Building Reliable Centralized Intra-domain Routing in Trustworthy and Controllable Network

Tan Jing, Luo Junzhou, Li Wei, Shan Feng
School of Computer Science and Engineering, Southeast University
Southeast University Road 2#, Jiangning, Nanjing 211189, P.R.China
{ tanjing, jluo, xchlw, shanfeng} @seu.edu.cn

Abstract— **Centralized control can improve the consistence of the network and reduce the load of routers. Trustworthy and Controllable Network takes centralized control as one of the basic control mechanisms and requires to build reliable centralized intra-domain routing. In this paper, we solved the problem of building reliable centralized intra-domain routing by finding the routing configuration which maximizes the disjoint paths of each ingress to all egresses. The problem is transformed to finding *K* paths for each ingress to the egresses. It is proved to be NP- hard to find the optimal solution when K≥2 if ingresses do not cross each other and when K≥3 if the ingresses cross each other. To solve the problem efficiently, a heuristic algorithm based on network flow theory called HANE is proposed and evaluated on different types of topologies. The experimental results show that HANE can achieve good performance.**

Keywords: Trustworthy and Controllable Network;centralzied routing;NP-hard;network flow

I. INTRODUCTION

The Internet is designed as a fully distributed network where each router computes and maintains the state needed for network control. Fully distributed network simplifies network protocol design and improves the reliability of network. It achieves great success in the early stages of the Internet. However, as the growth of the Internet scale, more and more control functions are added to the routers which increase the load of routers and restrict their further performance improvement. The fully distributed control also causes IP network fragile and hard to control and management. As considerable complexity is introduced to network control and management, more and more problems are caused by the inconsistence brought by distributed control. For example, by the description of Paxson,V, routing loops are universal in current network and it is energy-consuming to eliminate them[1]. Moreover, in distributed control, the state of network is computed and maintained by each router and there is not any unified network view. It is hard to monitor and predict the network state and make network-wide decisions. For example, a minor local event may cause the oscillation of the whole network.

In order to overcome the problem of fully distributed control, centralized control proposes to separate control logic from routers and build a centralized intra-domain control platform to make control decisions as shown in Fig.1. The routers don't make decisions any more and only receive instructions from the control platform and forward packets. Centralized intra-domain control can reduce the load of routers, improve the consistence of network and provide network-wide decision-making. There have been some works[2-12] about centralized intra-domain control, such as RCP[2], 4D[3] etc. RCP proposed to separate the inter-domain decision logic from routers and build a routing control platform to make inter-domain routing decision for all routers in a domain. Based on RCP, 4D proposed to separate all control logic from routers and build a separated control plane to directly control the underlying network. RCP and 4D are agreed by many researchers and some related works have been proposed[4-12].

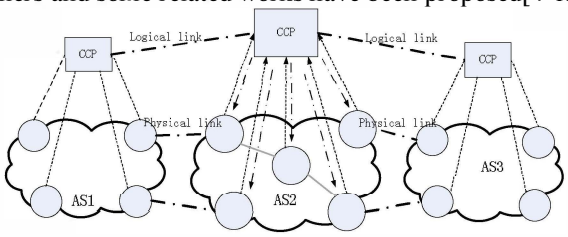


Fig. 1. Centralized intra-domain control. CCP means centralized control platform.

Due to the advantages of centralized control in more powerful processing ability and more concrete control, in our research of building Trustworthy and Controllable Network[7-12], centralized control is taken as one of the basic mechanisms. Trustworthy and Controllable Network hopes to improve the controllability and trustworthiness of network, its basic idea is to build trust-based control in intra-domain control platform. However, in centralized routing, reliability is a key problem. Prebuilding multiple paths, especially disjoint paths, is a common way to improve the reliability of routing[3][10].

In this paper, we discussed building reliable intra-domain routing in Trustworthy and Controllable Network. We still take hop-by-hop routing and single-path routing as the basic mechanisms. In intra-domain routing, for some destination address, some border routers are ingresses and some are outgresses. In different routing configurations, the reliability of network is diverse. For example in Fig.2, I_1 and I_2 are two ingresses and E_1, E_2 and E_3 are three egresses. In the configuration of Fig.2(b), I_1 has two disjoint (node-disjoint and edge-disjoint) paths while I_2 has only one disjoint path to the egresses, the minimum number of each ingress's paths is 1. However, if the network is configured as Fig.2(c), I_1 and I_2 all have two disjoint paths to the egresses, and the minimum number of each source's paths to the egresses is 2. Considering fairness and reliability, the configuration of Fig.2(c) is obviously better than Fig.2(b). The problem is to find the exact routing configuration which maximizes the minimum number

of each ingress's disjoint paths to all egresses in intra-domain routing. The problem is called MAny-to-Many DIsjoint paths problem in Single-path routing problem (MAMDIS problem) in this paper. We discussed the problem in two possible ways of building intra-domain routing. In the first way, the border routers just import and export packets inside and outside the domain; in the second way, the border routers also act as the intermediate nodes to forward traffic from one intra-domain router to another intra-domain router. The main contributions of this paper are: (1) a new problem to find the routing configuration which maximizes the disjoint paths of each ingress to all egresses in centralized routing; (2) NP-hard proof of finding optimal solution to this problem; (3) an efficient heuristic algorithm for this problem; (4) evaluation of the proposed algorithm on realistic and simulated topologies.

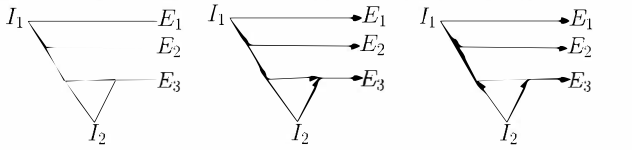


Fig.2. An example of routing configuration

The remainder of this paper is organized as follows. Section II introduces related work to centralized routing and disjoint path. Section III presents that finding optimal solution to MAMDIS is NP-hard. Section IV presents a heuristic algorithm for this problem. In section V the proposed algorithm is evaluated in several topologies. Section VI concludes this paper and points out the future work.

II. RELATED WORK

A. Intra-domain Centralized Control

Intra-domain centralized control can reduce the load of routers and improve the consistence of network. There have been many works about centralized control.

