第Ⅴ部

経路情報の解析および
次世代経路制御技術の検討

第 5 部
経路情報の解析および次世代経路制御技術の検討

Routing ワーキンググループはインターネットのルーティングシステムを研究するワーキンググループである。本報告書は、Routing ワーキンググループの 2008 年活動報告である。

第 1 章　Routing ワーキンググループ 2008 年の活動概要

継続して主に Drouting アーキテクチャの提案を行っている。今年は基礎についてまとめ、研究範囲をネットワーク最適化に拡張した。

- 2008 年 3 月 WIDE 合宿における BOF（3/5 13:00–15:50）
  - Potential Based Routing for ad-hoc network
    プレナリで発表された研究について、より詳細な議論を行った。（Hideya Ochiai @ 東大）
  - BGP accidents in Japanese ISPs
    国内 ISP 運用に携わる者から、大規模障害とその原因などについて報告があった。（takada @ MEX、ash @ KDDI）
    議論し、Internet-Draft（本文書第 2 章参照）にまとめた。
  - MARA/Drouting
    Drouting アーキテクチャの解説。（yasu @ KEIO-SFC）
  - Multicast routing using MA-Ordering
    MA-Ordering を利用したマルチキャストの研究に関するブレインストーミング。（qoo @ KEIO-SFC）
- 2008 年 9 月 WIDE 合宿における研究発表（9/10 20:30–21:00）
  “Drouting アーキテクチャにおけるネットワーク最適化と耐故障性の同時実現/Simultaneous Realization of Failure Recovery and Network Optimization on Drouting Architecture”
  （yasu @ JAIST）
  Drouting アーキテクチャにおけるネットワーク最適化手法の提案。論文として執筆中である（本文書第 3 章）
- 2008 年 9 月 WIDE 合宿における BOF（9/11 13:00–14:20）
  - New routing architecture for resiliency and optimization
    上記研究発表のより詳細な議論。（yasu @ JAIST）
  - Route visualization
    SimRouting というルーティングシミュレータの経路可視化ツールとしての紹介。（yasu @ JAIST）

第 2 章　Practical Report for BGP-Specification and Implimentation

前述の大規模障害の報告、その原因の解説、回避手法の提案、BGP プロトコルに求められる開発の思想を、Internet-Draft としてまとめた。IETF 72nd idr WG において、この議題を発表した。

Yasuhiro Ohara, Kenichi Nagami, Akira Kato “Practical Request for BGP Specification and Implementation” `draft-ohara-idr-practical-request-00.txt`, January, 2009.

（nagami @ Intec NetCore、kato @ KEIO）

第 3 章　Drouting アーキテクチャにおけるネットワーク最適化について

Abstract

For the purpose of failure recovery, a new simple multipath routing architecture called *Drouting architecture* has been proposed previously. Drouting architecture is expected to improve the probability of recovery of an operative communication path, even when a failure exists that the routing system is not aware of.

This paper proposes a straightforward method to implement a network optimization using a linear programming in the Drouting architecture, examines the optimality, and further examines the property of simultaneous implementation of both failure recovery and network optimization using simulation. The result shows that it is feasible to provision both features simultaneously, while the network optimization is slightly compromised for improvement of probability of failure recovery.

3.1 Introduction

Failure recovery and traffic engineering on the Internet have been the most important functionalities to support the healthy communication infrastructure. If either no communication path is available due to failures or the communication path is highly congested because of massive use in the Internet, human activities that are shifting on the Internet today, including emergency phone calls and critical commerce transactions, are suspended frequently for a long duration, or totally impossible. Also from the perspective of network administrators in large Internet Service Providers (ISPs), the task of manipulate network traffic to improve the efficiency of utilization of the network capacity (i.e. traffic engineering) is one of the major objectives, since inefficient use of the network capacity degrades their benefit, such as Return Of Investment (ROI). Routing system is responsible both to recover from failures and to decide the efficiency of the utilization of the network.

Current routing systems such as OSPF[123], IS-IS[75], and BGP[146], recover from failures in the duration of seconds to minutes. Since this is not deemed sufficiently quick, Bidirectional Forwarding Detection (BFD)[73] and MPLS[152] Fast ReRoute (FRR)[139] have been proposed, and are in partial deployment. However, there is no expectation that failures in routing systems such as seen in major large-scale internet failures[45, 131, 151] will be recovered by these technologies.

