$ $x_5$ $x_6$ $x_7$ $x_8$ $x_9$ $x_{10}$ $x_{11}$
$\boldsymbol{G}_7$
$s_{12}$ $s_{13}$ $s_{14}$
$x_{12}$ $x_{13}$ $x_{14}$
$\boldsymbol{G}_{12}$
$\boldsymbol{G}_3$

**Figure 3.** Multi-branch distribution network.

The mathematical description of the subnetwork fault location result is as follows: Let $\boldsymbol{G}_i = (\boldsymbol{V}_i, \boldsymbol{E}_i)$ be a subnetwork and $\boldsymbol{X}_i \subseteq \boldsymbol{E}_i \neq \varnothing$ be the set of fault sections. It holds that $\forall \boldsymbol{X}' \subseteq \boldsymbol{E}_i \neq \boldsymbol{X}_i \neq \varnothing$, $\forall \boldsymbol{R} \subseteq \boldsymbol{E} - \boldsymbol{E}_i$, $\boldsymbol{X}' \cup \boldsymbol{R}$ is a feasible solution to the inequality constraint. If $\boldsymbol{X}' \cup \boldsymbol{R}$ satisfies the inequality constraint and there is always $f(\boldsymbol{X}_i \cup \boldsymbol{R}) < f(\boldsymbol{X}' \cup \boldsymbol{R})$, then $\boldsymbol{X}_i$ is the fault location result of the subnetwork $\boldsymbol{G}_i$, and $\boldsymbol{X}_i$ is called the fault candidate set for $\boldsymbol{G}_i$.

We prove the formal consistency of the original problem with the subproblem. According to the numbering rules in this paper, section $x_1$ is a feeder section surrounded by the circuit breaker on the distribution side of the main power supply and its adjacent intelligent switch; thus, $x_1$ is on the road of all sections. According to Theorem 1, $\forall x_k \in \boldsymbol{E}$, there is $\boldsymbol{P}_1 = \boldsymbol{P}_1 * \boldsymbol{P}_k$. Thus, the whole distribution network can be considered a subnetwork starting from section $x_1$, i.e., $\boldsymbol{G} = \boldsymbol{G}_1 = (\boldsymbol{V}_1, \boldsymbol{E}_1)$. Let $\boldsymbol{X}_1$ be the fault candidate set for $\boldsymbol{G}_1$. It holds that $\forall \boldsymbol{X}' \subseteq \boldsymbol{E} \neq \boldsymbol{X}_1$, $\forall \boldsymbol{R} = \boldsymbol{E} - \boldsymbol{E}_1$, $\boldsymbol{X}' \cup \boldsymbol{R}$ is a feasible solution to the inequality constraint. According to the definition of the fault candidate set, $\boldsymbol{X}_1 \cup \boldsymbol{R}$ satisfies the inequality constraint, and there is always $f(\boldsymbol{X}_1) < f(\boldsymbol{X}')$. Considering $\boldsymbol{E} = \boldsymbol{E}_1$, $\boldsymbol{R} = \boldsymbol{E} - \boldsymbol{E}_1 = \varnothing$, $\boldsymbol{X}_1 \cup \boldsymbol{R} = \boldsymbol{X}_1$, so $\boldsymbol{X}_1$ is a feasible solution satisfying the inequality constraint. Additionally, because $\forall \boldsymbol{X}' \subseteq \boldsymbol{E} \neq \boldsymbol{X}_1$, we have $f(\boldsymbol{X}_1) < f(\boldsymbol{X}')$, $\boldsymbol{X}_1$ therefore is the optimal solution of the objective function. It can be seen that the distribution network fault location problem can be solved by solving the fault candidate set $\boldsymbol{X}_1$ for its subnetwork $\boldsymbol{G}_1$. This proves the formal consistency of the original problem with the subproblem. The solution of the fault candidate set for the subnetwork is studied as follows.

### 3.2.2. Base Case

For a subnetwork that does not contain a T-type section (e.g., the subnetwork $\boldsymbol{G}_7$, $\boldsymbol{G}_{12}$, $\boldsymbol{G}_{15}$ shown in Figure 3), no recursion is required, and its fault candidate set consists of the section with the largest value of the approximation gain in that network, as described in Theorem 8.

**Theorem 8.** *Let $\boldsymbol{G}_i = (\boldsymbol{V}_i, \boldsymbol{E}_i)$ be a subnetwork that does not contain a T-type section, and suppose $x_k \in \boldsymbol{E}_i$; if, $\forall x_r \neq x_k \in \boldsymbol{E}_i$, there is always $\Delta_k > \Delta_r$, then the fault section set $\boldsymbol{X}_i = \{x_k\}$ is the fault candidate set for $\boldsymbol{G}_i$.*

The proof of Theorem 8 is given in Appendix D.

### 3.2.3. Recursive Case

For a subnetwork containing a T-type section (e.g., subnetwork $\boldsymbol{G}_3$ shown in Figure 3), a recursive solution is required. Its fault candidate set is constructed from the fault candidate set of a subnetwork smaller than its size, as described in Theorem 9.

**Theorem 9.** *Let the subnetwork $\boldsymbol{G}_i = (\boldsymbol{V}_i, \boldsymbol{E}_i)$ contain T-type sections, and let section $x_t$ be the T-type section in $\boldsymbol{G}_i$ that is closest to the main power supply, assuming that xu and xv are the downstream sections of $x_t$, as shown in Figure 4. Let $\boldsymbol{G}_u = (\boldsymbol{V}_u, \boldsymbol{E}_u)$ and $\boldsymbol{G}_v = (\boldsymbol{V}_v, \boldsymbol{E}_v)$ be subnetworks and set $\boldsymbol{E}_{i-t} = \boldsymbol{E}_i - \boldsymbol{E}_u - \boldsymbol{E}_v$. Suppose that for section $x_k \in \boldsymbol{E}_{i-t}$, and $\forall x_q \neq x_k \in \boldsymbol{E}_{i-t}$, there is always $\Delta_k > \Delta_q$. Let $\boldsymbol{X}_{i-t} = \{x_k\}$, and let its corresponding approximation gain be $\Delta_{\boldsymbol{X}_{i-t}} = \Delta_k$. Suppose that $\boldsymbol{X}_u$ and $\boldsymbol{X}_v$ are the fault candidate sets for subnetworks $\boldsymbol{G}_u$ and $\boldsymbol{G}_v$ respectively, and construct the set $\boldsymbol{X}_{u\&v} = \boldsymbol{X}_u \cup \boldsymbol{X}_v$. Let $\Delta_{\boldsymbol{X}_{i-t}}$, $\Delta_{\boldsymbol{X}_u}$, $\Delta_{\boldsymbol{X}_v}$, and $\Delta_{\boldsymbol{X}_{u\&v}}$ be the approximation gains of sets $\boldsymbol{X}_{i-t}$, $\boldsymbol{X}_u$, $\boldsymbol{X}_v$, and $\boldsymbol{X}_{u\&v}$ respectively. Then, the set corresponding to the largest of $\Delta_{\boldsymbol{X}_{i-t}}$, $\Delta_{\boldsymbol{X}_u}$, $\Delta_{\boldsymbol{X}_v}$, and $\Delta_{\boldsymbol{X}_{u\&v}}$ is the fault candidate set $\boldsymbol{X}_i$ of $\boldsymbol{G}_i$.*

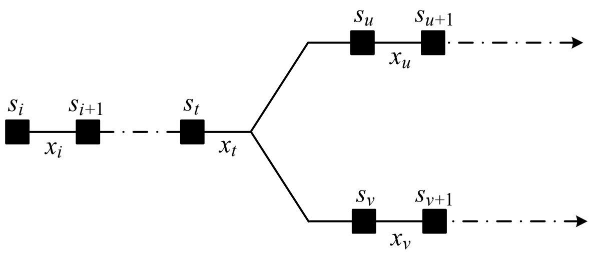


**Figure 4.** Subnetwork with T-type section.

The distribution network shown in Figure 3 is taken as an example to illustrate the meaning of Theorem 9. Subnetwork $\boldsymbol{G}_3$ contains T-type section $x_6$. The fault location result of this network is constructed from the fault candidate sets of subnetworks $\boldsymbol{G}_7$ and

$\boldsymbol{G}_{12}$, which are smaller than its size. Let the fault candidate sets of $\boldsymbol{G}_7$ and $\boldsymbol{G}_{12}$ be $\boldsymbol{X}_7$ and $\boldsymbol{X}_{12}$; their corresponding approximation gains are $\Delta_{\boldsymbol{X}_7}$ and $\Delta_{\boldsymbol{X}_{12}}$. Assume that the maximum approximation gain section found in the set of sections $\boldsymbol{E}_{3-6} = \boldsymbol{E}_3 - \boldsymbol{E}_7 - \boldsymbol{E}_{12} = \{x_3, x_4, x_5, x_6\}$ is $x_k$; construct the set $\boldsymbol{X}_{3-6} = \{x_k\}$, and its corresponding approximation gain is $\Delta_{X_{3-6}} = \Delta_k$. The set $\boldsymbol{X}_{7\&12} = \boldsymbol{X}_7 \cup \boldsymbol{X}_{12}$ is constructed, and its corresponding approximation gain is $\Delta_{\boldsymbol{X}_{7\&12}}$. Then, the set corresponding to the largest of $\Delta_{X_{3-6}}$, $\Delta_{\boldsymbol{X}_7}$, $\Delta_{\boldsymbol{X}_{12}}$, and $\Delta_{\boldsymbol{X}_{7\&12}}$ is the fault location result of $\boldsymbol{G}_3$.

The proof of Theorem 9 is given in Appendix D.

## 4. Fault Section Location Method for Distribution Networks Based on Divide-and-Conquer

In the previous section, we demonstrate the recursive structure of the numerical model for fault location. In this section, we propose a fault location algorithm based on DAC and analyze the computational efficiency of the algorithm.

### *4.1. Algorithm Design*

The fault location method closely follows the divide-and-conquer paradigm. Intuitively, it operates as follows.

- Divide. Divide the distribution network into smaller subnetworks using the T-type section closest to the main power source as the boundary (the decomposition effect is shown in Figure 4).
- Conquer. Recursively invoke the algorithm to solve the fault candidate set of the subnetwork. If the subnetwork no longer contains a T-type section, the recursion is stopped, and the fault candidate set is constructed directly according to Theorem 8.
- Combine. Combine the fault candidate set of subnetworks to create the fault candidate set of the original network according to Theorem 9.

The topological information of the network is the basis of the fault location for the distribution network, and the method of determining the topological information required by the algorithm is described below.

- Node number and section number. By applying the depth-first search algorithm, the section is traversed with the main power supply as the source node, and the sections are numbered in the order they are accessed so that the section numbers of any subnetwork in the network are consecutive and conform to the numbering rules in this paper.
- Section type vector ***T***. Construct the section type vector *T* to determine the type of the section. When $x_k$ is a T-type section, let ***T***(*k*) = 1; otherwise, ***T***(*k*) = 0.
- Adjacency list(*Adj*). Construct the adjacency list of the section to record the downstream section of each section for a fast search of the adjacency section of the T-type section, which helps implement the "divide" step in the recursive case.

When the distribution network topology changes, this operation is re-executed to make the algorithm flexible to adapt to the topology change.

The pseudocode DACFL gives the implementation of applying DAC to solve the subnetwork $\boldsymbol{G}_i$ fault candidate set. Because the subnetwork section numbers are consecutive, the diagnostic range is determined by the starting section number *i* and the maximum section number *j*. DACFL accepts *i*, *j*, network topology information Graph (containing adjacency list and section type vector ***T***), and the fault information vector ***I*** reported by FTUs as input. The fault candidate set ***X*** and its approximation gain $\Delta_X$ are returned. As described in Section 3.2.1, the distribution network fault location problem can be solved by solving the fault candidate set $\boldsymbol{X}_1$ of its subnetwork $\boldsymbol{G}_1$. Therefore, the algorithm initially set *i* = 1 and *j* to take the maximum number of the distribution network sections. The pseudocode for the algorithm is shown as Algorithm 1.

**Algorithm 1** DACFL (*i*, *j*, Gragh, ***I***, *D*)

```
 1: $k = i$
 2: $\Delta_k = D + 2\boldsymbol{I}(k) - 1$
 3: while $\boldsymbol{T}(k) \neq 0$ and $\Delta_k = D + 2\boldsymbol{I}(k) - 1$ do
 4:     $k = k + 1$
 5:     $\Delta_k = D + 2\boldsymbol{I}(k) - 1$
 6: end while
 7: if $k = j$ then                                                  // base case
 8:     return (FIND-MAX ($\Delta[i:j]$))
 9: end if
10: $ui = Adj(k)(1)$                                                  // divide
11: $vi = Adj(k)(1)$
12: if $ui > vi$ then
13:     $uj = j$
14:     $vj = ui - 1$
15: else then
16:     $vj = j$
17:     $uj = vi - 1$
18: end if
19: $(X_u, \Delta_{X_u}) = \mathrm{DACFL}(ui, uj, \mathrm{Gragh}, \boldsymbol{I}, \Delta_k)$    // conquer
20: $(X_v, \Delta_{X_v}) = \mathrm{DACFL}(vi, vj, \mathrm{Gragh}, \boldsymbol{I}, \Delta_k)$
21: $(X_{i-t}, \Delta_{X_{i-t}}) = \text{FIND-MAX}(\Delta[i:k])$                    // combine
22: $\boldsymbol{X}_{u\&v} = \boldsymbol{X}_u \cup \boldsymbol{X}_v$
23: $\Delta_{X_{u\&v}} = \Delta_{X_u} + \Delta_{X_v} - \Delta_k$
24: if $\Delta_{X_u} = \max(\Delta_{X_{i-t}}, \Delta_{X_u}, \Delta_{X_v}, \Delta_{X_{u\&v}})$ then
25:     return ($X_u, \Delta_{X_u}$)
26: elseif $\Delta_{X_v} = \max(\Delta_{X_{i-t}}, \Delta_{X_u}, \Delta_{X_v}, \Delta_{X_{u\&v}})$ then
27:     return ($X_v, \Delta_{X_v}$)
28: elseif $\Delta_{X_{i-t}} = \max(\Delta_{X_{i-t}}, \Delta_{X_u}, \Delta_{X_v}, \Delta_{X_{u\&v}})$ then
29:     return ($X_{i-t}, \Delta_{X_{i-t}}$)
30: else then
31:     return ($X_{u\&v}, \Delta_{X_{u\&v}}$)
32: end if
```

In detail, the DACFL procedure works as follows. Lines 1~2 of DACFL set the initial pointer position and calculate the approximation gain of the starting section, where the value of D is 0 when the algorithm is started, and the approximation gain of the upstream T-type section of the starting section xi is taken at each recursion thereafter (the calculation of the approximation gain of section $x_k$ in lines 2 and 5 is described in the last paragraph of this section). Lines 3~6 traverse the section from the starting section $x_i$, determine whether there is a T-type section in $\boldsymbol{G}_i$, and complete the calculation of the approximation gain of the T-type section and its upstream section. If $\boldsymbol{G}_i$ does not contain a T-type section, the traversal process is not interrupted and ends with *k* = *j*. Otherwise, the process ends with $k$ recording the T-type section number in $\boldsymbol{G}_i$ that has the shortest distance from the main power supply.

Lines 7~9 correspond to the base case, where $\boldsymbol{G}_i$ does not contain a T-type section. According to Theorem 8, the algorithm returns the section with the largest approximation gain as the location result at this time. Among them, the subroutine FIND-MAX returns the position and value of the largest element in the array.

Lines 10~32 correspond to the recursive case of $\boldsymbol{G}_i$ containing T-type sections. Lines 10~17 are the "Divide" steps that determine the diagnostic range of the subnetwork $\boldsymbol{G}_u$ and $\boldsymbol{G}_v$. Lines 10~11 identify the starting section $x_{ui}$ and $x_{vi}$ of $\boldsymbol{G}_u$ and $\boldsymbol{G}_v$ from the adjacency list of section $x_k$. Because the section numbers are consecutive, lines 12~18 find the sections $x_{uj}$ and $x_{vj}$ with the largest number in $\boldsymbol{G}_u$ and $\boldsymbol{G}_v$ by comparing the sizes of $ui$ and $vi$. Lines 19~20 are the "Conquer" steps, which recursively invoke the algorithm to find the fault candidate sets $\boldsymbol{X}_u, \boldsymbol{X}_v$ of $\boldsymbol{G}_u, \boldsymbol{G}_v$ and their approximation gains $\Delta_{\boldsymbol{X}_u}, \Delta_{\boldsymbol{X}_v}$. Lines 21~32 are the "Combine" steps. Line 21 finds the section with the largest approximation gain in the set of sections $\boldsymbol{E}_{i-t}$ to form $\boldsymbol{X}_{i\text{-}t}$ and its approximation gains $\Delta_{\boldsymbol{X}_{i-t}}$. Lines 22~23 construct $\boldsymbol{X}_{u\&v}$ and calculate $\Delta_{\boldsymbol{X}_{u\&v}}$. Lines 24~32 find largest one from $\Delta_{\boldsymbol{X}_{i-t}}, \Delta_{\boldsymbol{X}_u}, \Delta_{\boldsymbol{X}_v}, \Delta_{\boldsymbol{X}_{u\&v}}$ and return its corresponding $\boldsymbol{X}$ and $\Delta$ as the result of the fault location for $\boldsymbol{G}_i$.

In summary, the flow chart of the algorithm in this paper is shown in Figure 5.

