Policy-based BGP Control Architecture for Autonomous Routing Management

Osamu Akashi† Kensuke Fukuda‡ Toshio Hirotsu§ Toshiharu Sugawara∗
NTT Network Innovation Laboratories† National Institute of Informatics‡ Toyohashi University of Technology/JST CREST§ NTT Communication Science Laboratories∗
akashi@core.ntt.co.jp† kensuke@nii.ac.jp‡ hirotsu@ics.tut.ac.jp§
sugawara@core.ntt.co.jp∗

ABSTRACT

Unexpected temporal and spatial changes of inter-AS routing behavior often lead to the necessity of on-demand inter-domain routing-adjustment. For resolving this problem, we apply the AISLE framework, which is a multi-agent-based model, to a policy-based routing-adjustment system for transit ISPs and their customer ASs. This paper describes the BGP-control architecture called VR (Virtual Router) that can dynamically change forwarding paths considering alternative paths, which are inferred from historical data and confirmed when they are actually applied. VR can control conventional multiple border routers in an AS without any protocol extensions. The policy description, which is interpreted by an agent, enables network operators to define autonomous actions for analyzing network status and adjusting inter-AS routing based on these observed results by issuing requests to VR. Some evaluation results indicate that VR can effectively change routing over BGP data on the actual Internet and some control scenarios based on policy descriptions demonstrate the validity of our basic design framework.

Categories and Subject Descriptors

C.2.3 [**Computer-Communication Networks**]: Network Operations—*Network management*; C.2.6 [**Computer-Communication Networks**]: Internetworking—*Routers*

General Terms

Management, Design

Keywords

BGP, routing, multi-agents, policy-based control

1. INTRODUCTION

The Internet consists of more than 12000 autonomous systems (ASs) that correspond to independent network management units. Inter-AS or inter-domain routing is currently controlled by BGP [12]. In this architecture, advertised reachability information flows in a hop-by-hop manner throughout ASs, being modified at each AS according to its own policy. Since there are more than 12,000 independent ASs, and routing information mutates spatially and temporally, verifying whether routing is performed as human operators intend using static analysis in advance is difficult and sometimes meaningless. In the worst case, inconsistency among ASs or unintended traffic flow easily occurs. The essential problem is that there are no cooperative frameworks among ASs for monitoring, analyzing, and controlling inter-AS routing. In addition to these problems, the limitation of human operators, who cannot repeatedly and continuously observe and analyze large amounts of routing information and adjust routing behavior according to these results, requires the support by autonomous policy-based routing adjustment systems.

Autonomous routing control at the inter-AS level should share and modify inter-AS routing information that is available outside the AS. Cooperative distributed problem solving (CDPS) can work adequately in terms of efficiency, scalability, and availability in this situation, as in the case of the inter-AS diagnostic system called ENCORE [3, 4], which has been applied to commercial operation in several ISPs. This is because 1) CDPS coincides with the control architecture, and observing methods should be managed on a request-and-acceptance basis rather than by centralized control approaches, 2) observed results including statistical analysis should be shared, after local calculation has been performed, for efficiency and scalability, and 3) operation availability increases when some problems exist in the network. For example, cooperative operations such as message relaying among agents is effective to deliver information to appropriate agents when direct communication is unavailable; this was shown to be effective through the experience of using our multi-agent-based system, ENCORE.

To achieve policy-based routing-control for transit ISPs and their customer ASs, we apply the AISLE framework [2], which is a multi-agent-based model to cooperatively monitor, analyze, and control BGP routing information among several ASs. In [2], we proposed a basic cooperative model for controlling inter-AS routing and showed some applicable examples. In this paper, we focus on the VR (Virtual Router) architecture for controlling BGP, which is embedded in the AISLE agent, and discuss about application for the routing adjustment used for transit ISPs and their customer ASs.

This paper is organized as follows. First, we briefly explain about background of our application domain, requirements for autonomous Inter-AS routing management, and the AISLE cooperative model. Then, we focus on the BGP-control architecture called VR in detail. VR manages conventional BGP routers in an AS utilizing iBGP (internal BGP) sessions without any protocol extensions. We then discuss the effectiveness of the system from some

SIGCOMM'06 Workshops September 11-15, 2006, Pisa, Italy.

viewpoints: evaluation results about routing-control over BGP data in the Internet, and application scenarios that can be performed by this AISLE framework.

