Fault Detection of Hierarchical Networks with Probabilistic Testing Algorithms

Keqin Li[1] and David Lee[2]

[1] Bell Labs Research, Lucent Technologies
[2] Department of Computer Science and Engineering, The Ohio State University

Abstract. As communications networks are expanding to larger areas the control and maintenance of routing information are becoming a formidable task. To cope with its size and complexity and to make the network reliable and scalable hierarchical network has been proposed with new features to support the information infrastructure. However, the network hierarchy adds more complications to the network design and implementations and that hampers the network reliability and quality of services. Conformance testing is known to be a powerful tool for network fault detection yet most of the works in the published literature are on networks without hierarchy. We present probabilistic algorithms for testing hierarchical networks along with the added features. Based on a formal model of the networks, we provide a formal analysis that shows that our probabilistic algorithms guarantee a high fault coverage with a feasible number of tests. To further reduce the number of tests we identify test equivalence classes and that enables us to significantly reduce the number of tests yet without losing the fault coverage. Experimental results on Internet OSPF protocol are reported.

1 Introduction

Networks are indispensable for our daily communications, including PSTN (Public Switched Telephone Network), ATM, wireless, and Internet. With the expanding networks and new and sophisticated services, which are demanded by the user applications, the networks become more complex, and their reliability and scalability pose a challenge yet are essential for the QoS (Quality of Services). For the reliability of a large network, a key function is to ensure correct routing of information, and, consequently, routing protocols play a critical role. In this work we investigate conformance testing of routing protocols that checks whether an implementation of a routing protocol conforms to its specification.

As networks grow in size, the control and maintenance of routing information become difficult if not impossible. In order to improve the scalability of routing systems, hierarchy is introduced into networks [10] where large networks are partitioned into several subdomains. The routing information within a subdomain is first aggregated and then shared with other subdomains; the detailed internal network structure of a subdomain is hidden from each other while networking devices in different subdomains are still reachable from each other. Hierarchy in PSTN [18], PNNI [2] in ATM network, and OSPF [15] [16] in Internet are typical hierarchical mechanisms in networks.

The telephone networks worldwide are classical hierarchical routing networks [18]. Telephone-switching offices or exchanges are classified according to their level in a hierarchy. Routing in PSTN is performed as follows. When a call is coming, the switch checks its routing database to match the prefix of the destination phone number. If there is a match, the call is routed to the next switch. Otherwise, the call is routed to the higher-level switch/exchange. When the call arrives at the destination switch, the suffix of the number is checked for ringing the callee phone.

The PNNI (Private Network-Network Interface) [2] protocol provides mechanisms to support scalable, QoS-based ATM routing and switch-to-switch Switched Virtual Connection (SVC) interoperability. To create a PNNI network hierarchy, ATM switches at the lowest hierarchical level can be organized into multiple peer groups, each of which elects a Peer Group Leader (PGL) and its parent node becomes active. The purpose of the active parent node, or Logical Group Node (LGN), is to represent the entire peer group to other LGNs. Within each peer group, all nodes exchange complete topology database information among them. However, the LGN reduces the amount of information shared with other peer groups by sending only a limited amount of aggregated information to its neighbor LGNs, which in turn flood that information down to all nodes within their child peer group.

In order to improve scalability, a two level hierarchy is proposed in OSPF [15] [16], which is a widely used routing protocol in the Internet. An Autonomous System (AS) is divided into areas. Each area has been assigned an area ID and contains a group of routers, called Internal Routers. In order to avoid routing loops, these areas are organized in a hub-and-spoke structure. Area 0 is the backbone area and all the other areas attach to area 0 by one or more Area Border Routers (ABR). Routers in area 0 is at level 1 and all the other routers at level 2; it is a two-level hierarchy.

Routers in one area operate as if there is no hierarchy imposed. The routers originate and exchange LSAs (Link State Advertisement) which contains the topology update information so that each router has an identical Link State Database (LSDB), which represents the topology of the area for routing table computation. In each routing table entry, destination, cost to the destination and nexthop are specified. Note that the destinations of routing table entries are all in this area. Since an ABR belongs to multiple areas, logically there is one routing table entry for each area to which an ABR belongs.

In order to make the destinations in one area at level 2, e.g., area 1, reachable for routers and hosts outside the area, Summary-LSAs are originated and advertised outside the area by ABRs. The main fields of Summary-LSA are destination and metric to the destination. For each entry of area 1's routing table, a Summary-LSA is originated, in which the destination field is the destination of the routing table entry, and the metric field is the cost. This procedure is referred to as summarization.

In order to reduce the control traffic, a procedure referred to as aggregation is used. In an ABR of an area at level 2, e.g., area 1, several address ranges can be configured, and each address range can cover several entries in a routing table. For each address range, the ABR originates one Summary-LSA, instead of several Summary-LSAs from these routing table entries.

Upon receiving a Summary-LSA originating by an ABR, the router performs inter-area route calculation. One inter-area routing table entry is generated for each Summary-LSA received. In the routing table entry, the destination is the one described in the Summary-LSA, the cost is the sum of cost to the ABR and the metric specified in the Summary-LSA, and the nexthop is the same as the nexthop to the ABR.

In summary, when an AS is divided into areas and ABRs are designated, the following additional operations are performed by routers in the AS:

- Summary-LSA origination, performed by ABRs; and
- Inter-area route calculation, performed by every router.