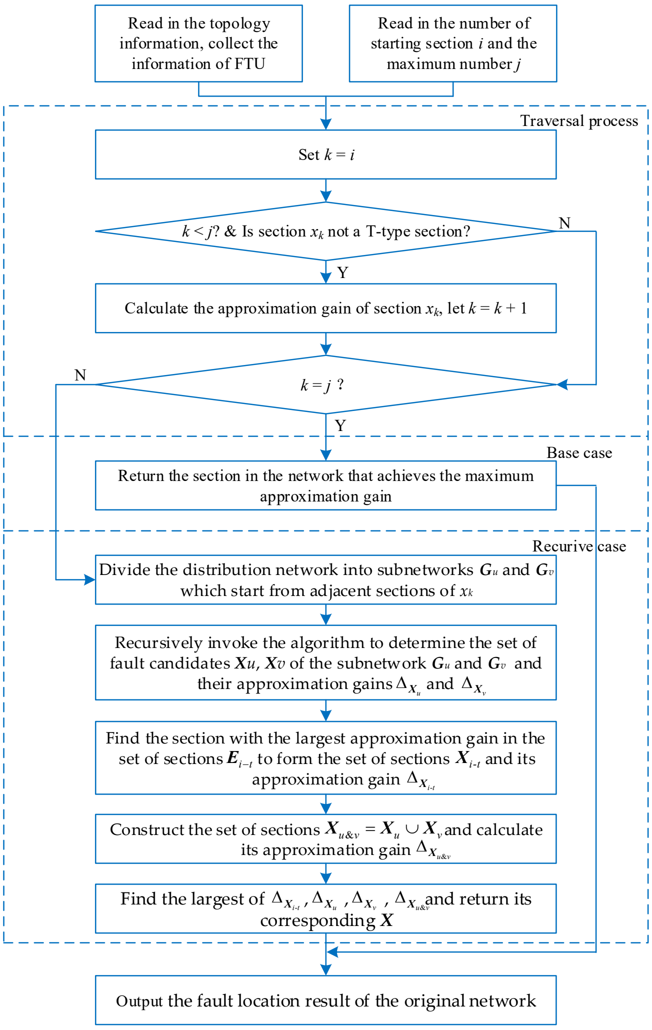

Read in the topology information, collect the information of FTU
Read in the number of starting section $i$ and the maximum number $j$
Traversal process
Set $k = i$
$k < j$? & Is section $x_k$ not a T-type section?
N
Y
Calculate the approximation gain of section $x_k$, let $k = k + 1$
N
$k = j$ ?
Y
Base case
Return the section in the network that achieves the maximum approximation gain
Recurive case
Divide the distribution network into subnetworks $\boldsymbol{G}_u$ and $\boldsymbol{G}_v$ which start from adjacent sections of $x_k$
Recursively invoke the algorithm to determine the set of fault candidates $\boldsymbol{X}u$, $\boldsymbol{X}v$ of the subnetwork $\boldsymbol{G}_u$ and $\boldsymbol{G}_v$ and their approximation gains $\Delta_{X_u}$ and $\Delta_{X_v}$
Find the section with the largest approximation gain in the set of sections $\boldsymbol{E}_{i-t}$ to form the set of sections $\boldsymbol{X}_{i-t}$ and its approximation gain $\Delta_{X_{i-t}}$
Construct the set of sections $\boldsymbol{X}_{u\&v} = \boldsymbol{X}_u \cup \boldsymbol{X}_v$ and calculate its approximation gain $\Delta_{X_{u\&v}}$
Find the largest of $\Delta_{X_{i-t}}, \Delta_{X_u}, \Delta_{X_v}, \Delta_{X_{u\&v}}$ and return its corresponding $\boldsymbol{X}$
Output the fault location result of the original network

**Figure 5.** Flowchart of the algorithm.

It should be noted that Formulas (32) and (33) calculating the approximation gains $\Delta_k$ and $\Delta_{\boldsymbol{x}_{u\&v}}$ have a linear time operation cost. To improve computational efficiency, lines 2, 5, and 23 of the algorithm use constant time operations to calculate $\Delta_k$ and $\Delta_{\boldsymbol{x}_{u\&v}}$ . The correctness of this calculation method is demonstrated in Appendixes E and F.

## 4.2. Efficiency Analysis

To theoretically prove that the algorithm can satisfy the requirement of fault location speed, this section describes application of the time complexity of the aggregation analysis algorithm pseudocode to argue the computational efficiency of the algorithm.

Lines 1~2 of DACFL set the initial pointer position and calculate the approximation gain of the starting section, which will be executed once in each recursion. Lines 3~6 traverse the sections of the distribution network, each of which takes constant time. The process will be interrupted by the appearance of T-type sections but can be continued in the subnetworks $\boldsymbol{G}_u$ and $\boldsymbol{G}_v$. According to aggregation analysis, lines 3~5 are executed once for each section of the distribution network. Line 7 determines the subnetwork type, executed once at each recursion. Line 8 and lines 10~32 correspond to the base and recursive cases, respectively, and are executed once per recursion at most. Each line takes constant time, except for lines 8 and 21. Lines 8 and 21 find the maximum value from the values of $j-i+1$ and $k-i+1$. Line 8 is executed for the base case. Line 21 is executed for the recursive case containing T-sections, but the same will be continued in the next recursion. Applying aggregation analysis, eventually, the approximation gain of each section will be taken out and compared once; thus, the overall number of operations in lines 8 and 21 is the number of sections in the distribution network.

Let the distribution network contain $n$ nodes, and the number of algorithm recursions is $l$. "Progressive notation" is applied to represent the operation cost and total running time of each line [32], and the analysis results of this algorithm efficiency are summarized in Table 4.

**Table 4.** Efficiency analysis.

| Steps | Number of Executions | Operation Cost | $f(X)$ |
|---|---|---|---|
| Lines 1~2 | $l$ | $\Theta(1)$ | $\Theta(l)$ |
| Lines 3~5 | $n$ | $\Theta(1)$ | $\Theta(n)$ |
| Line 6~7 | $l$ | $\Theta(1)$ | $\Theta(l)$ |
| Lines 8, 21 | $< l$ | $\Theta(j\text{-}i)$ or $\Theta(k\text{-}i)$ | $\Theta(n)$ |
| Line 9 | $l$ | $\Theta(1)$ | $\Theta(l)$ |
| Lines 10~20 | $< l$ | $\Theta(1)$ | $O(l)$ |
| Lines 22~32 | $< l$ | $\Theta(1)$ | $O(l)$ |

According to the "Divide" step of the pseudocode DACFL, the number of recursions depends on the number of downstream sections of the T-type sections in the distribution network; thus, $l < n$. Combining the analysis results of Table 4, the time complexity of DACFL is

$$\Theta(l)+\Theta(n)+\Theta(l)+\Theta(n)+\Theta(l)+O(l)+O(l)= \Theta(n) \tag{34}$$

It can be seen that the pseudocode DACFL has a linear level of time complexity, thereby proving the efficient computational capability of the algorithm in this paper. It can be foreseen that the method proposed in this section has great potential for improving the computational efficiency of fault location in the context of large-scale distribution networks.

### *4.3. Active Distribution Network Application Note*

For active distribution networks containing distributed generations (DGs), the short-circuit current after a fault will flow in both directions, and then, the encoding of fault information described in Equation (1) would no longer applicable. In this paper, we refer to the sections that reach the substation power supply through node $s_i$ as the downstream sections of node $s_i$ and the remaining sections in the distribution network as the upstream sections of node $s_i$. The fault information of node $s_i$ is determined by Equation (35) [33].

$$J_i(\boldsymbol{X})=\left\{\prod_{G_u}\left[K_{G_u}\left(1-\prod_{x_{Gu}\in\boldsymbol{E}_{i,G_u}}y_{Gu}\right)\right]\right\}\prod_{x_d\in\boldsymbol{E}_{i,d}}y_d-\left\{\prod_{G_d}\left[K_{G_d}\left(1-\prod_{x_{Gd}\in\boldsymbol{E}_{i,G_d}}y_{Gd}\right)\right]\right\}\prod_{x_u\in\boldsymbol{E}_{i,u}}y_u \tag{35}$$

where $G_u$ denotes the power source connected to the upstream section of node $s_i$, which is referred to as the upstream power source of node *si*. $K_{G_u}$ denotes the throw-off factor of $G_u$, when $G_u$ is put into operation $K_{G_u}=1$; otherwise, $K_{G_u}=0$. $y_{Gu}$ denotes the operating state of the feeder section $x_{Gu}$ between $s_i$ and $G_u$. In contrast, $G_d$, $K_{G_d}$, and $y_{Gd}$ denote the variables associated with the downstream power supply of node $s_i$. $G_d$ denotes the downstream power supply of $s_i$, $K_{G_d}$ denotes the throw-off factor of $G_d$, and $y_{Gd}$ denotes the operating state of the feeder section $x_{Gd}$ between $s_i$ and $G_d$. In addition, $y_d$ and $y_u$ indicate the operation status of the downstream section of and the upstream section of $s_i$, respectively, and " $\prod$ " indicates the logical "OR" superposition operation.

In Equation (35), it can be seen that each grid-connected power source will only provide fault current in a single direction to the node after a distribution network fault, and the fault state of the node can be regarded as the superposition of the states under the action of each power source individually. Therefore, the coding rule of node $s_i$ to grid-connected power sources *S* is proposed as

$$I_{i,S}=\begin{cases}1 & \text{Short circuit current is detected in the same direction} \\ & \text{as the short circuit current supplied by power supply S} \\ 0 & \text{Others}\end{cases} \tag{36}$$

Meanwhile, because the road from the section to each power source is unique, the distribution network with each power source as the main power source has a radial structure. In view of this, the following fault section location process for an active distribution network based on DAC is proposed.

- Step 1: Store the network topology information (including node section number, adjacency list, and section type vector *T*) of the distribution network with each power source as the main power source.
- Step 2: After the failure of the distribution network, take the grid-connected power supply as the main power supply, and the corresponding network topology information and node fault information vector determined by Equation (36) are taken as the input. The fault candidate sets $\boldsymbol{X}$ of each power supply as the main power supply are determined by the fetching algorithm.
- Step 3: The fault candidate sets $\boldsymbol{X}$ obtained from Step 2 is combined to obtain the final fault location result.

### *4.4. Fault Location Case*

To understand the fault location process of the method in this paper, this section selects various types of fault examples to elaborate on the operation process of the algorithm.

### 4.4.1. Test Case 1: Single Fault

In Figure 6, assume that section $x_9$ in the multi-branch distribution network has a fault. The nodes that detect the fault current report the fault information, which is marked with red boxes.

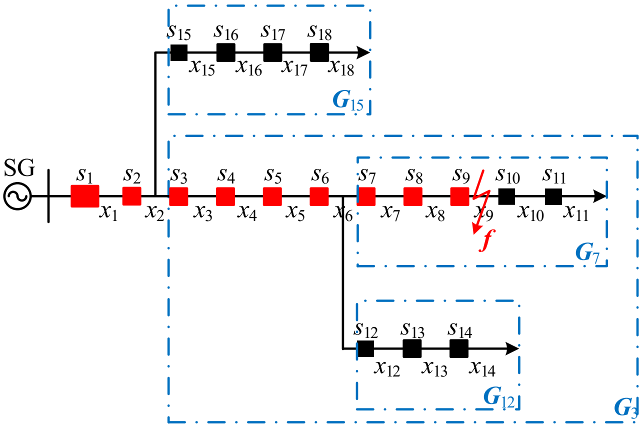


**Figure 6.** Single fault in the multi-branch distribution network.

The algorithm traverses the sections in the network from the starting section $x_1$, calculates the approximation gain of the sections, and determines the section's type. When accessing section $x_2$, $x_2$ is detected as a T-type section, and the network is divided into subnetwork $\boldsymbol{G}_3$ and subnetwork $\boldsymbol{G}_{15}$. The algorithm is called recursively for $\boldsymbol{G}_{15}$, and when the traversal end $k = 18$ $\boldsymbol{G}_{15}$ is determined to be the base case that does not contain a T-type section. The section with the maximum approximation gain in $x_{15}$~$x_{18}$ is $x_{15}$. According to Theorem 8, $\boldsymbol{X}_{15} = \{x_{15}\}$ and $\Delta_{\boldsymbol{X}_{15}} = \Delta_{15} = 1$ are taken as the fault location results of $\boldsymbol{G}_{15}$. Call the algorithm recursively for $\boldsymbol{G}_3$, traverse the sections in $\boldsymbol{G}_3$ from $x_3$, detect that $x_6$ is a T-type section, and divide the network into $\boldsymbol{G}_7$ and $\boldsymbol{G}_{12}$. Call the algorithm recursively for $\boldsymbol{G}_7$, and when the traversal end $k = 11$, determine $\boldsymbol{G}_7$ as the base case and return the result of fault location: $\boldsymbol{X}_7 = \{x_9\}$, $\Delta_{\boldsymbol{X}_7} = \Delta_9 = 9$. Similarly, $\boldsymbol{X}_{12} = \{x_{12}\}$ and $\Delta_{\boldsymbol{X}_{12}} = \Delta_{12} = 5$ are taken as the fault location results of $\boldsymbol{G}_{12}$. The divide process of the algorithm and the calculation results of each section approximation gain are shown in Figure 7a.

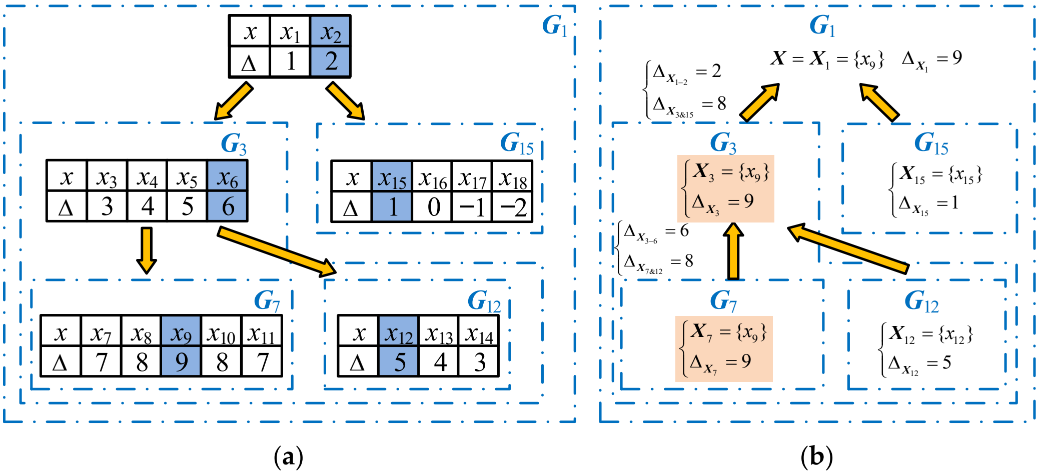

$G_1$
$x$ | $x_1$ | $x_2$
Δ | 1 | 2
$G_3$
$x$ | $x_3$ | $x_4$ | $x_5$ | $x_6$
Δ | 3 | 4 | 5 | 6
$G_{15}$
$x$ | $x_{15}$ | $x_{16}$ | $x_{17}$ | $x_{18}$
Δ | 1 | 0 | −1 | −2
$G_7$
$x$ | $x_7$ | $x_8$ | $x_9$ | $x_{10}$ | $x_{11}$
Δ | 7 | 8 | 9 | 8 | 7
$G_{12}$
$x$ | $x_{12}$ | $x_{13}$ | $x_{14}$
Δ | 5 | 4 | 3
(a)
$G_1$
$\boldsymbol{X} = \boldsymbol{X}_1 = \{x_9\}$ $\Delta_{\boldsymbol{X}_1} = 9$
$\Delta_{\boldsymbol{X}_{1\text{-}2}} = 2$
$\Delta_{\boldsymbol{X}_{3\&15}} = 8$
$G_3$
$\boldsymbol{X}_3 = \{x_9\}$
$\Delta_{\boldsymbol{X}_3} = 9$
$G_{15}$
$\boldsymbol{X}_{15} = \{x_{15}\}$
$\Delta_{\boldsymbol{X}_{15}} = 1$
$\Delta_{\boldsymbol{X}_{3\text{-}6}} = 6$
$\Delta_{\boldsymbol{X}_{7\&12}} = 8$
$G_7$
$\boldsymbol{X}_7 = \{x_9\}$
$\Delta_{\boldsymbol{X}_7} = 9$
$G_{12}$
$\boldsymbol{X}_{12} = \{x_{12}\}$
$\Delta_{\boldsymbol{X}_{12}} = 5$
(b)

**Figure 7.** The execution process of the algorithm under a single fault. (**a**) Divide process; (**b**) combine process.

The combine process of the algorithm is shown in Figure 7b, and the detailed running procedure is as follows. Combine the fault candidate sets $\boldsymbol{G}_3$ and $\boldsymbol{G}_{12}$ to create the fault location result $\boldsymbol{G}_3$. Find the section $x_6$ with the largest approximation gain from $x_3$ to $x_6$ and construct the set of sections $\boldsymbol{X}_{3-6} = \{x_6\}$ with the approximation gain $\Delta_{\boldsymbol{X}_{3-6}} = \Delta_6 = 6$. Construct the set $\boldsymbol{X}_{7\&12} = \boldsymbol{X}_7 \cup \boldsymbol{X}_{12} = \{x_9, x_{12}\}$, whose approximation gain is $\Delta_{\boldsymbol{X}_{7\&12}} = 8$. The largest approximation gain of $\Delta_{\boldsymbol{X}_{3-6}}$, $\Delta_{\boldsymbol{X}_7}$, $\Delta_{\boldsymbol{X}_{12}}$, and $\Delta_{\boldsymbol{X}_{7\&12}}$ is $\Delta_{\boldsymbol{X}_7}$; thus, $\boldsymbol{X}_7$ and $\Delta_{\boldsymbol{X}_7}$ are taken as the fault location results of $\boldsymbol{G}_3$, i.e., $\boldsymbol{X}_3 = \boldsymbol{X}_7 = \{x_9\}$, $\Delta_{\boldsymbol{X}_3} = \Delta_{\boldsymbol{X}_7} = 9$. Combine the fault candidate sets of $\boldsymbol{G}_3$ and $\boldsymbol{G}_{15}$ to create the fault location result $\boldsymbol{G}_1$. Find the section $x_2$ with the largest approximation gain from $x_1$ to $x_2$ and construct the set of sections $\boldsymbol{X}_{1-2} = \{x_2\}$ with the approximation gain $\Delta_{\boldsymbol{X}_{1-2}} = \Delta_2 = 2$. Construct the set $\boldsymbol{X}_{3\&15} = \boldsymbol{X}_3 \cup \boldsymbol{X}_{15} = \{x_9, x_{15}\}$, whose approximation gain is $\Delta_{\boldsymbol{X}_{3\&15}} = 8$. The largest approximation gain of $\Delta_{\boldsymbol{X}_{1-2}}$, $\Delta_{\boldsymbol{X}_3}$, $\Delta_{\boldsymbol{X}_{15}}$, and $\Delta_{\boldsymbol{X}_{3\&15}}$ is $\Delta_{\boldsymbol{X}_3}$; thus, $\boldsymbol{X}_3$ and $\Delta_{\boldsymbol{X}_3}$ are taken as the fault location results of $\boldsymbol{G}_1$, i.e., $\boldsymbol{X}_1 = \boldsymbol{X}_3 = \{x_9\}$, $\Delta_{\boldsymbol{X}_1} = \Delta_{\boldsymbol{X}_3} = 9$. The algorithm is executed, and $x_9$ is determined to be the fault section.

#### 4.4.2. Test Case 2: Multiple Faults

In Figure 8, assume that both sections $x_9$ and $x_{18}$ have a fault simultaneously. The nodes that reported the fault information are marked with red boxes.