M. Caesar attempted intra-domain centralized control for the first time by building a routing control platform(RCP) which makes all BGP routing decisions for routers[2]. In RCP, all decision logic of BGP on routers is separated and realized by a centralized platform. However, RCP is not concerned with other routing decisions other than BGP. Based on RCP, A. Greenberg proposed a more radical centralized control model called 4D[3]. In 4D, all control logic is divided from routers and a separated "decision plane" is built to control underlying network directly.

4D and RCP are agreed by many researchers and many following works are proposed[4-12]. Jacobus E built an intelligent intra-domain routing control platform which can choose BGP routes dynamically[4]. Jin Fu proved that centralized intra-domain routing does not need more time to reach convergence comparing with distributed routing[5]. Hemant Gogineni built a secure communication model by combining source routing and onion encryption[6]. H.Peterson proposed a loop-free routing updating algorithm in centralized routing[7]. Some researchers applied 4D in enterprise network and data centers and achieved great success[8][9]. Based on 4D, our team proposed Trustworthy and Controllable Network (TCN)[10-12]. TCN hopes to improve the controllability and trustworthiness of the network. Its basic idea is to realize centralized control and strengthen the trust by the consistent view provided by the centralized control platform.

In centralized routing, traditional hop-by-hop routing and label-based routing such as MPLS are all candidates. Hop-by-hop routing can make use of existing routers farthest and reduce the cost of centralized routing. Moreover, single-path routing is still the primary routing mechanism in currently network. In this paper, we still take hop-by-hop and single-path as the basic routing mechanisms in centralized intra-domain routing.

B. Disjoint Paths

Disjoint paths are of great significance in routing because they can improve the diversity of paths and further reduce the risk of simultaneous failure and the congestion of paths. Disjoint paths can avoid the risk of Shared Risk Link Group and provide high reliability. They can also balance the traffic to different links and avoid the congestion of the network. Node-disjoint and link-disjoint are two common ways to build disjoint paths.

Finding optimized disjoint paths between two nodes *s* and *t* has been widely investigated. Ford and Fulkson proposed a polynomial-time algorithm to compute two paths with minimum total length[13]. Li et al. proved that all four versions of the problem of finding two disjoint paths between *s* and *t* such that the length of the longer path is minimized are strongly NP-complete[14].

Moreover, Y.PERL increases the number of destination nodes and discusses the problem of finding two disjoint paths from s_1 to t_1 and from s_2 to t_2 given four vertices s_1, t_1, s_2 and t_2[15]. Based on the study of Y.PERL, Even shows that finding *k+1* pairwise edge(vertex) disjoint paths, *k* paths between s_1 and t_1 and one path between s_2 and t_2 is NP-complete[16].

C. Many-to-Many Maximum Disjoint Paths in Single-Path Routing

In intra-domain routing, it is required to improve the reliability between the ingresses and egresses of an AS and build multiple paths between them. Moreover, since disjoint paths can improve the reliability and congestion avoidance in network, the built paths are much meaningful if they are disjoint. The problem is to find the optimal configuration in a network so that every ingress has maximum disjoint(node-disjoint or link-disjoint) paths to all egresses. Moreover, since single-path routing is still the primary routing mechanism, the intermediate nodes are required to have only one out-degree. As far as we know, this many-to-many maximum disjoint paths problem in single-path routing is first addressed and researched.

III. NP-HARD PROOF FOR DISJOINT PATHS PROBLEM

We denote the ingresses as sources and the egresses as destinations. The problem can be described as follows: given a network topology, some nodes are sources and some nodes are destinations, find the exact routing configuration which can

maximize the minimum number of disjoint paths of every source to the destinations. In this paper, we discussed this problem in two feasible ways of building centralized intra-domain routing. In the first way, the border routers don't transmit packets from one intra-domain router to another intra-domain router and they only transmit packets entering or leaving the domain; in the second way, a border router also works as an intra-domain router and transmits packets from one intra-domain router to another intra-domain router. For the sake of simplicity, in this paper we assume that sources and destinations don't have intersection.

In this section, we first give a formal definition of the problem and then prove that it is NP-hard to find the optimal solution to the problem.

A. Problem Formalization

Definition 1. (Undirected connected graph) Given an undirected connected graph $G = (V, E)$, $V = \{v_1, v_2, ..., v_n\}$ is the set of nodes and $E = \{e_1, e_2, ..., e_m\}$ is the set of edges.

Intuitively, the network topology with *n* nodes and *m* links can be modeled by an undirected connected graph *G* containing *n* nodes and *m* links.

Definition 2. (Routed graph) Routed graph $G_r = (V_r, E_r)$ is obtained from $G = (V, E)$ by assigned directions to edges $E'(E' \subseteq E)$.

Definition 3. (MAny-to-Many DIsjoint paths problem in Single-path routing, called MAMDIS problem) the problem can be described formally as follows: given an undirected connected graph $G = (V, E)$ where $T = \{t_i | t_i \in V, i = 1, 2, ...p(p < n)\}$ are the destinations and some other nodes $S = \{s_i | s_i \in V - T, i = 1, 2, ...l(l < n)\}$ are marked as sources, find a routed graph G_r of G which maximizes the minimum of K_i, where nodes other than the sources have at most one out-degree and K_i denotes the number of disjoint paths from s_i to T in G_r.

To distinguish the two different ways of building intra-domain routing, we denote the problem as uc-MAMDIS if sources don't cross each other and the problem as c-MAMDIS if sources can cross each other.

B. NP-hard Proof

In this section, we analyze the hardness of the MAMDIS optimization problem by showing the corresponding decision problem to be NP-hard. The NP-hard proof exists for both node-disjoint and edge-disjoint problem.

Definition 4. (*k*-MAMDIS problem) Given an undirected connected graph *G*, the many-to-many *k* disjoint paths problem (*k*-MAMDIS problem) is to answer whether there exists an orientation assignment *F* to transform *G* to G_r in which nodes other than the sources have at most one out-degree and every source node has *k* disjoint paths to *T*.