To establish hierarchy in networks, new features are added into routing systems of networking devices. For example, when a hierarchy is structured in OSPF, router needs to originate a new type of Link State Advertisement (LSA) and perform different routing table calculations. These new features and operations are essential for the reliability of hierarchical networks. On the other hand, the implementations of the hierarchical OSPF are rather complex, and practical experiences show [15] that the hierarchy of OSPF is also a source of implementation faults and that often leads to the degradation of Internet performances. Consequently, their conformance testing is essential for the correct implementations of the OSPF routing protocol.

Currently, most testing tools conduct a test of routers in an isolated environment and check the conformance of router's behaviors in accordance with RFCs. Available commercial tools include Agilent RouterTester [1], Spirent AX/4000 and SmartBits TeraRouting Tester [19] and IXIA IxANVL [9]. A main function of these tools is to generate a set of tests corresponding to each of the requirements in the design/RFC for "typical" network configurations/topologies. Most of these tools also test on the hierarchical features of OSPF, yet in an ad hoc way and on a static network environment.

We study conformance testing of hierarchy features of routing protocols of networking devices. We propose probabilistic testing algorithms on routers connected to the networks and in a dynamic environment. Furthermore, we study test equivalence class of network configurations to selectively test representative configurations; we can significantly cut down the configurations to test yet without sacrificing the fault coverage. We provide a formal analysis of the fault coverage of our probabilistic algorithms and show that a high fault coverage can be guaranteed yet with a reasonable testing cost.

Given its importance in the current Internet, the testing of hierarchy of OSPF is our focus of investigation, and we take it as a case study of our general theory.

The rest of the paper is organized as follows. In Section 2, we describe a formal model of hierarchical networks and the basics of conformance testing with OSPF as a case study. We then discuss in Section 3 equivalence classes of network topologies and present our probabilistic algorithms for testing the hierarchy features with a fault coverage analysis. Experimental results on Cisco and Zebra OSPF implementations are reported in Section 4.

2 Conformance Testing and Modeling

In recent years, there are a lot of activities in the area of protocol conformance testing. We only mention a few related publications here. For instance, [17] highlights works in the area of algorithmic test generation from formal specifications with fault model-driven test derivation, [4], [5] and [6] describe methods for testing real time systems with fault coverage analysis, and [3] and [12] contain a survey.

As an important and complex routing protocol, testing of OSPF has been studied [8] [21], however, the approaches are on OSPF protocols without hierarchy. In this case, a bipartite graph $G_b =< R, W, E_b >$ is used to model the network in which RUT (Router Under Test) locates. As in Figure 1, Router Nodes in the set R represent routers in the network, and Network Nodes in the set W are used to model networks or LANs (Local Area Network). Edges in set E_b connect Router Nodes and Network Nodes, i.e., a router is in a LAN. A router can be in more than one LAN and a LAN can contain more than one router. For the completeness of tested network configurations and topologies, the number of Network Nodes $m = |W|$ has to satisfy $m = \lfloor \frac{n}{2} \rfloor \lceil \frac{n}{2} \rceil$ where $n = |R|$ is the number of Router Nodes [8].

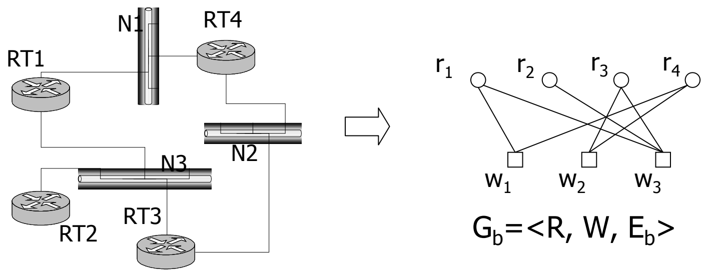

Fig. 1. Basic Model

In order to test hierarchical OSPF, a two-level bipartite graph can be used to model the network in which RUT locates. It can be considered as an extension of the basic model in Figure 1.

At the first level, a bipartite graph $G_0 =< R_0, W_0, ABR, E_0 >$ is used to model the backbone area 0 as in Figure 2 where

- $R_0 = \{r_x^{(0)} | x = 1, 2, \cdots, n_0\}$ is the set of $n_0 = |R_0|$ Internal Router Nodes in area 0.

- $W_0 = \{w_x^{(0)} | x = 1, 2, \cdots, m_0\}$ is the set of $m_0 = |W_0|$ Network Nodes in area 0.
- $ABR = \{abr_x | x = 1, 2, \cdots, k\}$ is the set of $k = |ABR|$ ABR Nodes in area 0.
- E_0 is the set of edges connecting router and network nodes.

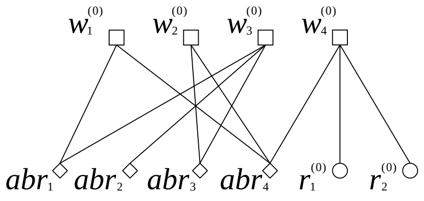

Fig. 2. The First Level Bipartite Graph

For instance, internal router nodes $r_1^{(0)}$ and $r_2^{(0)}$ are connected together with ABR router node abr_4 by network (LAN) node $w_4^{(0)}$; and ABR nodes abr_3 and abr_4 are connected by network node $w_2^{(0)}$.

A second level bipartite graph $G_i =< R_i, W_i, ABR_i, E_i >$ is used to model a non-backbone area $i (i \neq 0)$. In Figure 3, area 1, which connects to abr_1 in Figure 2, is expanded as an example.