Traffic engineering and the optimization of entire network utilization (i.e., network optimization) for link-state routing protocols (i.e., OSPF and IS-IS) are found computationally intractable[53]. BGP and MPLS can be used for traffic engineering, although they are for local optimality (See [166] for BGP. MPLS is only for LSPs provisioned in advance), and are done manually.

Drouting architecture[134, 136] is proposed previously to enable failure recovery even when some part of the routing system have problems. It provides probabilistic failure recovery function to any components involved in a communication such as end-host's transport protocol or user application software, since just changing the packet tag will change the path of the communication. It is expected that the recovery is sufficiently quick, because the time to verify and change the communication path is totally left up to each components.

This paper proposes and verifies a straightforward method to execute network optimization on the Drouting architecture. Simulations are conducted to exhibit the feasibility of network optimization on Drouting architecture using a linear programming, and to verify the expectation of realizing both failure recovery and network optimization simultaneously.

The organization of this paper is as follows. Section 3.2 describes the related works and their problems. Section 3.3 revisits the summary of Drouting architecture. Section 3.4 exhibits the simple application of a linear programming for the network optimization on Drouting architecture, and examines the optimality. Section 3.5 extends the network optimization model to sustain also the failure recovery property, and examines the result in simulation. Section 3.6 gives the concluding summary.

3.2 Related Work

Many traffic engineering and network optimization technique have been proposed. For a few

example, Cohen and Nakibly used loose source routing approach to minimize the maximum load in the network[32]. Basu et al.[18] proposed a new routing system for congestion avoidance that assigns a scalar to routers and uses gradient defined by the scalar to forward traffic. None of them are deployed as of writing, possibly due to lack of controllability, consideration to failure recovery, and compatibility to existing system.

Current Internet tends to employ BGP[22] or MPLS[48, 92] for traffic engineering. They are rather manual, or available only in a specific domain, such as in an interdomain or among provisioned MPLS LSPs.

This paper tackles to network optimization on fundamental IP network, using simple multipath IP routing approach.

Some multipath route calculation algorithms have been proposed in the past. Multipath routing methods proposed in the past are based on, and are extensions of, the shortest path routing. Thus, although the optimization of the routing metrics is computationally hard as mentioned earlier, they require the routing metric setting in advance. MPDA[184] is a link state routing algorithm which distributes only partial topology information. MDVA[186] is a distance vector routing algorithm that uses diffusing computation[40]. MPATH[185] is another distance vector routing algorithm that distributes predecessor node information of paths. MPDA, MDVA and MPATH calculate multipath routes that are loop-free at every instance, using the Loop Free Invariant (LFI) condition on the routing metrics.

FIR[102] computes per network interface routing tables by executing the Shortest Path First (SPF) calculations separately for each of its neighbors in order to route around the failure. Yang and Wetherall[195] proposed Deflection, which extends the LFI condition by utilizing the identity of the previous hop to produce an increased number of nexthops. FIR and Deflection are multipath routing methods for the purpose of failure avoidance. They provide backtracking paths that transit the same node twice, which is not efficient in terms of network utilization.

Drouting architecture proposes to construct the multipath routes to utilize all links in the network. They proposes a family of multipath route calculation algorithms called Maximum Alternative Routing Algorithm (MARA)[135]. MARA and its employer Drouting architecture cooperatively provides a number of diverse communication paths on the hop-by-hop network, without being restricted by the classic shortest path routing or the routing metrics. We revisit the Drouting architecture in the next section.

3.3 Drouting Architecture

Drouting architecture is a proposal that constructs and utilizes the multipath routes as the Directed Acyclic Graphs (DAGs) which include all links in the network. For the purpose of Internet routing, DAGs that include all links in the network have not been studied until the proposal.

IP packets carry packet tags that are set by the end host. The packet tag is assumed to be stored in the IPv6 flowlabel[145]. The packet tags are used to select a network path from the multipath routes. The tag forwarding enables end hosts to dynamically change a path based on user preferences. A packet tag is assigned to a network path deterministically without having to maintain any states on routers. The packet tag is randomly chosen. A source host changes its packet tag only when it desires to use another network path. In order to avoid packet reordering and degradation of TCP performance, source hosts are assumed to assign the same packet tag for all the packets in one TCP session.

