INTERNET DRAFT
September 2002
Expiration Date: March 2003

Jun-Hong Cui
Mario Gerla
Khaled Boussetta
/UCLA
Michalis Faloutsos
/UC Riverside
Aiguo Fei
Jinkyu Kim
Dario Maggiorini
/UCLA

Aggregated Multicast: A Scheme to Reduce Multicast States

<draft-cui-multicast-aggregation-01.txt>

Status of this Memo

Abstract

In this document, we present a novel scheme, called aggregated multicast, to reduce multicast states ([FeiGI01] and [FeiNGC01]). The key idea is that multiple groups are forced to share one distribution tree, which we call aggregated tree. In our scheme, core routers need to keep states only per aggregated tree instead of per group. This can significantly reduce the total number of trees in the network and thus reduce forwarding states.

We investigate the implementation issues of aggregated multicast in different network scenarios. We also discuss the effects of aggregated multicast on some important issues, such as QoS multicast provisioning, mobility support and fault tolerance.

The scope of this paper is not to propose a detailed protocol,

but present the idea of aggregated multicast at high level and show its merits.

Table of content

1. Introduction

IP multicast utilizes a tree delivery structure, on which data packets are duplicated only at fork nodes and are forwarded only once over each link. This approach makes IP multicast resource-efficient in delivering data to a group of members simultaneously and scalable in supporting very large multicast groups. However, a multicast distribution tree requires all tree nodes to maintain per-group (or even per-group/source) forwarding state, and the number of forwarding state entries grows with the number of "passing-by" groups. As multicast gains widespread use and the number of concurrently active groups grows, more and more forwarding state entries will be needed, specially in transit domains. More forwarding entries translate into more memory requirements, and may also lead to slower forwarding process since every packet forwarding involves an address look-up. Thus, though IP multicast scales well to the number of members within a single multicast group, it suffers from scalability problems when the number of simultaneous active multicast groups is very large. The scalability regarding to multicast state is often referred as state scalability.

State scalability problem is a very challenging for multicast routing, especially in backbone networks where the number of

active groups may be huge. Recently, this problem has prompted some research proposals, which can be classified into two categories: (1) suppressing multicast states at some routers; (2) aggregating multicast states.

Examples of the first category are [Tian98], [Chu00], and [Francis00]. In [Tian98], Tian and Neufeld propose a scheme that attempts to reduce forwarding states by establishing dynamic unicast tunnels over non-branching paths of a multicast distribution tree. By doing so, the multicast forwarding states

at non-branching routers could be eliminated. However, this scheme requires a complex method in order to identify non-branching routers and it only applies in sparse networks. Other schemes proposed in [Chu00] and [Francis00] aim to completely eliminate multicast states at all routers. In these schemes, a self-organizing tree (and/or mesh) among group members is constructed to provide multi-point communications among them without network-layer multicast support, which pushes complexity to end-points. These solutions might be good alternatives for small-scale multicast applications; however, it is difficult for them to scale up to support multicast applications with millions of group members like Internet TV.

Examples of the second category are [Rad99] and [Thaler00]. Thaler and Handley analyze the aggregatability of forwarding states using an input/output filter model of multicast forwarding in [Thaler00]. Radoslavov et al. propose algorithms to aggregate forwarding states and study the bandwidth-memory tradeoff with simulations in [Rad99]. However, these state aggregation schemes attempt to aggregate routing state after the distribution trees have been established, and they tend to change the state format maintained at routers, which is generally not desired by many service providers [ID-MAT]. Furthermore, the state aggregatability of this type of schemes heavily depends on multicast address allocation.

In this document, we present a novel scheme, called aggregated multicast, to reduce multicast states ([FeiGI01] and [FeiNGC01]). Unlike previous approaches, our scheme forces multicast multiple groups to share one distribution tree, which we call an aggregated tree. In our scheme, core routers need to keep states only per aggregated tree instead of per group. This can significantly reduce the total number of trees in the network and thus reduce forwarding states. While this approach significantly reduces the number of forwarding state entries and alleviates overhead

associated with tree management, it may also waste bandwidth as it delivers data to non-group-members. There is thus a trade-off between control overhead reduction via aggregation and bandwidth wastage introduced by common tree sharing. To find a good compromise point, we use a group-tree matching algorithm.