- $R_i = \{r_x^{(i)} | x = 1, 2, \cdots, n_i\}$ is the set of $n_i = |R_i|$ Internal Routers in area i.
- $W_i = \{w_x^{(i)} | x = 1, 2, \cdots, m_i\}$ is the set of $m_i = |W_i|$ Network Nodes in area i.
- ABR_i is the set of $b_i = |ABR_i|$ ABR Nodes in area i. In OSPF, all ABRs must attach to the backbone area, thus, $ABR_i \subseteq ABR$.
- E_i is the set of edges connecting router nodes and network nodes in area i.

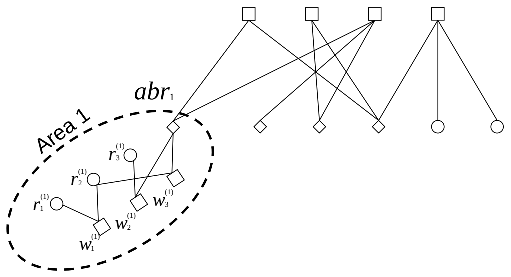

Fig. 3. A Second Level Bipartite Graph

When all the ABR Nodes are expanded the whole two-level bipartite graph is in Figure 4. Note that more than one ABR router can be in a same area.

Suppose that there are l non-backbone areas. Then, there are n_0 internal non-ABR router nodes in the backbone area 0, $\sum_{i=1}^{l} n_i$ internal non-ABR router nodes in the l non-backbone areas at the second level, and $\sum_{i=1}^{l} b_l = k$ ABR routers.

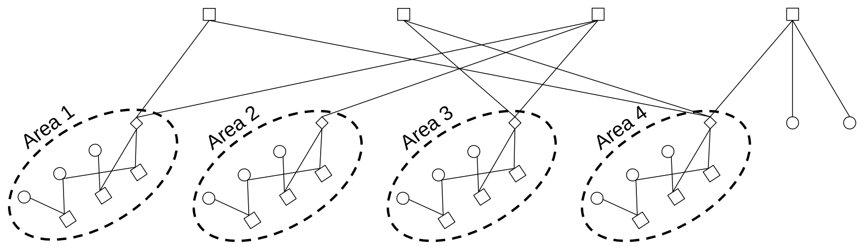

Fig. 4. The Two-Level Bipartite Graph Model

For clarity, we make the following assumptions.

No Range Assumption *An ABR only advertises metrics to Internal Router Nodes in the areas to which it attaches, i.e., no address range is configured at ABR.*

Link Cost Assumption *All the link costs from Router Node to Network Node are 1, and all the link costs from Network Node to Router Node are 0.*

Single Entry Assumption *There is only one ABR per non-backbone area.* Note that the above assumptions are only for clarity and that they can be relaxed or modified with due changes in our algorithms.

With these assumptions, the whole hierarchical network contains k ABR Nodes, the number of non-backbone areas is also k, i.e., $l = k, b_i = 1 (1 \leq i \leq k)$, the total number of routers is $\sum_{i=0}^{k} n_i + k$, and the total number of networks is $\sum_{i=0}^{k} m_i$.

Based on the formal model, we now present our probabilistic testing algorithms.

3 Probabilistic Testing Algorithms

In order to test the behaviors of the RUT in dynamic environment in a structured manner, we need to generate all the possible network topologies, and, taking each network topology as a test case, we check the RUT. However, it is formidable to generate and test all the possible topologies. We need to reduce the number of test cases without losing the fault coverage. We achieve this by the following two approaches: identifying test equivalence classes and randomization.

A randomized algorithm is an algorithm that uses random numbers to influence the choices it makes in the course of its computation. Once viewed as a tool in computational number theory, it has by now found widespread applications, fueled by the two major benefits of randomization: simplicity and speed. Randomized algorithms have been used for conformance testing. For instance, [11] applies a random walk for fault detection. For many applications, a randomized algorithm is the fastest algorithm available, or the simplest, or both [7]. [13] and [14] provide a comprehensive introduction survey of paradigms underlying randomized algorithms.

We are focused on testing the hierarchy of OSPF. Since Summary-LSA origination and inter-area route calculation are two new functions/features from the hierarchy of OSPF, we conduct tests on their implementations and analyze the fault coverage. According to the different positions and hence functions of RUT in an AS, different testing algorithms are needed, as enumerated in Figure 5:

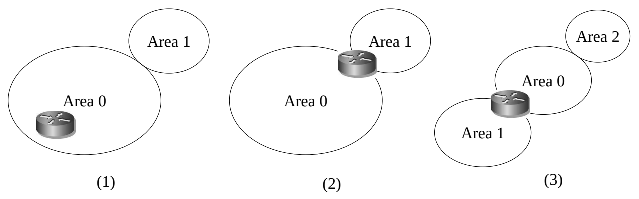

Fig. 5. Positions of RUT

1. RUT is an internal router. In this case, RUT receives Summary-LSA and performs inter-area route calculation. When RUT is an internal router in a non-backbone (level 2) area, it receives Summary-LSAs originated by one ABR. And when it is in the backbone (level 1) area, it receives Summary-LSAs originated by multiple ABRs. Thus, the testing algorithm for internal router in the backbone (level 1) area can test internal router in non-backbone (level 2) area. In the following, we only consider the case in which RUT is an internal router in the backbone (level 1) area.
2. RUT is an ABR and there is only one non-backbone (level 2) area. In this case, RUT originates Summary-LSA, but no inter-area route calculation is performed.
3. RUT is an ABR and there are two or more non-backbone (level 2) area. In this case, RUT performs Summary-LSA origination and also inter-area route calculation.

