Breadth-first 검색 알고리즘을 이용한 와이어 오류 검출에 관한 연구

Error Wire Locating Technology with Breadth-first Search Algorithm

서건*, 이정표**, 이재철***, 김일구****, 박재홍*****
Xu Jian, JeungPyo Lee, Jaechul Lee, Ealgoo Kim, Jaehong Park

Abstract - Nowadays the automotive circuit design becomes more complicated a practical modern car circuit usually contains thousands of wires. So the error connection between connector and pins becomes more difficult to be located. This paper proposes a general way to locate all error wires in an automotive circuit design. Firstly, we give an exact definition of error wire to guide our job. This definition also composes the core part of our algorithm. Then we limit the area of the error wires by several steps. During these steps, we apply breadth-first search method to step over all wires under consideration of reducing time cost. In addition, we apply bidirectional stack technique to organize the data structure for algorithm optimization. This algorithm can get a result with all error wires and doubtful wires in a very efficient way. The analysis of this algorithm shows that the complexity is linear. We also discuss some possible improvement of this algorithm.

Key Words :circuit design, error wire locating, breadth-first, bidirectional stack

1. Introduction

This paper's industrial background is to detect error wires in a complicated car model design figure. In modern car, there are thousands of wires in the circuit, if there is some problem with even one single one, it may cause the whole system to be unable to run properly.

Among all the possible errors, two of them take a great percentage, which is missing wire and wrong wire. In this paper, we will focus on the second error type - wrong wire, which means some wires should not be there and their existence cause some degree of error to the whole system.

Due to its complexity, this kind of searching job should be done in some effect way, which requires good performance in time cost. After carefully studied the topological characteristic of some typical circuit figure, I successfully solved this problem in an effect way by applying the breadth-first algorithm [1] with bidirectional stacks [2]. Our solution is proved to work well in a large-size case from a wiring company.

The basic idea of this method is to remove correct wires out of consideration step by step until we can locate the error wire in a small area. Generally, there are four steps: get all single wires, get all connecting wires, get all end-to-end wires, and get error wires.

In the first step, we make a set of all existing single wires to be filtered in the following steps. So all the possible error wires are included in the complete set.

In the second step, we use the stack of single wires to make a new stack of connecting wires by referring to the layer stack and the figure. The connecting wire means multiple continuous wires.

In the third step, we remove all the connecting wires that do not connect both points that are in one net in target net. So we get all the unique connecting wires that make the whole net connection.

In the final fourth step, we apply the exact definition of error wires to find out which single wires are error wires.

Through the four steps, we can efficiently get all the error wires within a linear complexity. In the Chapter 4 of this paper, I will give a detail analysis of the performance.

This paper is structured as follows: section 2 describes the problem model and gives the definitions of error wires section 3 describes the solution of applying breadth-first search algorithm with bidirectional stack; section 4 makes some analysis on this method; section 5 makes a conclusion of the paper and shows the future work.

* 서건 : 서울大學 전기공學科 碩士課程
** 이정표 : 서울大學 전기공學科 碩士課程
*** 이재철 : 서울大學 전기공學科 碩士課程
**** 김일구 : 서울大學 전기공學科 博士課程
***** 박재홍 : 서울大學 전기공學科 敎授 · 工博

2. Problem Model

2.1 "Error Wire" Problem in Circuit Industry

Nowadays, many electric engineers are facing one common problem that many small error wires exist in a complicated circuit design figure.

Figure 1 shows a part of real circuit figure. There are many wires connected in the figure, making it very complicated.

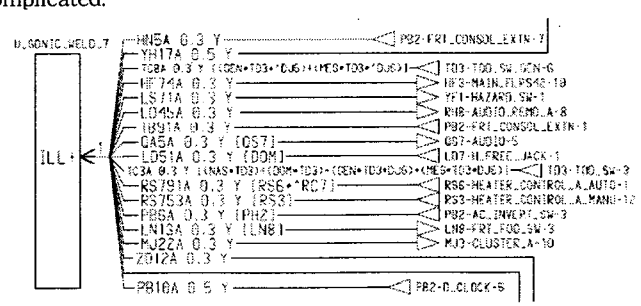

Figure 1. A Part of Real Modern Circuit Design Figure

From Figure 1, it is easy to understand that in such complicated handmade circuit error wires are very possible to exist. So the problem is to find out where are the error

wires.

2.2 Problem Model

A general problem model contains such parts: answer net and target net, connector and pin, wire. [3]

Many connectors and pins are connected together by wires, making several independent nets. Each net is separated from other nets. Answer net and target net are two lists of nets.

Answer net is ideal design without any error. We need to do is to point out all error wires in target net comparing it to answer net. If all jobs are done, the target net should be exactly same as answer net.