The answer of MAMDIS problem can be obtained by answering *k*-MAMDIS problem for *k*=1,2...*N* successively. If the answer of *k*-MAMDIS problem is YES when *k*=N and NOT when *k*=N+1, the solution to the MAMDIS problem is N. Accordingly, the *k*-MAMDIS problem is denoted as *k*-uc-MAMDIS if sources don't cross each other; otherwise, it is denoted as *k*-c-MAMDIS.

In the following description, we show that finding the optimal solution to *k*-uc-MAMDIS is NP-hard when $k \geq 2$, and it is NP-hard for *k*-c-MAMDIS when $k \geq 3$. Whether finding the optimal solution to k-c-MAMDIS when $k = 2$ is NP-hard is still an open question. We don't prove the *k*-MAMDIS directly, we first show that finding optimal solution to the MAny-to-One DIsjoint paths in Single-path routing problem(MAODIS) is NP-hard and then the NP-hard proof of MAMDIS can be obtained by the proof of MAODIS easily. The only difference between MAODIS and MAMDIS problem is that there is only one destination in MAODIS and there are multiple destinations in MAMDIS.

Theorem 3.1 Finding the optimal solution to *k*-uc-MAODIS for $k \geq 2$ is NP-hard.

Proof. We prove that finding the optimal solution to 2-uc-MAODIS is NP-hard by constructing a polynomial-time reduction from 3-Satisfiability problem(3-SAT) to 2-uc-MAODIS. And the proof for *k*-uc-MAODIS ($k \geq 3$) can be obtained from 2-uc-MAODIS easily.

Lemma 3.1 The mapping f: $3CNF \rightarrow G$ is polynomial-time computable.

Proof: Let Φ be a 3CNF formula that consists of *s* variables, $x_1, x_2, \ldots, x_s$, and *q* clauses. We label the *q* clauses as $C_1, C_2, \ldots, C_q$. For example, $\Phi = (x_1 \vee x_2 \vee x_3) \wedge (\overline{x_1} \vee \overline{x_2} \vee x_4) \wedge (x_1 \vee x_3 \vee \overline{x_4})$ consists of 4 variables and 3 clauses, where $C_1 = x_1 \vee x_2 \vee x_3, C_2 = \overline{x_1} \vee \overline{x_2} \vee x_4, C_3 = x_1 \vee x_3 \vee \overline{x_4}$.

1) For each variable x_i $(1 \leq i \leq s)$, construct a lobe L_i as shown in Fig.3, where a_i and $\overline{a_i}$ are two source nodes.

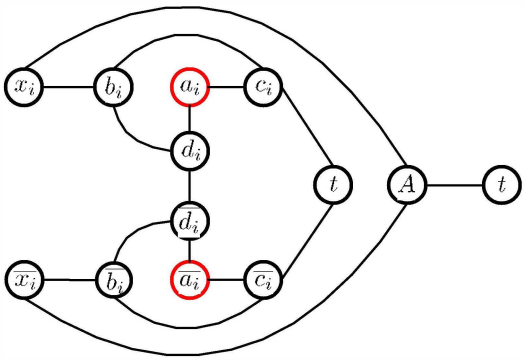


Fig. 3. The construction of L_i.

2) For each clause C_j, $1 \leq j \leq q$, create a lobe H_j which contains a node e_j and three links connecting with the three literals in C_j as shown in Fig.4, where e_j is a source node.

3) Put L_i ($i = 1...s$) and H_j ($j = 1...q$) together, an undirected graph G is constructed. The instance of G for $\Phi = (x_1 \vee x_2 \vee \overline{x_3}) \wedge (\overline{x_1} \vee x_2 \vee x_4) \wedge (\overline{x_2} \vee x_3 \vee \overline{x_4})$ can be obtained by eliminating the direction in Fig.7.

It is obvious that the mapping is polynomial-time computable. This completes the proof of *Lemma 3.1.* ■

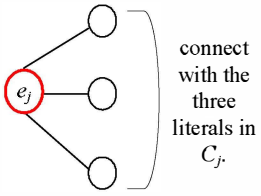


Fig. 4. The construction of H_j.

Lemma 3.2 If there is a truth assignment which satisfies 3CNF Φ, then we can construct a routed graph G_r in which there are two disjoint paths from every source to t.

Proof. Let π be a satisfying truth assignment for Φ, we construct G_r according to π by rule 1 and rule 2.

Rule 1: If the value of x_i is 1, then the edge $(d_i, \overline{d_i})$[1] in $H_i (1 \le i \le s)$ is given the direction from d_i to $\overline{d_i}$ and other edges are orientated as shown in Fig.5(a). It is obvious that two disjoint paths to t can be found for a_i and $\overline{a_i}$ (paths for a_i: $a_i \to c_i \to t$ and $a_i \to d_i \to \overline{d_i} \to \overline{b_i} \to \overline{x_i} \to A \to t$; paths for $\overline{a_i}$: $\overline{a_i} \to \overline{c_i} \to t$; and $\overline{a_i} \to \overline{d_i} \to \overline{b_i} \to \overline{x_i} \to A \to t$). Conversely, if the value of x_i is 0, the edge $(d_i, \overline{d_i})$ is given the direction from $\overline{d_i}$ to d_i and other edges are oriented as shown in Fig.5(b). It can be seen a_i and $\overline{a_i}$ all have two disjoint paths to t in this orientation (paths for a_i: $a_i \to c_i \to t$ and $a_i \to d_i \to b_i \to x_i \to A \to t$; paths for $\overline{a_i}$: $\overline{a_i} \to \overline{c_i} \to t$ and $\overline{a_i} \to \overline{d_i} \to d_i \to b_i \to x_i \to A \to t$). Obviously, whether the value of x_i is 0 or 1, sources a_i and $\overline{a_i}$ both have two disjoint paths to t.

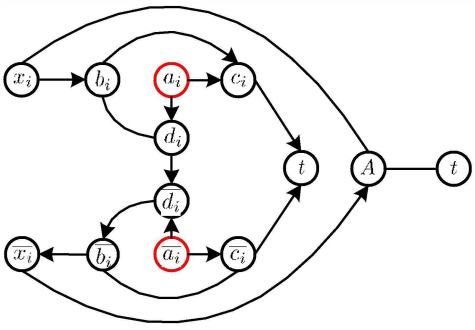
Rule 2: The directions of the edges linked with $e_j (1 \le j \le q)$ are set outward from $e_j (1 \le j \le q)$ (as shown in Fig.6).