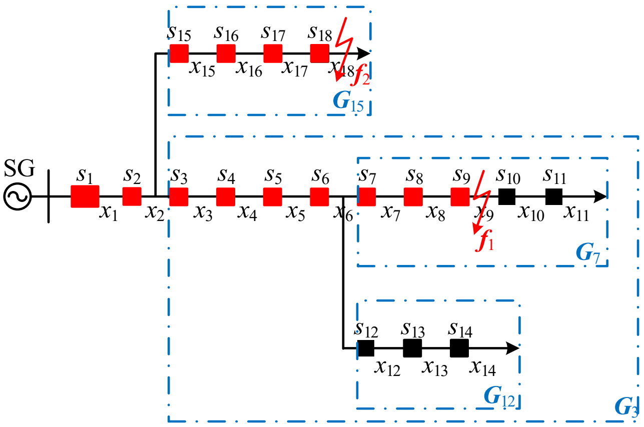


**Figure 8.** Multiple faults in the multi-branch distribution network.

The divide process of the algorithm is similar to that of Test Case 1. The divide process of the algorithm and the calculation results of each section approximation gain are shown in Figure 9a.

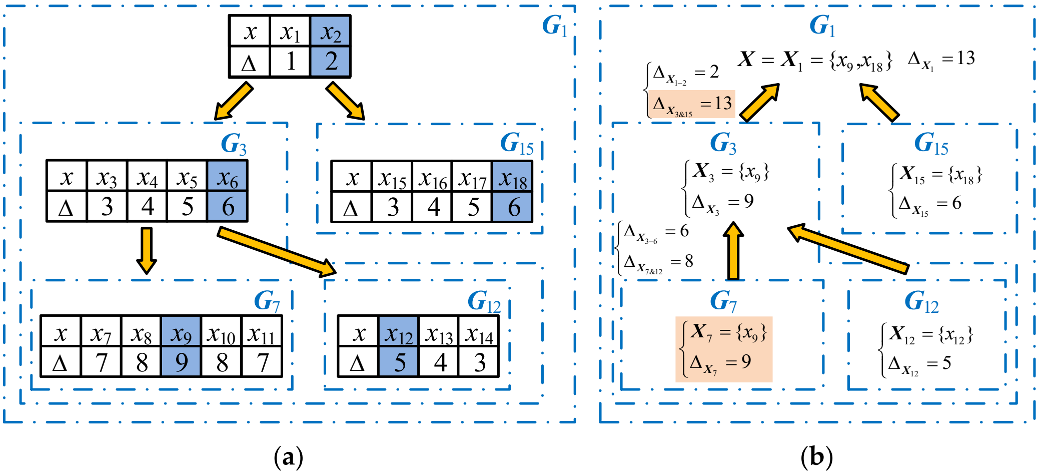


**Figure 9.** Execution process of the algorithm under multiple failures. (**a**) Divide process; (**b**) combine process.

The combine process of the algorithm is shown in Figure 9b, and the detailed running procedure is as follows. Combine the fault candidate sets $\boldsymbol{G}_7$ and $\boldsymbol{G}_{12}$ to create the fault location result $\boldsymbol{G}_3$. Find the section $x_6$ with the largest approximation gain from $x_3$ to $x_6$ and construct the set of sections $\boldsymbol{X}_{3-6} = \{x_6\}$ with the approximation gain $\Delta_{\boldsymbol{X}_{3-6}} = \Delta_6 = 6$. Construct the set $\boldsymbol{X}_{7\&12} = \boldsymbol{X}_7 \cup \boldsymbol{X}_{12} = \{x_9, x_{12}\}$, whose approximation gain is $\Delta_{\boldsymbol{X}_{7\&12}} = 8$. The largest approximation gain of $\Delta_{\boldsymbol{X}_{3-6}}$, $\Delta_{\boldsymbol{X}_7}$, $\Delta_{\boldsymbol{X}_{12}}$, and $\Delta_{\boldsymbol{X}_{7\&12}}$ is $\Delta_{\boldsymbol{X}_7}$; thus, $\boldsymbol{X}_7$ and $\Delta_{\boldsymbol{X}_7}$ are taken as the fault location results of $\boldsymbol{G}_3$, i.e., $\boldsymbol{X}_3 = \boldsymbol{X}_7 = \{x_9\}$, $\Delta_{\boldsymbol{X}_3} = \Delta_{\boldsymbol{X}_7} = 9$. Combine the fault candidate sets $\boldsymbol{G}_3$ and $\boldsymbol{G}_{15}$ to create the fault location result $\boldsymbol{G}_1$. Find section $x_2$ with the largest approximation gain from $x_1$ to $x_2$ and construct the set of sections $\boldsymbol{X}_{1-2} = \{x_2\}$ with the approximation gain $\Delta_{\boldsymbol{X}_{1-2}} = \Delta_2 = 2$. Construct the set $\boldsymbol{X}_{3\&15} = \boldsymbol{X}_3 \cup \boldsymbol{X}_{15} = \{x_9, x_{15}\}$, whose approximation gain is $\Delta_{\boldsymbol{X}_{3\&15}} = 13$. The largest approximation gain of $\Delta_{\boldsymbol{X}_{1-2}}$, $\Delta_{\boldsymbol{X}_3}$, $\Delta_{\boldsymbol{X}_{15}}$, and $\Delta_{\boldsymbol{X}_{3\&15}}$ is $\Delta_{\boldsymbol{X}_{3\&15}}$; thus, $\boldsymbol{X}_{3\&15}$ and $\Delta_{\boldsymbol{X}_{3\&15}}$ are taken as the fault location results of $\boldsymbol{G}_1$, i.e., $\boldsymbol{X}_1 = \boldsymbol{X}_{3\&15} = \{x_9, x_{18}\}$, $\Delta_{\boldsymbol{X}_1} = \Delta_{\boldsymbol{X}_{3\&15}} = 13$. After the algorithm is executed, $x_9$ and $x_{18}$ are determined to be the fault sections.

### 4.4.3. Test Case 3: Fault Information Distortion

In actual projects, FTUs are mostly installed outdoors, where the working environments are harsh and the reported fault information may be misreported, missed, or experiencing other distortions. Therefore, it should be possible to output correct positioning results as much as possible in the case of small-scale distortion of information. This case sets up the small-scale distortion of information to test the information fault tolerance of the algorithm.

In Figure 10, assume that section $x_{18}$ has a fault. Node $s_{10}$ misreports the fault information (i.e., $I_{10}$: 0 → 1), and node $s_2$ misses the fault information (i.e., $I_2$: 1 → 0). The nodes that reported the fault information are marked with red boxes.

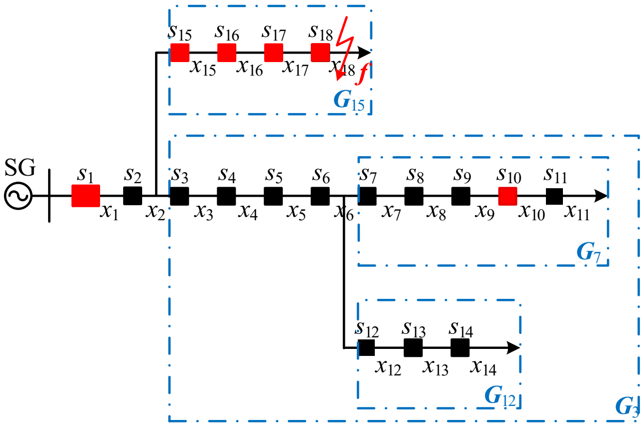
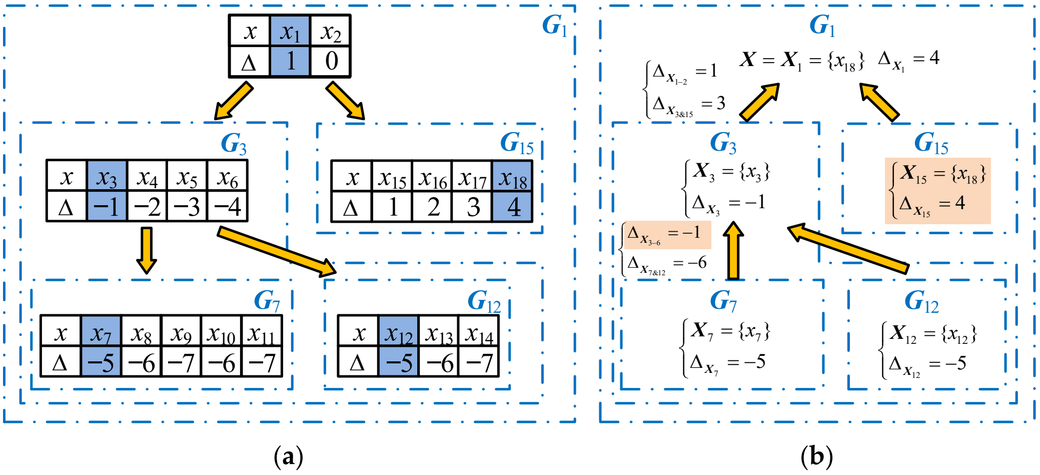

$s_{15}$ $s_{16}$ $s_{17}$ $s_{18}$
$x_{15}$ $x_{16}$ $x_{17}$ $x_{18}$
$f$
$G_{15}$
SG
$s_1$ $s_2$ $s_3$ $s_4$ $s_5$ $s_6$ $s_7$ $s_8$ $s_9$ $s_{10}$ $s_{11}$
$x_1$ $x_2$ $x_3$ $x_4$ $x_5$ $x_6$ $x_7$ $x_8$ $x_9$ $x_{10}$ $x_{11}$
$G_7$
$s_{12}$ $s_{13}$ $s_{14}$
$x_{12}$ $x_{13}$ $x_{14}$
$G_{12}$
$G_3$

**Figure 10.** Fault information distortion in the multi-branch distribution network.

The divide process of the algorithm is similar to Test Case 1. The divide process of the algorithm and the calculation results of each section approximation gain are shown in Figure 11a.



| $x$ | $x_1$ | $x_2$ |
|---|---|---|
| $\Delta$ | 1 | 0 |

| $x$ | $x_3$ | $x_4$ | $x_5$ | $x_6$ |
|---|---|---|---|---|
| $\Delta$ | −1 | −2 | −3 | −4 |

| $x$ | $x_{15}$ | $x_{16}$ | $x_{17}$ | $x_{18}$ |
|---|---|---|---|---|
| $\Delta$ | 1 | 2 | 3 | 4 |

| $x$ | $x_7$ | $x_8$ | $x_9$ | $x_{10}$ | $x_{11}$ |
|---|---|---|---|---|---|
| $\Delta$ | −5 | −6 | −7 | −6 | −7 |

| $x$ | $x_{12}$ | $x_{13}$ | $x_{14}$ |
|---|---|---|---|
| $\Delta$ | −5 | −6 | −7 |

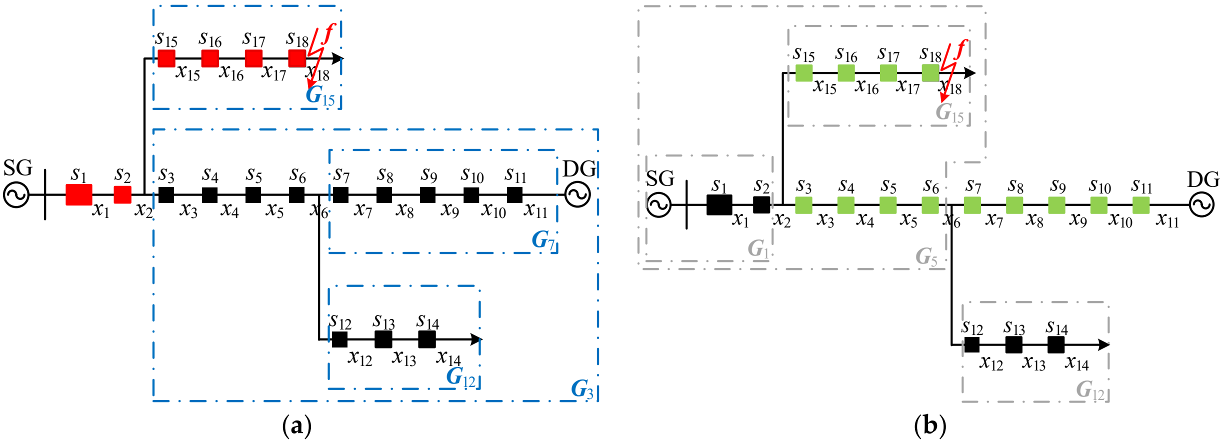
**Figure 11.** Execution process of the algorithm under fault information distortion. (**a**) Divide process; (**b**) combine process.

The combine process of the algorithm is shown in Figure 11b, and the detailed running procedure is as follows. Combine the fault candidate sets $\boldsymbol{G}_7$ and $\boldsymbol{G}_{12}$ to create the fault location result $\boldsymbol{G}_3$. Find the section $x_3$ with the largest approximation gain from $x_3$ to $x_6$ and construct the set of sections $\boldsymbol{X}_{3-6} = \{x_3\}$ with the approximation gain $\Delta_{\boldsymbol{X}_{3-6}} = \Delta_3 = -1$. Construct the set $\boldsymbol{X}_{7\&12} = \boldsymbol{X}_7 \cup \boldsymbol{X}_{12} = \{x_7, x_{12}\}$, whose approximation gain is $\Delta_{\boldsymbol{X}_{7\&12}} = -6$. The largest approximation gain of $\Delta_{\boldsymbol{X}_{3-6}}$, $\Delta_{\boldsymbol{X}_7}$, $\Delta_{\boldsymbol{X}_{12}}$, and $\Delta_{\boldsymbol{X}_{7\&12}}$ is $\Delta_{\boldsymbol{X}_{3-6}}$; thus, $\boldsymbol{X}_{3-6}$ and $\Delta_{\boldsymbol{X}_{3-6}}$ are taken as the fault location results of $\boldsymbol{G}_3$, i.e., $\boldsymbol{X}_3 = \boldsymbol{X}_{3-6} = \{x_3\}$, $\Delta_{\boldsymbol{X}_3} = \Delta_{\boldsymbol{X}_{3-6}} = -1$. Combine the fault candidate sets $\boldsymbol{G}_3$ and $\boldsymbol{G}_{15}$ to create the fault location result $\boldsymbol{G}_1$. Find the section $x_1$ with the largest approximation gain from $x_1$ to $x_2$ and construct the set of sections $\boldsymbol{X}_{1-2} = \{x_1\}$ with the approximation gain $\Delta_{\boldsymbol{X}_{1-2}} = \Delta_1 = 1$. Construct the set $\boldsymbol{X}_{3\&15} = \boldsymbol{X}_3 \cup \boldsymbol{X}_{15} = \{x_9, x_{15}\}$, whose approximation gain is $\Delta_{\boldsymbol{X}_{3\&15}} = 3$. The largest approximation gain of $\Delta_{\boldsymbol{X}_{1-2}}$, $\Delta_{\boldsymbol{X}_3}$, $\Delta_{\boldsymbol{X}_{15}}$, and $\Delta_{\boldsymbol{X}_{3\&15}}$ is $\Delta_{\boldsymbol{X}_{15}}$; thus, $\boldsymbol{X}_{15}$ and $\Delta_{\boldsymbol{X}_{15}}$ are taken as the fault location results of $\boldsymbol{G}_1$, i.e., $\boldsymbol{X}_1 = \boldsymbol{X}_{15} = \{x_{18}\}$, $\Delta_{\boldsymbol{X}_1} = \Delta_{\boldsymbol{X}_{15}} = 4$. After the algorithm is executed, $x_{18}$ is determined to be the fault section.

As can be seen from Test Case 3, the algorithm proposed in this paper uses the redundancy of fault information in the whole network to tolerate the distorted information, which can accurately locate the fault with small-scale distortion of fault information.

### 4.4.4. Test Case 4: Active Distribution Network

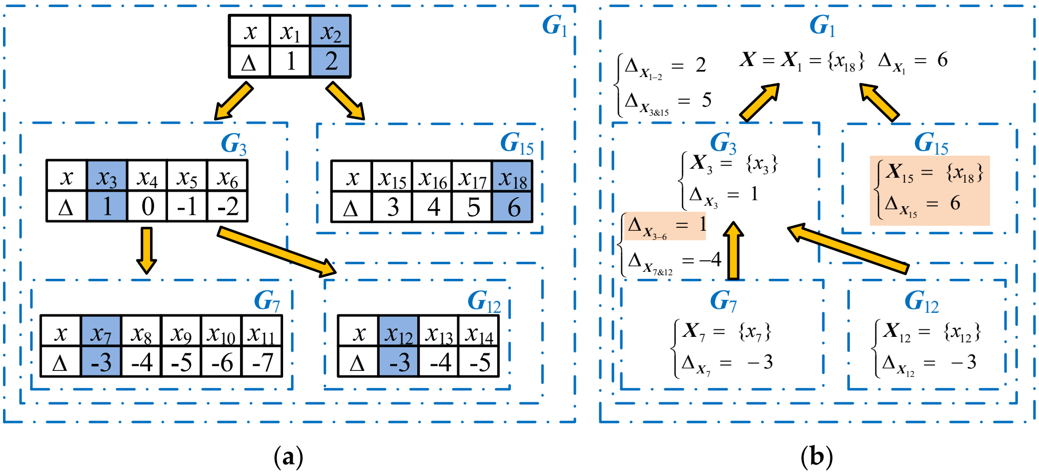
In Figure 12, assume that section $x_{18}$ in the active distribution network has a fault. SG and DG are the main power sources, respectively, and the corresponding nodes' fault information is marked in red and green boxes in Figure 12a and Figure 12b, respectively.



SG
DG
$s_1$ $s_2$ $s_3$ $s_4$ $s_5$ $s_6$ $s_7$ $s_8$ $s_9$ $s_{10}$ $s_{11}$
$x_1$ $x_2$ $x_3$ $x_4$ $x_5$ $x_6$ $x_7$ $x_8$ $x_9$ $x_{10}$ $x_{11}$
$s_{12}$ $s_{13}$ $s_{14}$
$x_{12}$ $x_{13}$ $x_{14}$
$s_{15}$ $s_{16}$ $s_{17}$ $s_{18}$
$x_{15}$ $x_{16}$ $x_{17}$ $x_{18}$
$f$
$G_{15}$
$G_7$
$G_{12}$
$G_3$
$G_1$
$G_5$
(a)
(b)

**Figure 12.** Multiple faults in the multi-branch active distribution network. (**a**) Fault information of SG as the main power source; (**b**) fault information of DG as the main power source.

The call algorithm determines the result of fault location with each power supply as the main power supply, and the algorithm's divide process and combine process are shown in Figure 13.