2. BACKGROUND

2.1 Application domain

In the current network topology, transit ISPs typically determine paths through which inbound packets for their customer ASs are sent. However, inconsistency or unintended traffic flow from the viewpoint of downstream customers sometimes occurs. In the example shown in Fig.1, cooperative actions for monitoring and controlling traffic outside the AS are required.

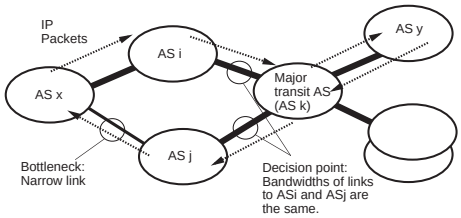

Figure 1: Routing adjustment at inter-AS level

In this example, AS_x is a multihomed AS that has two BGP peers. Therefore, AS_x has two possible paths for forwarding packets destined for AS_y, which are a path via AS_i and another path via AS_j. AS_x can designate AS_i as a next hop for outgoing packets, because AS_x knows that the bandwidth of the link between AS_x and AS_i is larger than that between AS_x and AS_j. This can be determined from only AS_x's local perspective. On the other hand, in the case of incoming packets, AS_x cannot control their routes. AS_k, which is a typical major transit AS, has two BGP paths for packets destined for AS_x, which are similarly via AS_i and AS_j. AS_k might select the route via AS_j even if AS_x wants to receive inbound packets via AS_i because those seem to be equal conditions from the viewpoint of AS_k. If AS_k selects the route via AS_j to forward packets to AS_x, the link between AS_x and AS_j becomes a bottleneck.

2.2 Requirements for inter-AS routing management

Inter-AS routing management has some difficulties such as spatial and temporal changes of routing information over different administrative domains. For diagnosis of inter-AS routing anomalies, we have proposed a CDPS approach [4] to cope with these problems. Simple centralized models are difficult to apply. The following functions are required to extend this system for the adaptive control of intra- and inter-AS routing based on network status.

1. **[Flexible BGP-control functions]** Router primitives only configure routing protocols at lower level network layers. Furthermore, there are no control interfaces for flexibly controlling BGP on existing border routers. Therefore, flexible BGP-control functions should be provided.

2. **[Feedback functions based on network status changes]** Functions for observing network status and analyzing its results are required. Feedback mechanisms based on these analyzed results are also required.

3. **[Policy description over router primitives]** A more abstract management policy is required to represent observation actions and feedback actions.

4. **[Cooperative framework among ASs]** The cooperative actions among ASs are required to control inbound traffic.

The objective of AISLE is to automate the use of these functions by multiple intelligent agents for policy-based routing management.

2.3 Cooperative management model in AISLE

To achieve this objective, we apply the AISLE cooperative model to a policy-based routing-adjustment system for transit ISPs and their customer ASs.

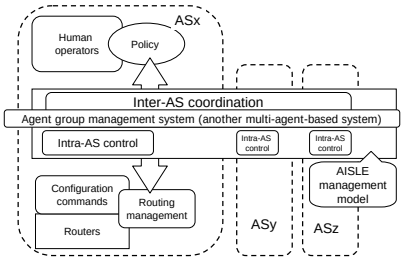

Figure 2: Cooperative management model in AISLE

As shown in Fig.2, each AS is independently managed, therefore the agent deployed in an AS performs intra-AS control functions. The agent is configured by operators in the AS and controls intra- and inter-AS routing information in the AS from its local perspective. The agent determines its actions according to the policy description, monitors BGP information from border routers, modifies the BGP information, and sends that information back to the border routers to adapt to network status changes.

The inter-AS control functions are performed through cooperative actions among agents over multiple ASs. Each agent is autonomous, so they can determine actions according to the local policy while considering requests from other agents. Therefore, the agent might refuse requests inconsistent with the agent's intention, namely the agent's routing-management policy. Location information about agents on the BGP topology map and agents' capabilities are managed by an agent group management system called ARTISTE [16], which is an independent system of AISLE, that can be applied to any other agent system on the Internet.