According to the above rules, a feasible orientation has been determined for G and G_r has been constructed. It is clear that G_r is loop-free.

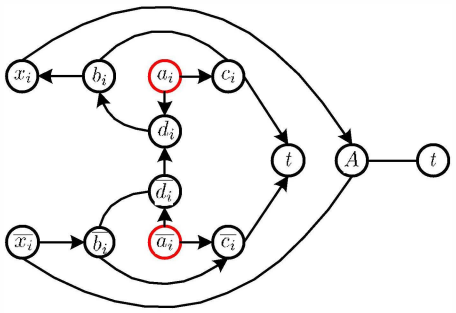
Now we prove that there are two disjoint paths for each source in G_r. In rule 1, we have shown that source a_i and $\overline{a_i} (1 \le i \le s)$ all have two disjoint paths. So, we just need to prove that e_j $(1 \le j \le q)$ has two disjoint paths. Assuming $C_j = x_{j_1} \wedge x_{j_2} \wedge x_{j_3}$, where $x_{ji} (i = 1, 2, 3)$ represents x_k or $\overline{x_k} (1 \le k \le s)$, that Φ is satisfied implies that each clause C_j $(1 \le j \le q)$ in Φ must contain at least one literal $\{x_{j_i} | (i \in (1,2,3)\}$ with true value. And the direction of (x_{j_i}, b_{j_i}) must be $< x_{j_i}, b_{j_i} >$. Thus, e_j has the following path to t: $e_j \to x_{j_i} \to b_{j_i} \to c_{j_i} \to t$. Moreover, whether the value of $x_{j_k} (k \ne i \&\& k \in (1,2,3))$ is 0 or 1, e_j can find another disjoint path: $e_j \to x_{j_k} \to b_{j_k} \to c_{j_k} \to t$ if $x_{j_k} = 1$ and $e_j \to x_{j_k} \to A \to t$ if $x_{j_k} = 0$. So, $e_j (1 \le j \le q)$ also has two disjoint paths to t.

Now the proof of *Lemma 3.2* is completed. ■

[1]In this paper, (u,v) denotes an undirected edge between node u and node v $<u,v>$ denotes a directed edge from u to v.



(a) The orientation for G_i when $x_i = 1$.



(b) The orientation for G_i when $\overline{x_i} = 1$.

Fig. 5. The one-to-one correspondence between the value of x_i and orientation of G_i.

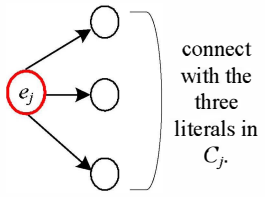


Fig. 6. The orientation of edges linking with e_j

Fig.7 illustrates an example of G_r for $\Phi = (x_1 \vee x_2 \vee x_3) \wedge (\overline{x_1} \vee \overline{x_2} \vee x_4) \wedge (x_1 \vee x_3 \vee \overline{x_4})$ when $x_1 = x_4 = 1$ and $x_2 = x_3 = 0$. This truth assignment satisfies Φ. We take e_1 for example. We can find a path traversing node A to t for e_1: $e_1 \to x_2 \to A \to t$ or $e_1 \to x_3 \to A \to t$. And since the value of x_1 is 1, another path which does not traverse A can also be found for e_1: $c_1 \to x_1 \to b_1 \to c_1 \to t$. In the same way, two disjoint paths can also be found for other sources.

Lemma 3.3 If every source has two disjoint paths to t and sources don't cross each other in the graph G_r, a truth assignment which satisfies Φ can be found in polynomial time.

Proof. We show that a feasible truth assignment for Φ can be found in polynomial time. The truth assignment can be constructed as follows: if the direction of $(d_i, \overline{d_i})$ is from d_i to $\overline{d_i}$ in G_r, x_i is assigned true value; conversely, if it is from $\overline{d_i}$ to d_i, x_i is assigned false value. If edge$(d_i, \overline{d_i})$ has no direction, x_i is assigned true or false randomly. Next we prove by contradiction that this truth assignment can satisfy the 3CNF Φ.

Since there is the precondition that sources cannot cross each other. So the two disjoint paths for each source $e_j (1 \le j \le q)$

must all cross $H_i(1 \le i \le s)$. Suppose that this truth assignment cannot satisfy Φ, there must exist at least one clause $C_j = x \vee y \vee z$ where the values of x, y and z are all false. Without loss of generality，we use X_i representing any node of {*x, y, z*}, let B_i represent b_i(or $\overline{b_i}$) which links with X_i and D_i represent d_i(or $\overline{d_i}$) linking with B_i. So, the direction of $(D_i, \overline{D_i})$ must be from $\overline{D_i}$ to D_i and the direction of (X_i, B_i) must be from B_i to X_i in G_r. Otherwise, the source A_i will has only one out-degree. However, in this case, the source node e_j surely cannot find two disjoint paths in G_r because its available paths all cross node A. It conflicts with the precondition that every source has two disjoint paths to *t*. Therefore, the supposition is false, i.e., the truth assignment must satisfy the 3CNF. Obviously the truth assignment is found in polynomial time. *Lemma 3.3* is proved. ■

