Available online at www.sciencedirect.com
ScienceDirect
Computer Communications 30 (2007) 1600–1612
computer communications
www.elsevier.com/locate/comcom

QoS-constrained core selection for group communication

Ayşe Karaman [a,*], Hossam Hassanein [b]

[a] Department of Computer Engineering, Çankaya University, Öğretmenler Cd. No: 14 06530 Ankara, Turkey
[b] Telecommunications Research Lab, School of Computing, Queen's University, Kingston, ON, Canada K7L 3N6

Received 6 April 2006; received in revised form 13 January 2007; accepted 20 January 2007
Available online 26 January 2007

Abstract

The core-based approach in multipoint communication enhances the solution space in terms of QoS-efficiency of solutions in inter- and intra-domain routing. In an earlier work [A. Karaman, H.S. Hassanein, Extended QoS-framework for Delay-constrained Group Communication, International Journal of Communication Systems, in press.], we showed that the constrained cost minimization solutions in core-based approach proposed to date are restrictive in their search to a subrange of solutions, and we proposed *SPAN*, a generic framework to process in our identified extended solution space. In this paper, we study the core selection component of *SPAN* and propose two novel algorithms, *SPAN/COST* and *SPAN/ADJUST*, which define the core-selection component of *SPAN*. *SPAN/COST* mainly optimizes the cost distances to be traveled between the source–core and core–receiver pairs on the multicast trees, while *SPAN/ADJUST* selects the cores based on the numbers of nodes they dominate and adjusting the set based on cost. Our algorithms consistently outperform their counterparts proposed to date and can be considered pioneering in their optimization range of multiple metrics and processing in the extended solution space.

Keywords: QoS routing; Multicasting; Core-based architecture; Core selection

1. Introduction

Multipoint communication is the simultaneous delivery of data stream from a group of sources to a set of receivers with the objective of achieving efficient transmission according to predetermined metrics. Multipoint communication has numerous applications in Internet group communications and distributed environments. Design objectives in Internet routing architectures are characterized under Quality-of-Service (QoS) constraints for the allocation of network resources particular to application demands. Broadband group applications are often delay-sensitive and demanding on network resources. The prominent QoS problem in multipoint communication is *constrained cost minimization* – minimization of total network resources in one metric while meeting a given end-to-end delay bound between source–receiver pairs as a second metric.

In multipoint communications, each packet from a source in the group needs to visit each one of its destinations exactly once. Loops are not required, neither are they allowed on their routes, and multipoint communication path has the structure of a tree or multiple trees. Routing for group communications in earliest protocols constructed source-rooted trees. Latter protocols proposed core-based trees rooted at a core node in the domain which is not necessarily a source. The core-based approach in multipoint communication enhances the solution space in terms of QoS-efficiency of solutions in inter- and intra-domain routing and for reduced path maintenance overhead potentially across autonomous domains. Core selection – the selection of the cores among the domain nodes to serve the communication group – is crucial for protocol performance. An efficient core-selection process is likely to improve the performance of multipoint communication protocols significantly.

* Corresponding author. Tel.: +90 312 2844500.
E-mail addresses: karaman@cankaya.edu.tr (A. Karaman), hossam@cs.queensu.ca (H. Hassanein).

0140-3664/$ - see front matter © 2007 Elsevier B.V. All rights reserved.
doi:10.1016/j.comcom.2007.01.011

The core-based approach for route construction provides improved efficiency for constrained applications for sparsely populated groups in the distributed routing platform [3,22]. A similar approach is recently applied for energy-efficient multicasting in SANETS [14] using face routing [7]. Core-based QoS-routing solutions proposed to date restricted the processing range to a subset of the solution space, which in turn restricted the efficiency of the potential results [12]. In [10], we proposed a basic core selection algorithm, *SPAN*, resulting in a distributed, QoS-constrained solutions in the extended solution space under the core-based architecture. *SPAN* processed solely on the connectivity information each candidate core provides the group members, yet consistently outperformed its counterparts in the literature. In this paper, we study the core selection algorithms in the extended solution space and propose two algorithms, *COST* and *ADJUST*. *COST* and *ADJUST* define the core-selection component of *SPAN* to account for the cost-approximation of the resulting multipoint communication path to achieve QoS guarantees. *COST* mainly optimizes the cost distances to be traveled between the source–core and core–receiver pairs on the multicast trees, ADJUST selects the cores based on the numbers of nodes they dominate and adjusting the set based on cost. In the next section, we layout the problem and the terminology before we examine the existing solutions in the literature. Previous work on the core-based approach in multipoint communication is presented in Section 3. In Section 4, we introduce *COST* and *ADJUST*. In Section 5, we evaluate the performance of *COST* and *ADJUST* against their counterparts in the literature. We conclude the paper in Section 6.

2. Preliminaries

We analyze a solution for a multipoint communication group under the core-based approach as a union of core-based trees each spanning a subset of receivers, and the source-based trees each spanning the set of cores connecting the receivers in respective core-trees to the source and thus serving the source for the group application. The trees are maintained separately and potentially share links.