According to the three different cases, we design and analyze the corresponding probabilistic testing algorithms.

3.1 Internal Router in Backbone Area 0

In this subsection, we consider the case when RUT is an Internal Router in the backbone area 0. In this case, RUT receives Summary-LSA and performs inter-area route calculation. We first identify equivalent Summary-LSAs advertised by ABRs; we only need to test on one Summary-LSA among each equivalence class and that significantly reduces the number of tests. We then present a probabilistic testing algorithms to test the inter-area route calculation by RUT with a fault coverage analysis. Recall that in this case the following functions of an ABR have to be tested. An RUT in area 0 receives Summary-LSAs originated by ABRs, and calculates/updates inter-area routes accordingly. Therefore, the inter-area route calculation is to be checked. We only need to consider ABRs, which are reachable from RUT.

Since an ABR summarizes the topology of a non-backbone area by Summary-LSAs, the topologies of this non-backbone area are invisible to RUT. The inter-area route calculation is conducted by the combination of the known topology of area 0 and Summary-LSAs originated by ABRs.

Under the *No Range Assumption*, ABR_i advertises metrics to the internal routers $r_1^{(i)}, r_2^{(i)}, \cdots, r_{n_i}^{(i)}$ in area i where $(1 \leq i \leq k)$. These Summary-LSAs can be taken as a Distance Vector $v^{(i)} = (v_1^{(i)}, v_2^{(i)}, \cdots, v_{n_i}^{(i)})$, advertised into area 0. For a reachable Internal Router Node $r_j^{(i)}(1 \leq j \leq n_i)$, $v_j^{(i)}$ is the length of the shortest path from ABR_i to this Router Node. Under the *Link Cost Assumption*, it is obvious that $v_j^{(i)} \geq 1(1 \leq i \leq k, 1 \leq j \leq n_i)$.

In order to generate all the possible topologies in area i, the number of LANs in this area is at least $m_i = \lfloor \frac{n_i+1}{2} \rfloor \lceil \frac{n_i+1}{2} \rceil$. It is obvious that $m_i \geq n_i (n_i \geq 1)$. Obviously, the longest path from ABR_i to an Internal Router is n_i. As a convention, if $r_j^{(i)}$ is not reachable from ABR_i, we set $v_j^{(i)} = \infty$.

Proposition 1. *If an internal router node $r_j^{(i)}$ is reachable from ABR_i, then $1 \leq v_j^{(i)} \leq n_i (1 \leq i \leq k, 1 \leq j \leq n_i)$. Otherwise, $v_j^{(i)} = \infty$.*

Consider the distance vector advertised by ABR_i from area i, i.e.,
$v^{(i)} = (v_1^{(i)}, v_2^{(i)}, \cdots, v_{n_i}^{(i)})$. We construct $u^{(i)} = (u_1^{(i)}, u_2^{(i)}, \cdots, u_{n_i}^{(i)})$ where
$u_j^{(i)} = \begin{cases} 1, & \text{if} v_j^{(i)} < \infty \\ \infty, & \text{if} v_j^{(i)} = \infty \end{cases} \quad 1 \leq j \leq n_i.$

Then we construct $\overline{u^{(i)}}$, which is obtained by permuting components of $u^{(i)}$.

Since we only consider ABR_i's, which are reachable from RUT, its cost (path length) is finite x. Therefore, the n_i internal nodes in area i correspond to n_i entries in the routing table of RUT, and their costs are $x + u_j^{(i)}, j = 1, \cdots, n_i$, respectively. It is natural to assume that the correctness of the routing table computation of RUT is not affected by the permutation of the internal nodes in an area i, which correspond to the identical n_i entries in the routing table; we claim that they are equivalent.

In summary, for each area i, we have a set of $n_i + 1$ Characteristic Distance Vectors, each of which represents an equivalence class of distance vectors:

$\left\{ \underbrace{(\infty, \infty, \cdots, \infty)}_{n_i}, \underbrace{(1, \infty, \cdots, \infty)}_{n_i}, \underbrace{(1, 1, \cdots, \infty)}_{n_i}, \cdots, \underbrace{(1, 1, \cdots, 1)}_{n_i} \right\}$. For each ABR_i, we only need to test these n_i+1 vectors. However, there are still $\prod_{i=1}^{k} n_i$ possible combinations of Characteristic Distance Vectors to test, and it is impossible to test on each of them in real OSPF networks. We apply probabilistic algorithms [8] with the following constants, parameters and variables:

1. k (input parameter): number of non-backbone areas;
2. $n_0, n_1, n_2, \cdots, n_k$ (input parameters): number of internal routers in area 0, 1, 2, $\cdots$, k respectively;
3. $0 \leq p_1, p_2, p_3, p_4 \leq 1$ (input parameters): probability of edge insertion, node insertion, edge deletion, and node deletion, respectively in area 0; $p_1 + p_2 + p_3 + p_4 = 1$.
4. $G_0 =< R_0, W_0, ABR, E_0 >$ (variable): topology graph of area 0 with internal router nodes R_0, network nodes W_0, set of ABR nodes ABR, and edges E_0;
5. $v^{(1)}, v^{(2)}, \cdots, v^{(k)}$ (variable): distance vectors advertised by ABR_1, ABR_2, $\cdots$, ABR_k, respectively, into area 0;
6. $v_0 \in R_0$ (constant): router under test.