3. SYSTEM ARCHITECTURE

3.1 Agent structure

The structure of the agent is shown in Fig.3. The main modules are the policy-control-engine, BGP-controller, and cooperative-action-controller. The first two modules achieve the VR functions that enable policy-based BGP-control as a whole AS. The policy-control-engine interprets a given policy description and invokes actions for observation and control. It also sends control requests to the BGP-controller to perform feedback actions according to the given description. The cooperative-action-controller organizes inter-AS cooperative actions for inter-AS routing monitoring and control.

The BGP-controller monitors and controls BGP information via iBGP sessions with border routers in the AS to reflect policy and to adjust routing behavior to environmental changes. For controlling BGP information, the BGP-controller modifies BGP attribute values such as `local_pref` and `next_hop` of received BGP entries and sends them back to border routers. The attribute `local_pref` value determines priority for selecting the best path and the `next_hop` designates a router to which packets bound for a prefix are forwarded. These routers then apply BGP best path selection rules

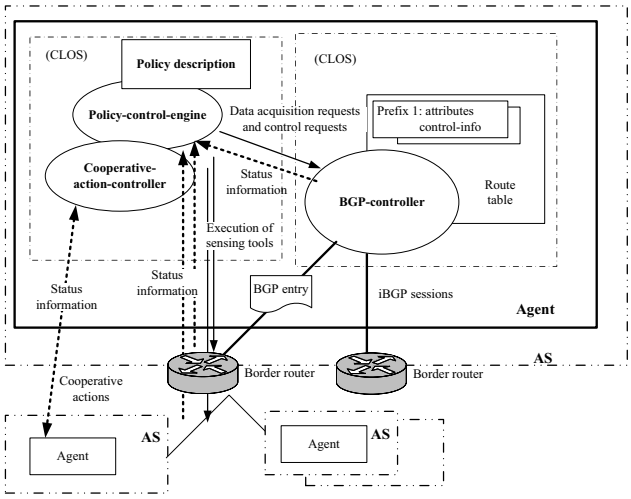

Figure 3: Structure of AISLE agent

```
Select the BGP entry:
  1) with the highest weight attribute
     (The weight attribute is Cisco proprietary)
  2) with the highest local_pref attribute
  3) that was locally originated
  4) with the shortest AS_path
  5) with the lowest origin attribute
  6) with the lowest MED
  7) learned via EBGP
  8) with the closest IGP neighbor
  9) with the lowest BGP router-ID
```

Figure 4: BGP best path selection rules

as shown in Fig. 4. Although the `weight` attribute has the highest priority, it is not defined as the standard [12] and cannot be distributed outside Cisco routers. Therefore, the BGP-controller utilizes the `local_pref` attribute for controlling the best path selection. When the BGP-controller controls any Cisco routers, it is assumed that these routers do not use the `weight` attribute internally since the use of this attribute protects control about concerning BGP entries from the outside.

These selection rules except rule (1) must be implemented in all border routers as the standard and should not be changed in designing the routing control architecture. Therefore, the VR architecture which modifies `local_pref` and `next_hop` attributes can control the forwarding addresses of the next hop in the path to the destination, and therefore can control conventional routers externally without any protocol extensions. Details of this process are described in section 3.3 and 3.4.

The policy-control-engine and the cooperative-action-controller are constructed on the agent platform which is a basic component of the ENCORE agent [4]. This platform provides basic primitive functions for distributed environments and is implemented using CLOS, which is a Common Lisp extension language. The BGP-controller is implemented as another process and written also in CLOS. The BGP-controller and other modules in an agent communicate by sending data acquisition requests, control requests, and status information by RPC/SSL/TCP.