The source host is assumed to detect problems on the communication path in some way such as a fixed timer for packet losses or dynamic bandwidth estimation[90]. Once the source host detects a problem, it randomly chooses a new packet tag. The new packet tag is expected to be assigned to a new communication path, which may stochastically avoid the problem.

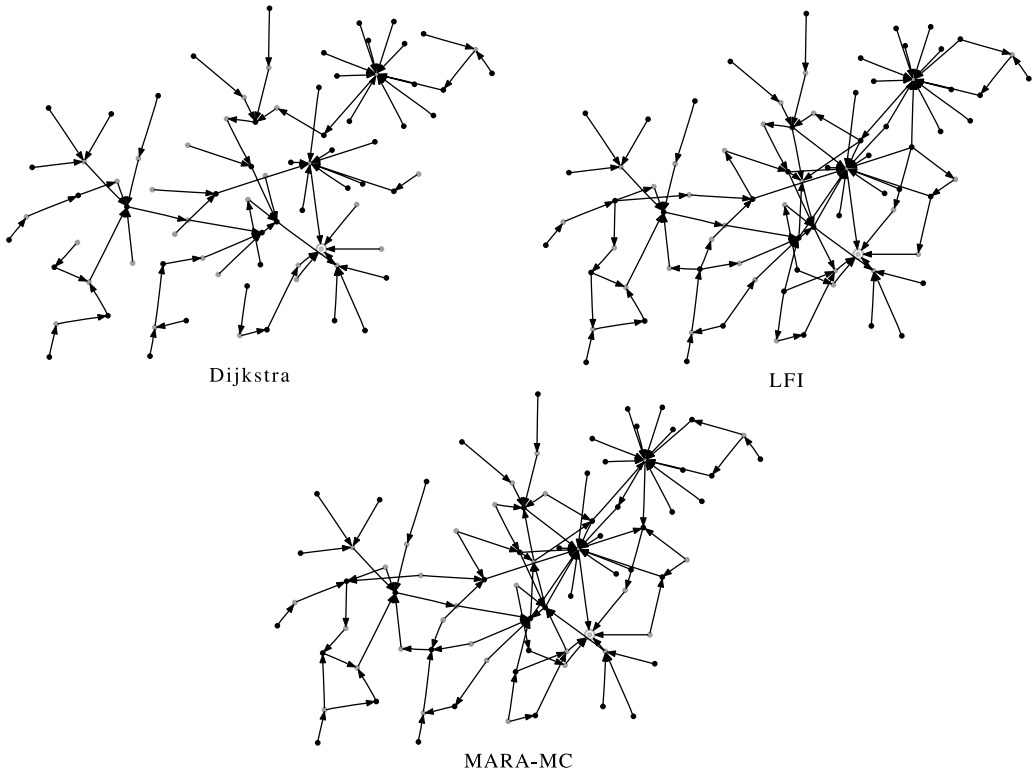

Fig. 3.1. Routing graph destined to a node: Dijkstra, LFI, and MARA

The source node can neither predict nor specify in advance which network path will be assigned for its packets. A network path is only randomly assigned to a packet as a result of forwarding the packet with the particular packet tag. However, a source node can specify the same network path for multiple communication sessions to the same destination, by using the same packet tag.

Changing the packet tag enables avoidance of a long failure even if the routing system fails to detect the network failure. If a failure occurs in the network and the routing system can detect the failure, the routing system will automatically recompute network routes, hence altering network paths to avoid the failure, the same as in the existing Internet. A source host can use an alternative path by changing the packet tag, regardless of whether or not the routing system detects (and hence will route around) the failure. By changing the packet tag, the network path which the packet will take may or may not be changed, depending on the randomly generated new packet tag.

Assuming that the traffic demands are the union of a massive number of micro flows, and also assuming that each of the micro flows have distinct random packet tags, Drouting enables traffic splitting at the unit of the micro flow over the multipath routes. It is expected that the smaller the traffic size of a micro flow becomes, the finer granularity of splitting we obtain. With these hypotheses, the traffic engineering issue is tackled in this paper.