The rest of this document is organized as follows. Section 2 introduces the aggregated multicast scheme. Section 3 addresses the implementation issues. Section 4 describes how aggregated multicast helps to achieve scalability in QoS multicast provisioning. Then section 5 and 6 discuss the impacts of our scheme on mobility support and fault tolerance separately.

2. Aggregated Multicast

Aggregated multicast is proposed to reduce multicast states, and it is targeted to intra-domain multicast provisioning. The key idea of aggregated multicast is that, instead of constructing a tree for each individual multicast group in the core network (backbone), multiple multicast groups are forced to share a single aggregated tree.

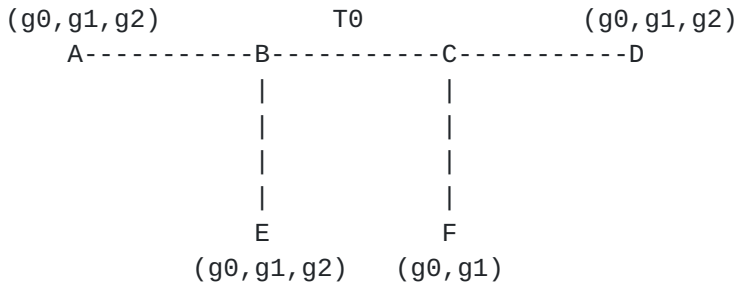

Fig. 1: An illustration of aggregated multicast

Fig. 1 illustrates the basic idea of aggregated multicast. We assume that A, B, C, D, E and F are routers in a transit domain (noted by DX). A, D, E and F are edge routers, while B and C are core routers. A multicast group g0 enters domain DX at router A and leaves domain DX at routers D, E, and F. Then routers A, B, C, D, E, and F form an intra-domain multicast tree within domain DX. Consider a second multicast group g1 that also has member requests from A, D, E and F. For this group, a tree with exactly the same

set of nodes will be established to carry its traffic within domain DX.

In conventional IP multicast, all the nodes in the above example that are involved within domain DX must maintain separate states for each of the two groups individually though their multicast trees are actually of the same "shape". Alternatively, in aggregated multicast, we can set up a pre-defined tree T0 (or establish on demand) that covers nodes A, B, C, D, E, and F using a single multicast group address (within domain DX). This tree is called an aggregated tree and it is shared by more than one multicast groups (two groups in the above example). Data from a specific group enters the transit domain at the traffic injecting nodes (or called incoming edge routers). It is then distributed over the aggregated tree and forwarded by the traffic exiting nodes (or called outgoing edge routers) to neighboring networks. This way, core routers B and C only need to maintain a single forwarding entry for the aggregated tree regardless how many groups are sharing it.

Thus, aggregated multicast can reduce the required multicast states. Core routers don't need to maintain states for individual groups; instead, they only maintain forwarding states for a smaller number of aggregated trees. The management overhead for the distribution trees is also reduced. First, there are fewer

trees that exchange refresh messages. Second, tree maintenance can be a much less frequent process than in conventional multicast, since an aggregated tree has a longer life span.

In aggregated multicast, we need to match groups to aggregated trees. The group-tree matching problem hides several subtleties. The set of the group members and the tree leaves are not always identical. A match is a perfect match for a group, if all the tree leaves have group members. For example, in Fig. 1, T0 is a perfect match for groups g0 and g1. A match may also be a leaky match, if there are leaves of the tree that do not have group members. In other words, we send data to parts of the tree that is not received by anyone. In Fig. 1, T0 is a leaky match for group g2. A disadvantage of the leaky match is that some bandwidth is wasted to deliver data to nodes that are not members for the group. Namely, we trade off bandwidth for state reduction.

In order to find a good compromise point between state and tree management overhead reduction and bandwidth waste, aggregated multicast uses a group-tree matching algorithm which selects

a multicast tree for each group carefully depending on a creteria function. An example algorithm can be found in [FeiNGC01].

3. Implementation Issues

To implement aggregated multicast, there are several issues to concern.

First of all, some scheme has to take the responsibility of distributing multicast traffic of different groups over a shared aggregated tree. For any implementation, there are two requirements: (1) original group addresses of data packets must be preserved somewhere and can be recovered by outgoing edge routers to determine how to further forward these packets; (2)some kind of identification for the aggregated tree which the group is using must be carried and core routers must forward packets based on that. One possibility is to use IP encapsulation, which could add complexity and processing overhead (at edge routers). Another possibility is to use MPLS (MultiProtocol Label Switching) ([RFC3031]), in which labels can identify different aggregated trees.