$G_1$

| $x$ | $x_1$ | $x_2$ |
|---|---|---|
| Δ | 1 | 2 |

$G_3$

| $x$ | $x_3$ | $x_4$ | $x_5$ | $x_6$ |
|---|---|---|---|---|
| Δ | 1 | 0 | -1 | -2 |

$G_{15}$

| $x$ | $x_{15}$ | $x_{16}$ | $x_{17}$ | $x_{18}$ |
|---|---|---|---|---|
| Δ | 3 | 4 | 5 | 6 |

$G_7$

| $x$ | $x_7$ | $x_8$ | $x_9$ | $x_{10}$ | $x_{11}$ |
|---|---|---|---|---|---|
| Δ | -3 | -4 | -5 | -6 | -7 |

$G_{12}$

| $x$ | $x_{12}$ | $x_{13}$ | $x_{14}$ |
|---|---|---|---|
| Δ | -3 | -4 | -5 |

(a)
$G_1$
$\begin{cases}\Delta_{X_{1-2}} = 2\\ \Delta_{X_{3\&15}} = 5\end{cases}$
$\boldsymbol{X} = \boldsymbol{X}_1 = \{x_{18}\}$ $\Delta_{X_1} = 6$
$G_3$
$\begin{cases}\boldsymbol{X}_3 = \{x_3\}\\ \Delta_{X_3} = 1\end{cases}$
$\begin{cases}\Delta_{X_{3-6}} = 1\\ \Delta_{X_{7\&12}} = -4\end{cases}$
$G_{15}$
$\begin{cases}\boldsymbol{X}_{15} = \{x_{18}\}\\ \Delta_{X_{15}} = 6\end{cases}$
$G_7$
$\begin{cases}\boldsymbol{X}_7 = \{x_7\}\\ \Delta_{X_7} = -3\end{cases}$
$G_{12}$
$\begin{cases}\boldsymbol{X}_{12} = \{x_{12}\}\\ \Delta_{X_{12}} = -3\end{cases}$
(b)

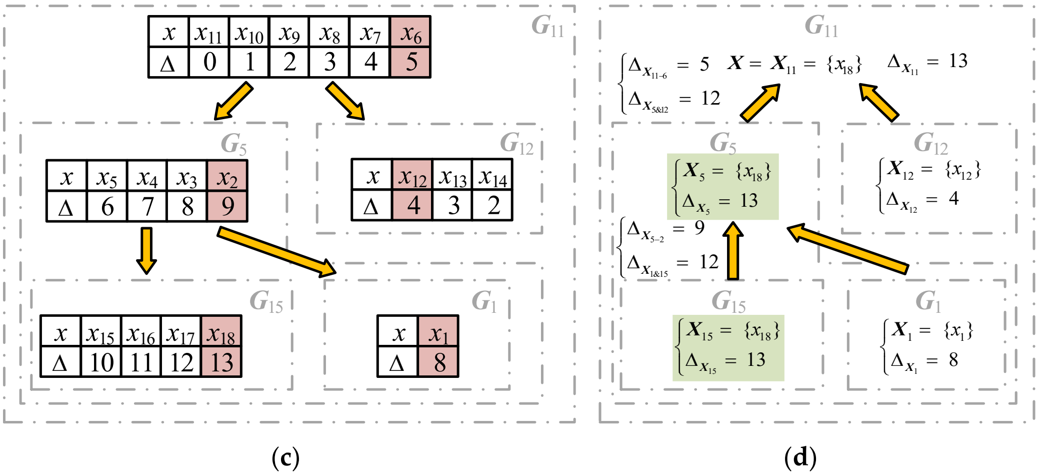


**Figure 13.** The execution process of the algorithm in an active distribution network. (**a**) Divide process with SG as the main power source; (**b**) combine process with SG as the main power source; (**c**) divide process with DG as the main power source; (**d**) combine process with DG as the main power source.

In Figure 13, it can be seen that the result of fault location with SG and DG as the main power source are $\boldsymbol{X}_1 = \{x_{18}\}$, $\boldsymbol{X}_{11} = \{x_{18}\}$, respectively. The final location result is obtained by taking the two together: $\boldsymbol{X} = \boldsymbol{X}_1 \cup \boldsymbol{X}_{11} = \{x_{18}\}$. After the algorithm is executed, $x_{18}$ is determined to be the fault section.

## 5. Simulation Analysis

In order to test the performance of the proposed method, this section describes, taking the IEEE 33-node power distribution system and IEEE 69-node power distribution system as examples, fault location simulation tests performed on the MATLAB simulation environment on an Intel Core i7-1260P CPU 4.70GHz platform, and the results are compared with the current typical fault location algorithms in terms of accuracy, fault tolerance, and computational efficiency. The compared algorithms include the binary particle swarm optimization algorithm (Algorithm 2) [20], bald eagle search algorithm (Algorithm 3) [31], and linear integer programming algorithm (Algorithm 4) [26]. The specific fault location methods for each algorithm are shown in Table 5.

**Table 5.** Fault location methods for various algorithms.

| Algorithm Number | Algorithm Type | Fault Location Method |
|---|---|---|
| Algorithm 1 | Algorithm in this paper | Algorithm in this paper. Construct a numerical computational model for fault location with the help of road vector, and apply DAC to locate the fault. |
| Algorithm 2 | Intelligent optimization algorithm | Assuming all possible fault scenarios in the distribution network, construct a solution space and apply the binary particle swarm optimization algorithm to locate the fault section in the solution space. |
| Algorithm 3 | Hierarchical fault location algorithm | Partition the distribution network according to the branch structure, apply the bald eagle search algorithm to locate the fault branch first, and then determine the fault section by applying the exhaustive method in the fault branch. |
| Algorithm 4 | Linear integer programming algorithm | Construct a linear integer programming model for fault location using ascending dimension and apply the branch and bound method to locate the fault. |

### 5.1. IEEE 33-Node Distribution Network

The IEEE 33-node distribution system is shown in Figure 14, which has 1 substation power supply, 1 circuit breaker, 32 intelligent switches, and 33 sections. The population size of the intelligent optimization algorithm was set to 50, and the maximum number of iterations was set to 30.

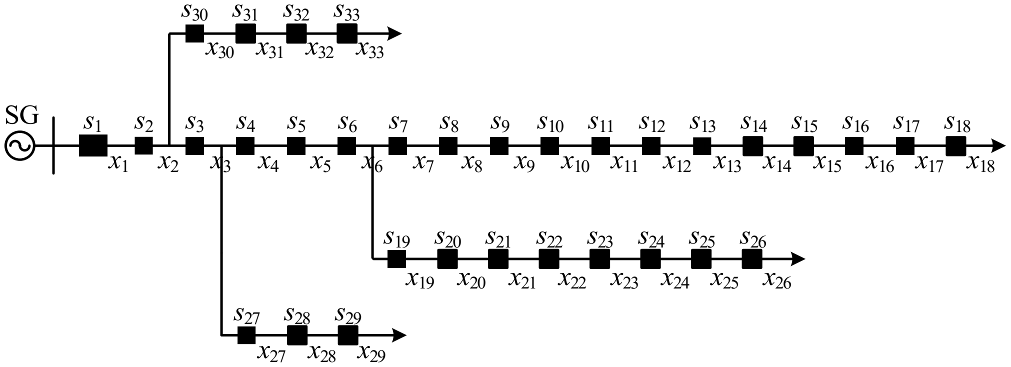

SG
$s_1$ $s_2$ $s_3$ $s_4$ $s_5$ $s_6$ $s_7$ $s_8$ $s_9$ $s_{10}$ $s_{11}$ $s_{12}$ $s_{13}$ $s_{14}$ $s_{15}$ $s_{16}$ $s_{17}$ $s_{18}$
$x_1$ $x_2$ $x_3$ $x_4$ $x_5$ $x_6$ $x_7$ $x_8$ $x_9$ $x_{10}$ $x_{11}$ $x_{12}$ $x_{13}$ $x_{14}$ $x_{15}$ $x_{16}$ $x_{17}$ $x_{18}$
$s_{30}$ $s_{31}$ $s_{32}$ $s_{33}$
$x_{30}$ $x_{31}$ $x_{32}$ $x_{33}$
$s_{19}$ $s_{20}$ $s_{21}$ $s_{22}$ $s_{23}$ $s_{24}$ $s_{25}$ $s_{26}$
$x_{19}$ $x_{20}$ $x_{21}$ $x_{22}$ $x_{23}$ $x_{24}$ $x_{25}$ $x_{26}$
$s_{27}$ $s_{28}$ $s_{29}$
$x_{27}$ $x_{28}$ $x_{29}$

**Figure 14.** IEEE 33-node distribution system.

### 5.1.1. Accuracy

To test the accuracy of the algorithm, it was run 100 times with multiple fault types, such as single faults, multiple faults, and end-of-supply faults. The accuracy of the algorithm is described by the ratio of the number of correct locations to the total number of runs. The test results are shown in Table 6.

**Table 6.** Accuracy test results.

| Equipment Number | Preset Fault Section | Accuracy (%) | | | |
|---|---|---|---|---|---|
| | | Algorithm 1 | Algorithm 2 | Algorithm 3 | Algorithm 4 |
| 1 | $x_8$ | 100 | 94 | 99 | 100 |
| 2 | $x_{26}$ | 100 | 80 | 100 | 100 |
| 3 | $x_{28}$ | 100 | 82 | 100 | 100 |
| 4 | $x_{30}$ | 100 | 78 | 99 | 100 |
| 5 | $x_4$, $x_{29}$ | 100 | 96 | 100 | 100 |
| 6 | $x_{11}$, $x_{28}$ | 100 | 84 | 96 | 100 |
| 7 | $x_{25}$, $x_{30}$ | 100 | 81 | 100 | 100 |
| 8 | $x_6$, $x_{27}$ | 100 | 95 | 100 | 100 |

As can be seen in Table 6, all four algorithms have correct result outputs; the difference lies in their ability to maintain consistency in their results. Algorithm 2 and Algorithm 3 cannot guarantee 100% accuracy. The accuracy of Algorithm 2 is around 85%, which is due to the fact that the algorithm applies an intelligent algorithm with stochastic search characteristics to locate faults, and along with the depth of iterations, the population tends to homogenize, leading to possible local convergence; thus, the consistency of results cannot be guaranteed. Algorithm 3 applies a hierarchical strategy to reduce the fault search dimension, and its accuracy is improved, but it still needs to locate the fault branch with the help of an intelligent algorithm; thus, the defect of insufficient convergence stability cannot be eradicated. With the pseudocode DACFL, the algorithm in this paper does not contain random operations, which makes the algorithm have the same execution process for the same fault situation; thus, it can guarantee consistency in its results. The test results confirm that Algorithm 1 achieved 100% accuracy for the fault cases in Table 6.

5.1.2. Fault Tolerance

In actual engineering, the distortion of fault information reported by FTU is divided into two cases of misreporting or omission of the reported information. Omission of the reported information refers to the fault current exceeding the set threshold and the FTU not reporting the fault information. Misreporting means that the FTU reports fault information, but the actual situation does not match it.

In this section, different numbers and locations of false alarm and omission signals are set to simulate a variety of information distortion situations in real projects to test the information tolerance of the algorithm.

As can be seen in Table 7, the accuracy of Algorithm 2 dropped to about 80%. Algorithm 3 requires high information reliability of the branch nodes and therefore produced misjudgment for fault cases 2 and 7, in which the branch nodes were distorted. The fault location model of the algorithm in this paper was derived using strict mathematical derivation and is fully equivalent to the classical localization model, which is fault-tolerant to distorted information with the help of redundancy of fault information in the whole network. The test results confirm that Algorithm 1 had 100% accuracy for the fault information distortion case in Table 7.

**Table 7.** Fault tolerance test results.

| Equipment Number | Preset Fault Section | Preset Information Distortion | Accuracy (%) | | | |
|---|---|---|---|---|---|---|
| | | | Algorithm 1 | Algorithm 2 | Algorithm 3 | Algorithm 4 |
| 1 | $x_3$ | $I_{16}$: 0 → 1 | 100 | 84 | 100 | 100 |
| 2 | $x_{18}$ | $I_7$: 1 → 0, $I_{32}$: 0 → 1 | 100 | 80 | Misjudgment | 100 |
| 3 | $x_{23}$ | $I_4$: 1 → 0 | 100 | 88 | 100 | 100 |
| 4 | $x_{33}$ | $I_{11}$: 0 → 1, $I_{20}$: 0 → 1 | 100 | 86 | 99 | 100 |
| 5 | $x_7$, $x_{22}$ | $I_{14}$: 1 → 0 | 100 | 90 | 100 | 100 |
| 6 | $x_{10}$, $x_{33}$ | $I_6$: 1 → 0, $I_{30}$: 1 → 0 | 100 | 70 | 98 | 100 |
| 7 | $x_{16}$, $x_{27}$ | $I_7$: 1 → 0, $I_{22}$: 0 → 1, $I_{32}$: 0 → 1 | 100 | 76 | Misjudgment | 100 |
| 8 | $x_{14}$, $x_{26}$ | $I_{10}$: 1 → 0 | 100 | 72 | 100 | 100 |

5.1.3. Computational Efficiency

The algorithms were run 100 times, and the average time consumed by the algorithms was taken as a measure of computational efficiency. The test results are shown in Table 8.

**Table 8.** Computational efficiency test results.

| Equipment Number | Preset Fault Section | Preset Information Distortion | Computational Efficiency (ms) | | | |
|---|---|---|---|---|---|---|
| | | | Algorithm 1 | Algorithm 2 | Algorithm 3 | Algorithm 4 |
| 1 | $x_{18}$ | | 13.27 | 131.43 | 75.27 | 118.18 |
| 2 | $x_{24}$ | $I_9$: 0 → 1, $I_{20}$: 1 → 0 | 13.86 | 128.29 | 79.34 | 116.76 |
| 3 | $x_6$ | $I_{12}$: 0 → 1 | 15.16 | 128.45 | 77.98 | 126.72 |
| 4 | $x_{33}$ | $I_2$: 1 → 0 | 16.85 | 129.23 | 76.25 | 142.61 |
| 5 | $x_{11}$, $x_{20}$ | $I_8$: 1 → 0, $I_{29}$: 0 → 1 | 14.33 | 133.67 | 103.56 | 121.98 |
| 6 | $x_{17}$, $x_{33}$ | $I_{22}$: 0 → 1 | 13.46 | 128.92 | 109.54 | 130.47 |
| 7 | $x_4$, $x_{29}$ | | 15.32 | 133.96 | 108.48 | 152.42 |
| 8 | $x_{12}$, $x_{23}$ | $I_5$: 1 → 0, $I_8$: 1 → 0, $I_{15}$: 0 → 1 | 15.05 | 149.75 | 106.26 | 112.01 |

From Table 8, it can be seen that the computational efficiency of Algorithm 2 was around 130 ms, which is due to the fact that the algorithm needs to perform iterative operations in a high-dimensional solution space. Algorithm 3 applies a hierarchical strategy to reduce the fault search dimension, and its computational efficiency was around 90 ms.

Algorithm 4 had a computational efficiency of about 127 ms. The analysis in Section 4.2 confirms that the algorithm proposed in this paper has a linear level of time complexity. The test results show that the computational efficiency of Algorithm 1 was below 20 ms, which is a significant advantage compared with the other three algorithms.

### *5.2. IEEE 69-Node Distribution Network*

The IEEE 69-node distribution system is shown in Figure 15, which has 1 substation power supply, 4 distributed power supplies, 1 circuit breaker, 4 DG grid-connected switches, 68 intelligent switches, and 69 sections. The population size of the intelligent optimization algorithm was set to 100, and the maximum number of iterations was set to 70.

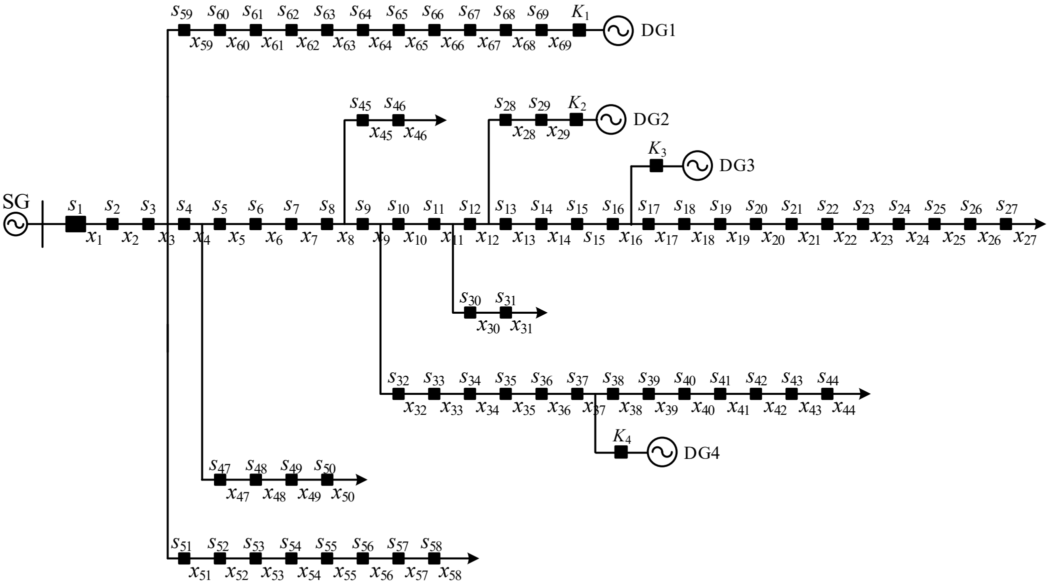

$s_{59}$ $s_{60}$ $s_{61}$ $s_{62}$ $s_{63}$ $s_{64}$ $s_{65}$ $s_{66}$ $s_{67}$ $s_{68}$ $s_{69}$ $K_1$
DG1
$x_{59}$ $x_{60}$ $x_{61}$ $x_{62}$ $x_{63}$ $x_{64}$ $x_{65}$ $x_{66}$ $x_{67}$ $x_{68}$ $x_{69}$
$s_{45}$ $s_{46}$
$x_{45}$ $x_{46}$
$s_{28}$ $s_{29}$ $K_2$
DG2
$x_{28}$ $x_{29}$
$K_3$
DG3
SG
$s_1$ $s_2$ $s_3$ $s_4$ $s_5$ $s_6$ $s_7$ $s_8$ $s_9$ $s_{10}$ $s_{11}$ $s_{12}$ $s_{13}$ $s_{14}$ $s_{15}$ $s_{16}$ $s_{17}$ $s_{18}$ $s_{19}$ $s_{20}$ $s_{21}$ $s_{22}$ $s_{23}$ $s_{24}$ $s_{25}$ $s_{26}$ $s_{27}$
$x_1$ $x_2$ $x_3$ $x_4$ $x_5$ $x_6$ $x_7$ $x_8$ $x_9$ $x_{10}$ $x_{11}$ $x_{12}$ $x_{13}$ $x_{14}$ $x_{15}$ $x_{16}$ $x_{17}$ $x_{18}$ $x_{19}$ $x_{20}$ $x_{21}$ $x_{22}$ $x_{23}$ $x_{24}$ $x_{25}$ $x_{26}$ $x_{27}$
$s_{30}$ $s_{31}$
$x_{30}$ $x_{31}$
$s_{32}$ $s_{33}$ $s_{34}$ $s_{35}$ $s_{36}$ $s_{37}$ $s_{38}$ $s_{39}$ $s_{40}$ $s_{41}$ $s_{42}$ $s_{43}$ $s_{44}$
$x_{32}$ $x_{33}$ $x_{34}$ $x_{35}$ $x_{36}$ $x_{37}$ $x_{38}$ $x_{39}$ $x_{40}$ $x_{41}$ $x_{42}$ $x_{43}$ $x_{44}$
$K_4$
DG4
$s_{47}$ $s_{48}$ $s_{49}$ $s_{50}$
$x_{47}$ $x_{48}$ $x_{49}$ $x_{50}$
$s_{51}$ $s_{52}$ $s_{53}$ $s_{54}$ $s_{55}$ $s_{56}$ $s_{57}$ $s_{58}$
$x_{51}$ $x_{52}$ $x_{53}$ $x_{54}$ $x_{55}$ $x_{56}$ $x_{57}$ $x_{58}$

**Figure 15.** IEEE 69-node distribution system.

To further demonstrate the performance advantages of the proposed method, this section describes taking the IEEE69-node distribution system as an example to test the accuracy and computational efficiency of the algorithms. The test cases in this section are shown in Table 9, which covers various fault scenarios such as single-fault, multiple-fault, sound fault information, and distorted fault information. Among them, the application vectors $K_1$, $K_2$, $K_3$, and $K_4$ represent the dynamic throwing cases of the distributed power supply. When DGi is put into operation, $K_i = 1$; otherwise, $K_i = 0$.