In order to show how Drouting architecture and MARA looks in contrast to others, Figure 3.1 illustrates routing graphs to a destination in the network of WIDE Project[190] as an example. Algorithms calculating the routing graph are Dijkstra, LFI, and MARA. Each arrow in the figure indicates the individual route to the destination. Dijkstra used in the Internet today calculates basically a simple tree[1], which revokes both the diversity of network path and the robustness

1　This is not precisely true when Equal-Cost Multi-Paths (ECMPs) are calculated.

of the graph from its density. LFI extends the shortest path tree only when the relation of the routing metrics allows. While in this case LFI successfully calculates a number of routes sufficiently on all edges, whether the number of calculated routes is sufficient depends on the setting of the routing metric. LFI may, and sometimes actually does, fail to calculate sufficient number of multipath routes. MARA always calculate routes on all edges.

Drouting architecture allows to split the traffic over these multipath routes, in the specified ratio. We will try to optimize the network utilization in the next section, by determining the traffic split ratio among multipath routes.

3.4 Network Optimization

A Linear Programming (LP) technique is utilized to determine the traffic split ratios, to adapt to a specific shape of the traffic demands. The LP problem in [53] is introduced and slightly changed to fit to the Drouting architecture. The purpose of the LP problem in [53] was to find optimal traffic splitting on each node in the base graph structure of the network, without hop-by-hop network restriction (i.e., routes on different nodes may not be consistent). The example application of this traffic splitting optimization is to the virtual circuit based network such as MPLS. The optimization by the LP model cannot be applied to the hop-by-hop network routing as is.

In contrast to the optimization on the base network, the purpose here is to find the optimal traffic split ratios in the routing graphs for each destination. The introduction of routing graphs enables straightforward application of LP optimization to the hop-by-hop network.

3.4.1 LP model

The network model used in the LP is as follows. A directed network $G = (N, A)$ with a capacity $cap(a)$ for each arc $a \in A$ and a demand matrix D is given, where D tells the demand $D(s,t)$ between s and t for each pair of nodes $(s,t) \in N \times N$. $f_a^{(s,t)}$ tells how much of the traffic flow from s to t goes over a. $l(a)$ represents the total load on arc a, i.e., the sum of the flows going over a. Φ_a is a piecewise linear cost function that is a function of the load $l(a)$ on arc a.

In Drouting architecture, multipath routes are calculated for each destination t. The multipath routes are represented by the directed set of links denoted as A_t for each $t \in N$.

Then the problem to find optimal traffic split ratio among the multipath routes is defined as follows.

Minimize:

$$\Phi = \sum_{a \in A} \Phi_a \tag{1}$$

subject to

$$\sum_{x:\,(x,y)\in A_t} f_{(x,y)}^{(s,t)} - \sum_{z:\,(y,z)\in A_t} f_{(y,z)}^{(s,t)} = \begin{cases} -D(s,t) & \text{if } y = s, \\ D(s,t) & \text{if } y = t, \\ 0 & \text{otherwise} \end{cases} \qquad y,s,t \in N, \tag{2}$$

$$l(a) = \sum_{(s,t)\in N\times N} f_a^{(s,t)} \qquad a \in A, \tag{3}$$

$$\Phi_a \geq l(a) \qquad a \in A, \tag{4}$$

$$\Phi_a \geq 3 \cdot l(a) - \frac{2}{3} \cdot cap(a) \qquad a \in A, \tag{5}$$

$$\Phi_a \geq 10 \cdot l(a) - \frac{16}{3} \cdot cap(a) \qquad a \in A, \tag{6}$$

$$\Phi_a \geq 70 \cdot l(a) - \frac{178}{3} \cdot cap(a) \qquad a \in A, \tag{7}$$

$$\Phi_a \geq 500 \cdot l(a) - \frac{1468}{3} \cdot cap(a) \qquad a \in A, \tag{8}$$

$$\Phi_a \geq 5000 \cdot l(a) - \frac{19468}{3} \cdot cap(a) \qquad a \in A, \tag{9}$$

$$f_a^{(s,t)} \geq 0 \qquad a \in A;\ s,t \in N, \tag{10}$$

$$\sum_{z:\,(y,z)\in A_t} r_{(y,z)}^t = 1 \qquad y,t \in N, \tag{11}$$

$$f_{(y,z)}^{(s,t)} = \left(\sum_{x:\,(x,y)\in A_t} f_{(x,y)}^{(s,t)} \right) \cdot r_{(y,z)}^t \qquad y,s,t \in N. \tag{12}$$

Equation (2) indicates that, y must produce the traffic of size $D(s,t)$ if y is the source, y must receive the traffic of size $D(s,t)$ if y is the destination, and otherwise y must relay the received traffic to other nodes in order for the traffic to

Table 3.1. Network optimization results.