3.2 Policy description

The syntax of policy description is shown in Fig. 5. `Rule` is a unit to describe a set of actions. It consists of an `acq` function that designates how to acquire data and an `eval` function that designates how to evaluate obtained results. `Rule` is called from `policy` or `strategy`. The `policy` form, which includes some `trigger-event` names, is directly invoked by the agent, other agents, or operators when a `trigger-event` name matches the issued event message. The `strategy` form is for describing timer-driven activation and is invoked by an internal timer.

```
goal ::= {policy}+ | {strategy}+
policy ::= {policy}* | {rule}+ | {reaction}*
  | {sub-goal}* [| trigger-event] [| priority]
strategy ::= {rule}+ | time-spec
reaction ::= function
rule ::= acq-proc | eval-proc [| type]
```

Figure 5: Syntax of action description on agent

When an agent sends requests to other agents, it must know where suitable agents are located. In this case, the agent uses the agent group organization system called ARTISTE[16] that manages capability or roles of agents combined with the BGP topology map. Hence, the agent can issue queries such as to find agents that are located at ASs that have many peer ASs, neighbor agents within n AS-hops from AS_x, or agents that are located downstream from AS_x

3.3 BGP control architecture

In this section, we explain how the BGP-controller works on the assumption that the BGP-controller can know the information about alternative routes for simplifying explanation, and then we describe how the BGP-controller infers and confirms alternative routes in section 3.4.

Suppose AS_x has two BGP peers, AS_i and AS_j. In this case, AS_x typically has two BGP entries for a destination AS_y. If the BGP entry via AS_i has a higher `local_pref` value lp_i than that via AS_j, packets destined for AS_y are forwarded to AS_i. By inserting a new BGP entry about AS_y, which has a higher `local_pref` value than that of lp_i and whose `next_hop` address is AS_j, the agent can change the next-hop AS from AS_i to AS_j for forwarding packets destined for AS_y. Therefore, outbound traffic can be controlled by adjusting these attribute values. Interface functions that are called from an agent to control BGP information are shown in Fig. 6. The BGP-controller has a table for all BGP entries and their attribute values and maintains various statistical values concerning its BGP table such as the number of the best paths per peer.

Usually border routers exchange eBGP (external BGP)-learned routes with other border routers in the AS by using iBGP full-mesh connections or a route reflector [6] because iBGP-learned routes must not be re-advertised to other BGP peers according to the definition in [12]. If a BGP-controller wants to modify `local_pref` attributes for changing the best path among candidate entries, the former full-mesh connection method is not suitable. Full-mesh connections allow the best path information to flow directly between any two routers. These flows prevent information about alternative route entries from flowing to the BGP-controller. As a result, the BGP-controller cannot change an alternative path to the best path in this architecture. Therefore, the BGP-controller adopts an architecture like that of the route reflector and connects to all target border routers in an AS. All BGP information controlled in an AS flows via a BGP-controller, which forwards the best-path information among BGP entries to other peers like route reflectors. Note that there is no iBGP connection between any two routers.

The following explains this operation for changing the best path in detail. At the initial state, as shown in Fig. 7, packets destined for an IP prefix, which is the network portion of the IP address,

```
;;; --- For controlling BGP routing table ---
(get-global-status)
  ; Gets statistical values such as the
  ; number of best paths per peer etc.
(search-entry &key prefix as-path);
  ; Gets BGP entry info: the best path and
  ; alternative paths, and their attributes.
; Changes the best paths according to
; given conditions.
(change-best-path-by-prefix
  prefix-list ; Target BGP entries
  peer-id )   ; New peer to be changed
(change-best-path
  peer-id  ; New peer to be changed
           ; for forwarding packets.
  &key ;;; The following vars are keyword optional.
  current-peer-id  ; Current peer
  origin-as ; Target entries whose origins are
            ; specified AS.
  as-path-from ; Target entries whose AS paths
               ; match specified regular expression.
  as-path-to   ;
  number ; Maximum number of entries
  )      ; to be operated.
; Resets applied policies according to given conditions.
(reset-best-path-by-prefix
  prefix ) ; Target BGP entries
(reset-best-path
  &key current-peer-id
  origin-as
  as-path ; Target entries whose AS paths
          ; match specified regular expression.
  number ); Maximum number of entries to be operated.
  ; All applied policies are cleared if invoked
  ; with no args.
(set-trap
  cond ; Conditional function which is applied
       ; with update messages.
  event ; Function invoked if cond is t.
  )     ; This function typically issues events.
```