We say that a receiver r is *dominated by* a core c for a particular source s if there exists a path p connecting s to r so that p passes through c without violating the delay bound. Equivalently, we say that a core c serves r_s. We use the notation $D(c,s)$ to indicate the set of receivers that are dominated by the core c for source s. Similarly, we indicate by $D(c,S')$ the domination of a set of receivers that are dominated by the core c for each source in a subset S' of the set of sources. We refer by *multipoint tree* the union of paths serving a $S' \subseteq S$ for all receivers in the group where each core tree constituting the path is identically serving all sources in S'. Consequently on such a multipoint path, the domination sets of the cores in the cluster of cores describing the path partitions the receiver set. Observe that a multipoint tree does not necessarily have a tree structure although it specifies distinct trees each of which is serving a particular source for the delivery of its stream to the receivers. In Fig. 1a, we present an example for a multipoint tree. The source and receiver sets in the multipoint communication group are $S = \{s_1, s_2\}$ and $\{r_1, r_2, r_3, r_4, r_5\}$, respectively. There are three cores in the group. Dominating sets are $D(c_1, S) = \{r_1, r_2\}$, $D(c_2, S) = \{r_3\}, D(c_3, S) = \{r_4, r_5\}$. There are three core-rooted trees, each spanning the receivers in the domination sets of their respective cores. The multipoint tree is serving both sources in the group. There are two source-rooted trees, one for each source, each spanning the entire cores in the core-cluster corresponding to the multipoint tree. The links on source and core trees are indicated by solid and double lines, respectively. All links except those indicated by dashed line are on the multipoint path. Fig. 1b presents an alternate solution to the same problem in which different trees with non-identical core trees serve the sources. The links serving sources s_1 and s_2 are indicated solid and dashed lines, respectively. Double lines indicate the links serving both sources. The domination sets are $D(c_1, S) = \{r_1, r_2\}, D(c_2, s_1) = \{r_3, r_4\}, D(c_2, s_2) = \{\}, D(c_3, s_1) = \{r_5\}, D(c_3, s_2) = \{r_3, r_4, r_5\}$. Unlike the case in Fig. 1a, the receiver set is partitioned differently by the core clusters for each source, with one partition and the dominating core, c_1, common to both. We name the range of solutions that involve, respectively, a unique core cluster and multiple core clusters for a particular multipoint communication problem as *singular* and *non-singular* solutions. Note that any union of paths constituting a solution to a constrained multipoint communication problem can be formulated as a set of core clusters.

The search for solutions in the singular solution space examines the domination of each receiver for all sources in the group. An algorithm to span the solutions in the non-singular space, on the other hand, examines the domination sets separately for each source rather than across all sources. The non-singular solution space expands the range of potential solutions to achieve more efficient results for constrained-group communication. Thus, a heuristic to provide a solution to constrained multipoint communication problem in the entire solution space is considerably preferable to a heuristic to provide a solution in the singular solution space [11].

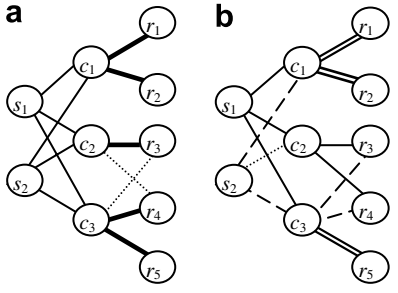

Fig. 1. An example constrained-multipoint communication problem. (a) A singular solution, (b) a non-singular solution for the same problem instance.

3. Related work

Routing for group communications has initially been studied in the multicasting domain for transmission from one source to a given set of receivers. Latter protocols proposed core-based approach, which develops core-rooted trees shared across multiple sources in the group for the delivery of their stream. Earlier core-based models, CBT [1,2] and PIM [5] restricted to single-core solutions. Further research focused on multi-core trees for the improvement of tree accessibility to the multicast sources and new receivers. OCBT [16] is the first multi-core routing protocol in the literature. In its design specifications, OCBT develops a unique shared tree which contains multiple cores, and emphasizes on maintaining the tree structure as the tree is updated during protocol operations, diverting from the efficiency concerns of path construction and maintenance throughout the protocol operations. The model disregards a delay bound of the application and is not applicable for constrained problems. CBT, PIM, and OCBT assume core selection as a process external to the protocol suite and operate on the already selected set of core(s) available to them. Core selection for core-based protocols is studied separately in literature for single-core and multi-core schemes. In the following subsection, we outline single-core selection algorithms proposed in literature to date. Section 3.2 examines the multi-core selection algorithms in literature and first examines *n-dominating* set and *k-center* studied in [23] for their *Senders-to-Many* and *Members-to-Many* protocols. We then look into *GREEDY* and *NAÏVE* [15] and *SPAN* [10] in this subsection which are the constrained multi-core selection algorithms in literature directly relating to our work. We refer the reader to [12] for a review and analysis of core-based multicast routing algorithms.

3.1. Single-core selection algorithms

Multicast routing evolved into core-based approach initially within the domain of single core scheme. In this class, a core is selected in the domain for a unique tree to be rooted at this core, spanning the receivers in the group. A source willing to deliver its stream to the group transmits the packets to the core from whereon forwarded to the receivers throughout the tree. Note that the classification of the solution domain as singular versus non-singular solutions applies through partitioning of the receivers across the cores and thus the single-core selection solutions are inherently in the singular space.

[20] proposed a set of single core selection algorithms to operate on hop-count as the minimization metric. [3] proposed the placement of a core in the center of the domain, and, alternatively, in the center of the subdomain induced by the communication group participants for the efficiency of group communication throughout the domain. With a similar approach, [17] suggested the selection of the core node at the center of the group members through a "tournament" locating the "midpoints" of the node pairs which are the "winners" of the previous level of the tournament. The single core selection algorithms studied by [3,17,20] disregard a delay-bound value for the application and are unconstrained. Salama [15] studied MinMaxD, and INITIAL as single core selection algorithms for constrained groups. Both models operate on delay metric to return a core node in the domain to be the root of the delivery tree meeting the delay-bound of the application. The single-core selection algorithms have the common property that they incur feasible amount of message exchange for their operations with no reliance on an external message transmission protocol and are distributed.

A greedy core selection algorithm based on CBT with the purpose of reducing delay and delay jitter was introduced in [18]. The scheme, known as Delay and Delay Variation Constraint Algorithm (DDVCA) selects the node with the least delay variation to all nodes in the network as the core. An extension that tightens multicast delay variation of the DDVCA scheme was recently proposed in [9]. The QoS core selection algorithm (QCSA) was introduced in [4]. QCSA takes into account the multiple QoS constraints between the core node and its members. The scheme runs in three stages. In the first stage, potential core nodes are marked. These are the nodes that can potentially guarantee delay constraints or member nodes. In the second stage, marked core nodes exchange QoS-constrained information. In the third stage, the node with the least maximum number of hops to nodes in the group (while maintaining QoS constraints) is chosen as the core node of the group.