Algorithm 1
Input: $k, n_0, n_1, n_2, \cdots, n_k, 0 \leq p_1, p_2, p_3, p_4 \leq 1$
Output: implementation fault in hierarchy of OSPF or conformance

```
1. repeat
2.     Construct initial network topology graph $G_0$ with
       $R_0 = \{v_0\}$, $W_0 = ABR = E_0 = \phi$;
3.     while ($G_0$ is not a complete graph)
4.         UPDATE($G_0$);
5.         if ROUTE($G_0$) = FALSE;
6.             return "faulty";
7.     end-while
8. end-repeat
9. return "conforms"
```

The algorithm is probabilistic in nature. Line 2 constructs an initial network topology graph G_0 of area 0 with only one router node: v_0 (RUT). The while-loop from Line 3 to Line 7 continues until a complete bipartite graph is obtained. Subroutine UPDATE(G_0) in Line 4 gets a new network topology of area 0. Subroutine ROUTE(G_0) in Line 5 generates distance vectors advertised by ABRs, and checks LSDB and routing table of RUT. If any faults are detected, the process is aborted and "faulty" is reported in Line 6. Otherwise, after sufficient repetition of the repeat-loop from Line 1 to Line 8, "conforms" is declared in Line 9 with a good confidence in the topologies and router behaviors that have been tested.

Subroutine UPDATE(G_0)

In Algorithm 1, while-loop is repeated until network topology graph of area 0 becomes a complete bipartite graph. Each repetition of the loop runs the subroutine UPDATE(G_0) in Line 4, which updates G_0 incrementally.

```
Subroutine UPDATE($G_0$)
Parameters: $n_0, m_0, 0 \le p_1, p_2, p_3, p_4 \le 1$
Variables: $G_0 =< R_0, W_0, ABR, E_0 >$

 1.  switch($p$)
 2.      case '$p_1$': if( $|E_0| < (|R_0| + |ABR|) * |W_0|$ )
 3.          /* graph is not complete */
 4.          insert an edge u.a.r. in $E_0$;
 5.      case '$p_2$': if( $|R_0| + |ABR| + |W_0| < n_0 + m_0$ )
 6.          /* nodes below upper bounds */
 7.          insert a node u.a.r. in $R_0 \cup ABR \cup W_0$;
 8.          Add all related physical links to $E_0$.
 9.      case '$p_3$': if( $|E_0| > 0$ )
10.          /* edge set is not empty */
11.          delete an edge u.a.r. from $E_0$;
12.      case '$p_4$': if( $|R_0| + |ABR| + |W_0| > 1$ )
13.          /* node set is not empty */
14.          delete a node u.a.r. from $R_0 \cup ABR \cup W_0$;
15.          Remove all related physical links from $E_0$;
16.  return
```

For a network topology, one of the four operations on edge or node insertion or deletion is performed with probability $0 \le p_1, p_2, p_3, p_4 \le 1$. We can partition the unit interval into four subintervals $I_1 = [a_0, a_1), I_2 = [a_1, a_2), I_3 = [a_2, a_3), I_4 = [a_3, a_4]$ with $|I_1| = p_1, |I_2| = p_2, |I_3| = p_3, |I_4| = p_4$. We then sample uniformly at random (u.a.r.) in the unit interval and obtain $0 \le p \le 1$. We then "switch" on the value of p in Line 1. Specifically, depending on $p \in I_i, i = 1, 2, 3, 4$, one of the cases is executed at Line 2, 5, 9, or 12. This subroutine is similar to the corresponding one described in [8] and we omit the details.

Subroutine ROUTE(G_0)

In Algorithm 1, once G_0 is updated, the subroutine ROUTE(G_0) is called. This subroutine has the following functions:

1. Generating distance vectors advertised by ABRs into area 0;
2. Calculating routing update information, i.e., Link State Update packets (LSU packets), and sending to RUT;
3. Obtaining LSDB and routing table from RUT, and checking correctness.

Function 2 and 3 are similar to the corresponding ones described in [8]. The subroutine ROUTE(G_0) is described in the following where Line 4 is for Function 2 and 3.

Subroutine ROUTE(G_0)

```
1.  for $i \leftarrow 1$ until $k$ do
2.          generate 1 possible value of $v^{(i)}$ u.a.r.;
3.  for each possible combination $< v^{(1)}, v^{(2)}, \cdots, v^{(k)} >$ do
4.          <Function 2 and 3>
```

In this subroutine, the one vector of area i is generated uniformly at random. Thus, the calculation and correctness checking operations are performed only once for each topology in area 0.

Combining Distance Vector

Recall that we only take into account ABRs which are reachable from RUT and there are k of them. For ABR_i,$i = 1, \cdots, k$, there are n_i distinct Characteristic Distance Vectors, which we have to test on, and there are a total of $\prod_{i=1}^{k} n_i$. It can be shown that each of them is to be tested by Algorithm 1 with a non-zero probability.

Fault Coverage

The inter-area route calculation/update is performed upon receiving each Summary-LSAs, and a reasonable fault model is that a calculation, which is based on a specific Summary-LSA, is performed incorrectly. This is often referred to as a single-fault model. A single fault involves an ABR Node ABR_x that is reachable from RUT and an internal router $r_y^{(x)}$ of area x. When ABR_x advertises a Summary-LSA destined for $r_y^{(x)}$ to area 0, RUT calculates inter-area route to $r_y^{(x)}$ incorrectly.

We present the following result on fault coverage. Due to space limit we omit the proof.