Figure 6: Control functions of the BGP-controller

e.g., `a.b.c.0`, are forwarded to `x.x.x.1`, which is the IP address of router x's BGP-peer because the #1 entry has the highest `local_pref` value, 1000. If this agent decides to change the next hop to `y.y.y.1`, which is the IP address of router y's BGP-peer and is an alternative route shown as the #2 entry, first, the BGP-controller copies the #2 entry and makes a new entry, #3. Then, the BGP-controller sets the highest value to `local_pref` of the #3 entry, whose value is 2000 in this example. Therefore, #3 becomes the best path, which is indicated by ' >' in the figure. Then, the BGP-controller sends #3 to router x and the best path in router x changes from #1 to #3. At the same time, the BGP-controller also sends a withdrawal message about `a.b.c.0` to router y. As a result, the best path in router y changes from #1 to #2. Therefore, in both routers, the next hop changes to `y.y.y.1`. This state information is maintained in the BGP-controller table. If the original route entry, #2, is withdrawn for some reason, the withdrawal message is delivered from router y to the BGP-controller by the BGP protocol. Then, the BGP-controller must delete the copied route because the path via `y.y.y.1` does not exist any more. In this case, the BGP-controller sends the withdrawal message about `a.b.c.0` to router x to delete #3 and the best path changes to #1. This operation should be done as soon as possible because forwarding packets destined for `a.b.c.0` to `y.y.y.1` results in a black hole. The BGP-controller must also send #1 to router y again because #1 had been deleted by the previous operation in router y and does not have any route to `a.b.c.0`. Therefore, damping [17] procedures should not be done in these actions.

3.4 Inference and confirmation of alternative routes

As described before, the BGP-controller can not have any alternative route information when it acts as the route reflector, because

(Initial state)

#	Dest IP	local_pref	next_hop	(ID, src)
1	> a.b.c.0	1000	x.x.x.1	(x, -)
2		500	y.y.y.1	(y, -)

(State after the best-path change operation)

#	Dest IP	local_pref	next_hop	(ID, src)
1	a.b.c.0	1000	x.x.x.1	(x, -)
2		500	y.y.y.1	(y, -)
3	>	2000	y.y.y.1	(vr, y)

Figure 7: BGP Global state as the whole AS

any border router that selects the BGP information from the BGP-controller as the best path must send the withdrawal message to the BGP-controller to tell the previously selected best path, which was received from one of peer ASs, is no more the best path. As the results, the BGP-controller can not have any alternative BGP entries to the same prefix. Therefore the BGP-controller performs additional actions to infer and confirm alternative paths to which it tries to change the best-paths. The detailed actions are as follows:

1. The BGP-controller maintains internal records about BGP entries even if these entries are withdrawn. In this case, the record is only marked as withdrawn. If the BGP-controller receives a withdrawal message just after it sends an advertise message which was sent by another router, it is also marked as a candidate.

2. For finding alternative route entry e_1 about a prefix p_1, the BGP-controller first extracts an internal record whose prefix is p_1 and whose state is marked as a candidate.

3. For confirming whether the alternative route e_1 currently exists, the BGP-controller sends a BGP update message to the router r_1 from which e_1 was sent. This message is an advertisement of the currently selected best-path for p_1 but the `local_pref` value is set to the lowest value among all routers. As the result, r_1 must send an advertisement message to the BGP-controller if r_1 has e_1, which was received from one of peer ASs and is the newly selected best path.

4. If the BGP-controller receives this entry e_1, whose `next_hop` differs from the current one, it has confirmed that the alternative path e_1 exists. Therefore, the BGP-controller can send the alternative entry to other routers except original r_1 after it modifies e_1's `local_pref` value with higher one as explained in section 3.3. The advertisement of p_1 to r_1 is withdrawn.

5. If the BGP-controller does not receive any BGP entry information within pre-defined time, it means this alternative route e_1 does not exist. Therefore the BGP-controller tries to find anther candidate for an alternative path.

6. If the BGP-controller receives a withdrawal message about p_1 from r_1 after confirmation, the BGP-controller similarly try to find another candidate.

4. DISCUSSION

4.1 Evaluation results of BGP control

This section discusses the effectiveness of the system from several viewpoints using some evaluation results. In these experiments, BGP full-route information, which consists of more than 160000 BGP entries, from three upstream Tier-1 peers was used. The quagga software [1] and the Cisco 7200 are used as border

routers in this evaluation environments. The VR ran on FreeBSD 4.11 / Pentium4 3.2GHz with 1 GB memory.