3.2. Multi-core selection algorithms

Single-core based approach restricts the solution domain into solutions involving a unique core. The main drawback of these algorithms is that this restriction rules out potential feasible solutions for constrained applications. Fig. 2 presents an example to demonstrate this case. T_1 and T_2 are network clusters each containing a subset of

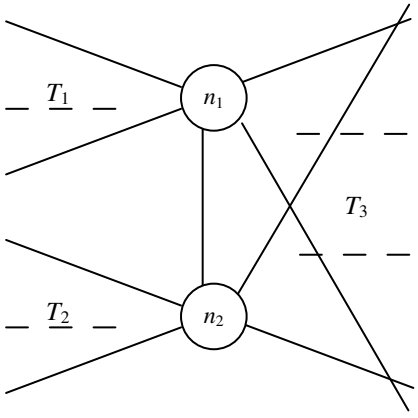

Fig. 2. Example case demonstrating the solution space restricted by single-core solutions.

sources, and T_3 is another cluster of receivers. T_1, T_2 and T_3 have no delay-bound connection between one another except through the nodes n_1 and n_2, as indicated in the figure. When the delay distance between the group members in cluster pairs T_1 and T_3, and T_1 and T_2 are maximum for the delay constraint, there is no single-core solution in the domain while there exist delay-bound solutions involving multiple cores. The restriction of the solution domain within single-core based approach further restricts the potential efficiency of the solutions. We continue in this section with the review of multi-core selection algorithms studied so far.

3.2.1. Singular multi-core selection algorithms

Zappala, Fabbri, and Lo [23] propose two multi-core selection algorithms for unconstrained communication groups:

(1) *n-dominating set*: locates a minimal dominating set as the set of cores, which is a subset of nodes in the domain so that all nodes in the multicast group are within n-hops of this set for a given n.
(2) *k-center*: places a given number of k cores in the domain so that the hop-distance from all nodes in the multicast group to the cores is minimum.

Note that neither the dominating set nor k-center algorithms have polynomial-time solutions [8]. Zappala et al. [23] used the greedy dominating set heuristic for n-dominating set solution iterates to select a core which dominates the highest number of un-dominated receivers for all sources in the group until all the receivers are dominated by the selected cores. Zappala et al. [23] also use this greedy algorithm for their k-center heuristics by parameterizing it with a constant k to specify the exact number of cores to be selected for a solution to k-center problem. It should be noted that Zappala et al. [23] are not specific on the algorithmic details of k-center heuristics.

Zappala et al. [23] tested each of these algorithms comparing their performances on delay to random core selection. Both k-center and dominating set algorithms demonstrated modest improvement compared to random core selection. k-center core placement with multiple cores resulted in lower delays than single-core tree with optimal core placement as the value of k increased. Zappala et al.'s results suggest that polynomial-time heuristics for dominating set, and especially k-center core placement contribute to the efficiency of multi-core protocols. Both algorithms select a unique core cluster in which each core spans a subset of receivers for all sources in the group. Thus, both algorithms process solely in the singular solution space.

Salama [15] studied the problem of multiple core selection for delay-constrained groups. Salama's multi-core selection algorithms are centralized, and assume a symmetric network:

(1) *GREEDY*: This algorithm assumes bi-directional utilization of the generated trees and attempts to construct shared trees each spanning a subset of receivers to be dominated for all sources in the group. In bi-directional utilization of the trees, the data stream from a particular source no longer needs to travel to the tree root, which is a core, to be delivered to the receivers spanned on the tree. Each source is necessarily on each core tree. A source willing to transmit its stream to the receivers acts as the tree root, and each on-tree node receiving this stream simply forwards it to each of its on-tree links except the one through which it received the stream. [15] considers the bi-directional utilization of the shared trees during tree design, and the resulting trees are those which provide a path between each source in the group and each receiver spanned on the tree so that each such path connects the corresponding source–receiver pair within the delay-bound of the application without necessarily passing through the tree root. The problem remains NP-complete in this modified version [15]. *GREEDY* assumes as input, for each node in the domain, the set of receiver nodes that the node is dominating for the delay bound for all sources in the group. The algorithm merely iterates to select the node that dominates the maximum number of un-dominated receivers as the next core until all receivers are dominated.
(2) *NAÏVE*: Applies iterative selection of the core set as the tree is constructed. Initially the selected core set is empty. When a new receiver attempts to join the multicast path, the algorithm checks iteratively the members of the core set to see whether there already exists a core so that the paths from each source to the new receiver via the core is within the delay bound. If the search fails, then the candidate core set is searched for a core to lead to a feasible solution. Salama suggests as the candidate core set either the set of receivers, or an ordered list of a set of nodes with respect to the maximum (*NAÏVE-MinMaxD*), or average (*NAÏVE-AVGD*) delay of the candidate to the multicast group members. When the receiver set constitutes the candidate cores, the next core to be selected is always the new receiver to join itself in case of inexistence of a feasible core among the selected ones. In all versions, *NAÏVE* processes for at most as many cores as the number of sources in the group be selected, and terminates without providing a solution with the idea that source-based trees would be more feasible when the number of cores and hence number of distinct trees exceed the number of source-based trees.

Both *GREEDY* and *NAÏVE* restrict their processing to the singular solution space and return a feasible solution whenever such a solution exists, provided that the candidate core set does not exclude some subset of domain nodes

that lead to a feasible solution whenever there is a solution to the problem instance. Both algorithms lack the consideration of cost within the core selection criterion. *NAÏVE* is an incremental algorithm, iterates on the receivers in the group for core selection and the construction of the set of paths to connect the incoming receiver to the group. However, *NAÏVE*, unlike *GREEDY*, processes with no regard to the direct on-tree paths connecting source receiver pairs for cost efficiency. Furthermore, *NAÏVE*'s incremental processing restricts effective selection of cores and *NAÏVE* underperforms *GREEDY* on the resulting core set size [15], and *GREEDY* is the prominent model for constrained, multi-core based solutions for group communications.