Proposition 2. *There exists a polynomial $P(k, n_0, n_1, \cdots, n_k)$ such that for any $0 < \varepsilon \leq 1$, with no more than $P(k, n_0, n_1, \cdots, n_k) \ln \frac{1}{\varepsilon}$ repetitions of the repeat-loop in Algorithm 1, any single-fault is to be detected with a probability at least $1 - \varepsilon$.*

It shows that with a polynomial number of tests Algorithm 1 detects any single fault with a high probability.

3.2 Area Border Router: Only One Non-backbone Area

In this subsection, we consider the case when RUT is an ABR and there is only one non-backbone level 2 area, i.e., area 1. In this case, RUT originates Summary-LSA, but no inter-area route calculation is needed since there is only one level 2 area. We present a probabilistic testing algorithm to test the Summary-LSA origination feature with a fault coverage analysis. Recall that in this case the following functions of an ABR have to be tested:

– Area 0 Summary-LSA origination. RUT originates Summary-LSA from area 1 into the backbone area. This function is only determined by the topology of area 1, more specifically, by the routing table of area 1.
– Area 1 Summary-LSA origination. RUT originates Summary-LSA from area 0 into the non-backbone area 1. This function is only determined by the topology of area 0, more specifically, by the routing table of area 0.

Testing Algorithm

We present a probabilistic testing algorithm with the following:

1. n_0, n_1(input parameters): number of internal routers in area 0 and area 1;
2. $0 \leq p_1, p_2, p_3, p_4 \leq 1$(input parameters): probability of edge insertion, node insertion, edge deletion, and node deletion, respectively in area 0; $p_1 + p_2 + p_3 + p_4 = 1$.
3. v_0 (constant): router under test.
4. $G_0 =< R_0, W_0, \{v_0\}, E_0 >$ (variable): topology graph of area 0 with internal router nodes R_0, network nodes W_0, an ABR node v_0, and edges E_0;
5. $G_1 =< R_1, W_1, \{v_0\}, E_1 >$ (variable): topology graph of area 1 with internal router nodes R_1, network nodes W_1, an ABR node v_0, and edges E_1.

Algorithm 2
input: $n_0, n_1, 0 \leq p_1, p_2, p_3, p_4 \leq 1$
output: implementation fault in hierarchy of OSPF or conformance

```
1.  repeat
2.      construct initial network topology graph $G_0$ with
        $R_0 = W_0 = E_0 = \phi$;
3.      while($G_0$ is not a complete graph)
4.          UPDATE( $G_0$ );
5.          GENERATE( $G_1$ );
6.          if ROUTE( $G_0, G_1$ ) = FALSE;
7.              return "faulty";
8.      end-while
9.  end-repeat
10. return "conforms"
```

The algorithm is probabilistic in nature. Line 2 constructs an initial network topology graph G_0 of area 0 with only one ABR node: v_0 (RUT). The while-loop from Line 3 to Line 8 continues until a complete bipartite graph is obtained. Subroutine UPDATE(G_0) in Line 4 gets a new network topology of area 0. It is the similar to the one in Algorithm 1. Subroutine GENERATE(G_1) in Line 5 generates a new network topology of area 1. Subroutine ROUTE(G_0, G_1) in Line 6 checks LSDB and routing table of RUT. If any faults are detected, the process is aborted and "faulty" is reported in Line 7. Otherwise, after sufficient repetition of the repeat-loop from Line 1 to Line 9, "conforms" is declared in Line 10 with a good confidence in the topologies and router behaviors that have been tested.

Subroutine GENERATE(G_1)

In Algorithm 2, while-loop is repeated until network topology graph of area 0 becomes a complete bipartite graph. In each repetition of the loop, subroutine UPDATE(G_0) is called to updates G_0 incrementally. After that, subroutine GENERATE(G_1) is called to generate a new topology graph G_1:

```
Subroutine GENERATE( $G_1$ )

1.  Construct initial graph $G_1 =< R_1, W_1, \{v_0\}, E_1 >$ with
    $R_1 = \{r_x^{(1)} | x = 1, 2, \cdots, n_1\}, W_1 = \{w_x^{(1)} | x = 1, 2, \cdots, m_1\}, E_1 = \phi$;
2.  Choose $l$ from $[n_1 + 1..(n_1 + 1)m_1]$ u.a.r.;
3.  Insert $l$ edges u.a.r. into $E_1$.
```

In Line 1, an initial G_1 is constructed with all the nodes yet without any edges. In Line 2, the number of edges to be inserted into G_1 is determined randomly. In Line 3, these edges are inserted into G_1 randomly to obtain G_1.

Subroutine ROUTE(G_0 , G_1)

In Algorithm 2, after G_0 is updated and G_1 is generated, subroutine ROUTE(G_0 , G_1) is called to check the correctness of RUT. It is similar to that in [8] and we omit the details. Note that LSU packets are calculated and sent to RUT for both areas, and LSDBs of the two areas are obtained, respectively. Routing table of RUT is also computed based on the LSDBs. If any of them is incorrect, "faulty" is returned.

Fault Coverage

The Summary-LSA origination is performed based on routing table entries one by one, a reasonable fault model is to assume that the origination based on a specific routing table entry is performed incorrectly. Since the Summary-LSA originations of the two areas are performed at the same time, and there may be interactions of them in an implementation, we need to consider the routing table entries in both areas. Again this is a single-fault model; a single fault about Summary-LSA origination involves an internal router $r_x^{(0)}$ of area 0 and an internal router $r_y^{(1)}$ of area 1. When both of them are reachable from RUT, RUT originates one or two Summary-LSAs incorrectly.