To handle aggregated tree management and matching between multicast groups and aggregated trees, a logical management entity called tree manager is introduced. A tree manager has the knowledge of established aggregated trees and is responsible for establishing new ones when necessary. It collects (inter-domain) group join messages received by border routers and assigns aggregated trees to groups. Once it determines which aggregated tree to use for a group, the tree manager can install corresponding state at those edge routers involved, or distribute corresponding label bindings if MPLS is used. Aggregated tree construction within

the domain can use an existing routing protocol such as PIM-SM, or use MPLS signaling protocols extensions proposed in [ID-MPLS-MCAST] which allow the establishment of pre-calculated trees. It should be noted that a tree manager is not necessary a single point, but it is a logical entity which can be implemented in either centralized or distributed manner depending on the specific protocol design.

In the rest part of this section, we will discuss the implementations of aggregated multicast in different network scenarios.

3.1. Implementations in IP networks without MPLS support

In IP networks without MPLS support, to implement aggregated multicast, we have to employ IP-encapsulation. In the following, we describe the main implementation issues for different multicast routing protocols.

3.1.1. Source specific multicast

In source specific multicast ([Holbrook99] [ID-SSM-ARCH] [ID-SSM-DEPLOY]), a group is identified by a combination of a source address and a D-class address. The main advantage is that address allocation is no longer a problem. Fig. 2 gives an example of a source specific tree. When dealing with source specific multicast in a transit domain, we only consider aggregating groups originating from the same edge router. In this way, the tree manager can be implemented distributedly: each edge router can have tree manager functionality, which is only responsible for managing groups and trees originating from the edge router itself. The encapsulation will be performed on the incoming edge router, which also operate as a tree manager. The decapsulation will be done in outgoing edge routers, which will forward multicast packets to the corresponding receivers.

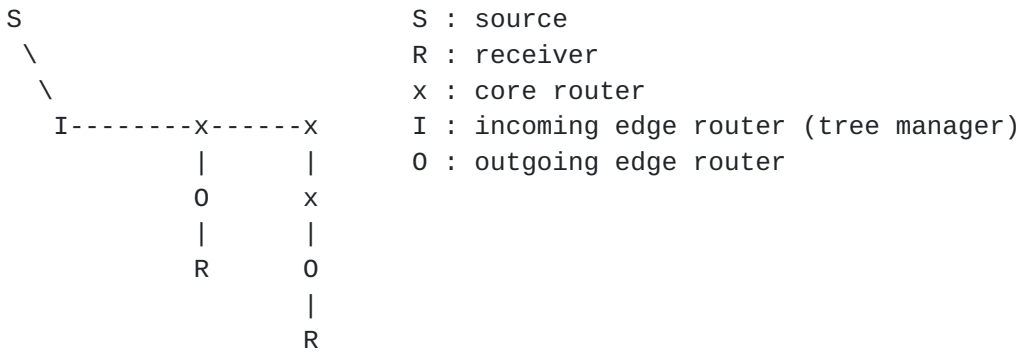

Fig. 2: Source-specific multicast

3.1.2. Uni-directional shared tree multicast

In uni-directional shared tree multicast routing protocols, such as PIM-SM ([RFC2362] [ID-PIM-SM-REVISED]), multiple sources of a

group (identified by a D-class address, which is different from source specific multicast) first unicast data to its RP (Rendezvous Point), then all the traffic will be delivered on a uni-directional tree rooted at the RP. In this scenario, we only aggregate groups which have the same RP. RPs will act as tree managers. Correspondingly, the encapsulation can be done in RPs,

and the decapsulation can be performed in outgoing edge routers. Fig. 3 gives an example of uni-directional shared tree.

If a shortest path tree is activated by a source of a group (eg. S2 in Fig. 3.), the source will not unicast data to its corresponding RP and simply utilize its shortest path tree to deliver data to the receivers. Upon this case, further aggregation can be realized for each source, which is very similar to source specific multicast except that the multicast addressing is different.