In reference [13], three methods for core selection and evaluation were introduced. These are k-maximum path count, k-maximum degree and k-minimum average delay, where k is the number of selected cores. No QoS guarantees where considered in the selection though. The use of clustering techniques in core selection was proposed in [6]. The schemes try to optimize a certain objective function, say max-sum vector portioning or minimum traffic concentration ratio. A Scheme to select a minimum number of clusters, and hence number of core nodes is proposed.

3.2.2. Non-singular multi-core selection algorithms

SPAN [10] is a distributed, asymmetric framework that operates in both non-singular and singular solution spaces for QoS-constrained group communication. *SPAN* initially identifies the potential cores into a pool as candidates by examining their domination characteristics. A domain node is a candidate core only if it connects at least one source–receiver pair within the delay-bound of the application. Each candidate core tests itself on the local state information for its domination for the group, and reports its results to the designated coordinator node in the domain for the selection of cores across these results. The ultimate core set selected among the candidate cores includes the cores that lead to a multipoint path to approximate the optimum solution. The construction of the trees follows core selection and is coordinated separately by the root of each tree.

Let $T_{(c,s)}$ be a core tree rooted at core c to serve source s for all receivers in $D(c,s)$, i.e., along the particular tree rooted at it, core c is serving all receivers in $D(c,s)$ for s. According to this definition, $T_{(c,s)}$ *totally* serves s. We name such a source s as the defining source of the core tree $T_{(c,s)}$. Consider that $T_{(c,s)}$ serves a source $s' \in S$, so that $\emptyset \subset D(c,s') \subset D(c,s)$ where $D(c,s') \neq \emptyset$ and $D(c,s') \neq D(c,s)$. In other words, $T_{(c,s)}$ serves s' not for all but some of the receivers it is spanning. We say, in this case, that $T_{(c,s)}$ serves s' *partially*. In Fig. 1b, T_{c_3,s_2} is serving s_2, its defining source totally, and also serving s_1 but partially since not all but part of the receivers in $D(c_3,s_2)$ are dominated by T_{c_3,s_2} for source s_2. T_{c_3,s_2} is serving its defining source totally and not serving s_2 at all. In an attempt to minimize the multipoint path structure in terms of the number of links on it, the approach is the maximization of combined partial and total utilization of multipoint paths for simplification of ultimate trees in conjunction with efficiency of the transmission. SPAN imposes that a tree $T_{(c,s)}$ constructed to serve its defining source s for all the receivers in $D(c,s)$ also serves, with no modification on the on-tree paths, the rest of the sources for all receivers in set $D(c,s) \cap D(c,s') \forall s' \in S$. For all $s' \in S$, a domination $r_{s'} \in T_{(c,s)}$ is served on $T_{(c,s)}$ along the path that serves r_s. According to this, $r_{s'} \in T_{(c,s)} \iff r \in D(c,s) \cap D(c,s')$. Consider the definition of *domination count* of a potential core-tree $T_{(c,s)}$ as follows:

$$dominationcount_{(c,s)} = \sum_{s' \in S} |D(c,s) \cap D(c,s')|$$

Literally, domination count of a (c,s) tuple specifies the number of source–receiver pairs the core tree $T_{(c,s)}$ is capable of serving when the receiver set being served is restricted to $D(c,s)$. The extended solution space offers, during core selection, efficient choice of a core in consideration of each source separately, diverting from the shared tree approach for the efficiency of path construction, update and management during the communication session. However, higher utilization of $T_{(c,s)}$ in terms of the number of sources it is partially or totally serving improves the overall multipoint path structure through tree sharing as the case in core-based approach literature. Furthermore, higher utilization of $T_{(c,s)}$ in terms of the number of receivers it is dominating improves the efficiency of transmission as the case in multicast routing literature. Therefore, high domination count is desirable for core-selection criterion. *SPAN* iterates to select the potential core tree $T_{(c,s)}$ to return the highest domination count for the currently un-dominated receivers. Note that, whenever there is a delay-bound solution, each source in the group is a potential core to serve itself for all the receivers in the group. This is the case when the core trees are $T_{(s,s)}$ for each $s \in S$ and each of these trees is serving s for all $r \in R$. SPAN is processing to establish a core tree at each iteration maximizing on the overall domination count of the tree to serve as many r_s as possible dominated totally and partially on the tree selected. This implies that the number of cores selected by *SPAN* is at most as great as $|S|$, the size of the source set.

4. Core selection algorithms

SPAN [10] conducts its search for cores based solely on the domination count of each core as the selection criterion. In this section, we present two core selection algorithms, *COST* and *ADJUST*. Both algorithms operate within the framework describing *SPAN* itself, and thus both algorithms are distributed and asymmetric. Each algorithm processes in the non-singular as well as singular solution spaces to further improve the efficiency of the solutions by also accounting the cost metric during core selection. *SPAN/COST* additionally examines the cost distances between the group members and candidate cores as part of the core selection criterion during the selection pro-

cess. *SPAN/ADJUST*, on the other hand, first executes the core selection component of *SPAN* and processes on the selected core trees to improve the efficiency of the already found solution.