The construction of a routing table in a BGP-controller by using BGP full-route information from two routers, both of which are quagga, took about 27 [s]. The construction of one from three routers, which are two quagga routers and one Cisco 7200, took about 61 [s], although the Cisco completed sending all data within 20 [s]. When the BGP-controller worked with no given policy, which means it acted like a route reflector [6] of these three routers except the inference and confirmation actions, the initialization phase requires additional 77 [s]. On the other hand, the BGP-controller took only about 6 [s] for receiving all data from a single quagga router. It contains time for generating Lisp objects from TCP binary streams, but does not contain time for constructing a local routing table, calculating the best paths, nor notifying selected best path information to other peers. The dominant bottleneck in the initialization phase is the cost for the route reflection.

The time for changing the BGP best paths of specified number of entries is shown in Fig.8. The additional cost of changing routes using the BGP-controller is relatively small compared with that initial cost. In the application area where upstream ISPs can receive route

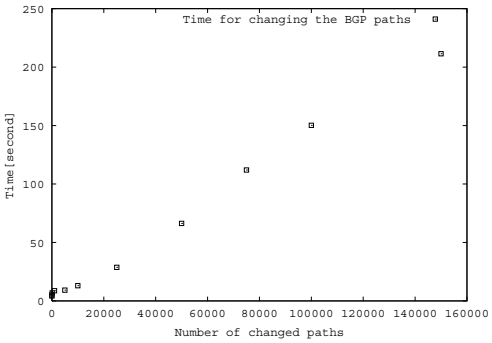

Figure 8: Times for changing the BGP best paths

change requests from downstream customers, the assumed number of changed paths would be less than 10000. Therefore, this cost is sufficiently small compared with one at the initial phase.

According to results in our AS observed for one month, the average numbers of BGP advertisement and withdrawal messages per minute are 59 and 9, respectively. They are sufficiently small compared with the number of full-route entries that the system must receive at the initialization phase. Although bursts in the number of updates, such as more than 20000 entries of updates per minute, were observed six times, most of the updates only modified the AS-path attributes and did not affect selection of the best paths. In the case where forwarding-peers of one thirds of the all BGP entries, namely 50000, are changed, this operation took about 66 [s]. On the other hand, each continuous policy adjustment did not affect the best path adjustment, because almost all update messages only indicated changes of AS paths, and they did not affect the currently selected best paths. Although withdrawal messages about currently selected BGP entries as the best paths should be checked to maintain consistency in this kind of policy applications, this cost is similarly small sufficiently.

Therefore, these evaluation results demonstrate the effectiveness of this VR BGP-control architecture as far as our target domain is concerned, where periodic adjustments at the coarse-grained level are performed. The problems of slow convergence and fluctuation would not be outstanding in this application domain. In future work, feasibility in the case where interactions among participant ASs are more complicated and/or stability when more number of

```
;;; --- For observation of network status ---
; Defines observation strategy
(def-strategy observe-table
  (:interval 1200) ; [s]
  ; Triggered by the internal timer
  (rule check-and-warn-best-path-balance) )
(def-rule check-and-warn-best-path-balance
  ; Gets statistical info of the local BGP table
  (acq get-global-status)
  ; Issues the event 'not-balanced-in-number
  ; if condition is satisfied
  (eval issue-warning-if-not-balanced) )
;;; --- For feedback actions ---
; Defines adjustment policy
(def-policy distribute-entries-in-number
  (trigger-event 'not-balanced-in-number)
  ; Triggered by received event
  (rule distribute-entries-in-number) )
; Defines adjustment function
(def-rule distribute-entries-in-number
  ; Gets statistical info of the local BGP table
  (acq get-global-status)
  (eval change-best-path-by-number
    (peer-id
     ; Peer-id which the smallest number of
     ; BGP entries uses for forwarding packets.
     current-peer-id
     ; Peer-id which the largest number of
     ; BGP entries uses.
     number ))
     ; Number of alternative paths that should be
     ; changed to the best paths for using this peer.
  ; Invokes (change-best-path peer-id :current-peer-id
  ;           current-peer-id :number number)
```