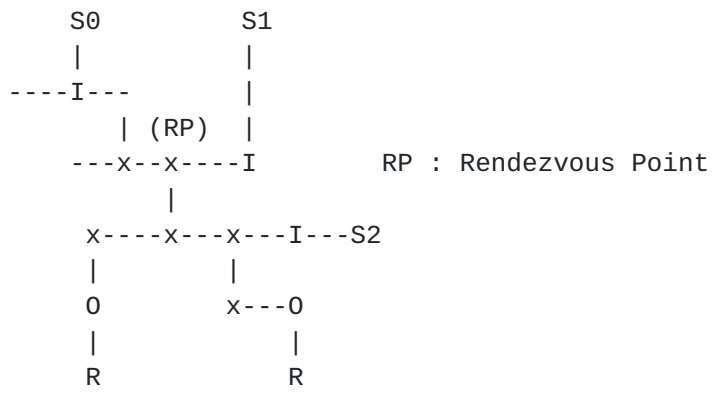

Fig. 3: Uni-directional shared tree multicast

3.1.3. Bi-directional shared tree multicast

In bi-directional shared tree multicast routing protocols, such as CBT ([RFC2189] [RFC2201]) and BIDIR-PIM ([ID-BIDIR-PIM]), a RP or core node only acts as a routing auxiliary node at which no unicast traffic concentrates. Every on tree node can receive data from a source and then forward it to the group members along the bi-directional tree. This type of routing protocol is specially good for multiparty interactive applications, such as video conferencing and netgames. Using bi-directional trees can improve aggregatability since any group which is covered by a bi-directional tree (that is, all the group members are on tree nodes) can share the tree no matter where the sources or receivers of the groups are. Fig. 4 shows an example of bi-directional shared tree.

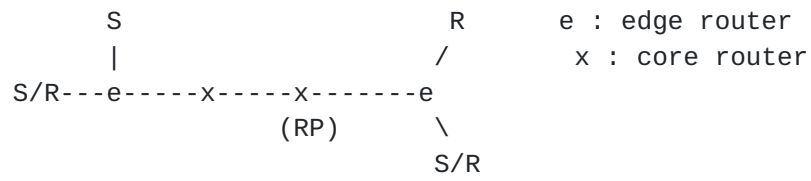

Fig. 4: Bi-directional shared tree

In this scenario, tree management can be handled at RP, while encapsulation and decapsulation need to be done in every involved edge router, which encapsulates every incoming packet from sources and decapsulates every outgoing packet.

It should be noted that, in the above descriptions, multiple tree managers (one tree manager for a set of groups which have the same source or RP) are employed instead of a centralized one. In order to achieve load balancing, different tree managers can communicate with each other and come up with better tree management policies.

3.2. Implementations in IP networks with MPLS support

If the target network supports MPLS, then IP encapsulation will be replaced by label switching, which is much more efficient.

In an MPLS domain, when a stream of data traverses a common path, a Label Switched Path (LSP) can be established using MPLS signaling protocols. At the ingress Label Switch Router (LSR), each packet is assigned a label and is transmitted downstream. At each LSR along the LSP, the label is used to forward the packet to the next hop.

To deliver multicast traffic, we need to establish MPLS trees (that is, labeled multicast trees). If MPLS trees are mapped directly from level 3, namely IP level, all the considerations about tree management in networks without MPLS support can be simply applied here. If MPLS trees are established by explicit routing, besides group-tree matching, a tree manager is also responsible for computing multicast trees. Then it uses MPLS signalling protocols extensions (eg. [ID-MPLS-MCAST]) to set up the MPLS trees.

4. QOS Considerations

As we have discussed, current IP multicast suffers from state scalability problems. In QoS multicast provisioning, the problem is exacerbated, since not only the forwarding state but also the multicast group's resource requirement must be kept at the router. In this section, we investigate how aggregated multicast helps to achieve scalability in QoS multicast support.

4.1. Aggregated multicast support in DiffServ domains

Differentiated Service architecture (DiffServ) ([RFC2475]) is proposed for scalable service differentiation in the Internet. In a DiffServ domain, packets of the same QoS requirements constitute an aggregated flow. At the ingress edge routers, the packets are classified and marked with a DiffServ Code Point (DSCP) according to their QoS requirements. At each core router, the DSCP is used to determine the behavior for each packet. In

this way, DiffServ reduces QoS states and puts complexity to edge routers.