4.1. SPAN/COST

SPAN/COST defines core-selection within *SPAN*, taking into account the cost distances to be traveled between the source–core and core–receiver pairs on the trees to be constructed for a better approximation of the resultant paths on the cost metric. As in *SPAN*, it still operates in the non-singular solution space and uses the domination count as a metric for core selection. We use the cost of the minimum delay-distance paths for our approximations. Let $cost(i,j)$ denote the cost of the minimum delay-distance path between the nodes i and j. We extend the selection criterion for a core tree as follows:

$$Criterion_{(c,s)} = R\_factor_{(c,s)} + S\_factor_{(c,s)} + D\_factor_{(c,s)}$$

where

$$R\_factor_{(c,s)} = \frac{\sum_{r\in D(c,s)}[(\cos t(c,r)+m) * |S_{(c,s),r}|]}{\sum_{r\in D(c,s)}|S_{(c,s),r}| * ave\_dist}$$

$$S\_factor_{(c,s)} = \frac{\sum_{r\in D(c,s)}\left[\frac{\sum_{s'\in S_{(c,s),r}}\cos t(c,s')}{|S_{(c,s),r}|}+m\right]}{\sum_{r\in D(c,s)}|S_{(c,s),r}| * ave\_dist}$$

$$D\_factor_{(c,s)} = 1 - \frac{\sum_{s'\in S}|D(c,s)\cap D(c,s')|}{|S| * |R| - \sum_{(a,b)}\sum_{b'\in S}|D_u(a,b)\cap D_u(a,b')|}$$

$$ave\_dist = \frac{\sum_{(c,s)\in CxS}[\cos t(c,r) * |S_{(c,s),r}|]}{\sum_{(c,s)\in CxS}|S_{(c,s),r}|} + m$$

R_factor and *S_factor* are estimates of the costs of, respectively, the core and source trees of the solution considered by the algorithm at its current iteration. The second term in the $D\_factor_{(c,s)}$ expression is literally the proportion of the domination count of the potential core tree $T_{(c,s)}$ to the currently un-dominated receivers across the sources. $D\_factor_{(c,s)}$ in return is merely a transformation of $domination\_count_{(c,s)}$ to a scale in a definite range in which it is to be minimized for the highest value of $domination\_count_{(c,s)}$. In the basic *SPAN*, $Criterion_{(c,s)} = D\_factor_{(c,s)}$. All of *D_factor*, *S_factor* and *D_factor* are normalized into $[0,1]$ range for their comparison on the same scale. The transformation of domination count to *D_factor* is made to adjust this part of the criterion into the range. Note that *Criterion* in turn can be tuned to

$$Criterion_{(c,s)} = K_R * R\_factor_{(c,s)} + K_S * S\_factor_{(c,s)} + K_D * D\_factor_{(c,s)}$$

where $K_R + K_S + K_D = 1$. However, our performance tuning tests yielded insignificant results and we set Criterion as linear sum of the three factors.

For the formulation of *Criterion*, we shift the link cost values by m which is the minimum of the link costs in the domain. We do so to avoid multiplication by 0 during the estimation of tree costs, which arises in *R_factor* and *S_factor*. In *R_factor*, $cost(c,r) = 0$ if $c = r$. Similarly, in *S_factor*, $cost(c,s) = 0$ if $c = s$. Shifting the link costs by the minimum value they can take allows equal consideration of tree cost values during their estimate. The value m appears in *S_factor*, *R_factor* and *ave_dist* in the simplified form of the formulae.

Let $S_{(c,s),r}$ be the set of all sources served by the tree $T_{(c,s)}$ to dominate the r. The $R\_factor_{(c,s)}$ accounts for the cost distances between a core c, and receivers dominated on its respective core tree. The value $cost(c,r)*|S_{(c,s),r}|$ accumulates the cost distances to be traveled between the core c and the receiver r separately for each source in $S_{(c,s),r}$ transmitting its stream to r via the tree $T_{(c,s)}$. The numerator of *R_factor*, $\sum_{r\in D(c,s)}[\cos t(c,r) * |S_{(c,s),r}|]$ in return is the sum of all core–receiver cost distances across the receivers dominated on the tree $T_{(c,s)}$. Note that this value the upper bound for the cost of core tree, and used in the formula an approximation of this cost, since the paths leading to multiple receivers on the tree $T_{(c,s)}$ are potentially shared, and thus the cost of transmission from a source to multiple receivers sharing paths leading to them on the core tree is less than their additional cost value. *R_factor* divides by $\sum_{r\in D(c,s)}|S_{(c,s),r}| = |\{r_{s'}|r_{s'}\in T_{(c,s)}\}|$ to take the simple average of the approximated cost. *ave_dist* in the denominator of *R_factor* is the approximated cost of all core trees in the investigated solution set, i.e., the value to approximate the average of all cost-distances to be traveled between the potential cores and the receivers to be dominated on their respective trees. Further dividing *R_factor* by *ave_dist* transforms it into a magnitude comparing the potential core tree directly to the rest of such trees.

S_factor measures the cost distances between core–source pairs. $\sum_{s'\in S_{(c,s),r}}\cos t(c,s')/|S_{(c,s),r}|$ represents the average of all transmission costs between each source served on $T_{(c,s)}$ for a given receiver r dominated on the tree. We accumulate the average source-to-core transmission costs for all the receivers dominated on the tree $T_{(c,s)}$. The packets transmitted from a source to a core are delivered to multiple receivers on the core-tree. Therefore, the more receivers are served for a particular source s' on $T_{(c,s)}$, the less the cost between s' and c is accounted for. Due to this, we diminish each $\sum_{s'\in S_{(c,s),r}}\cos t(c,s')/|S_{(c,s),r}|$ further by the size of the set $\{s'|r_{s'}\in T_{(c,s)}, s'\in S_{(c,s),r}\}$, and divide *S_factor* by $\sum_{r\in D(c,s)}|S_{(c,s),r}| = |\{r_{s'}|r_{s'}\in T_{(c,s)}\}|$ rather than $|D(c,s)|$ to take the simple average across the receivers served on $T_{(c,s)}$. We further divide *S_factor* by *ave_dist* as above in *R_factor*.