**Table 9.** Test cases for IEEE 69-node distribution network.

| Equipment Number | $[K_1, K_2, K_3, K_4]$ | Preset Fault Section | Preset Information Distortion |
|---|---|---|---|
| 1 | [1, 1, 1, 1] | $x_{52}$ | $I_8$: −1 → 0 |
| 2 | [0, 1, 1, 1] | $x_{15}$ | $I_{49}$: 0 → 1 |
| 3 | [1, 1, 1, 1] | $x_{57}$ | $I_{20}$: 0 → −1, $I_{51}$: 1 → 0 |
| 4 | [1, 0, 1, 1] | $x_{16}$, $x_{32}$ | $I_8$: 1 → 0, $I_{25}$: 0 → 1, $I_{39}$: 0 → −1 |
| 5 | [0, 1, 1, 1] | $x_{64}$ | |
| 6 | [1, 1, 0, 1] | $x_{10}$, $x_{36}$ | $I_{15}$: 0 → −1 |
| 7 | [1, 1, 1, 0] | $x_{12}$, $x_{35}$ | $I_7$: 1 → 0, $I_{32}$: 1 → 0, $I_{41}$: 0 → −1 |
| 8 | [1, 1, 1, 1] | $x_{56}$, $x_{67}$ | $I_{35}$: −1 → 0, $I_{63}$: 1 → −1 |
| 9 | [1, 1, 1, 1] | $x_{14}$, $x_{23}$ | $I_{17}$: 1 → 0, $I_{34}$: −1 → 1, $I_{43}$: 0 → 1 |
| 10 | [1, 1, 1, 1] | $x_9$, $x_{59}$ | $I_{24}$: 0 → 1, $I_{63}$: −1 → 0 |
| 11 | [1, 0, 0, 1] | $x_{27}$, $x_{34}$ | |
| 12 | [1, 0, 1, 1] | $x_7$ | $I_{34}$: −1 → 1, $I_{62}$: −1 → 0 |

### 5.2.1. Accuracy

As the size of the network increases, the solution space of the fault location problem increases, which causes an intelligent algorithm with randomized search operations to be more prone to local convergence. As can be seen in Figure 16a, the accuracy of Algorithm 2 dropped below 50% for some of the test cases. The accuracy of Algorithm 3 was higher than that of Algorithm 2, but it produced misjudgment for fault cases 3, 7, and 9, where the branch nodes were distorted. As mentioned previously, the algorithm in this paper does not contain random operation and makes use of the redundancy of fault information in the whole network to tolerate the distorted information; thus, its accuracy was almost not affected by the network size. The test results confirm that Algorithm 1 maintained an accuracy of 100% for the various fault cases, as seen in Table 9.

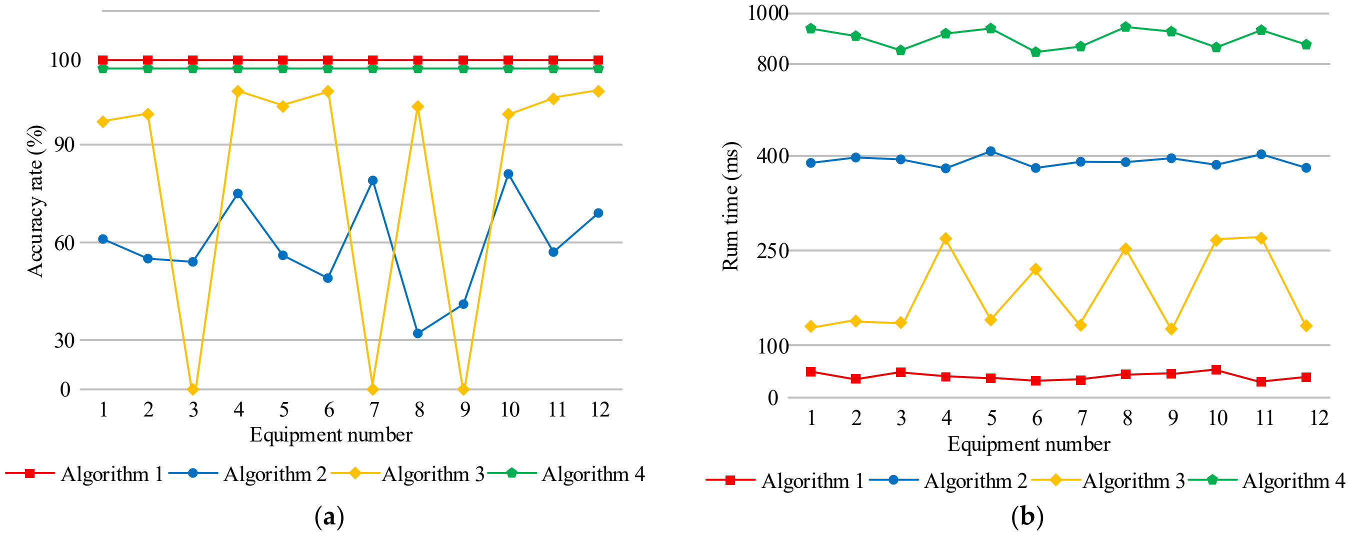


**Figure 16.** The test results of the IEEE 69-node distribution system. (**a**) Accuracy comparison; (**b**) comparison of computational efficiency.

### 5.2.2. Computational Efficiency

From Figure 16b, it can be seen that the computational efficiency of Algorithm 1 was below 50 ms for a 69-node scale distribution network, thanks to its linear time complexity, which indicates that the advantage of the computational efficiency of the algorithm proposed in this paper is more obvious in large distribution networks. For the intelligent optimization algorithm represented by Algorithm 2, the population size of the algorithm is usually no less than the number of sections of the distribution network to ensure the correct rate. Assuming that the distribution network contains $n$ sections, Algorithm 2 needs to operate on at least $n$ individuals in the iterative process, and the cost of individual operation is $\Theta(n)$. Even with the convergence condition of "no update of global extremum" [20], the asymptotic lower bound of $\Omega(n^2)$ is available in the best case. The test results show that the computational efficiency of Algorithm 2 was around 400 ms, which is more than eight times different from the algorithm in this paper. The fault area determination process of Algorithm 3 still requires iterative operations, and its section location requires exhaustive fault scenarios, which affects location speed. The test results show that the computational efficiency of Algorithm 3 was about 200 ms, which is more than four times different from the algorithm in this paper. The solution of Algorithm 4 involves the computation of its continuous relaxation LP model several times, and the time complexity of the LP problem reached $O(n^4)$ [27]. From Figure 16b, we can see that the algorithm had the lowest computational efficiency, which reached the second level and had a large gap compared with the computational efficiency of the algorithm in this paper.

### *5.3. Comparative Analysis of Performance*

In summary, Table 10 gives a comparison of the results of the algorithms' performance.

**Table 10.** Comparative results of algorithm performance.

| Algorithm Number | Algorithm Type | Accuracy | Computational Efficiency |
|---|---|---|---|
| Algorithm 1 | Algorithm in this paper | Highest | Highest |
| Algorithm 2 | Intelligent optimization algorithm | Low | Low |
| Algorithm 3 | Hierarchical fault location algorithm | Higher | Higher |
| Algorithm 4 | Linear integer programming algorithm | Highest | Lowest |

In terms of accuracy, as described in Section 5.1.1, all of the algorithms in the comparison output correct localization results, but the difference lies in their ability to maintain the consistency of their output results during repeated testing. In other words, the accuracy of the algorithms is closely related to the convergence stability. The intelligent optimization algorithm represented by Algorithm 2 has a random population generation and update, and the population form tends to be homogeneous with the depth of iteration; thus, it sometimes converges to the "local extreme value point" in the process of repeated testing, and it cannot maintain consistency in its output results. When the scale of distribution network is expanded, the scale of solution space increases, and these problems become more prominent. The hierarchical fault location algorithm, represented by Algorithm 3, uses a hierarchical strategy to reduce the size of the solution space and improve the convergence stability. However, this algorithm only uses the FTU information of the branch port in determining the fault area, which may lead to misjudgment when the FTU information of the port is distorted. The linear integer programming algorithm, represented by Algorithm 4, applies the classical integer programming algorithm to locate faults, in which the "optimality check" step can make it gradually bound through iteration and eventually converge to the same "global optimal solution"; thus, it has higher convergence stability. From the pseudocode DASFL in Section 4.1, we can see that the algorithm proposed in this paper does not contain random operations and has the same execution process and calculation results in the same fault situation; thus, it can maintain the consistency of the results, which makes the algorithm of this paper have high convergence stability.

The computational efficiency of the algorithm was determined using the time complexity of the algorithm. Assuming that the distribution network contains n sections, as described in Section 5.2.2, Algorithm 4 had a time complexity of $O(n^4)$, and therefore has the lowest computational efficiency. Algorithm 2 requires iterative search operations in a high-dimensional solution space. If it adopts the convergence condition of "no update at the global extremum", it ideally has the asymptotic lower bound of $\Omega(n^2)$, and in the extreme case, the number of iterations of Algorithm 2 is the maximum number of iterations, at which time the algorithm has the asymptotic upper bound of $O(n^3)$. Therefore, its computational efficiency is higher than that of Algorithm 4. Algorithm 3 applies a hierarchical strategy to transform the single-layer, high-dimensional search operation into a double-layer, low-dimensional search operation; thus, its computational efficiency is higher than that of Algorithm 2. As described in Section 4.2, the algorithm in this paper has a linear level of time complexity $\Theta(n)$, and it therefore has the highest computational efficiency.

## 6. Conclusions

The existing distribution network fault section location model contains a large number of logic operations and does not meet the requirements of rapidity and accuracy of fault diagnosis. Therefore, a fault location method based on divide-and-conquer is proposed in this paper. In addition, through a simulated test, the following conclusions can be drawn.

- DAC reduces the operational dimension of fault location to have a linear level of time complexity, which improves the computational efficiency of fault location more significantly. The test results show that for distribution networks with a scale of 33 and 69 nodes, the computational efficiency of the algorithm in this paper is below 20 ms and 50 ms, respectively, which is more than eight times faster than the traditional intelligent optimization algorithms.
- The algorithm in this paper does not contain random search operation; thus, it has strong numerical stability. The test results show that the accuracy of the algorithm in this paper is not affected by the size of the network, and it is more than 19.25% higher than the traditional intelligent optimization algorithm.
- The numerical calculation model of fault location proposed in this paper uses the redundancy of fault information in the whole network to tolerate the distorted information, which can accurately locate the fault with a small amount of distortion in the fault information.

This paper was based on the fault alarm information uploaded by FTU to locate distribution network faults, which has the advantages of clear and simple information link and convenient implementation. However, when the amount of fault information distortion exceeds the acceptable amount of redundancy, misjudgment could occur. Future research could be conducted on the fault criterion of multi-source information fusion to further improve the accuracy of distribution network fault location.

**Author Contributions:** Conceptualization, Z.W.; methodology, Q.Z.; software, Q.Z.; validation, Q.Z.; formal analysis, X.L.; investigation, G.L.; resources, Z.W.; data curation, Y.W.; writing—original draft preparation, Q.Z.; writing—review and editing, G.L.; visualization, X.L.; supervision, Z.W.; project administration, Z.W.; funding acquisition, Z.W. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research received no external funding.

**Conflicts of Interest:** The authors declare no conflicts of interest.

## Appendix A. The Proofs of the Properties of Road Vectors

**Proof of Theorem 1.** (1) ⇒ (2): According to $\boldsymbol{P}_j(i) = 1$, node $s_i$ is on the road of section $x_j$. According to the numbering rule and positive direction of this paper, node $s_i$ is the originating point of section $x_i$, that is, the connection relationships between node $s_i$, section $x_i$ and section $x_j$ are only possible as shown in Figure A1. In Figure A1, section $x_i$ is on the road of section $x_j$.

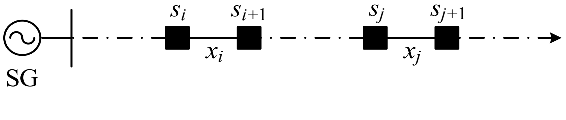


**Figure A1.** Connection relationships between nodes and sections.

(2) ⇒ (3): According to Equation (7), the road vector is a 0–1 vector. According to the description of the operation " $*$ " in this paper, the result of the " $*$ " operation on two 0–1 vectors is that the elements of which both are 1 are taken as 1, and the other elements are taken as 0. Analyze the values of elements in $\boldsymbol{P}_i$ and $\boldsymbol{P}_j$ : assuming that section $x_i$

reaches the main power source through the set of nodes {$s_{i1}$, $s_{i2}$,…}, then the corresponding position of {$s_{i1}$, $s_{i2}$,…} in $\boldsymbol{P}_i$ is taken as 1. Because section $x_i$ is on the road of section $x_j$, assuming that section $x_j$ reaches section $x_i$ through the set of nodes {$s_{j1}$, $s_{j2}$,…}, then the corresponding position of {$s_{i1}$, $s_{i2}$,…} and {$s_{j1}$, $s_{j2}$,…} in $\boldsymbol{P}_j$ is taken as 1. The position where the elements of $\boldsymbol{P}_i$ and $\boldsymbol{P}_j$ are the same is 1, which is the same as that of $\boldsymbol{P}_i$ , and so $\boldsymbol{P}_i = \boldsymbol{P}_i * \boldsymbol{P}_j$ .

(3) ⇒ (1): Suppose for the sake of contradiction that $\boldsymbol{P}_i = \boldsymbol{P}_i * \boldsymbol{P}_j$ but $\boldsymbol{P}_j(i) = 0$ . Then, the ith element of $\boldsymbol{P}_i * \boldsymbol{P}_j$ takes the value 0. Because node $s_i$ is the originating point of section $s_i$, node $s_i$ is on the road of section $x_i$., i.e., $\boldsymbol{P}_i(i) = 1$ , so $\boldsymbol{P}_i \neq \boldsymbol{P}_i * \boldsymbol{P}_j$ , contradicting the assumption $\boldsymbol{P}_i = \boldsymbol{P}_i * \boldsymbol{P}_j$ . □

**Proof of Theorem 2.** Suppose for the sake of contradiction that neither Equation (12) nor (13) holds. Let $\boldsymbol{P}(a) = \boldsymbol{P}_i * \boldsymbol{P}_j * (\boldsymbol{e} - \boldsymbol{P}_r)(a) = 1$; thus, node $s_a$ is then on the road of $x_i$ and $x_j$ but not on the road of $x_r$. Let $\boldsymbol{P}'(b) = \boldsymbol{P}_i * \boldsymbol{P}_r * (\boldsymbol{e} - \boldsymbol{P}_j)(b) = 1$; thus, node $s_b$ is then on the road of $x_i$ and $x_r$ but not on the road of $x_j$. Therefore, the only possible connections of node $s_a$, node $s_b$, section $x_j$, and section $x_j$ are those shown in Figure A2. In Figure A2, it can be seen that there is no path between nodes $s_a$ and $s_b$; thus, it is impossible to be on the road of section $x_i$ at the same time, contradicting the assumption that $s_a$ and $s_b$ are both on the road of $x_i$. □

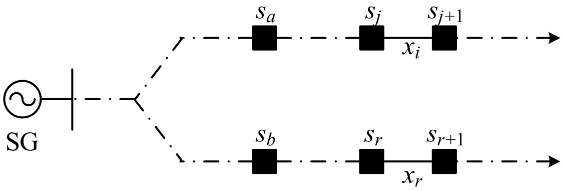


**Figure A2.** Position relationship between node $s_a$ and node $s_b$.

**Proof of Theorem 3.** Let the set $\boldsymbol{X}$ contain $p$ elements. When $p$ = 1, $\boldsymbol{P}_i * (\boldsymbol{e} - \boldsymbol{P}_{k1}) = \boldsymbol{0}$ ; thus, $\boldsymbol{P}_i = \boldsymbol{P}_i * \boldsymbol{P}_{k1}$ . Because $x_{k1} \in \boldsymbol{X}$ , the conclusion holds. When $p$ > 1, by Theorem 2, $\forall x_j, x_r \in \boldsymbol{X}$ , $\boldsymbol{P}_i * \boldsymbol{P}_j * (\boldsymbol{e} - \boldsymbol{P}_r) = \boldsymbol{0}$ or $\boldsymbol{P}_i * \boldsymbol{P}_r * (\boldsymbol{e} - \boldsymbol{P}_j) = \boldsymbol{0}$ . Apply the data structure “Stack” to $\boldsymbol{X}$, $\boldsymbol{X}$ = {$x_{k1}$, $x_{k2}$, …, $x_k$, $x_{k+1}$, …} and rearrange the order of sections in $\boldsymbol{X}$ so that for $\forall x_k, x_{k+1} \in \boldsymbol{X}$ , there is $\boldsymbol{P}_i * \boldsymbol{P}_k * (\boldsymbol{e} - \boldsymbol{P}_{k+1}) = \boldsymbol{0}$ . Thus,

$$\begin{aligned} \boldsymbol{P}_i * \prod_{x_k \in \boldsymbol{X}}^{*} (\boldsymbol{e} - \boldsymbol{P}_k) &= \boldsymbol{P}_i * (\boldsymbol{e} - \boldsymbol{P}_{kp}) * \prod_{x_k \in \boldsymbol{X} - x_{kp}}^{*} (\boldsymbol{e} - \boldsymbol{P}_k) \\ &= \boldsymbol{P}_i * \prod_{x_k \in \boldsymbol{X} - x_{kp}}^{*} (\boldsymbol{e} - \boldsymbol{P}_k) - \boldsymbol{P}_i * \boldsymbol{P}_{kp} * \prod_{x_k \in \boldsymbol{X} - x_{kp}}^{*} (\boldsymbol{e} - \boldsymbol{P}_k) \\ &= \boldsymbol{P}_i * \prod_{x_k \in \boldsymbol{X} - x_{kp}}^{*} (\boldsymbol{e} - \boldsymbol{P}_k) - \boldsymbol{P}_i * \boldsymbol{P}_{kp} * (\boldsymbol{e} - \boldsymbol{P}_{kp-1}) * \prod_{x_k \in \boldsymbol{X} - x_{kp} - x_{kp-1}}^{*} (\boldsymbol{e} - \boldsymbol{P}_k) \end{aligned} \tag{A1}$$

From the order of sections in $\boldsymbol{X}$, we know that $\boldsymbol{P}_i * \boldsymbol{P}_{kp} * (\boldsymbol{e} - \boldsymbol{P}_{kp-1}) = \boldsymbol{0}$; thus, the latter term in Equation (A1) is equal to $\boldsymbol{0}$. Thus, $\boldsymbol{P}_i * \prod_{x_k \in \boldsymbol{X}}^{*} (\boldsymbol{e} - \boldsymbol{P}_k) = \boldsymbol{P}_i * \prod_{x_k \in \boldsymbol{X} - x_{kp}}^{*} (\boldsymbol{e} - \boldsymbol{P}_k)$. This argument shows the process of removing the last element from $\boldsymbol{X}$, etc., taking the elements in $\boldsymbol{X}$ in reverse order, and substituting the known conditions:

$$\boldsymbol{P}_i * \prod_{x_k \in \boldsymbol{X}}^{*} \left(\boldsymbol{e} - \boldsymbol{P}_k\right) = \boldsymbol{P}_i * \left(\boldsymbol{e} - \boldsymbol{P}_{k1}\right) = \boldsymbol{0} \tag{A2}$$

where $\boldsymbol{P}_{k1}$ denotes the road vector of the first element $x_{k1}$ in *X*.