Similar to the fault coverage analysis of Algorithm 1, we have the following:

Proposition 3. *There exists a polynomial $P(n_0, n_1)$ such that for any $0 < \varepsilon \leq 1$, with no more than $P(n_0, n_1) \ln \frac{1}{\varepsilon}$ repetitions of the repeat-loop in Algorithm 2, any single-fault is to be detected with a probability at least $1 - \varepsilon$.*

It shows that with a polynomial number of tests Algorithm 2 detects any single fault with a high probability.

3.3 Area Border Router: More than One Non-backbone Area

In this subsection, we consider the case when RUT is an ABR and there is more than one non-backbone (level 2) area, i.e., $k \geq 2$. Specifically, suppose that RUT is ABR_1 which connects the backbone area and the non-backbone (level 2) area 1. In this case, both Summary-LSA origination and inter-area route calculation are performed, since there are two or more level 2 areas. Obviously, it is a combination of the previous two cases, and we can apply both Algorithm 1 and 2 to test the two functions as follows. Initially network topology graph G_0 of area 0 is constructed with only one router node: v_0 (RUT). Then G_0 is updated until it becomes a complete bipartite graph. With each G_0, network topology G_1 of non-backbone area 1 is generated using Algorithm 2, and characteristic distance vectors $v^{(i)}, i = 2, \cdots, k$ of the other non-backbone areas are originated using Algorithm 1. We check the valid performance of RUT using Algorithm 2. On the other hand, the corresponding LSDB and routing table of RUT are also tested using Algorithm 1. Upon detecting any faults, the process is aborted and "faulty" is reported. Otherwise, after sufficient repetition, "conform" is declared. Obviously, the fault coverage of both Algorithm 1 and 2 apply, and faults in this case can be detected with a high probability in polynomial number of repetitions.

In summary:

Theorem 1. *For testing the hierarchy features of an IP router OSPF protocol with the probabilistic algorithms, any single fault can be detected with a high probability and in a number of tests that is polynomial in the size of the network.*

4 Experiments

We implemented both probabilistic algorithms in a software tool to test IP routers. For this experiment we use a software tool, Socrates. It was developed at Bell Labs [8], and can simulate IP network topologies. We further enhance the software to simulate hierarchical IP network topologies for our testing. When an RUT is connected to simulator it perceives itself is connected to a real network of IP routers and interacts as if it is a router connected with Internet, performing due operations: it exchanges messages, including LSAs, with other routers and computes routing tables with each network topology update.

In order to test the inter-area route calculation, Algorithm 1 is implemented and integrated with the simulator with the following configuration in Figure 6.

In this configuration, RUT is $r_1^{(0)}$, and $w_1^{(0)}$ and $w_2^{(0)}$ are physical networks connecting RUT and the software tool. The other routers and networks in area 0 are simulated by the software tool. For this experiment, we set $k = 1$, i.e., there is only one ABR with one non-backbone area. The distance vectors advertised by ABR_1 into area 0 are generated by the software tool.

In order to test the summary-LSA origination function, Algorithm 2 is applied, and the experiment configuration is in Figure 7.

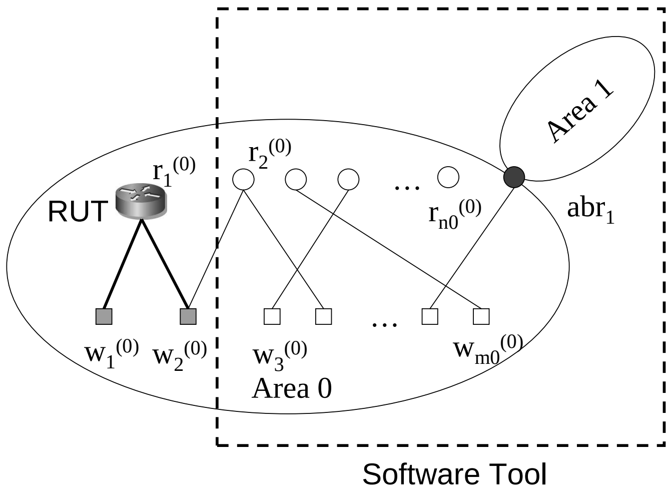

Fig. 6. Experiment Configuration 1

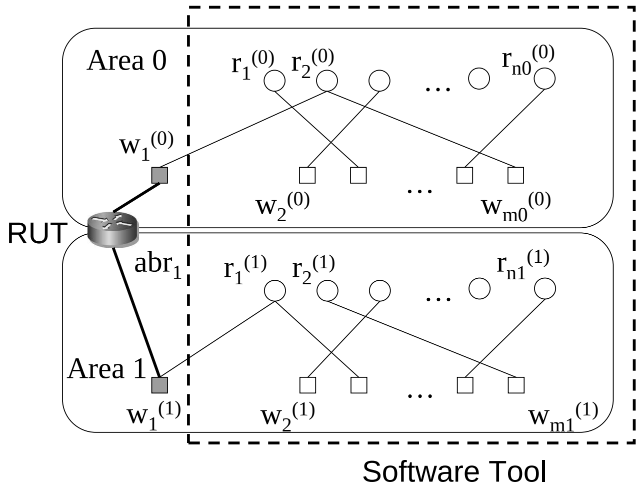

Fig. 7. Experiment Configuration 2

In this configuration, RUT is abr_1, and $w_1^{(0)}$ and $w_1^{(1)}$ are physical networks connecting RUT and the software tool. The other routers and networks in area 0 and 1 are simulated.