In Fig. 3 we present the *SPAN/COST* algorithm. D_u and C_u, respectively, denote the ultimate domination and core sets for the selected core trees. Note that $D_u(core,s)$

```
C1.. for (each c∈C)  for (each s∈S)  {   // O(|S||C|)   [O(|S|²|R||C|)]
C2..    // compute domination_counts
C3..     domination count(c,s) = |D(c,s)|;
C4..     for (each s'∈S, s'≠s)                    // O(|S|)   [O(|S||R|)]
C5..        domination count(c,s) += |D(c,s) ∩ D(c,s')|;   // O(|R|)
C6.. }
C7.. for (each c∈C)  for (each s∈S)  for (each r∈D(c,s))  for (each s'∈S)   // O(|S|²|R||C|).
        //  compute S(c,s),r values
C8..      if (r∈D(c,s'))  {
C9..         S(c,s),r = S(c,s),r ∪{s'}
C10..        S(c,s),r .cost += cost(s',c);
C11..        |S(c,s),r|++;
C12..     };
C13.. compute ave_dist = Σ_{r∈D(c,s), (c,s)∈CxS} ⌊cos t(c,r)*|S(c,s),r|⌋ / Σ_{r∈D(c,s), (c,s)∈CxS} |S(c,s),r| + m;   // O(|S||R||C|)
```

```
C14.. total_count = 0;
C15.. repeat   // O(|C_u|) = O(|S|)   [O(|S|³|R||C|)]
C16..   if (|C| =|S|) exit;
C17..   min_value = ∞
C18..   for (each c∈C)  for (each s∈S)  {  // O(|S||C|)   [O(|S||R||C|)]
        // compute criterion for each candidate core and select the best scoring one
C19..      N_R = 0;  N_S = 0;  D = 0;
C20..      for (each r∈D(c,s))  {   //  O(|R|)
C21..         N_S += S(c,s),r.cost/|S(c,s),r|;   // numerator of S_factor(c,s);
C22..         N_R += cost(c,r)*|S(c,s),r|;   // numerator of R_factor(c,s);
C23..         D += |S(c,s),r|;   // denominator of both factors
C24..      };
C25..      R_factor(c,s) =  N_R / (D*ave_cost_dist)
C26..      S_factor(c,s) =  N_S / (D*ave_cost_dist)
C27..      D_factor(c,s)=1- domination count(c,s) /(|R|*|S|-total_count)
C28..      criterion(c,s)  = R_factor(c,s) + S_factor(c,s) + D_factor(c,s);
C29..      if (criterion(c,s) <min_value)  {
C30..         min_value = criterion(c,s);
C31..            core = c;
C32..            source = s;
C33..      };
C34..   };
C35..   total_count += domination count(core,source);
C36..   domination count(core,source) = 0;
C37..   D_u(core,source) = D(core,source);
C38..   C_u = C_u ∪{core, source)}
C39..   C = C \ {core};
C40..   for (each s∈S)  for (each r∈D(core,source))   // O(|S||R|)   [O(|S|²|R||C|)]
C41..      if (r∈D(core,s))   {
C42..         D_u(core,s) = D_u(core,s) ∪{r};
C43..         for (each c∈C, c ≠core)   {   // O(|C|)   [O(|S||C|)]
C44..            S(c,s),r = ∅
C45..            for (each s'∈S)    S(c,s'),r = S(c,s'),r \ {s};    // O(|S|)
C46..            if (r∈D(c,s))   domination count(c,s) --;
C47..            D(c,s) = D(c,s) \{r};
C48..         };
C49..   };
C50.. until (total_count = |R|*|S|)
```

Fig. 3. Pseudo-code of *SPAN/COST*.

$\forall s \in S$ for specify a selected core tree $T_{(core,source)}$. The algorithm starts out with the initialization of the sources set attributes (lines C7–12) and domination counts (lines C1–6) for all possible core trees. The domination counts and the attributes of each $S_{(c,s),r}$ for the remaining cores are updated whenever a new core is selected throughout the algorithm (lines C40–49). Line 13 indicates the one-time computation of *ave_dist*. *SPAN/COST* does not have an inherent bound on the number of cores selected. We force the upper bound on the ultimate core set for *SPAN* for *SPAN/COST* as well, and terminate the algorithm whenever the core set size exceeds the number of sources and assume *SPAN* to be invoked for the core selection for the particular problem instance (line C16). The block in lines C18–35 computes $Criterion_{(c,s)}$ for each candidate-core and source tuple (lines C18–28) and compares the value to the current minimum value, *min_value* (lines C29–33). *total_count* keeps track of the number of current dominations, i.e., the number of distinct r_s dominated by an already selected core tree. The main loop (lines C15–50)

iterates to select exactly one core tree each time, and terminates when all the receivers are dominated for all sources.

For the deployment of our algorithms, we assume direct addressing which is feasible considering the sizes of the associated composite variables. The computational complexity of each computation step is depicted on the line as comments. The lines of which the computation time is bound by a constant are not specified. The values indicated in square brackets denote the overall complexity of the loop-statements when the nested blocks are accounted. The *repeat* loop of *SPAN/COST* (lines C15–50) iterates a total number of $|Cu| \leqslant |S|$ times. The inner loop within in lines C40–49 iterates on the source and receiver sets to update the domination sets for further iterations, executing in time $O(|S||C|)$ (lines C43–48) at each step. The overall complexity of the block C43–48 is $O(|S|^2|R||C|)$, governing the complexity of the block C16–49. The *repeat* loop in turn executes in time $O(|S|^3|R||C|)$ which is the computational complexity of *SPAN/COST*.

4.2. SPAN/ADJUST

SPAN selects the cores based on their domination counts. The resulting cores combine all the currently undominated receivers on their respective trees with no regard to the cost-proximity of the receivers to the cores. In this section, we introduce an alternative non-singular algorithm, which we call *ADJUST*. *ADJUST* first runs *SPAN* to select the cores using domination count as the sole criterion, then adjusts the existing trees by moving the receivers between the trees for their less-costly dominations. Let the function *delay* return the minimum delay between the given pair of nodes in its parameters. The cost function relevant for this case to measure the cost distances between the core–receiver pairs needs also to account Δ, the delay bound of the application, in order to determine whether the path being examined meets the delay bound:

$$cost'(c,r_s) = \begin{cases} cost(c,r); & \text{if } delay(c,r) + delay(s,c) < \Delta \\ \infty; & \text{otherwise.} \end{cases} \quad \text{(I)}$$