According to Equation (A2), $\boldsymbol{P}_i = \boldsymbol{P}_i * \boldsymbol{P}_{k1}$. Because $x_{k1} \in \boldsymbol{X}$, the conclusion holds when *p* > 1. □

**Appendix B. The Proofs of the Properties of Compatible Sets**

**Proof of Theorem 4.**

(1) Sufficiency. $\forall x_i, x_k \in \boldsymbol{X}_a$, there are obviously $x_i, x_k \in \boldsymbol{X}_a \cup \boldsymbol{X}_b$. Because $\boldsymbol{X}_a \cup \boldsymbol{X}_b$ satisfies the inequality constraint, $\boldsymbol{P}_i \neq \boldsymbol{P}_i * \boldsymbol{P}_k$, i.e., $\boldsymbol{X}_a$ satisfies the inequality constraint. Similarly, $\boldsymbol{X}_b$ satisfies the inequality constraint. Then, $\forall x_i \in \boldsymbol{X}_a$, $\forall x_k \in \boldsymbol{X}_b$, and there are $x_i, x_k \in \boldsymbol{X}_a \cup \boldsymbol{X}_b$. Because $\boldsymbol{X}_a \cup \boldsymbol{X}_b$ satisfies the inequality constraint, $\boldsymbol{P}_i \neq \boldsymbol{P}_i * \boldsymbol{P}_k$ and $\boldsymbol{P}_k \neq \boldsymbol{P}_i * \boldsymbol{P}_k$. According to the compatibility set definition, $x_k \in \boldsymbol{C}_i$. Because $x_k$ is any element of $\boldsymbol{X}_b$, $\boldsymbol{X}_b \subseteq \boldsymbol{C}_i$.

(2) Necessity. $\forall x_i, x_k \in \boldsymbol{X}_a \cup \boldsymbol{X}_b$, and if $x_i, x_k \in \boldsymbol{X}_a$ or $x_i, x_k \in \boldsymbol{X}_b$, $\boldsymbol{P}_i \neq \boldsymbol{P}_i * \boldsymbol{P}_k$ due to $\boldsymbol{X}_a, \boldsymbol{X}_b$ satisfying the inequality constraint. If $x_i \in \boldsymbol{X}_a$, $x_k \in \boldsymbol{X}_b$, due to $x_k \in \boldsymbol{X}_b \subseteq \boldsymbol{C}_i$, according to the definition of the following compatible set, $\boldsymbol{P}_i \neq \boldsymbol{P}_i * \boldsymbol{P}_k$. In summary, $\forall x_i, x_k \in \boldsymbol{X}_a \cup \boldsymbol{X}_b$, and we have $\boldsymbol{P}_i \neq \boldsymbol{P}_i * \boldsymbol{P}_k$. Thus, $\boldsymbol{X}_a \cup \boldsymbol{X}_b$ satisfies the inequality constraint. □

**Proof of Theorem 5.** Transform the problem: $\forall x_r \in \boldsymbol{C}_i$, prove $x_r \in \boldsymbol{C}_k$, i.e., $\boldsymbol{P}_k \neq \boldsymbol{P}_k * \boldsymbol{P}_r$ and $\boldsymbol{P}_r \neq \boldsymbol{P}_k * \boldsymbol{P}_r$.

(1) Suppose for the sake of contradiction that $x_r \in \boldsymbol{C}_i$ but $\boldsymbol{P}_k = \boldsymbol{P}_k * \boldsymbol{P}_r$. It is known that section $x_i$ is on the road of section $x_k$, and according to Theorem 1, $\boldsymbol{P}_i = \boldsymbol{P}_i * \boldsymbol{P}_k$, substituting the assumption $\boldsymbol{P}_i = \boldsymbol{P}_i * \boldsymbol{P}_k = \boldsymbol{P}_i * (\boldsymbol{P}_k * \boldsymbol{P}_r) = (\boldsymbol{P}_i * \boldsymbol{P}_k) * \boldsymbol{P}_r = \boldsymbol{P}_i * \boldsymbol{P}_r$ contradicts the assumption $x_r \in \boldsymbol{C}_i$.

(2) Suppose for the sake of contradiction that $x_r \in \boldsymbol{C}_i$ but $\boldsymbol{P}_r = \boldsymbol{P}_k * \boldsymbol{P}_r$. According to Theorem 2, $\boldsymbol{P}_k * \boldsymbol{P}_r * (\boldsymbol{e} - \boldsymbol{P}_i) = \boldsymbol{0}$ or $\boldsymbol{P}_k * \boldsymbol{P}_i * (\boldsymbol{e} - \boldsymbol{P}_r) = \boldsymbol{0}$. If the former equation holds, then $\boldsymbol{P}_k * \boldsymbol{P}_r = \boldsymbol{P}_k * \boldsymbol{P}_r * \boldsymbol{P}_i$, substituting the assumptions $\boldsymbol{P}_r = \boldsymbol{P}_k * \boldsymbol{P}_r = \boldsymbol{P}_k * \boldsymbol{P}_r * \boldsymbol{P}_i = \boldsymbol{P}_r * \left(\boldsymbol{P}_k * \boldsymbol{P}_i\right) = \boldsymbol{P}_r * \boldsymbol{P}_i$, and contradicting the assumption $x_r \in \boldsymbol{C}_i$. If the latter equation holds, then $\boldsymbol{P}_k * \boldsymbol{P}_i = \boldsymbol{P}_k * \boldsymbol{P}_i * \boldsymbol{P}_r$. Because $x_i$ is on the road of $x_k$, $\boldsymbol{P}_i = \boldsymbol{P}_k * \boldsymbol{P}_i$, $\boldsymbol{P}_i = \boldsymbol{P}_i * \boldsymbol{P}_k = \boldsymbol{P}_k * \boldsymbol{P}_i * \boldsymbol{P}_r = \boldsymbol{P}_r * \left(\boldsymbol{P}_k * \boldsymbol{P}_i\right) = \boldsymbol{P}_r * \boldsymbol{P}_i$, contradicting the assumption $x_r \in \boldsymbol{C}_i$. □

**Proof of Theorem 6.** According to Theorem 5, $\boldsymbol{C}_i \subseteq \boldsymbol{C}_k$; we need to further prove $\boldsymbol{C}_k \subseteq \boldsymbol{C}_i$. Transform the problem: $\forall x_r \in \boldsymbol{C}_k$, prove $x_r \in \boldsymbol{C}_i$, i.e., $\boldsymbol{P}_r \neq \boldsymbol{P}_i * \boldsymbol{P}_r$ and $\boldsymbol{P}_i \neq \boldsymbol{P}_i * \boldsymbol{P}_r$.

(1) Suppose for the sake of contradiction that $x_r \in \boldsymbol{C}_k$ but $\boldsymbol{P}_r = \boldsymbol{P}_i * \boldsymbol{P}_r$. It is known that section $x_i$ is on the road of section $x_k$, and according to Theorem 1: $\boldsymbol{P}_i = \boldsymbol{P}_k * \boldsymbol{P}_i$. Substitute the assumption $\boldsymbol{P}_r = \boldsymbol{P}_i * \boldsymbol{P}_r = (\boldsymbol{P}_k * \boldsymbol{P}_i) * \boldsymbol{P}_r = \boldsymbol{P}_k * (\boldsymbol{P}_i * \boldsymbol{P}_r) = \boldsymbol{P}_k * \boldsymbol{P}_r$, contradicting the assumption $x_r \in \boldsymbol{C}_k$.

(2) Suppose for the sake of contradiction that $x_r \in \boldsymbol{C}_k$ but $\boldsymbol{P}_i = \boldsymbol{P}_i * \boldsymbol{P}_r$. According to Theorem 1, section $x_i$ is on the road of section $x_r$. We claim that there is a path between section $x_r$ and section $x_k$. Otherwise, under the premise that section $x_i$ is simultaneously on the road of $x_r$ and $x_k$, the only possibility that there is no path between $x_r$ and $x_k$, as shown in Figure A3. As can be seen in Figure A3,

section $x_j$ is a T-type section, contradicting the condition that the path from $x_k$ to $x_i$ does not contain a T-type section; thus, there is a path between section $x_r$ and section $x_k$. That is, $x_r$ is on the road of $x_k$ or $x_k$ is on the road of $x_r$. According to Theorem 1, $\boldsymbol{P}_r = \boldsymbol{P}_r * \boldsymbol{P}_k$ or $\boldsymbol{P}_k = \boldsymbol{P}_r * \boldsymbol{P}_k$, contradicting the assumption $x_r \in \boldsymbol{C}_k$. □

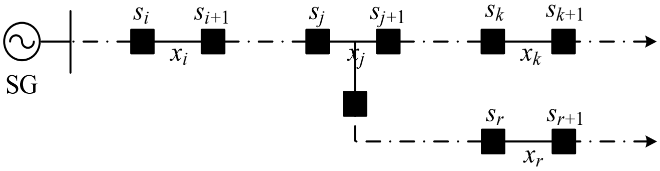


**Figure A3.** Positional relationships between sections $x_i$, $x_r$, and $x_k$.

## Appendix C. The Proofs of the Properties of Approximation Gains

**Proof of Theorem 7.** The proof of Theorem 7 requires the help of two conclusions, which are stated and proved in this section.□

**Conclusion A1**. Let $\boldsymbol{C}_i \subseteq \boldsymbol{C}_k$; then, $\forall x_r \in \boldsymbol{C}_i$, $\boldsymbol{P}_i * \boldsymbol{P}_r = \boldsymbol{P}_k * \boldsymbol{P}_r$.

**Proof of Conclusion A1**. We claim that there is a path between section $x_i$ and section $x_k$. Otherwise, according to Theorem 1, $\boldsymbol{P}_k \neq \boldsymbol{P}_i * \boldsymbol{P}_k$ and $\boldsymbol{P}_i \neq \boldsymbol{P}_i * \boldsymbol{P}_k$; thus, $x_k \subseteq \boldsymbol{C}_i$. Because $\boldsymbol{C}_i \subseteq \boldsymbol{C}_k$, $x_k \in \boldsymbol{C}_k$. Because $\boldsymbol{P}_k = \boldsymbol{P}_k * \boldsymbol{P}_k$, $x_k \notin \boldsymbol{C}_k$. These two conclusions contradict each other. Therefore, there is a path between section $x_i$ and section $x_k$. There are two possible positional relationships between $x_i$ and $x_k$, which are argued for separately.

(1) $x_i$ is on the road of $x_k$. According to Theorem 1, $\boldsymbol{P}_i = \boldsymbol{P}_i * \boldsymbol{P}_k$. According to Theorem 2, $\boldsymbol{P}_k * \boldsymbol{P}_i * (\boldsymbol{e} - \boldsymbol{P}_r) = \boldsymbol{0}$ or $\boldsymbol{P}_k * \boldsymbol{P}_r * (\boldsymbol{e} - \boldsymbol{P}_i) = \boldsymbol{0}$. The previous equation cannot hold; otherwise, $\boldsymbol{P}_k * \boldsymbol{P}_i = \boldsymbol{P}_k * \boldsymbol{P}_i * \boldsymbol{P}_r$, substituting $\boldsymbol{P}_i = \boldsymbol{P}_i * \boldsymbol{P}_k$, $\boldsymbol{P}_i = \boldsymbol{P}_i * \boldsymbol{P}_k = \boldsymbol{P}_k * \boldsymbol{P}_i * \boldsymbol{P}_r$ $= \boldsymbol{P}_i * \boldsymbol{P}_r$; thus, $x_r \notin \boldsymbol{C}_i$, contradicting what is known. According to the latter equation, $\boldsymbol{P}_k * \boldsymbol{P}_r = \boldsymbol{P}_r * \boldsymbol{P}_i * \boldsymbol{P}_k$, substituting $\boldsymbol{P}_i = \boldsymbol{P}_i * \boldsymbol{P}_k$ into $\boldsymbol{P}_k * \boldsymbol{P}_r = \boldsymbol{P}_i * \boldsymbol{P}_r$.

(2) $x_k$ is on the road of $x_i$. According to Theorem 1, $\boldsymbol{P}_k = \boldsymbol{P}_i * \boldsymbol{P}_k$. According to Theorem 2, $\boldsymbol{P}_i * \boldsymbol{P}_k * (\boldsymbol{e} - \boldsymbol{P}_r) = \boldsymbol{0}$ or $\boldsymbol{P}_i * \boldsymbol{P}_r * (\boldsymbol{e} - \boldsymbol{P}_k) = \boldsymbol{0}$. The previous equation cannot hold; otherwise, $\boldsymbol{P}_i * \boldsymbol{P}_k = \boldsymbol{P}_i * \boldsymbol{P}_k * \boldsymbol{P}_r$, substituting $\boldsymbol{P}_k = \boldsymbol{P}_i * \boldsymbol{P}_k$, $\boldsymbol{P}_k = \boldsymbol{P}_k * \boldsymbol{P}_r = \boldsymbol{P}_i * \boldsymbol{P}_k * \boldsymbol{P}_r$, $= \boldsymbol{P}_i * \boldsymbol{P}_r$; thus, $x_r \notin \boldsymbol{C}_k$. Because $\boldsymbol{C}_i \subseteq \boldsymbol{C}_k$ and $x_r \in \boldsymbol{C}_i$, $x_r \in \boldsymbol{C}_k$. The two conclusions contradict each other. According to the latter equation, $\boldsymbol{P}_i * \boldsymbol{P}_r = \boldsymbol{P}_i * \boldsymbol{P}_k * \boldsymbol{P}_r$, substitute $\boldsymbol{P}_k = \boldsymbol{P}_i * \boldsymbol{P}_k$ into $\boldsymbol{P}_i * \boldsymbol{P}_r = \boldsymbol{P}_k * \boldsymbol{P}_r$.□

**Conclusion A2.** Suppose we have section $x_i \in \boldsymbol{E}$, set of sections $\boldsymbol{X} \subseteq \{x_k \in \boldsymbol{E} \mid \boldsymbol{C}_i \subseteq \boldsymbol{C}_k\}$, and $\boldsymbol{X} \neq \varnothing$. Then, $\forall x_r \in \boldsymbol{C}_i$, $\boldsymbol{P}_i * \boldsymbol{P}_r = \boldsymbol{J}(\boldsymbol{X}) * \boldsymbol{P}_r$.

**Proof of Conclusion A2.** According to Equation (19):

$$\boldsymbol{J}(\boldsymbol{X}) * \boldsymbol{P}_r = \left[\boldsymbol{e} - \prod_{x_k \in \boldsymbol{X}}^{*}\left(\boldsymbol{e} - \boldsymbol{P}_k\right)\right] * \boldsymbol{P}_r = \boldsymbol{P}_r - \prod_{x_k \in \boldsymbol{X}}^{*}\left(\boldsymbol{P}_r - \boldsymbol{P}_k * \boldsymbol{P}_r\right) \tag{A3}$$

According to Conclusion A1, substitute $\boldsymbol{P}_i * \boldsymbol{P}_r = \boldsymbol{P}_k * \boldsymbol{P}_r$ into Equation (A3): $\boldsymbol{J}(\boldsymbol{X}) * \boldsymbol{P}_r = \boldsymbol{P}_r - \left(\boldsymbol{P}_r - \boldsymbol{P}_i * \boldsymbol{P}_r\right) = \boldsymbol{P}_i * \boldsymbol{P}_r$.