Instance	Φ	MaxUtil	AvgUtil	MINOS result
Dijkstra	49586400	0.830248	0.134635	optimal solution found.
LFI	27498700	0.73225	0.108235	optimal solution found.
MARA	40350600	0.929208	0.116519	the current point cannot be improved.

eventually reach its destination. This is called the flow conservation constraint, and ensures that there is no drop in forwarding traffic. Equation (3) calculates the load for each arcs by summing all flows traversing the arc a. From Equation (4) to (9) the model specifies the piecewise linear cost function in relation to the load on the arc. Each equation determines individual piece of the function, by specifying the slope and the y-intercept of the function for the range. The constant coefficients appeared in these equations are determined in [53].

For the part from Equation (1) to (10), the only difference from LP problem in [53] is that in Equation (2), the links on which the traffic can be flowed is restricted by the routing graph, A_t, rather than the original A.

$r^t_{(x,y)}$ denotes the traffic split ratio on the node x to the nexthop node y for the traffic destined to t, in the multipath routing graph A_t. Equation (11) states that the sum of the traffic split ratios on a node x for a destination t must be 1. Equation (12) constraints that the amount of traffic flow outgoing from y to z for a s-t flow (i.e., $F^{(s,t)}_{(y,z)}$) must obey the traffic split ratio $r^t_{(y,z)}$, in relation to the sum of the incoming traffic flows on y.

3.4.2 Optimization Results

The network optimization using the LP model described in Section 3.4.1 is applied to the network graph structure of WIDE Project as an example. The number of nodes $|N|$ was 80 and the number of edges $|A|$ was 105. The routing metrics for each edges are retrieved by accessing OSPF LSDB in a working router, and is used to calculate routing graphs by Dijkstra and LFI algorithms. The traffic demands are randomly generated on each s-t pairs between 0 and 1,500, assuming less than 1.5 Mbps. The bandwidth capacities for each links are defined uniformly 1,000,000, assuming 1 Gbps. MINOS[171] via AMPL[8] is used to solve the LP problems.

The optimization results are summarized in Table 3.1. The instance column describes the optimization problem instance. All instances utilize the same synthetic traffic demands and the network topology described above. Dijkstra, LFI, MARA instances are the ones using the LP model described in Section 3.4.1, with the routing graph calculated by the algorithms Dijkstra, LFI, and MARA, respectively.

In the results, Dijkstra gets the objective Φ as 49,586,400, the utilization of the maximum loaded link (MaxUtil) as 0.830248, and the average utilization of links (AvgUtil) as 0.134635. LFI gets the better result, Φ as 27,498,700, MaxUtil as 0.73225, AvgUtil as 0.108235. This indicates that multipath routing can improve the efficiency of network utilization significantly. For MARA instance, the LP solver MINOS could not get optimal solution. A possible reason of this result is that MARA produce huge number of possible communication path, while LFI does not. Another possible reason is that the configuration of LP model may not fit to the MINOS solver. This result indicates that further improvement in LP model is desired.

While the MINOS cannot get optimal solution for this case, MARA improved the objective in the LP model, compared to the Dijkstra (40350600 against 49586400). The interesting point is that while the global objective is better, the MaxUtil is worse (0.929208 against 0.830248). We consider that a better cost function might exist, to find the better split ratios for routing to improve the MaxUtil.

3.5 Failure Recovery

To see the failure recovery property of routing algorithms, simulations are conducted in this section. Simulations are executed using SimRouting[164] routing simulation tool.

Incorporating the split ratio optimized in Section 3.4, the failure simulation is executed following the scenario below.

1. 3 nodes are chosen randomly as the failed nodes.
2. For each source destination pairs (s, t) that neither s nor t is the failed node:
   (a) Find a random path from s to t, obeying the routing graph and the probabilities of traffic split ratio on each hops. This trial is executed 10 times.
   (b) Count the number of occurrence that the path does not include a failure node (i.e., valid available path).
3. The entire procedure is repeated 10 times.