Consider *main domination* and *ordinary domination* to define $r_s \in T_{(c,s)}$ and $r_{s' \neq s} \in T_{(c,s)}$, respectively. Observe that an ordinary domination for a particular receiver can be on a given tree only if the main domination for that receiver is on that tree. *ADJUST* distinctively considers the following cases for the movement of dominations between the trees:

(a) *From ordinary to main domination*: let $r_{s'}$ be an ordinary domination on a core tree $T_{(c,s)}$ so that $s' \neq s$. The necessary condition for the movement of $r_{s'}$ to a core tree $T_{(c'',s'')}$ for main domination requires that $s' = s''$ and $cost'(c'', r_{s'}) < \infty$.

```
//  from ordinary to main
A1.. for (each core tree T(c,s))      // O(|C_u|) = O(|S|)   [O(|S|^2|R|)]
A2..    for (each r∈R)              // O(|R|)   [O(|S||R|)]
A3..      if (r_s∉T(c,s))   {
A4..          locate T(c',s') where r_s∈T(c',s');     // O(|C_u|) = O(|S|)
A5..          if ( (cost'(c, r_s)< cost'(c', r_s) ) & (s'≠s) )
A6..                    { move r_s from T(c',s') to T(c,s)  }
A7..      }
// from main to any
A8.. for (each core tree T(c,s))   for (each r∈R)  // O(|C_u||R|)=O(|S||R|)   [O(|S|^2|R|^2)]
A9..    if ( (r_s∈T(c,s)) & (r_s is not marked) ) {
A10..       min_value = cost'(c, r_s);
A11..       for (each core tree T(c',s'))       // O(|C_u|) = O(|S|)     [O(|S||R|)]
A12..          if ( (r_s'∈T(c',s')  ||  s'=s) && (cost'(c', r_s) < min_value) )   {
A13..              dependants = true;
A14..              for (each r_s''∈T(c,s))   //   O(|R|)
A15..                  if (cost'(c', r_s'') = ∞) dependants = false;
A16..              if (dependants)     { min_value = cost'(c', r_s);  toTree = T(c',s');  }
A17..          }
A18..          mark and move r_s' ∀'s ∈S from T(c,s) to toTree;  // O(|S|)
A19..    }
// from ordinary to ordinary
A20.. for (each s∈S)  for (each r∈R)   if (r_s is not marked) { //O(|S||R|)  [O(|S|^2|R|)]
A21..    locate T(c',s') where r_s∈T(c',s');     // O(|C_u|) = O(|S|)
A22..    if (s'≠s){    // if not a main domination
A23..       min_value = cost'(c', r_s);
A24..       for (each core tree T(c'',s''))       // O(|C_u|) = O(|S|)
A25..          if ( (r_s'' ∈T(c'',s'')) & (cost'(c'', r_s) < min_value) )
A26..                { min_value = cost'(c'', r_s);  toTree = T(c'',s''); }
A27..          mark and move r_s from T(c',s') to toTree;
A28..    }
A29.. }
```

Fig. 4. Pseudo-code of *SPAN/ADJUST*.

(b) *From main to main or ordinary domination*: let r_s be a main domination on a core tree $T_{(c,s)}$. The necessary condition for the movement of r_s to a core tree $T_{(c',s')}$ for any domination requires that $cost'(c', r_{s''}) < \infty$ for all $r_{s''}$ where $r_{s''} \in T_{(c,s)}$. If $s \neq s'$, further requirement is $r_{s'} \in T_{(c',s')}$. When a main domination r_s is moved to another tree, all ordinary dominations for the r are moved with it and dominated in their new location. If the targeted tree's defining source is different from the source for the domination itself, then it is further checked whether the r is already being dominated on that tree so that its ordinary domination is allowed.

(c) *From ordinary to ordinary domination*: let $r_{s'}$ be an ordinary domination on a core tree $T_{(c,s)}$ so that $s' \neq s$. The necessary condition for the movement of $r_{s'}$ to a core tree $T_{(c'',s'')}$ for ordinary domination requires that $r_{s''} \in T_{(c'',s'')}$ and $cost'(c'', r_{s'}) < \infty$.

ADJUST (Fig. 4) applies the intuition to initially establish the main dominations based on their ultimate locations on the core trees for which the distance of the receiver in the domination is closest in cost to the core of the tree. The algorithm first examines ordinary dominations for their main dominations on the selected core trees (Case (a), lines A1–7). When there is a core tree rooted at a core, say c, satisfying the necessary condition for this case, the cost distance of the receiver to its currently dominating core and the core c are compared, and the domination is moved if the cost distance on the newly found tree turns out to be less. In case (a), the first tree found to provide a better cost is selected to move the domination. Case (b) (lines A8–19) is examined after, and complements case (*a*) in that this case examines the main dominations for their least costly dominations across *all* satisfactory core-trees. In this case, all the core trees satisfying the necessary condition are examined for the least-costly core tree. If the cost $cost'(c,r_s)$ of a main domination r_s is less in another one of the existing trees, say $T_{(c',s')}$ so that $cost'(c,r_s) > cost'(c',r_s)$ then r_s is moved to $T_{(c',s')}$ regardless of whether or not $s = s'$, i.e., r_s is a main domination on $T_{(c',s')}$ or an ordinary one. At the end of this stage, all main dominations are set at their ultimate trees to provide potential ordinary dominations. At the final stage (lines A20–29), the algorithm adjusts the ordinary dominations for their least-costly locations on the set of core-trees.

Lines A8–19 of *ADJUST*, examining the movement of the main dominations, additionally iterate on the receivers set to see whether there exists a "dependant" ordinary domination on the main domination to be moved to violate the necessary condition in this case. Due to this, the computational complexity of the block executing Case (*b*) is higher by a factor of $|R|$ than the rest of the algorithm and is $O(|S|^2|R|^2)$, which then also is the overall complexity of the algorithm.