On this basis, Theorem 7 is proved. Apply the data structure "Queue" to represent ***R***, ***R*** = {*r*1, *r*2, …, *r*u, …}, and let $\boldsymbol{R}_u$ be the queue consisting of the first *u* elements in ***R***.

$$\boldsymbol{R}_u = \{r_z \in \boldsymbol{R} \mid 1 \le z \le u\} \tag{A4}$$

Let ***R*** have *p* elements and $\boldsymbol{R}_0 = \varnothing$. According to Equation (33), $\Delta_{\boldsymbol{R}}^{\boldsymbol{X}}$ is transformed into a superposition of approximation gains of the sections in ***R***.

$$\Delta_{\boldsymbol{R}}^{\boldsymbol{X}} = \Delta_{r_1}^{\boldsymbol{X}\cup\boldsymbol{R}_0} + \Delta_{r_2}^{\boldsymbol{X}\cup\boldsymbol{R}_1} + \ldots + \Delta_{r_u}^{\boldsymbol{X}\cup\boldsymbol{R}_{u-1}} + \ldots = \sum_{u=1}^{p} \Delta_{r_u}^{\boldsymbol{X}\cup\boldsymbol{R}_{u-1}} \tag{A5}$$

Substitute Equations (14) and (30) into Equation (A5).

$$\begin{aligned}
\Delta_{\boldsymbol{R}}^{\boldsymbol{X}} &= \sum_{u=1}^{p}\left\{\left[(2\boldsymbol{I}-\boldsymbol{e}) * \prod_{r\in\boldsymbol{R}_{u-1}}^{*}(\boldsymbol{e}-\boldsymbol{P}_r) * \prod_{x_k\in\boldsymbol{X}}^{*}(\boldsymbol{e}-\boldsymbol{P}_k)\right]^T \boldsymbol{P}_{r_u}\right\} \\
&= \sum_{u=1}^{p}\left\{\left[(2\boldsymbol{I}-\boldsymbol{e}) * \prod_{r\in\boldsymbol{R}_{u-1}}^{*}(\boldsymbol{e}-\boldsymbol{P}_r)\right]^T \prod_{x_k\in\boldsymbol{X}}^{*}(\boldsymbol{e}-\boldsymbol{P}_k) * \boldsymbol{P}_{r_u}\right\} \\
&= \sum_{u=1}^{p}\left\{\left[(2\boldsymbol{I}-\boldsymbol{e}) * \prod_{r\in\boldsymbol{R}_{u-1}}^{*}(\boldsymbol{e}-\boldsymbol{P}_r)\right]^T \left[\boldsymbol{e}-\boldsymbol{J}(\boldsymbol{X})\right] * \boldsymbol{P}_{r_u}\right\} \\
&= \sum_{u=1}^{p}\left\{\left[(2\boldsymbol{I}-\boldsymbol{e}) * \prod_{r\in\boldsymbol{R}_{u-1}}^{*}(\boldsymbol{e}-\boldsymbol{P}_r)\right]^T \left[\boldsymbol{P}_{r_u} - \boldsymbol{J}(\boldsymbol{X}) * \boldsymbol{P}_{r_u}\right]\right\}
\end{aligned} \tag{A6}$$

According to Conclusion A2, $\boldsymbol{P}_i * \boldsymbol{P}_{r_u} = \boldsymbol{J}(\boldsymbol{X}) * \boldsymbol{P}_{r_u}$.

$$\begin{aligned}
\Delta_{\boldsymbol{R}}^{\boldsymbol{X}} &= \sum_{u=1}^{p}\left\{\left[(2\boldsymbol{I}-\boldsymbol{e}) * \prod_{r\in\boldsymbol{R}_{u-1}}^{*}(\boldsymbol{e}-\boldsymbol{P}_r)\right]^T \left(\boldsymbol{P}_{r_u} - \boldsymbol{P}_i * \boldsymbol{P}_{r_u}\right)\right\} \\
&= \sum_{u=1}^{p}\left\{\left[(2\boldsymbol{I}-\boldsymbol{e}) * \prod_{r\in\boldsymbol{R}_{u-1}}^{*}(\boldsymbol{e}-\boldsymbol{P}_r) * (\boldsymbol{e}-\boldsymbol{P}_i)\right]^T \boldsymbol{P}_{r_u}\right\} \\
&= \sum_{u=1}^{m} \Delta_{r_u}^{i\cup\boldsymbol{R}_{u-1}} = \Delta_{r_1}^{i\cup\boldsymbol{R}_0} + \Delta_{r_2}^{i\cup\boldsymbol{R}_1} + \ldots + \Delta_{r_u}^{i\cup\boldsymbol{R}_{u-1}} + \ldots = \Delta_{\boldsymbol{R}}^{i}
\end{aligned} \tag{A7}$$

□

## Appendix D. The Proofs of Recursive Structure

**Proof of Theorem 8.** It holds that $\forall \boldsymbol{X}' \subseteq \boldsymbol{E}_i \neq \{x_k\}, \forall \boldsymbol{R} \subseteq \boldsymbol{E} - \boldsymbol{E}_i$, $\boldsymbol{X}' \cup \boldsymbol{R}$ satisfies the inequality constraint. According to Theorem 4, $\boldsymbol{X}'$, $\boldsymbol{R}$ satisfy the inequality constraint. We claim that $\boldsymbol{X}'$ contains only one section. Otherwise, let $x_a, x_b \in \boldsymbol{X}'$; because $\boldsymbol{G}_i$ does not contain a T-type section, there is a path between $x_a$ and $x_b$, i.e., $x_a$ on the road of $x_b$ or $x_b$ on the road of $x_a$. According to Theorem 1, $\boldsymbol{P}_a = \boldsymbol{P}_a * \boldsymbol{P}_b$ or $\boldsymbol{P}_b = \boldsymbol{P}_a * \boldsymbol{P}_b$; thus, $\boldsymbol{X}'$ does not satisfy the inequality constraint, contradicting what is known.

Let $\boldsymbol{X}' = \{x_r\}$, $x_r \neq x_k \in \boldsymbol{E}_i$. According to Theorem 4, $\boldsymbol{R} \subseteq \boldsymbol{C}_r$. Because there is no T-type section on the path between $x_k$ and $x_r$, according to Theorem 6, $\boldsymbol{C}_r = \boldsymbol{C}_k$; thus, $\boldsymbol{R} \subseteq \boldsymbol{C}_k$. Because $\boldsymbol{X}_i$ contains only one section, it must satisfy the inequality constraint. According to Theorem 4, $\boldsymbol{X}_i \cup \boldsymbol{R}$ satisfies the inequality constraint.

Applying the approximation gain definition Equations (31) and (33) to rewrite $f(\boldsymbol{X}_i \cup \boldsymbol{R})$:

$$\begin{aligned} f(\boldsymbol{X}_i \cup \boldsymbol{R}) &= f(x_k \cup \boldsymbol{R}) = \left[f(x_k \cup \boldsymbol{R}) - f(x_k)\right] + \left[f(x_k) - f(\varnothing)\right] + f(\varnothing) \\ &= -\Delta_{\boldsymbol{R}}^{k} - \Delta_k + f(\varnothing) \end{aligned} \tag{A8}$$

Similarly, we obtain $f(\boldsymbol{X}' \cup \boldsymbol{R}) = -\Delta_{\boldsymbol{R}}^{r} - \Delta_r + f(\varnothing)$. Because $\boldsymbol{C}_r = \boldsymbol{C}_k$, $\boldsymbol{R} \subseteq \boldsymbol{C}_k$, according to Theorem 7: $\Delta_{\boldsymbol{R}}^{k} = \Delta_{\boldsymbol{R}}^{r}$. Because $\Delta_k > \Delta_r$, $f(\boldsymbol{X}_i \cup \boldsymbol{R}) < f(\boldsymbol{X}' \cup \boldsymbol{R})$. Therefore, $\boldsymbol{X}_i = \{x_k\}$ is the fault candidate set for $\boldsymbol{G}_i$. □

**Proof of Theorem 9.** It holds that $\forall \boldsymbol{X}' \subseteq \boldsymbol{E}_i$, $\forall \boldsymbol{R} \subseteq \boldsymbol{E} - \boldsymbol{E}_i$, $\boldsymbol{X}' \cup \boldsymbol{R}$ satisfies the inequality constraint. The proof of Theorem 9 is divided into the following three steps.

Step 1. Prove that $\boldsymbol{X}_{i-t} \cup \boldsymbol{R}$, $\boldsymbol{X}_u \cup \boldsymbol{R}$, $\boldsymbol{X}_v \cup \boldsymbol{R}$, $\boldsymbol{X}_{u\&v} \cup \boldsymbol{R}$ satisfy the inequality constraint. Because $\boldsymbol{X}' \cup \boldsymbol{R}$ satisfies the inequality constraint, according to Theorem 4, $\boldsymbol{R}$ satisfies the inequality constraint. Therefore, it remains to be proved that (1) $\boldsymbol{X}_{i-t}$, $\boldsymbol{X}_u$, $\boldsymbol{X}_v$, $\boldsymbol{X}_{u\&v}$ satisfy the inequality constraint; (2) $\forall x_w \in \boldsymbol{X}_u$, $\forall x_y \in \boldsymbol{X}_v$, $\forall x_z \in \boldsymbol{X}_{u\&v}$, there should be $\boldsymbol{R} \subseteq \boldsymbol{C}_k$, $\boldsymbol{R} \subseteq \boldsymbol{C}_w$, $\boldsymbol{R} \subseteq \boldsymbol{C}_y$, $\boldsymbol{R} \subseteq \boldsymbol{C}_z$. The proofs are given separately below.

(1) Because $\boldsymbol{X}_{i-t}$ contains only one element, $\boldsymbol{X}_u$ and $\boldsymbol{X}_v$ are fault candidate sets; thus, $\boldsymbol{X}_{i-t}$, $\boldsymbol{X}_u$, $\boldsymbol{X}_v$ satisfy the inequality constraint. Then, $\forall x_w \in \boldsymbol{X}_u$, $\forall x_y \in \boldsymbol{X}_v$, we claim that $\boldsymbol{P}_w \neq \boldsymbol{P}_w * \boldsymbol{P}_y$. Otherwise, by $x_w \in \boldsymbol{X}_u \subseteq \boldsymbol{E}_u$: $\boldsymbol{P}_u = \boldsymbol{P}_w * \boldsymbol{P}_u$. Substitute $\boldsymbol{P}_w = \boldsymbol{P}_w * \boldsymbol{P}_y$ for $\boldsymbol{P}_u = \boldsymbol{P}_w * \boldsymbol{P}_u = (\boldsymbol{P}_w * \boldsymbol{P}_y) * \boldsymbol{P}_u = \boldsymbol{P}_y * (\boldsymbol{P}_w * \boldsymbol{P}_u) = \boldsymbol{P}_y * \boldsymbol{P}_u$; that is, $x_y \in \boldsymbol{E}_u$. According to Figure 4, $\boldsymbol{E}_u \cap \boldsymbol{E}_v = \varnothing$; thus, $x_y \notin \boldsymbol{E}_v$, contradicting what is known, $x_y \in \boldsymbol{X}_v \subseteq \boldsymbol{E}_v$. Similarly, $\boldsymbol{P}_z \neq \boldsymbol{P}_w * \boldsymbol{P}_z$. Therefore, $x_y \in \boldsymbol{C}_w$, $\boldsymbol{X}_v \subseteq \boldsymbol{C}_w$. According to Theorem 4, $\boldsymbol{X}_{u\&v} = \boldsymbol{X}_u \cup \boldsymbol{X}_v$ satisfies the inequality constraint.

(2) Let section *x<sub>i</sub>* be the starting section of $\boldsymbol{G}_i$. Then, $\forall x_r \in \boldsymbol{R}$, $\forall x_q \in \boldsymbol{X}'$, according to Theorem 4, $x_r \in \boldsymbol{C}_q$. We claim that $\boldsymbol{P}_i \neq \boldsymbol{P}_i * \boldsymbol{P}_r$ and $\boldsymbol{P}_r \neq \boldsymbol{P}_i * \boldsymbol{P}_r$. If $\boldsymbol{P}_i = \boldsymbol{P}_i * \boldsymbol{P}_r$, then $x_r \in \boldsymbol{E}_i$, contradicting what is known, $x_r \in \boldsymbol{R} \subseteq \boldsymbol{E} - \boldsymbol{E}_i$. If $\boldsymbol{P}_r = \boldsymbol{P}_i * \boldsymbol{P}_r$, by $x_q \in \boldsymbol{E}_i$, $\boldsymbol{P}_i = \boldsymbol{P}_i * \boldsymbol{P}_q$, then $\boldsymbol{P}_r = \boldsymbol{P}_i * \boldsymbol{P}_r = (\boldsymbol{P}_i * \boldsymbol{P}_q) * \boldsymbol{P}_r = \boldsymbol{P}_q * (\boldsymbol{P}_i * \boldsymbol{P}_r) = \boldsymbol{P}_q * \boldsymbol{P}_r$; thus, $x_r \notin \boldsymbol{C}_q$, contradicting what is known, $x_r \in \boldsymbol{C}_q$. Thus, $\boldsymbol{P}_i \neq \boldsymbol{P}_i * \boldsymbol{P}_r$ and $\boldsymbol{P}_r \neq \boldsymbol{P}_i * \boldsymbol{P}_r$ hold, i.e., $x_r \in \boldsymbol{C}_i$, $\boldsymbol{R} \subseteq \boldsymbol{C}_i$. Due to $x_k, x_w, x_y, x_z \in \boldsymbol{E}_i$: $\boldsymbol{P}_i = \boldsymbol{P}_i * \boldsymbol{P}_k$, $\boldsymbol{P}_i = \boldsymbol{P}_i * \boldsymbol{P}_w$, $\boldsymbol{P}_i = \boldsymbol{P}_i * \boldsymbol{P}_y$, $\boldsymbol{P}_i = \boldsymbol{P}_i * \boldsymbol{P}_z$, according to Theorem 1, section *x<sub>i</sub>* is on the road of $x_k, x_w, x_y, x_z$. According to Theorem 5, $\boldsymbol{C}_i \subseteq \boldsymbol{C}_k$, $\boldsymbol{C}_i \subseteq \boldsymbol{C}_w$, $\boldsymbol{C}_i \subseteq \boldsymbol{C}_y$, $\boldsymbol{C}_i \subseteq \boldsymbol{C}_z$. Combined with $\boldsymbol{R} \subseteq \boldsymbol{C}_i$: $\boldsymbol{R} \subseteq \boldsymbol{C}_k$, $\boldsymbol{R} \subseteq \boldsymbol{C}_w$, $\boldsymbol{R} \subseteq \boldsymbol{C}_y$, $\boldsymbol{R} \subseteq \boldsymbol{C}_z$.

Step 2. Prove that the fault candidate set of $\boldsymbol{G}_i$ is only one of $\boldsymbol{X}_{i-t}$, $\boldsymbol{X}_u$, $\boldsymbol{X}_v$, $\boldsymbol{X}_{u\&v}$. Combined with the conclusion of Step 1, the proposition is transformed: if $\boldsymbol{X}' \neq \boldsymbol{X}_{i-t}, \boldsymbol{X}_u, \boldsymbol{X}_v, \boldsymbol{X}_{u\&v}$, then one of the following equations must hold: $f(\boldsymbol{X}_{i-t} \cup \boldsymbol{R}) < f(\boldsymbol{X}' \cup \boldsymbol{R})$, $f(\boldsymbol{X}_u \cup \boldsymbol{R}) < f(\boldsymbol{X}' \cup \boldsymbol{R})$, $f(\boldsymbol{X}_v \cup \boldsymbol{R}) < f(\boldsymbol{X}' \cup \boldsymbol{R})$, or $f(\boldsymbol{X}_{u\&v} \cup \boldsymbol{R}) < f(\boldsymbol{X}' \cup \boldsymbol{R})$.

Because all sections in $\boldsymbol{E}_{i-t}$ are on the road of sections in $\boldsymbol{E}_u$ and $\boldsymbol{E}_v$, there are four cases of $\boldsymbol{X}'$ that meet the inequality constraint: $\boldsymbol{X}' \subseteq \boldsymbol{E}_{i-t}$, $\boldsymbol{X}' \subseteq \boldsymbol{E}_u$, $\boldsymbol{X}' \subseteq \boldsymbol{E}_v$, and $\boldsymbol{X}' \subseteq \boldsymbol{E}_u \cup \boldsymbol{E}_v$. The proofs are given separately below.

(1) $\boldsymbol{X}' \subseteq \boldsymbol{E}_{i-t}$. Same procedure as the proof of Theorem 8: the $\boldsymbol{X}'$ satisfying the inequality constraint contains only one section, and let $\boldsymbol{X}' = \left\{x_q\right\}$. Apply Equations (31) and (33) to rewrite $f(\boldsymbol{X}_{i-t} \cup \boldsymbol{R})$: $f(\boldsymbol{X}_{i-t} \cup \boldsymbol{R}) = -\Delta_{\boldsymbol{R}}^{k} - \Delta_k + f(\varnothing)$ (the derivation process is similar to Equation (A8)). Similarly, $f(\boldsymbol{X}' \cup \boldsymbol{R}) = -\Delta_{\boldsymbol{R}}^{q} - \Delta_q + f(\varnothing)$. Because

there is no T-type section on the path between $x_k$ and $x_q$, according to Theorem 6, $\boldsymbol{C}_k = \boldsymbol{C}_q$. $\boldsymbol{R} \subseteq \boldsymbol{C}_k$, according to the conclusion of step 1, and according to Theorem 7, $\Delta_{\boldsymbol{R}}^{k} = \Delta_{\boldsymbol{R}}^{q}$. Combined with $\Delta_k > \Delta_q$: $f(\boldsymbol{X}_{i-t} \cup \boldsymbol{R}) < f(\boldsymbol{X'} \cup \boldsymbol{R})$.

(2) $\boldsymbol{X'} \subseteq \boldsymbol{E}_u$. Because $\boldsymbol{R} \subseteq \boldsymbol{E} - \boldsymbol{E}_i \subseteq \boldsymbol{E} - \boldsymbol{E}_u$, $\boldsymbol{X}_u$ is the fault candidate set of $\boldsymbol{G}_u$, according to the definition of fault candidate sets, $f(\boldsymbol{X}_u \cup \boldsymbol{R}) < f(\boldsymbol{X'} \cup \boldsymbol{R})$.

(3) $\boldsymbol{X'} \subseteq \boldsymbol{E}_v$. Because $\boldsymbol{R} \subseteq \boldsymbol{E} - \boldsymbol{E}_i \subseteq \boldsymbol{E} - \boldsymbol{E}_v$, $\boldsymbol{X}_v$ is the fault candidate set of $\boldsymbol{G}_v$, according to the definition of fault candidate sets, $f(\boldsymbol{X}_v \cup \boldsymbol{R}) < f(\boldsymbol{X'} \cup \boldsymbol{R})$.

(4) $\boldsymbol{X'} \subseteq \boldsymbol{E}_u \cup \boldsymbol{E}_v$. Let the sets of sections $\boldsymbol{X}_a = \boldsymbol{X'} \cap \boldsymbol{E}_u$, $\boldsymbol{X}_b = \boldsymbol{X'} \cap \boldsymbol{E}_v$. Because $\boldsymbol{E}_u \cap \boldsymbol{E}_v = \varnothing$, $\boldsymbol{X'} = \boldsymbol{X}_a \cup \boldsymbol{X}_b$. $f(\boldsymbol{X'} \cup \boldsymbol{R})$ can be equivalently expressed as $f(\boldsymbol{X'} \cup \boldsymbol{R}) = f(\boldsymbol{X}_a \cup \boldsymbol{X}_b \cup \boldsymbol{R})$. It is known that $\boldsymbol{X}_u$ is a fault candidate set of $\boldsymbol{G}_u$, $\boldsymbol{X}_a \subseteq \boldsymbol{E}_u$, $\boldsymbol{X}_b \cup \boldsymbol{R} \subseteq \boldsymbol{E} - \boldsymbol{E}_u$, and therefore, $f(\boldsymbol{X}_u \cup \boldsymbol{X}_b \cup \boldsymbol{R}) < f(\boldsymbol{X}_a \cup \boldsymbol{X}_b \cup \boldsymbol{R})$. It is known that $\boldsymbol{X}_v$ is a fault candidate set of $\boldsymbol{G}_v$, $\boldsymbol{X}_b \subseteq \boldsymbol{E}_v$, $\boldsymbol{X}_u \cup \boldsymbol{R} \subseteq \boldsymbol{E} - \boldsymbol{E}_v$; therefore, $f(\boldsymbol{X}_u \cup \boldsymbol{X}_v \cup \boldsymbol{R}) < f(\boldsymbol{X}_u \cup \boldsymbol{X}_b \cup \boldsymbol{R})$. Combined with $\boldsymbol{X}_{u\&v} = \boldsymbol{X}_u \cup \boldsymbol{X}_v$, $\boldsymbol{X'} = \boldsymbol{X}_a \cup \boldsymbol{X}_b$: $f(\boldsymbol{X}_{u\&v} \cup \boldsymbol{R}) < f(\boldsymbol{X'} \cup \boldsymbol{R})$.