We tested OSPF implementations of Cisco router and Zebra [20]. We used several combinations of n_0 and n_1 in the experiments and most of the tests went well without reporting any faults.

In order to verify the fault detection capability, we intentionally introduced some errors into the implementation of Zebra. For example, one of the added errors was that during Summary-LSA origination when both $r_{n_0}^{(0)}$ and $r_{n_1}^{(1)}$ were reachable from RUT and that the modified implementation of RUT originated summary-LSAs with wrong value in the metric field. We applied our algorithms against the faulty implementation, and all the faults were detected. Specifically, for $n_0 = n_1 = 4$, after a large number of runs of the testing algorithm, the average time to detect the fault was 10.25 minutes. For $n_0 = n_1 = 6$, the average time was 22 minutes. When $n_0 = n_1 = 8$, the average time was 33.5 minutes. Note that our algorithms detected the faults in the first run of the repeat-loop before the network topology became a complete graph.

5 Conclusion

We study testing of hierarchical networks with Internet OSPF routing protocol as a case study. Due to the size and complexity of all the possible network topologies it is impossible to test on each network configuration. We discuss network topology equivalence and reduce the testing to the characteristic topology representation for each equivalence class. We then provide probabilistic algorithms for testing the hierarchy features and show that a high fault coverage can be achieved with a polynomial number of tests. The basic ideas and mechanisms can be applied to the testing of the hierarchy features of PSTN, ATM PNNI and other hierarchical networks.

We have analyzed the fault coverage with a single fault model. Apparently, multiple faults are easier to detect since they result in more violations of the network protocol specifications. However, a rigorous analysis is yet to be obtained. One of the difficulties is that different faults might ”cover up” each other, and how to model their interactions and show rigorously the fault coverage remains to be investigated.

Acknowledgement

We are indebted to Caixia Chi for part of the analysis of the fault coverage and to Ruibing Hao and Dawei Wang for their help with the experiments.

References

1. Agilent Technologies: URL=http://advanced.comms.agilent.com/routertester/
2. ATM Forum: Private Network-Network Interface Specification Version 1.1 (PNNI 1.1). 2002

3. Gregor v. Bochmann, Alexandre Petrenko: Protocol Testing: Review of Methods and Relevance for Software Testing. International Symposium on Software Testing and Analysis, August 1994, Seattle, Washington, USA
4. Abdeslam En-Nouaary, Ferhat Khendek, Rachida Dssouli: Fault Coverage in Testing Real-Time Systems. Proceedings of the Sixth International Conference on Real-Time Computing Systems and Applications, 1999
5. Ahmed Khoumsi, Mehdi Akalay, Rachida Dssouli, Abdeslam En-Nouaary, Louis Granger: An Approach for Testing Real Time Protocol Entities. Proceedings of the IFIP TC6/WG6.1 13th International Conference on Testing Communicating Systems: Tools and Techniques, 2000
6. Abdeslam En-Nouaary, Rachida Dssouli, Ferhat Khendek: Timed Wp-Method: Testing Real-Time Systems. IEEE Transactions on Software Engineering, Volume 28, Issue 11, November 2002
7. Rajiv Gupta, Scott A. Smolka, Shaji Bhaskar: On Randomization in Sequential and Distributed Algorithms. ACM Computing Surveys, Vol. 26, No. 1, 1994
8. Ruibing Hao, David Lee, Rakesh Sinha, Dario Vlah: Testing IP Routing Protocols - From Probabilistic Algorithms to Software Tool. FORTE/PSTV 2000
9. Ixia: URL= http://www.ixiacom.com/products/conformance_applications/
10. Leonard Kleinrock, Farouk Kamoun: Hierarchical Routing for Large networks Performance Evaluation and Optimization. Computer Networks, Vol. 1, No. 3, (1977) 155-174
11. David Lee, K. K. Sabnani, D. M. Kristol and Sanjoy Paul: Conformance Testing of Protocols Specified as Communicating Finite State Machines - a Guided Random Walk Based Approach. IEEE Trans. on Communications, Vol. 44, No. 5, (1996) 631-640
12. David Lee, Mihalis Yannakakis: Principles and Methods of Testing Finite State Machines - a Survey. Proceedings of the IEEE, vol. 84, pp. 1090–1123, Aug 1996
13. Rajeev Motwani, Prabhakar Rafhavan: Randomized Algorithms. Cambridge University Press, New York, 1995
14. Rajeev Motwani, Prabhakar Raghavan: Randomized Algorithms. ACM Computing Surveys, Vol. 28, No. 1, 1996
15. John Moy: OSPF - Anatomy of an Internet Routing Protocol. Addison-Wesley, 1997
16. John Moy: OSPF Version 2. Internet RFC 2328
17. Alexandre Petrenko: Fault Model-Driven Test Derivation from Finite State Models: Annotated Bibliography. In the Proceedings of Modelling and Verification of Parallel Processes (MOVEP'2k). Nantes, France, June 19-23, 2000
18. Misha Schwartz: Telecommunication Networks: Protocol, Modeling and Analysis. Addison-Wesley, 1987
19. Spirent Communications: URL=http://www.spirentcom.com/
20. Zebra: URL=http://www.zebra.org/
21. Yixin Zhao, Xia Yin, Bo Han, Jianping Wu: OnLine Test System Applied in Routing Protocol Test. International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS), 2001