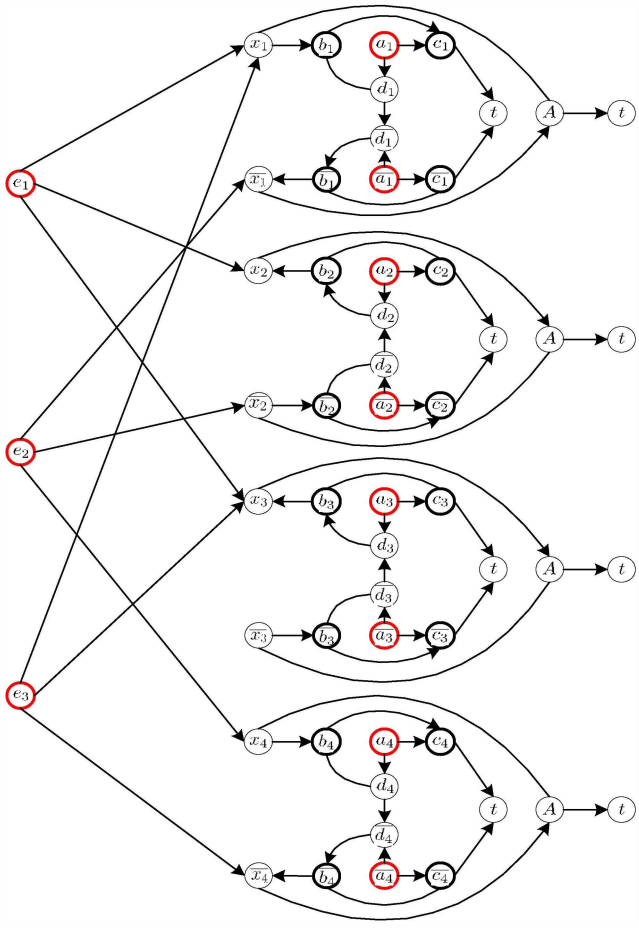


Fig. 7. The graph G_r constructed for $\Phi = (x_1 \vee x_2 \vee x_3) \wedge (\overline{x_1} \vee \overline{x_2} \vee x_4) \wedge (x_1 \vee x_3 \vee \overline{x_4})$ with $x_1 = x_4 = 1$ and $x_2 = x_3 = 0$.

From *Lemma 3.1*, *Lemma 3.2* and *Lemma 3.3*, finding the optimal solution to 2-uc-MAODIS is proved to be NP-hard. And finding the optimal solution to *k*-uc-MAODIS can also be proved to be NP-hard by directly adding (*k*-2) edges from every source to *t*. So theorem 3.1 is proved. ■

Theorem 3.2. Finding the optimal solution to *k*-uc-MAMDIS is NP-hard when $k \ge 2$.

Proof. This theorem for *p* destinations can be proved by replacing node *t* by adding *p-1* nodes $t_1, t_2 ... t_{p-1}$ in G_i as shown in Fig.8. It can be easily proved similarly as Theorem 3.1. ■

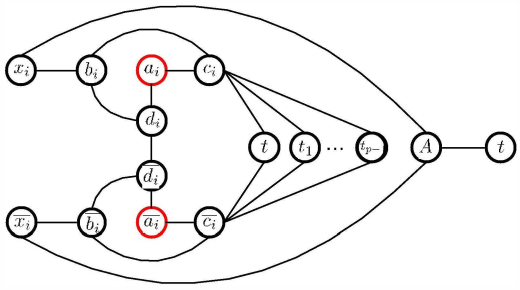


Fig. 8. G_i for *k*-uc-MAMDIS problem

Theorem 3.3. Finding the optimal solution to *k*-c-MAMDIS when $k \ge 3$ is NP-hard.

Proof. *3*-c-MAODIS can be proved to be NP-hard as theorem 3.1 by replacing G_i of 2-uc-MAODIS with the G_i shown in Fig.9. And 3-c-MAMDIS can be proved by adding $t_1, t_2 ... t_{p-1}$ to G_i of *3*-c-MAODIS as theorem 3.2. And *k*-c-MAMDIS when $k \ge 3$ can also be proved to be NP-hard by directly adding (*k*-3) edges from every source to *t* in G_r of 3-c-MAMDIS. However, whether finding the optimal solution to 2-c-MAMDIS is NP-hard is still an open question. ■

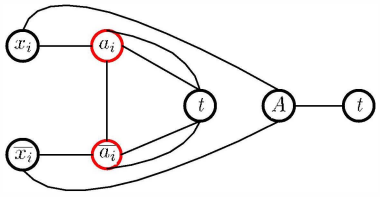


Fig. 9. G_i for k-c-MAMDIS

IV. A HEURISTIC ALGORITHM FOR MAMDIS

In this section, we proposed a Heuristic Algorithm based on NEtwork flow theory(HANE) for MAMDIS problem. In the following subsection, firstly, we introduce the basic idea of HANE; secondly, we describe HANE in detail; thirdly, we give an example of HANE.

A. The basic idea of HANE

The basic idea of HANE is very simple: a new path is found for each source circularly until there is no available path for some source. And when the new path is found for a source in each round, previous paths of this source can be adjusted and the paths of other sources are kept unchanged.

B. Algorithm description

HANE is composed of two phases: *initialization phase* and *calculation phase*. In *initialization phase*, a super destination is added to the original graph so that network flow theory can be

applied; in *calculation phase*, disjoint paths are found for each source. In this subsection, the algorithm for finding edge-disjoint paths is given; in subsection D, the algorithm for node-disjoint paths is discussed.

1) Initialization phase: In this phase, a super destination t is added to the given graph. The super destination t links with all original destinations $t_i(1 \leq i \leq p)$ in the resulted graph.

2) Calculation phase: This is the core part of our algorithm. In this phase, a new disjoint path is found for each source circularly. When we find a path for s_i in this phase, the previous paths of s_i can be adjusted while the paths of $s_j(j \neq i)$ must be kept.

To facilitate the description, a new graph named *p-residual graph* is proposed in this paper. The main difference between *p-residual graph* and *residual graph*[18] is as follows. Residual graph is built for a topology with only one source s while p-eresidual is built for a topology with multiple sources. When residual graph for s is built, all previous flows in the graph can be undoed and every edge in the graph has an backward edge with non-zero capacity. When the p-residual graph for s_i is built, only the flows of s_i can be undoed, the edges on the paths of s_i have non-zero backward edge and the edges on any path of other sources all have backward edges with zero capacity and forward edges with non-zero capacity. The main steps of the calculation phase is given in algorithm 1 and the detailed algorithm of building p-residual graph is given in algorithm 2.

Algorithm 1: *Calculating disjoint paths*

input: $G_r(V_r, E_r)$ in which there are a source node set $S = \{s_i | s_i \in V - T\}, i = 1, 2, ..., n\}$ and a super destination node t.

output: $P = \{\{P_i\} | i = 1, 2, .., n\}$

1) For source node s_i, construct p-residual graph. If a new path $path$ can be found in the p-residual graph, add $path$ to $\{P_i\}$, set $i = i + 1$ and repeat step 1) ; else goto step 2).
2) Return $P = \{\{P_i\} | i = 1, 2, .., n\}$.