2.3 Model Component in Detail

As shown in Figure 1, one net is a collection of connectors and pins which are connected by wires.

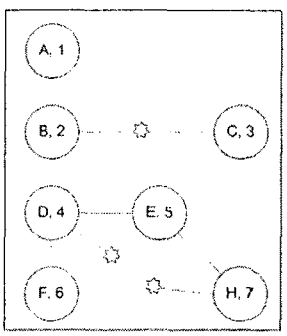

Figure 2. Three Nets

"A ~ H" mean several connectors, "1 ~ 7" mean several pins. One pair of connector and pin makes one point in the figure.

The connecting lines mean wires.

All the points connected by wires make one independent net. One separate point can also make one net.

The little stars mean some other points we don't care. Their existence is just for indicating the corresponding two points are not directly connected.

Answer net and target net are two lists of independent nets. The difference of there two net lists are: answer net is the correct standard nets list, while target net may have some error wires (the definition of error wire will be given later) in it. In fact, what we need to do is to find out all the error wires in target net by comparing it to answer net.

Connector and pin make one point in the connection figure. One connector may have (with very high possibility) more than one pins, for example, "A 1" and "A 2". But in our research job, we are going to regard these two points as totally different ones, which means we suppose there is no connection between these two points unless under special indication.

Wire is our main research object. Two points can be connected by one single wire, and can also be connected by multiple connected wires. If some connection should not exist, we think there is an error. If error exists in multiple wires situation, we are going to find out the exact error wire instead of considering all the involved wires are error wires.

Here we are going to give exact definition of error wire.

Definition of Error Wire

One wire is called error wire when and only when it matches both of the two conditions:

Condition 1: It is not included in correct wires.

Condition 2: It is directly connected to correct wire.

2.4 Problem Analysis

The main difficulty in this problem comes from two facts. One is that the connection of net can be very complicated figure, so we have to find some algorithm to step over all the points and also to meet the time requirement. The other is that we need to point out the exact error wire instead of suppose all the connected wires to be error ones curtly.

3. Solution of Applying Breadth-first Algorithm with Bidirectional Stack

3.1 Get All Single Wires by Stepping Whole Figure

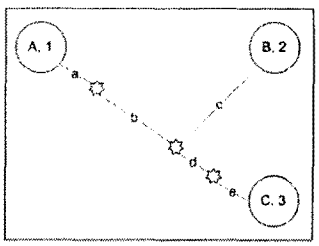

Figure 3. Example net

Take Figure 3 for example. In this figure, there are 3 points connected by 5 single wires.

In addition, we have the answer net of (A1, B2, D4) and target net of (A1, B2, C3).

Firstly, we use breadth-first search algorithm to step over all single wires, and we use two bidirectional stacks to store those wires, one for the wire names, and one for their layer numbers.

Therefore, we get the following two bidirectional stacks shown in Figure 4.

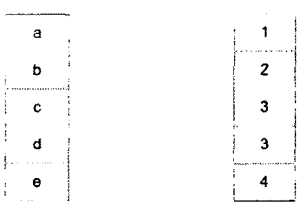

Figure 4. Two bidirectional stacks

Left stack is for wire names, right stack is for wires' layer numbers.

3.2 Get All Connecting Wires by Referring to Two Bidirectional Stacks

The next step is to get all connecting wires from the two bidirectional stacks.

First let's consider the layer stack. From highest layer - layer "1", we get the wire "a". We search its following layer, and we only find "b" with layer 2, so we put "ab"into a new stack.

Then let's consider the second high layer - layer "2". By searching its following layer, we find two wires: "c" and "d" with layer 3, so we put both "abc" and "abd" into the stack.

Then let's consider the third high layer - layer "3". There are two wires, so we do it one by one. To "c", we find it has already reached its end, so we skip it. To "d", we get "abde". Similarly, we put "abde" into the stack.

Then the lowest layer - layer "4". There is only "e", and we find it has reached its end, so we skip it. This time we have nothing to be put into the stack.

At last we get all the connecting wires and store them into a new stack as shown below in Figure 5.

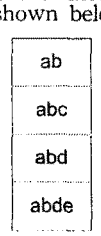

Figure 5. Stack of all connecting wires

3.3 Get All End-to-end Wires By Checking The Connecting Points in Target Net

This step is to remove those wires which are not connecting two points in the target net. After this step, all remaining wires are directly connecting two points, so

they are all objects to be studied.

By checking the connecting points in target net, we find that "ab" and "abd" are not what we want. Finally we get the follow stack as shown below in Figure 6.