The result of the simulation is summarized in Table 3.2. MARA significantly improves the failure recovery probability as 87.470% against the original Dijkstra's 75.527%.

We consider still that the MARA's failure recovery is not adequate. This is because that the traffic split ratio optimized for the network utilization does not fit for the failure recovery purpose. In what follows we modify the LP model to improve the failure recovery further, compromising the network optimality slightly.

Table 3.2. Failure recovery results.

Algorithm	Total trials	#Success	rate
Dijkstra	585200	441988	75.527%
LFI	585200	442637	75.638%
MARA	585200	511880	87.470%

The LP model is modified as follows.

Minimize:

$$\sum_{a \in A} \Phi_a + \text{equality}$$

subject to

$$\text{equality} = \sum_{(y,z) \in A_t} q^t_{(y,z)}, \tag{13}$$

$$q^t_{(y,z)} \geq -200 \cdot r^t_{(y,z)} + 100 \qquad (y,z) \in A_t, \tag{14}$$
$$q^t_{(y,z)} \geq -500 \cdot r^t_{(y,z)} + 200 \qquad (y,z) \in A_t, \tag{15}$$
$$q^t_{(y,z)} \geq -1000 \cdot r^t_{(y,z)} + 300 \qquad (y,z) \in A_t, \tag{16}$$
$$q^t_{(y,z)} \geq -2000 \cdot r^t_{(y,z)} + 400 \qquad (y,z) \in A_t, \tag{17}$$
$$q^t_{(y,z)} \geq -5000 \cdot r^t_{(y,z)} + 500 \qquad (y,z) \in A_t, \tag{18}$$
$$q^t_{(y,z)} \geq 200 \cdot r^t_{(y,z)} - 100 \qquad (y,z) \in A_t, \tag{19}$$
$$q^t_{(y,z)} \geq 500 \cdot r^t_{(y,z)} - 200 \qquad (y,z) \in A_t, \tag{20}$$
$$q^t_{(y,z)} \geq 1000 \cdot r^t_{(y,z)} - 300 \qquad (y,z) \in A_t, \tag{21}$$
$$q^t_{(y,z)} \geq 2000 \cdot r^t_{(y,z)} - 400 \qquad (y,z) \in A_t, \tag{22}$$
$$q^t_{(y,z)} \geq 5000 \cdot r^t_{(y,z)} - 500 \qquad (y,z) \in A_t. \tag{23}$$

Constraints Equation (13) sums all penalty for split ratios, $q^t_{(y,z)}$. This is incorporated in the global objective of the LP model. Equation (14) to (23) describes a piecewise linear cost function for the value of split ratio. This penalty $q^t_{(y,z)}$ tries to make the split ratio $r^t_{(y,z)}$ closer to 0.5.

The results of optimization using modified LP model are shown in Table 3.3. Note that the solution to the MARA instance was made optimally, by the modification to the LP model. In summary, there was no major improvement from Table 3.1.

However, failure recovery probability is improved considerably. The results are given in Table 3.4. All instances improve the failure recovery probability. Among those, MARA exhibited the highest probability to recover from failures. It was improved to 93.460% from the previous result (87.470% in Table 3.2).

Table 3.3. Modified network optimization results.

Instance	Φ	equality	MaxUtil	AvgUtil	MINOS result
Dijkstra	49586400	28345000	0.830248	0.134635	optimal solution found.
LFI	28675000	28191800	0.732845	0.112089	optimal solution found.
MARA	40208100	27553700	0.929208	0.115977	optimal solution found.

Table 3.4. Modified failure recovery results.

Algorithm	Total trials	#Success	rate
Dijkstra	585200	529729	90.521%
LFI	585200	524134	89.564%
MARA	585200	546933	93.460%

3.6 Conclusion

The feasibility of traffic engineering and network optimization on Drouting architecture was verified in this paper. Although the LP model must further be improved, the straightforward method to optimize the network utilization was shown feasible.

To implement simultaneous realization of both failure recovery and network optimization, existing LP model is modified so that the failure recovery probability is improved compromising the network optimization.

Traffic engineering capability with failure recovery in Drouting architecture is expected to contribute to implement the Dependable Internet.