The key point of algorithm 1 is to build p-residual graph for each source s_k and it is described in detail in algorithm 2. For convenience, a function *opposite()* is defined to reverse the direction of a given edge. As shown in Fig.11(b), e and e' are two related edges with the opposite directions, there are e'=*opposite*(e) and e=*opposite*(e').

Algorithm 2: *constructing p-residual-graph for* s_k

input: $G_r(V_r, E_r)$ in which there are a source node set $S = \{s_i | s_i \in V - T, i = 1, 2, ..., n\}$, $s_k \in S$ *and* a super destination node t, and previously found paths $P = \{\{P_i\} | i = 1, 2, .., n\}$ in G_r

output: *p-residual graph* $G_r^{S_k}$ of G_r with respect to P and s_k

step1) let the node set of $G_r^{S_k}$ be the same as that of G_r

step2) for each edge $e = (u, v)$ or $e = \langle u, v \rangle$ of G_r, accordingly add a *forward edge* $e_r = \langle u, v \rangle$ and a *backward edge* $e'_r = \langle v, u \rangle$ in $G_r^{S_k}$.

step3) If $e = \langle u, v \rangle$ is crossed by $p(p \in \{P_i\}$ &&$i \neq k)$ in G_r, set $capacity(e_r) = 1$ and $capacity(e'_r) = 0$ where e'_r is the backward edge of e_r in $G_r^{S_k}$.

Note: Since the precondition of the adjustment is that paths of other sources cannot be influenced. When we find an augmenting path for s_k, the flow of other sources cannot be pushed backward. So the direction of the edges which are traversed by any path of $s_i(i \neq k)$ in $G_r^{S_k}$ should be kept unchanged and the capacities of their backward edge are set to be 0. However, the edges on the path of $s_i(i \neq k)$ can also be shared by s_k in the forward direction, so the capacities of their forward edges are set to be 1.

Step4) If $e = \langle u, v \rangle$ is a directed edge in G_r and e is traversed by $p(p \in \{P_i\}$&&$i \neq k)$, for $e_r = \langle u, v_r \rangle (v_r \neq v)$ in $G_r^{S_k}$, set $capacity(e_r)$= 0.

Note: Because it is required that intermediate nodes have one out-degree and the previous paths of other sources cannot be adjusted, so if $e = \langle u, v \rangle$ is a directed edge in G_r and it is on a path of source $s_j(j \neq k)$, other edges linking with u cannot outflow from u when finding the augmenting path.

Step5) If $e = (t_i, t)$ or $e = \langle t_i, t \rangle$ in G_r, set $capacity(e_r)$= 1 and $capacity(e'_r) = 0$ in $G_r^{S_k}$.

Note: The edges between the original destinations and the super destination can always be crossed by any source, so the capabilities of their forward edges are always set to be 1.

Step6) If $e = \langle u, v \rangle$ is only crossed by the paths of s_k in G_r, set $capacity(e_r)$= 0 and $capacity(e'_r) = 1$ in $G_r^{S_k}$, where e'_r is the backward edge of e_r.

Note: Since we need to find an augmenting path for s_k, previous flows of s_k on any edge can be undoed by pushing them on the backward edge. So, in this step, we set the capacities of the backward edges to be 1, and the capacities of the forward edges to be 0.

Step7) For any other edge $e = (u, v)$ in G_r, set $capacity(e_r)$= 1 and $capacity(e'_r) = 1$.

Note: the other edges can be crossed by s_k in arbitrary direction, so the capacities of the forward edge and backward edges are all set to be 1.

C. An Example

An example of finding edge-disjoint paths is shown in Fig.10. The original graph G with two source nodes s_1 and s_2 and three destinations t_1, t_2, t_3 is shown in Fig.10(a) and the green color parts are the super destination and the links added in the initialization phase. In Fig.10(b)-(h), two paths are found for s_1 and s_2 respectively. Fig.10(i) shows the final result. For simplicity, in Fig.10(b)-(i), nodes t_1, t_2, t_3 are neglected.

D. Node-disjoint

As mentioned before, finding edge-disjoint and node-disjoint paths are all NP-hard in MAMDIS. HANE can also solve the node-disjoint problem by splitting each node into two nodes[17]. The only difference is in *initialization* phase.

In the edge-disjoint path problem, nodes are preserved and each edge is transformed to two directed edges with the opposite directions as shown in Fig.11(b). The conversion in node-disjoint problem is a little bit different from that in the edge-disjoint problem. In the node-disjoint problem, each node is split into two halves with a directed edge in between, and each undirected link is transformed to two directed edges as shown in Fig.11(c).

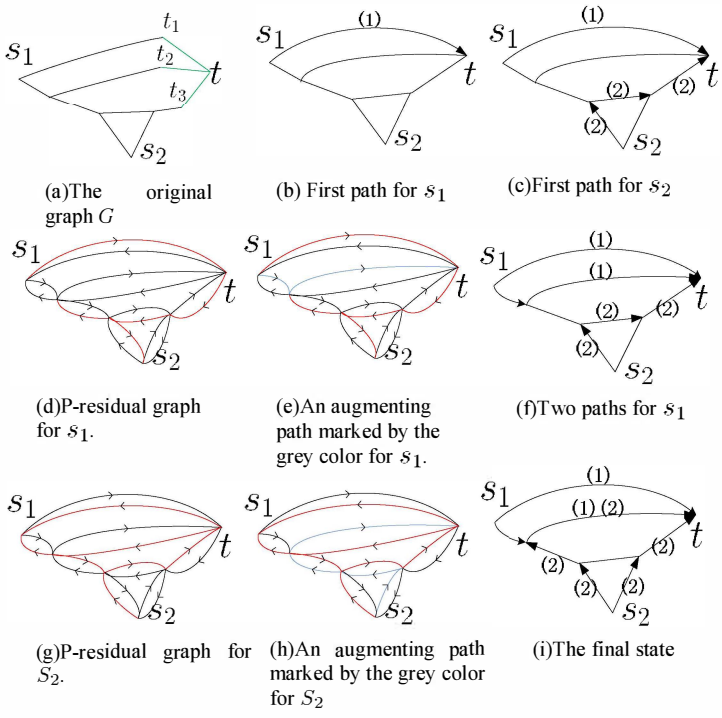


Fig. 10. An example of finding edge-disjoint paths. The capacities of the red color edges are 0 and the capacities of the black colr edges are 1. Symbol (1) on an arc represents a path of s_1 traverses this edge and (2) represents a path of s_2 passes through this edge.