Though DiffServ achieves scalability for unicast, it does not solve routing state scalability problem for multicast. Using aggregated multicast can simplify traffic management and facilitate QoS provisioning for multicast by: (1) pushing per group multicast state to network edges and reducing multicast state in the network core; (2) pre-assigning resource/bandwidth (or reserve on demand) only for a small number of aggregated trees. Note that, since, in DiffServ, core routers determine the behavior for each packet simply based on its DSCP (without keeping each flow's QoS information), different groups routed on the same aggregated tree can have different QoS requirements. In other words, core routers do not need to differentiate groups which share one single tree. Thus, to aggregate multicast groups in a DiffServ domain, the additional functionalities are group-tree matching handled by tree managers and en/decapsulation performed by edge routers. How to implement tree managers depends on specific multicast routing protocols employed. Moreover, call admission control module in Bandwidth Broker can cooperate with tree managers to achieve load balancing.

4.2. Aggregated multicast support in IntServ domains

Integrated Service architecture (IntServ) ([RFC1633]) is proposed for guaranteed services in the Internet. It is a micro-flow based QoS architecture and requires per-flow reservation and data packet handling. Thus, IntServ has QoS state scalability problem at network core routers. When supporting multicast, IntServ will face serious multicast routing state scalability problems. Using aggregated multicast, the routing state scalability can be improved: by aggregating multiple groups into one tree, the number of multicast flows maintained at core routers can be reduced significantly, and thus the corresponding overhead of reservation and packet handing is reduced too. However, in IntServ, only groups which have same QoS requirements can share the same tree, which is different from the aggregation policy in DiffServ. This is because there no other way to identify flows with different QoS requirements in IntServ.

4.3. Incorporation with Unicast

Aggregated multicast scheme may have some effects on unicast traffics. State reduction is a main advantage for unicast flows; the saved look-up time of multicast traffics will benefit unicast

packet routing. On the other hand, in order to achieve more aggregation, some bandwidth might be wasted, which could affect the admission control of unicast traffic. Thus, to maximize the utilization of network resources, a good tree management policy has to fall into place, specially taking into account the load balancing issue.

5. Mobility Support

As we have mentioned, in leaky match cases, there is bandwidth waste since some data packets are delivered to non-group-members. There is a trade-off between aggregation and bandwidth wastage. Actually, our aggregated multicast scheme may offer additional advantages even in leaky match cases.

We look at the following scenario. Assume the designated router (denoted by DR) of an end user (denoted by M) is a leaf node of an aggregated tree shared by group G, and DR is not a member router of G before M subscribes. Then when M sends subscription request to G, it will observe lower join latency in our scheme than in conventional multicast since DR is already on the distribution tree. This feature can efficiently support multicast mobile users.

As the mobile member moves, it may happen that its first foreign IP router of the mobile is not able to forward data. In this case, an IP level handoff must be performed. IP level handoff for unicast traffic is often complex because it require the discovery of the new care-of IP address. However, since individual destination address is not needed for multicast support, the difficulty mainly lies in the multicast tree reconfiguration. Performing this operation inside the mobile network domain may also implies a tree reconfiguration in the transit domain. In this case, besides the signaling overhead, the mobile may experience a long distribution period, resulting in high packet losses. With aggregated multicast scheme, IP level handoff for multicast mobile receivers may be much simpler under leaky match cases, since many tree configurations can be just simply omitted.

6. Fault Tolerance

In some applications (e.g. battlefield communications, distributed visualization and control, etc.), it is important to provide fault tolerant multicast. For example, consider the control of a space launch carried out from different ground stations interconnected by an Internet multicast tree. This control scenario may require the exchange of real time, interactive data and streams. For these applications, pre-planned restoration mechanism is necessary, since it potentially reduce restoration time. In pre-planned mechanism, a backup tree of each multicast tree is set up before hand. Using aggregated multicast, we can facilitate fault tolerance: by forcing multiple groups to share one single tree, we can reduce the number of multicast trees, and thus the number of backup trees need to set up is reduced also. In this way, the computation resource will be saved and the restoration overhead will be decreased.

7. References

[Chu00] Y. Chu, S. Rao and H. Zhang, "A Case for End System Multicast", Proceedings of ACM Sigmetrics, Santa Clara,CA, June 2000.

[FeiGI01] A. Fei, J. Cui, M. Gerla, M. Faloutsos. "Aggregated Multicast: an Approach to Reduce Multicast State", Proceedings of Sixth Global Internet Symposium (GI2001), San Antonio, Texas, USA, November 2001.