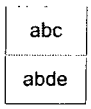

Figure 6. All end-to-end wires

3.4 Get Error Wires by Their Definition

Now we consider the remaining wires after 3 steps filtration. By comparing the target net with the answer net, we can easily find that (C, 3) is a wrong point. Therefore, the wire that is used to connect (C, 3) into the net is wrong. From the stack, we find wire "abde" has some problem inside, which "abc" is totally correct wire.

But according the definition of error wire, we need to find the exact wire in the wires path. We can just directly apply the definition.

From condition 1, we know that "a" and "b" can not be error wires, because they are included in correct wires "abc". Now only "d" and "e" are left.

Form condition 2, we can not consider "e" to be error wire. Because "d" is directly connected to correct wire "b", and "e"has no direct connection to any known correct wires. So we get the final answer of the example - wire "d".

3.5 Some Additional Explanation

In the previous section, we pointed out that "e" can not be considered as an error wire. To make it can be understood more clearly, I add another point (E, 5) to and a wire "f" to connect it to the point (C. 3), as shown in Figure 7.

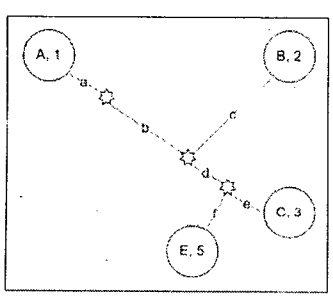

Figure 7. Example Net with One Additional Point

Also the answer net changed from ((A1, B2, D4), (C3, E 5)), and the target net changed from (A1, B2, C3, E5).

Now you may notice that there are two independent nets in answer net, which means "e"is also used to connect (C, 3) and (E, 5). So in this case, if we are considering only the net of (A1, B2, D4) without knowing other net, we cannot guarantee that "e" is an error wire.

In practical case, we usually process nets from net list one by one. So we don't know how the wrong points will be connected in the following net. If we simply consider all the wires that are not included in correct wire to be wrong, it may turn out to be a mistake. However, if we make our criteria stricter, we can avoid such mistake. That's why we add condition 2 to the definition of error wire.

4. Algorithm Analysis

Breadth-first algorithm on a graph with M vertices and N edges takes O (M + N) time, which determines the complexity of this method is O (M + N).

Figure 8 shows how time cost increases when the scale of figure increases. From the figure, we can see that the line is almost linear, which matches our complexity analysis of O (M + N).

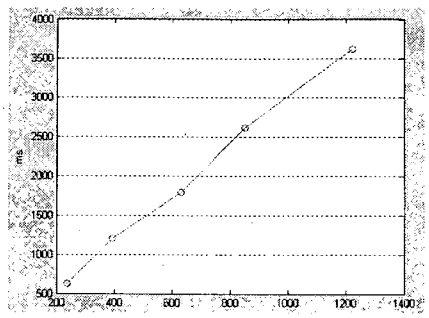

Figure 8. Program Performance Analysis

The X value is the number of points (connector and pin) plus the number of wires, and the Y value is the time cost (in millisecond).

There is still some improvement can be made which is left for future research.

The algorithm steps can be optimized according to different case [4]. For example, if in the real case the points are usually connected by many continuous wires, the depth-first search algorithm can be considered to replace breadth-first one [5].

In the given example, the start point (A, 1) is chosen randomly. We believe that if start point is carefully chosen after some further research on optimizing the stack, the algorithm complexity can be reduced even more.

5. Conclusion

Breadth-first search algorithm with bidirectional stack can be easily programmed by many common languages, for example, Visual Basic, Visual C++. To detect error wires in a complicated figure, an effective programmable algorithm is much more useful than those ones that can not be programmed.

As we know, the breadth-first search algorithm is very efficient. Inheriting this attribute from it, this specified algorithm can achieve a high performance comparing with other ones without adopting breadth-first search algorithm.

Limited by time and research domain, I only focus on how to apply it to detect error wires. Because of the flexibility of this algorithm, it may also have a great developing space in other field where the basic topological instinct is same.

Reference

[1] Zhou, R., and Hansen, E. 2006. Breadth-first heuristic search. Artificial Intelligence 170(4 - 5):385 - 408.

[2] V. Senk and P. Radivojac, "The Bidirectional Stack Algorithm," Proceedings of the 1997 IEEE International Symposium on Information Theory, ISIT'97, p. 500, Ulm, Germany, July 1997.

[3] Zhou, R., and Hansen, E. 2003a. Sparse-memory graph search. In Proceedings of the 18th International Joint Conference on Artificial Intelligence (IJCAI-03), 1259 - 1266.

[4] R. M.Haralick and G.L. Elliott. Increasing tree search efficiency for constraint satisfaction problems. Artificial Intelligence, 14:263-313, 1980.

[5] Awerbuch, B. (1985) A new distributed depth-first search algorithm. Inform. Process. Lett., 20, 147 - 150.