Figure 9: Policy description for observation and adjustment of outgoing packets

entries are controlled under unintended burst traffic flow will be examined

4.2 Application Scenarios

4.2.1 Outbound traffic control

When an operator in an AS that has several BGP peers distributes outbound packets uniformly among the best paths, the agent in the AS periodically acquires statistical information about its BGP table, which includes the number of BGP best paths per peer. In other words, a `strategy` periodically invokes a `rule` for observation. If the number of best paths to a peer exceeds a calculated average by a given threshold value, this `rule` issues a warning to the agent. The `policy` that matches this warning message is invoked and a rule for adjustment is performed, as shown in Fig.9.

Load balancing based on the amount of traffic requires an agent to know the traffic status of each interface, by which this AS is peering with a neighbor AS. In this case, the traffic rate, such as bits per second at interfaces of border routers can be used. According to this result, the agent tries to adjust the number of BGP best-path entries per peer. Therefore, the agent searches the best path entries whose forwarding paths use the interface that has the highest traffic rate and makes other alterative entries, which use lines that have lower traffic rates, the best paths. Although distributing traffic uniformly at the fine-grained level is difficult, especially when traffic fluctuates frequently, periodic adjustments at the coarse-grained level could be helpful for operators in assumed application domain.

4.2.2 Inbound traffic control

In the case for inbound traffic control shown in Fig. 1, an agent in AS_x can send a `route-preference` request to an agent in AS_k. The request contains an AS list such as $\{AS_i, AS_j..AS_n\}$. The agent in AS_k tries to use the route via AS_i according to the `route-preference` list from AS_x because AS_k has two BGP routes destined for AS_x via AS_i and AS_j and their costs are the

same from the viewpoint of AS_k. This best-path control mechanism performed in AS_k is similar to that for outbound packets. This process is only performed at the preference level because AS_x and AS_k are managed by different authoritative organizations. We assumed the agent in AS_k is not prohibited by AS_i's policies from selecting the route via AS_i as the best path. Although the actual BGP topology is more complicated and an optimal solution does not necessarily exist in all cases, a part of the inconsistencies can be resolved by detecting them and coordinating routing through cooperation among multiple ASs. In this case, AS_k can provide its customers and their downstream customers with a route selection service, where a customer such as AS_x can send its route preference to its upstream ISP such as AS_k under their contract. The group management and authentication functions are similar with ones used in ENCORE-2.

If a requested AS has a policy that is inconsistent with a requested action, a resolving process is required. applied. First, the policy from a policy description concerning the local AS precedes all other policies. Then, requests for routing destined directly to the requesting AS take precedence. Requests concerning downstream ASs are allowed to be executed if there is no direct request from downstream ASs. Introducing these priority rules can resolve inconsistent states. As a result, some requests might be refused. The requesting AS should be notified of this situation so it can try the following policy candidates.

5. RELATED WORK

The Routing Control Platform (RCP) [8, 7] and the 4D architecture [9] have the concept of separation of the data-plane and control-plane and are consistent with our approach based on the control using the multi-agent architecture [2]. The main difference is that our VR architecture has functions for the best path changing at the on demand basis, which are enabled by alternative route inference and confirmation functions.

Although some extensions of the community attribute for policy control were proposed [14], only the mechanism to distribute additional values on BGP is defined, and inter-AS routing adjustment or coordination functions are not discussed. The path selection mechanism of BGP paths and overlay routing was reported in [5], but that discusses routing at the fine-grained level such as a unit of a session or a packet. Inter-AS control is not treated. Our system focuses on control traffic at the macro level or coarser-grained level including inter-AS control, considering observed results, and the given policy description. Intelligent routers [13, 11] are also capable of controlling outgoing packets by modifying received BGP information according to the given policy description, but they do not provide cooperative actions among multiple ASs and they require special devices. On the other hand, the VR architecture works with conventional routers without any protocol extensions. RAML[10], a metadescription approach was reported, but it cannot represent feedback control according to observed network status. Active network approaches [15] provide similar control functions, but they do not consider control structures like ASs. The CDPS approach in AISLE adopts the cooperation on a request-and-acceptance basis and that coincides with the actual network management structure.