Step 3: Prove that the set corresponding to the largest of $\Delta_{\boldsymbol{X}_{i-t}}$, $\Delta_{\boldsymbol{X}_u}$, $\Delta_{\boldsymbol{X}_v}$ and $\Delta_{\boldsymbol{X}_{u\&v}}$ is the fault candidate set $\boldsymbol{X}_i$ for $\boldsymbol{G}_i$. Apply Equations (31) and (33) to rewrite $f(\boldsymbol{X}_{i-t} \cup \boldsymbol{R})$, $f(\boldsymbol{X}_u \cup \boldsymbol{R})$, $f(\boldsymbol{X}_v \cup \boldsymbol{R})$, and $f(\boldsymbol{X}_{u\&v} \cup \boldsymbol{R})$ (the derivation process is similar to Equation (A8)).

$$\begin{cases} f(\boldsymbol{X}_{i-t} \cup \boldsymbol{R}) = -\Delta_{\boldsymbol{R}}^{\boldsymbol{X}_{i-t}} - \Delta_{\boldsymbol{X}_{i-t}} + f(\varnothing) \\ f(\boldsymbol{X}_u \cup \boldsymbol{R}) = -\Delta_{\boldsymbol{R}}^{\boldsymbol{X}_u} - \Delta_{\boldsymbol{X}_u} + f(\varnothing) \\ f(\boldsymbol{X}_v \cup \boldsymbol{R}) = -\Delta_{\boldsymbol{R}}^{\boldsymbol{X}_v} - \Delta_{\boldsymbol{X}_v} + f(\varnothing) \\ f(\boldsymbol{X}_{u\&v} \cup \boldsymbol{R}) = -\Delta_{\boldsymbol{R}}^{\boldsymbol{X}_{u\&v}} - \Delta_{\boldsymbol{X}_{u\&v}} + f(\varnothing) \end{cases} \tag{A9}$$

According to the argument process in Step 1, for $x_k \in \boldsymbol{X}_{i-t}$, $\forall x_w \in \boldsymbol{X}_u$, $\forall x_y \in \boldsymbol{X}_v$, and $\forall x_z \in \boldsymbol{X}_{u\&v}$, there are $\boldsymbol{C}_i \subseteq \boldsymbol{C}_k$, $\boldsymbol{C}_i \subseteq \boldsymbol{C}_w$, $\boldsymbol{C}_i \subseteq \boldsymbol{C}_y$, and $\boldsymbol{C}_i \subseteq \boldsymbol{C}_z$; thus, $\boldsymbol{X}_{i-t}, \boldsymbol{X}_u, \boldsymbol{X}_v, \boldsymbol{X}_{u\&v} \subseteq \{x_j \in \boldsymbol{E} \mid \boldsymbol{C}_i \subseteq \boldsymbol{C}_j\}$. Combined with $\boldsymbol{R} \subseteq \boldsymbol{C}_i$, according to Theorem 7, $\Delta_{\boldsymbol{R}}^{\boldsymbol{X}_{i-t}} = \Delta_{\boldsymbol{R}}^{\boldsymbol{X}_u} = \Delta_{\boldsymbol{R}}^{\boldsymbol{X}_v} = \Delta_{\boldsymbol{R}}^{\boldsymbol{X}_{u\&v}} = \Delta_{\boldsymbol{R}}^{i}$. Therefore, the largest of $\Delta_{\boldsymbol{X}_{i-t}}$, $\Delta_{\boldsymbol{X}_u}$, $\Delta_{\boldsymbol{X}_v}$, and $\Delta_{\boldsymbol{X}_{u\&v}}$ corresponds to the smallest objective function on the left side. Therefore, the set corresponding to the largest of $\Delta_{\boldsymbol{X}_{i-t}}$, $\Delta_{\boldsymbol{X}_u}$, $\Delta_{\boldsymbol{X}_v}$ and $\Delta_{\boldsymbol{X}_{u\&v}}$ is the fault candidate set $\boldsymbol{X}_i$ for $\boldsymbol{G}_i$. □

## Appendix E. Explanation of the Fast Method for Calculating the Section Approximation Gain in Lines 2 and 5 of the Pseudocode DACFL Recursive Structure

The equivalent mathematical description of the fast computation method described is as follows: let the set of nodes in the distribution network be $\boldsymbol{V}$, let $s_k \in V$ be a node, let the upstream section of the node be $x_{k-1}$, and let the downstream section of the node be $x_k$; then, the approximation gains $\Delta_{k-1}$ and $\Delta_k$ of $x_{k-1}$ and $x_k$ satisfy the following expression: $\Delta_k = \Delta_{k-1} + 2\boldsymbol{I}_k - 1$.

The proof of this conclusion follows. According to the known conditions, the connection relationship of node $s_k$, section $x_{k-1}$, and section $x_k$ is shown in Figure A4. It can be seen in A1 that section $x_{k-1}$ is on the road of section $x_k$. Assuming that section $x_{k-1}$ reaches the main power source through the set of nodes $\boldsymbol{SI} = \{sk_1, sk_2, \ldots\}$, then the corresponding position of $\{sk_1, sk_2, \ldots\}$ in $\boldsymbol{P}_{i-1}$ is taken as 1, and the corresponding position of $\{sk_1, sk_2, \ldots\}$ and $s_i$ in $\boldsymbol{P}_i$ is taken as 1.

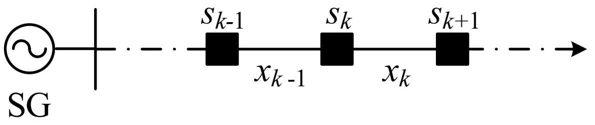


**Figure A4.** Connection relationships between nodes and sections.

Expand $\Delta_{k-1}$ and $\Delta_k$ by applying Equation (32).

$$\begin{cases} \Delta_{k-1} = (2\boldsymbol{I}-\boldsymbol{e})^T \boldsymbol{P}_{k-1} = \sum\limits_{s_i \in \boldsymbol{SI}} [2\boldsymbol{I}_i - 1] \\ \Delta_k = (2\boldsymbol{I}-\boldsymbol{e})^T \boldsymbol{P}_k = \sum\limits_{s_i \in \boldsymbol{SI} \cup s_k} [2\boldsymbol{I}_i - 1] = [2\boldsymbol{I}_k - 1] + \sum\limits_{s_i \in \boldsymbol{SI}} [2\boldsymbol{I}_i - 1] \end{cases}$$

Subtract these two equations to obtain $\Delta_k = \Delta_{k-1} + 2\boldsymbol{I}_k - 1$.

## Appendix F. Explanation of the Fast Method for Calculating the Set Approximation Gain in Line 23 of the Pseudocode DACFL

The equivalent mathematical description of the fast computation method described is as follows: let the subnetwork $\boldsymbol{G}_i$ contain a T-type section, and the names of sections and variables are the same as described in Figure 4. Then, $\Delta_{X_{u\&v}} = \Delta_{X_u} + \Delta_{X_v} - \Delta_t$.

The proof of the conclusion follows. According to Equation (33):

$$\Delta_{\boldsymbol{X}_{u\&v}} = f(\varnothing) - f(\boldsymbol{X}_u \cup \boldsymbol{X}_v) = [f(\varnothing) - f(\boldsymbol{X}_u)] + [f(\boldsymbol{X}_u) - f(\boldsymbol{X}_u \cup \boldsymbol{X}_v)] = \Delta_{\boldsymbol{X}_u} + \Delta_{\boldsymbol{X}_v}^{\boldsymbol{X}_u} \quad \text{(A10)}$$

Combining $\boldsymbol{X}_u \subseteq \{x_w \in \boldsymbol{E} \mid \boldsymbol{C}_u \subseteq \boldsymbol{C}_w\}$, $\boldsymbol{X}_v \subseteq \{x_y \in \boldsymbol{E} \mid y \in C_u\}$, according to Theorem 7: $\Delta_{\boldsymbol{X}_v}^{\boldsymbol{X}_u} = \Delta_{\boldsymbol{X}_v}^{u}$. According to Equations (31) and (33):

$$\begin{aligned} \Delta_{\boldsymbol{X}_v}^{u} &= f(u) - f(u \cup \boldsymbol{X}_v) = [f(u) - f(\varnothing)] + [f(\varnothing) - f(\boldsymbol{X}_v)] + [f(\boldsymbol{X}_v) - f(\boldsymbol{X}_v \cup u)] \\ &= -\Delta_u + \Delta_{\boldsymbol{X}_v} + \Delta_u^{\boldsymbol{X}_v} \end{aligned} \quad \text{(A11)}$$

Substituting Equation (A11) into (A10):

$$\Delta_{\boldsymbol{X}_{u\&v}} = \Delta_{\boldsymbol{X}_u} - \Delta_u + \Delta_{\boldsymbol{X}_v} + \Delta_u^{\boldsymbol{X}_v} \quad \text{(A12)}$$

Combining $\boldsymbol{X}_v \subseteq \{x_y \in \boldsymbol{E} \mid \boldsymbol{C}_v \subseteq \boldsymbol{C}_y\}$, $u \in \boldsymbol{C}_v$, according to Theorem 7: $\Delta_u^{\boldsymbol{X}_v} = \Delta_u^v$. According to Equation (30):

$$\Delta_u^v = \left[(2\boldsymbol{I}-\boldsymbol{e}) * (\boldsymbol{e} - \boldsymbol{P}_v)\right]^T \boldsymbol{P}_u = (2\boldsymbol{I}-\boldsymbol{e})^T \boldsymbol{P}_u - (2\boldsymbol{I}-\boldsymbol{e})^T (\boldsymbol{P}_v * \boldsymbol{P}_u) \quad \text{(A13)}$$

Because section $x_t$ is on the road of section $x_u$ and section $x_v$, assuming that section $x_t$ reaches the main power source through the set of nodes $\{st_1, st_2, \ldots\}$, then the corresponding position of $\{st_1, st_2, \ldots\}$ and $s_u$ in $\boldsymbol{P}_u$ are taken as 1, and the corresponding position of $\{st_1, st_2, \ldots\}$ and $s_v$ are $\boldsymbol{P}_v$ is taken as 1. The elements with both $\boldsymbol{P}_u$ and $\boldsymbol{P}_v$ being 1 are $\{st_1, st_2, \ldots\}$; thus, $\boldsymbol{P}_v * \boldsymbol{P}_u = \boldsymbol{P}_t$. Substitute $\boldsymbol{P}_v * \boldsymbol{P}_u = \boldsymbol{P}_t$ into Equation (A13):

$$\Delta_u^{\boldsymbol{X}_v} = \Delta_u^v = (2\boldsymbol{I}-\boldsymbol{e})^T \boldsymbol{P}_u - (2\boldsymbol{I}-\boldsymbol{e})^T (\boldsymbol{P}_v * \boldsymbol{P}_u) = (2\boldsymbol{I}-\boldsymbol{e})^T \boldsymbol{P}_u - (2\boldsymbol{I}-\boldsymbol{e})^T \boldsymbol{P}_t = \Delta_u - \Delta_t \quad \text{(A14)}$$

Substitute (A14) into (A12): $\Delta_{X_{u\&v}} = \Delta_{X_u} + \Delta_{X_v} - \Delta_t$.

## References

1. Li, G.; Chen, Q.; Zhang, J. Novel faulted section location method for distribution network based on status information of fault indicating equipment. *Appl. Sci.* **2020**, *10*, 5910.
2. He, W.; Wang, S.; Xu, T.; Shen, H.; Su, Y.; Liu, Z.; Xiong, Z.; Wang, X. A construction and development path of the urban resilient distribution network. *Power Syst. Technol.* **2022**, *46*, 680–690.
3. Galvez, C.; Abur, A. Fault location in active distribution networks containing distributed energy resources (DERs). *IEEE Trans. Power Deliv.* **2021**, *36*, 3128–3139.
4. Zhao, F.; Meng, Z.; Li, Y.; Liu, C. Pilot protection scheme for active distribution network based on fault components. *High Volt. Eng.* **2019**, *45*, 3092–3100.
5. Wang, S.; Liu, Q.; Zhao, Q.; Wang, H. Connotation Analysis and Prospect of Distribution Network Elasticity. *Autom. Electr. Power Syst.* **2021**, *45*, 1–9.
6. Shen, G.; Zhang, Y.; Qiu, H.; Wang, C.; Wen, F.; Salam, M.; Weng, L.; Zhang, L.; Zhang, S. Fault diagnosis with false and/or missing alarms in distribution systems with distributed generators. *Energies* **2018**, *11*, 2579.
7. Wang, S.; Song, L.; Shu, X. Adaptive overcurrent protection of active distribution network with high penetration of distributed generations and multiple loads. *High Volt. Eng.* **2019**, *45*, 1783–1794.
8. Zheng, T.; Ma, L.; Zhang, B. Fault tolerant fast fault section location method for active distribution network. *J. N. China Electr. Power Univ.* **2022**, *49*, 12–20.
9. Zhao, Q. *Research on Wide Area Protection of Smart Distribution Grid*; North China Electric Power University: Beijing, China, 2017.
10. Xu, B.; Yin, X.; Zhang, Z.; Pang, S.; Li, X. Fault location for distribution network based on matrix algorithm and optimization algorithm. *Autom. Electr. Power Syst.* **2019**, *43*, 152–158.
11. Ji, X.; Zhang, S.; Zhang, Y.; Han, X.; Xiao, Y.; Zeng, R. Fault section location for distribution network based on improved electromagnetism-like mechanism algorithm. *Autom. Electr. Power Syst.* **2021**, *45*, 157–165.
12. Teng, J.; Huang, W.; Luan, S. Automatic and fast faulted line-section location method for distribution systems based on fault indicators. *IEEE Trans. Power Syst.* **2014**, *4*, 1653–1662.
13. Ma, T.; Gao, L. Fault location algorithm for active distribution network with multi micro-grids. *Power Syst. Prot. Control* **2017**, *45*, 64–68.
14. Wang, Q.; Jin, T.; Mohamed, A. A fast and robust fault section location method for power distribution systems considering multisource information. *IEEE Syst. J.* **2022**, *16*, 1954–1964.
15. Jiao, Y.; Du, S.; Wang, Q.; Chen, C. Information aberrance correction and fault-section location for distribution networks based on the information contradiction theory. *Power Syst. Prot. Control* **2014**, *42*, 43–48.
16. Zheng, T.; Ma, L.; Li, B. Fault section location of active distribution network based on feeder terminal unit information distortion correction. *Power Syst. Technol.* **2021**, *45*, 3926–3934.
17. Kong, B.; Liu, J.; Zhou, J.; Zhou, Y.; Song, Z. Fault-tolerant algorithm for fault location in distribution network based on integer linear programming. *Power Syst. Prot. Control* **2020**, *48*, 27–35.
18. Wei, Z.; He, H.; Zheng, Y. A refined genetic algorithm for the fault sections location. *Proc. CSEE* **2002**, *22*, 127–130.
19. Guo, Z.; Chen, B.; Liu, C.; Xu, K.; Li, J. Fault location of distribution network based on genetic algorithm. *Power Syst. Technol.* **2007**, *31*, 88–92.
20. Zhao, Q.; Wang, Z.; Dong, W.; Bao, W. Research on fault location in a distribution network based on an immune binary particle swarm algorithm. *Power Syst. Prot. Control* **2020**, *48*, 83–89.
21. Zhang, Y.; Zhou, R.; Zhong, K. Application of improved ant colony algorithm in fault-section location of complex distribution network. *Power Syst. Technol.* **2011**, *35*, 224–228.
22. Liu, B.; Wang, F.; Chen, C.; Huang, H.; Dong, X. Harmony search algorithm for solving fault location in distribution networks with DG. *Trans. China Electrotech. Soc.* **2013**, *28*, 280–284.
23. Wang, Q.; Jin, T.; Tan, H.; Zhu, S.; Liu, S. A complete analytic model of section location in distribution network based on multi-factor dimensionality deduction. *Trans. China Electrotech. Soc.* **2019**, *34*, 3012–3024.
24. Guo, Z.; Xu, Q.; Hong, J.; Mao, X. Integer linear programming based fault section diagnosis method with high fault-tolerance and fast performance for distribution network. *Proc. CSEE* **2017**, *37*, 786–795.
25. He, R.; Hu, Z.; Li, Y.; Wang, T. Fault section location method for DG-DNs based on integer linear programming. *Power Syst. Technol.* **2018**, *42*, 3684–3690.
26. Li, Z.; Wang, Z.; Zhang, Y.; Qiao, X. Fault section location method for active distribution network based on linear programming with ascending dimension. *Autom. Electr. Power Syst.* **2021**, *12*, 122–131.
27. Yin, Y.; Wang, D.; Yu, Y. *Integer Programming*, 1st ed.; Science Press: Beijing, China, 2022; pp. 56–66.
28. Wang, Q.; Jin, T.; Tan, H.; Li, Z. The technology on fault location of distribution network based on hierarchical model and intelligent checking algorithm. *Trans. China Electrotech. Soc.* **2018**, *33*, 5327–5337.
29. Zheng, C.; Zhou, H.; Zheng, D.; Lin, Z.; Zhang, X. An active distribution network fault location method based on improved multi-universe algorithm. *Power Syst. Prot. Control* **2023**, *51*, 169–179.
30. Gao, F.Y.; Li, Z.J.; Yuan, C.; Qi, X.D.; Li, X.F.; Zhuang, S.X. Fault location for active distribution network based on quantum computing and immune optimization algorithm. *High Volt. Eng.* **2021**, *47*, 396–406.

31. Yang, G.; Feng, J.; Liu, X.; Chen, R.; Pan, H.; Yang, Q. Fault location of a distribution network hierarchical model with a distribution generator based on IBES. *Power Syst. Prot. Control* **2022**, *50*, 1–9.
32. Cormen, T.H.; Leiserson, C.E.; Rivest, R.L.; Stein, C. *Introduction to Algorithms*, 3rd ed.; The MIT Press: London, UK, 2009; pp. 24–37.
33. Li, Y.; Wang, Z.; Zhao, Q. Distributed fault section location for ADN based on bayesian complete analytic model and multifactor dimension reduction. *Power Syst. Technol.* **2021**, *45*, 3917–3925.

**Disclaimer/Publisher's Note:** The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.