[FeiNGC01] A. Fei, J. Cui, M. Gerla, M. Faloutsos. "Aggregated Multicast with Inter-Group Tree Sharing", Proceedings of Third International Workshop on Networked Group Communications (NGC2001), UCL, London, UK, November 2001.

[Francis00] P. Francis, "Yoid: Extending the Internet Multicast Architecture", http://www.aciri.org/yoid/docs/index.html

[ID-BIDIR-PIM] Mark Handley and et al., "Bi-directional Protocol Independent Multicast (BIDIR-PIM)", Work in progress.

[ID-MAT] J. Crowcroft, "Multicast Address Translation", Work in progress.

[ID-MPLS-MCAST] D. Ooms, R. Hoebeke, P. Cheval, and L. Wu, "MPLS Multicast Traffic Engineering", Work in progress.

[ID-PIM-SM-REVISED] B. Fenner and et al., "Protocol Independent Multicast-Sparse Mode (PIM-SM): Protocol Specification (REVISED)", Work in progress.

[ID-SSM-ARCH] H. Holbrook and B. Cain, "Source-Specific Multicast for IP", Work in progress.

[ID-SSM-DEPLOY] S. Bhattacharyya and et al., "An Overview of Source-Specific Multicast (SSM) Deployment", Work in progress.

[Holbrook99] H. Holbrook and D. Cheriton, "IP Multicast Channels: EXPRESS Support for Large Scale Single-source Applications", Proceedings of SIGCOMM'99, Cambridge, MA, September 1999.

[Rad99] P. I. Radoslavov, D. Estrin, and R. Govindan, "Exploiting the bandwidth-memory tradeoff in multicast state aggregation", Technical report, USC Dept. of CS Technical Report 99-697, July 1999.

[RFC1633] R. Braden, D. Clark, and S. Shenker, "Integrated services in the internet architecture: an overview", IETF RFC 1633, 1994.

[RFC2189] A. Ballardie, "Core Based Trees (CBT version 2)

Multicast Routing: Protocol Specification", IETF RFC 2189.

[RFC2201] A. Ballardie, "Core Based Trees (CBT) Multicast Routing Architecture", IETF RFC 2201, 1997.

[RFC2362] D. Estrin and et al., "Protocol Independent Multicast-Sparse Mode (PIM-SM): Protocol Specification", IETF RFC 2362, 1998.

[RFC2475] S. Blake, D. Black, and et al, "An Architecture for Differentiated Services", IETF RFC 2475, 1998.

[RFC3031] E. Rosen, A. Viswanathan, and R. Callon, "Multiprotocol Label Switching Architecture", IETF RFC 3031, 2001.

[Thaler00] D. Thaler and M. Handley, "On the Aggregatability of Multicast Forwarding State", Proceedings of IEEE INFOCOM'00, March 2000.

[Tian98] J. Tian and G. Neufeld., "Forwarding State Reduction for Sparse Mode Multicast Communications", Proceedings of IEEE INFOCOM'98, March 1998.

8. Full Copyright Statement

9 Authors' Addresses

Jun-Hong Cui

Computer Science Department
University of California
Los Angeles, CA 90095-1596

Phone: +1 310 825-4623
Fax: +1 310 825-7578
Email: jcui@cs.ucla.edu

Mario Gerla
Computer Science Department
University of California
Los Angeles, CA 90095-1596
Phone: +1 310 825-4367
Fax: +1 310 825-7578
Email: gerla@cs.ucla.edu

Khaled Boussetta
Computer Science Department
University of California
Los Angeles, CA 90095-1596
Phone: +1 310 825-4623
Fax: +1 310 825-7578
Email: boukha@cs.ucla.edu

Michalis Faloutsos
Computer Science Department
University of California
Riverside, CA 92521
Phone: +1 909 787-2480
Fax: +1 909 787-4643
Email: michalis@cs.ucr.edu

Aiguo Fei
Computer Science Department
University of California
Los Angeles, CA 90095-1596
Phone: +1 310 825-4623
Fax: +1 310 825-7578

Jinkyu Kim
Computer Science Department
University of California
Los Angeles, CA 90095-1596
Phone: +1 310 206-6518
Fax: +1 310 825-7578
Email: jinkyu@cs.ucla.edu

Dario Maggiorini
Computer Science Department
University of California
Los Angeles, CA 90095-1596
Phone: +1 310 824-1547

Fax: +1 310 825-7578
Email: dario@cs.ucla.edu