6. CONCLUSION

For autonomous and adaptive routing control, we have proposed an inter-AS policy-based routing control system called AISLE that uses multiple cooperative agents. Each AISLE agent has the BGP-controller and the policy-control-engine for monitoring and adjusting BGP information according to the observed network status and given policy description. This VR architecture controls conventional BGP routers under a given policy description without any protocol extensions. Some evaluation results demonstrate that the AISLE framework with VR can effectively perform routing adjustment on our target domains.

7. REFERENCES

[1] The quagga routing suite. http://www.quagga.net.

[2] O. Akashi, K. Kourai, K. Sato, T. Hirotsu, M. Maruyama, and T. Sugawara. "Agents Support for Flexible Inter-AS Policy Control". In *Proc. of SAINT'03 Workshops / AI on the Internet*, pages 294–298. IEEE / IPSJ, Jan 2003.

[3] O. Akashi, T. Sugawara, K. Murakami, M. Maruyama, and N. Takahashi. "Multiagent-based Cooperative Inter-AS Diagnosis in ENCORE". In *IEEE/IFIP Network Operations and Management Symposium*, pages 521 – 534. IEEE Press, Apr 2000. ISBN0-7803-5928-3.

[4] O. Akashi, A. Terauchi, K. Fukuda, T. Hirotsu, M. Maruyama, and T. Sugawara. "Detection and Diagnosis of Inter-AS Routing Anomalies by Cooperative Intelligent Agents". In *16th IFIP/IEEE Int'l Workshop on Distributed Systems: Operations and Management*, pages 181–192, Oct 2005. LNCS 3775.

[5] A. Akella, J. Pang, B. Maggs, S. Seshan, and A. Shaikh. "A Comparison of Overlay Routing and Multihoming Route Control". In *Proc. of SIGCOMM*, pages 93–105. ACM, Aug-Sep 2004.

[6] T. Bates and R. Chandra. "BGP Route Reflection", 1996. RFC1966.

[7] M. Caesar, D. Caldwell, N. Feamster, , J. Rexford, A. Shaikh, and J. Merwe. "Design and Implementation of a Routing Control Platform ". In *Proc. of Networked Systems Design and Implementation (NSDI)*. USENIX, May 2005.

[8] N. Feamster, H. Balakrishnan, J. Rexford, A. Shaikh, and K. Merwe. "The Case for Separating Routing from Routers". In *Proc. of ACM SIGCOMM Workshop on Future Directions in Network Architecture (FDNA)*. ACM, Sep 2004.

[9] A. Greenberg, G. Hjalmtysson, D. A. Maltz, A. Myers, J. Rexford, G. Xie, H. Yan, J. Zhan, and H. Zhang. "A Clean Slate 4D Approach to Network Control and Management". *Computer Communication Review*, 35(5), Oct 2005.

[10] T. G. Griffin and J. L. Sobrinho. "Metarouting". In *Proc. of SIGCOMM*, pages 1–12. ACM, Aug 2005.

[11] radware. Peer Director. http://www.radware.com/contents/products/pd/.

[12] Y. Rekhter and T. Li. "A Border Gateway Protocol 4 (BGP-4)", 1995. RFC1771.

[13] RouteScience Technologies, Inc. RouteScience PathControl. http://www.routescience.com/products.

[14] S. Sangli, D. Tappan, and Y. Rekhter. "BGP Extended Communities Attribute", 2005. draft-ietf-idr-bgp-ext-communities-08.txt.

[15] J. M. Smith and S. M. Nettles. Active Networking: One View of the past, Present, and Future. *IEEE Transaction on System, Man, and Cybernetics*, 34(1):4–18, 2004.

[16] A. Terauchi, O. Akashi, M. Maruyama, K. Fukuda, T. Sugawara, T. Hirotsu, and S. Kurihara. "ARTISTE: An Agent Organization Management System for Multi-agent Systems". In *8th Pacific Rim Int'l Workshop on Multi-Agents (PRIMA)*, pages 245–259. IFMAS, Sep 2005.

[17] C. Villamizar, R. Chandra, and R. Govindan. "BGP Route Flap Damping", 1998. RFC2439.