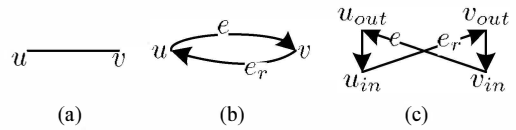


Fig. 11. Convert undirected edge to directed edges. (a) The original undirected edge. (b) The conversion for edge-disjoint (c)The conversion for node-disjoint.

V. EVALUATION

We evaluated HANE on different network topologies to verify its performance in different node scales and link degrees. In this section, firstly, we introduce two reference algorithms, and then describe our experiment environment. Finally, we show and comment the experimental results.

A. *Reference Routing algorithm*

To evaluate the performance of HANE, we compare it with MARA[19] and r-Dijkstra. MARA is proposed to build maximum number of paths for each source, however, the built paths may intersect and make little sense. Moreover, we build r-Dijkstra for comparison. In r-Dijkstra, a shortest path for a source to any destination is found successively until no satisfying path exists in the graph.

B. *Experimental Settings*

We ran experiments on two typical network topologies: AS1239 and Sim300. AS1239 is a real network topology from Rocketfuel project[20] which is of lower connectivity and

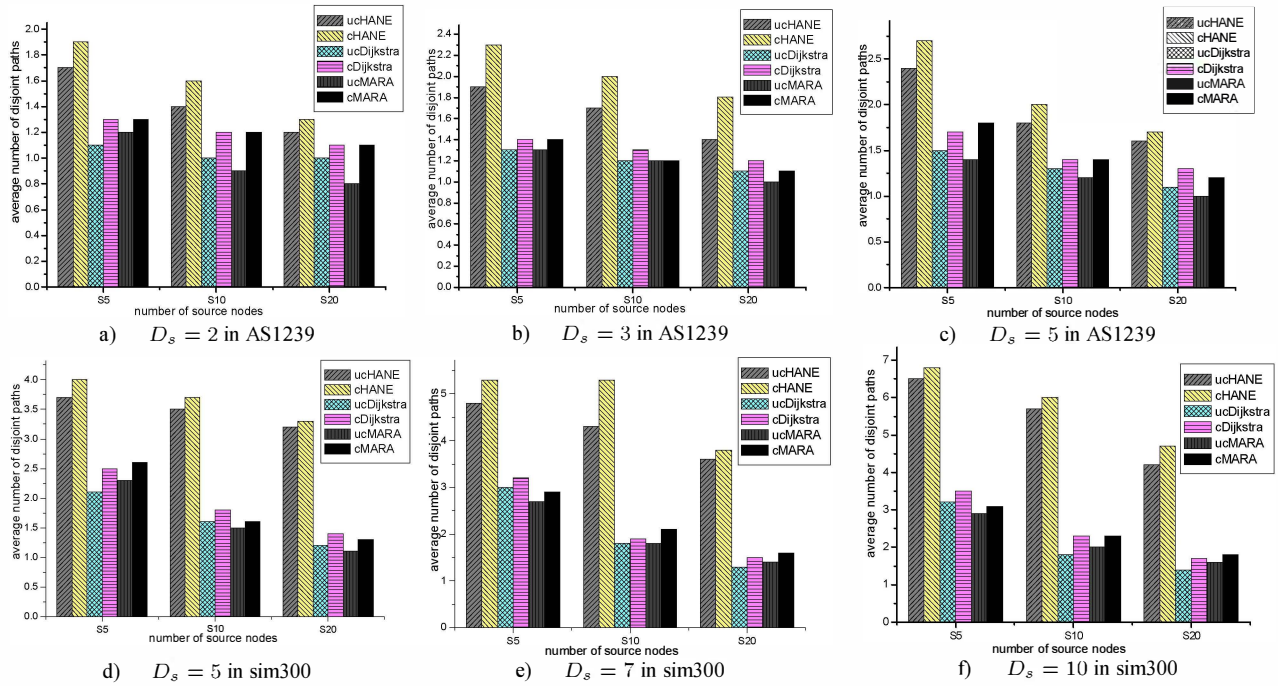


Fig.12. Comparison on average minimum number of disjoint paths. ucHANE and cHANE respectively mean the sources don't cross each other and one source can cross each other in HANE. ucDijkstra and cDijkstra respectively mean sources cannot cross each other and sources can cross each other in r-Dijkstra. ucMARA and cMARA respectively mean sources cannot cross each other and sources can cross each other in MARA. S5, S10 and S20 mean that the number of source nodes are 5, 10, 20 respectively. $D_s = K$ means that the outdegrees of all sources are larger than K.

consists of 315 nodes and 972 links. To evaluate the performance of HANE on high-connectivity topologies, such as enterprise network, we generated a topology called Sim300 by BRITE[21] based on Waxman's probability model[22] which contains 300 nodes and 1500 links.

To verify the performance of HANE under different numbers of source nodes, we select 5, 10, 20 nodes respectively as sources and 5 nodes as destinations in each topology. Moreover, if the node degrees of the sources are too low, it is worthless to compute disjoint paths because the number of disjoint paths is impossible to exceed the sources' node degree. So, we respectively select nodes whose degree is larger than 2, 3 and 5 as sources in AS1239 and select nodes whose degree is larger than 5, 7 and 10 as sources in sim300. The experiment at the same source node scale and node degree in every topology is executed for fifty times and the average value is taken as the result.

C. *Results*

HANE is evaluated by comparing with r-Dijkstra and MARA on AS1239 and Sim300 respectively. The experiment results are shown in Fig.12. The figure presents that HANE always outperforms r-Dijkstra and MARA. It can also be seen that the higher the connectivity of the topology is, the more disjoint paths are available for each source. And in every topology, the larger the proportion of source nodes is, the fewer average disjoint paths are built for each source.

As shown in Fig.12, when $D_s \geq 3$ in AS1239, about two disjoint paths can always be found by HANE. It is very meaningful in improving the reliability of routing. And in sim300, HANE achieves better performance. More than three node disjoint paths can always be found for each source, it is enough for the reliability of routing.

VI. CONCLUSION AND FUTURE WORK

We addressed the problem of building reliable intra-domain routing in trustworthy and controllable network for the first time in this paper and discussed two possible ways to implement it. Finding the optimal solution to this problem is proved to be NP-hard and a heuristic algorithm called HANE is given. The evaluation results show that HANE can achieve good performance. Our work not only promotes the centralized routing theory, but also gives a practical performance reference in live network. In the experiments, we find that topologies have a potent effect on the time complexity of computing. In the future work, we want to check in what types of topologies, two disjoint paths can be found in polynomial time.

ACKNOWLEDGMENTS

This work is supported by National Natural Science Foundation of China under Grant No. 60903161 and No. 60903162 and No. 90912002 and No. 61003311, China Specialized Research Fund for the Doctoral Program of Higher Education under Grant No. 200802860031, Jiangsu Provincial Natural Science Foundation of China under Grant BK2008030, National Key Basic Research Program of China under Grant No. 2010CB328104，Jiangsu Provincial Key Laboratory of Network and Information Security under Grant No. BM2003201 and Key Laboratory of Computer Network and Information Integration of Ministry of Education of China under Grant No. 93 K-9.

REFERENCES

[1] Paxson, V. End-to-End Routing Behavior in the Internet. IEEE/ACM Transactions on Networking 5, 601-618 (1997).

[2] M. Caesar, D. Caldwell, N. Feamster, etc al.. Design and Implementation of a Routing Control Platform [C]. Proceedings of the 2nd Symposium on Networked Systems Design & Implementation (NSDI'05), 2005, 15-28.

[3] A. Greenberg, G. Hjalmtysson, D.A. Maltz. etc al.. A Clean Slate 4D Approach to Network Control and Management [J]. ACM SIGCOMM Computer Communication Review, 2005, 35(5): 41-54.

[4] Jacobus E. van der Merwe et al. Dynamic Connectivity Management with an Intelligent Route Service Control Point. ACM SIGCOMM INM, October 2006.

[5] J Fu, P Sjödin, G Karlsson. Intra-Domain Routing Convergence with Centralized Control. Computer Networks, 2009 – Elsevie.

[6] Hemant Gogineni, Albert Greenberg, David A. Maltz, T. S. Eugene Ng, Hong Yan, and Hui Zhang. MMS: An Autonomic network-Layer Foundation for Network Management. Rice University Technical Report TR08-11,December 2008.

[7] H.Peterson, S.Sen, J.Chandrashekar, L.Gao, R.Guerin, Z.Zhang. Message-Efficient Dissemination for loop-free centralized routing. ACM SIGCOMM Computer Communication Review, vol. 38, no. 3, July 2008.

[8] Martin Casado, Michael J. Freedman, Justin Pettit, Jianying Luo, Nick McKeown, Scott Shenker. Ethane: Taking Control of the Enterprise. ACM SIGCOMM'07, August 2007, Kyoto, Japan.

[9] Al-Fares, M., Loukissas, A., and Vahdat, A. (2008). A scalable, commodity data center network architecture. SIGCOMM CCR, 38(4):63–74.

[10] J. Luo, Z. Han, L. Wang. Trustworthy and controllable network architecture and protocol framework. Chinese journal of computer. Vol. 3, No. 3, pp 391—404, 2009.

[11] P. Wang, J.Z. Luo, W. Li, Y.S. Qu. Control Information Description Model and Processing Mechanism in the Trustworthy and Controllable Network [C]. Proceedings of the 11th IFIP/IEEE International Symposium on Integrated Network Management (IM09), Long Island, New York, USA, June 1-5, 2009

[12] Yansheng Qu, Junzhou Luo, Wei Li, Peng Wang, RCM: A New Resource Control Model Based on Trustworthy and Controllable Network Architecture. 2010 IEEE/IFIP Network Operations and Management Symposium (NOMS2010). April 2010, Osaka, Japan, pp 201-208.

[13] L.R. Ford and D.R. Fulkerson. Flows in Networks. Princeton, 1962.

[14] C. L. Li, S. T. McCormick, and D. Simchi-Levi. The Complexity of Finding Two Disjoint Paths with Min-Max Objective Function. Discrete Appl. Math. 26(1) (1990), 105-115.

[15] Y. Perl and Y. Shiloach. Finding two disjoint paths between two pairs of vertices in a graph. J.Assoc. Comput. Mach. 25:1–9. Assoc. for Computing Machinery, 1978.

[16] S. Even, A. Itai, and A. Shamir. On the Complexity of Timetable and Multicommodity Flow Problems. SIAM Journal of Computing, vol. 5, pp. 691-703, 1976.

[17] D. Xu, Y. Chen, Y. Xiong et al., On finding disjoint paths in single and dual link cost networks, in Proc. INFOCOM, Mar. 2004, pp. 715–725.

[18] J.Kleinberg and Éva Tardo, Algorithm Design, Addison Wesley, 2005.

[19] Y. Ohara, S. Imahori, and R. V. Meter, "MARA: Maximum alternative routing algorithm," in Proceedings of IEEE INFOCOM, April 2009.

[20] N. Spring, R. Mahajan, D. Wetherall, and T. Anderson, Measuring ISP topologies with rocketfuel, IEEE/ACM Trans. Netw., vol. 12, no. 1, pp.2–16, 2004.

[21] Alberto Medina , Anukool Lakhina, Ibrahim Matta , John Byers, BRITE: An Approach to Universal Topology Generation, Proceedings of the Ninth International Symposium in Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS'01), p.346, August 15-18, 2001.

[22] B. Waxman. Routing of Multipoint Connections. IEEE J. Select. Areas Commun., December 1988.