5. Performance evaluation

As we pointed out in Section 3, *GREEDY* [15] is the prominent model for QoS-constrained multipoint communication in terms of architectural structure and performance. Hence it is used as basis for comparison with our algorithms. In its architectural description, *GREEDY* operates on bi-directional trees so that every on-tree node on a given tree forwards every incoming data packet to everyone of its on-tree links except the one where it received the packet regardless of whether the outgoing link(s) lead to receivers as further destinations. We modify *GREEDY*, into *m-GREEDY*, which, in account of the bi-directional utilization of the generated trees, traverses the tree for the source of the current data stream assumed as the root, and forwards the stream *only* to those links leading to downstream receivers in the traversed version. Observe that our modification performs at least as good as *GREEDY* itself at the cost of tree maintenance overhead additional to that of *GREEDY*. We also consider *m-GREEDY* as a potential alternative to our models in terms of cost-efficiency compared to its original version.

Within our framework [10], we separate the process of tree construction from core selection for modular efficiency of the components in the construction and maintenance of paths in a distributed manner. For tree construction, we modify the SMT heuristic in [19] into a constrained version. The heuristic itself starts out with a single node as the original tree, and iterates to add the un-spanned node which is closest in its cost distance to the current tree until all the given set of nodes is spanned. For its processing, the SMT algorithm in [19] makes use of a distance function to measure the cost distance between the given pair of nodes. With the modification of the cost function in a way similar to formula (I), the algorithm turns into a constrained-SMT heuristic for tree construction [10] which we use to generate the trees for the full solution to be tested on for the evaluation of our core selection algorithms.

For a direct comparison of our model to its distributed counterpart in the singular solution space, we generated a model, *SINGULAR*, which applies the entire architecture of *SPAN* this time to process in the singular solution space. The core selection component of *SINGULAR* differs from that of *SPAN* in its domination count, which now is an attribute of a core rather than a core–source pair as the attribute uniquely describing a core-tree, i.e., $r_s \in T_{(c,s)} \iff r \in D_u(c,S) \forall r \in R, s \in S,\ c \in C_u$. According to this, the domination count is described for a core rather than core–source tuple, and specifies the number of receivers dominated by the core for all sources in the group. The candidate core c returning the highest value for $|D(c,S)|$ as the primary criterion and c being closest in average cost-distance to source set as the secondary criterion is selected to be the next member of the core set.

We tested the performance of all heuristics for the cost metric which is the sum of the costs of all the links traversed by exactly one data packet from each source to each

receiver in the group. We used sample domains of size 60, and tested groups sparsely distributed throughout the domain. We used Waxman's model [21] for sampling the domains. Our domains have the average node degree in range (3.5, 5) and the average of average node degrees of our sample domains is 4.46. In all cases, the source and receivers in the group are randomly distributed in the domain. We maintained a 90% confidence level and 10% confidence interval in our measurements.

In order to maintain the same "scale" for the symmetric (*GREEDY* and *m-GREEDY*) and asymmetric (*SPAN* variations and *SINGULAR*) models being evaluated, we avoided asymmetric domains and tested our samples on symmetric networks in which the link costs and delays are equal in both directions of the link. Our results compare *GREEDY* and *m-GREEDY* to their asymmetric alternatives within their intended design setup with no restriction on the functionality of the asymmetric models.

We conducted all measurements on normalized delay-bound values for each case. Consider a source and receivers sets, respectively, S and R in a group. We define *critical-delta*, Δ_{critical}, for the group in a given domain as $\max_{s\in S, r\in R}\{d(s,r)\}$ where $d(s,r)$ is the minimum delay-distance between the nodes s and r in the domain. In other words, critical-delta is the minimum delay-bound that leads to a successful solution for the group, and the domain provides no solution for the given multipoint communication group sample if the delay-bound is any smaller than

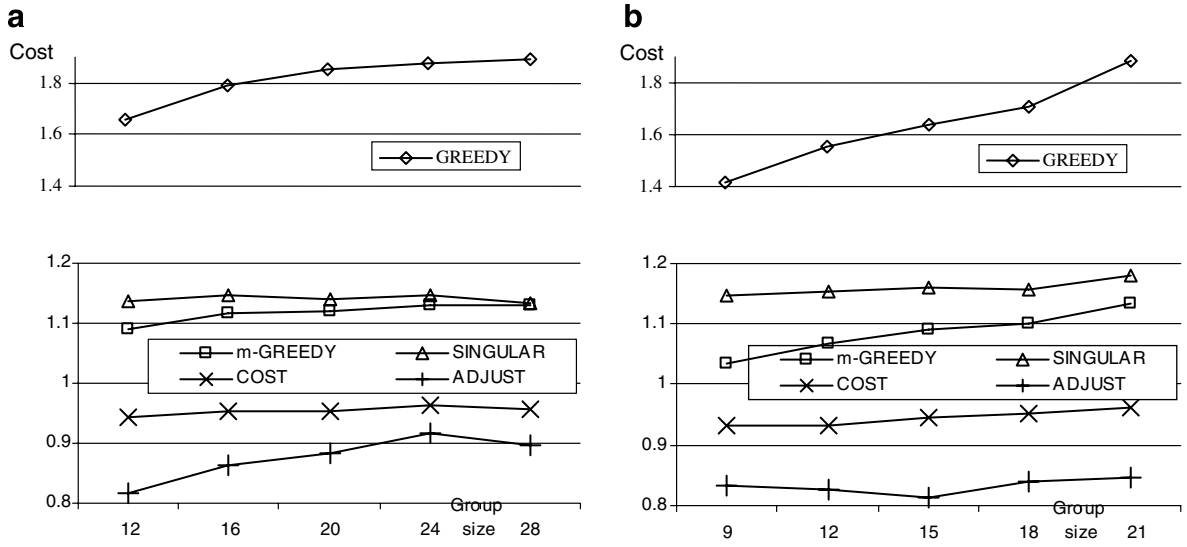

Fig. 5. Evaluation for cost performance on group size, $|R \cup S|$. $|R| \in [9,12,15,18,21]$, $|S| \in [3–7]$, $|S|/|R| = 1/3$. $\Delta = \Delta_{\text{critical}}$. (a) No sources are also receivers. (b) All sources are also receivers.

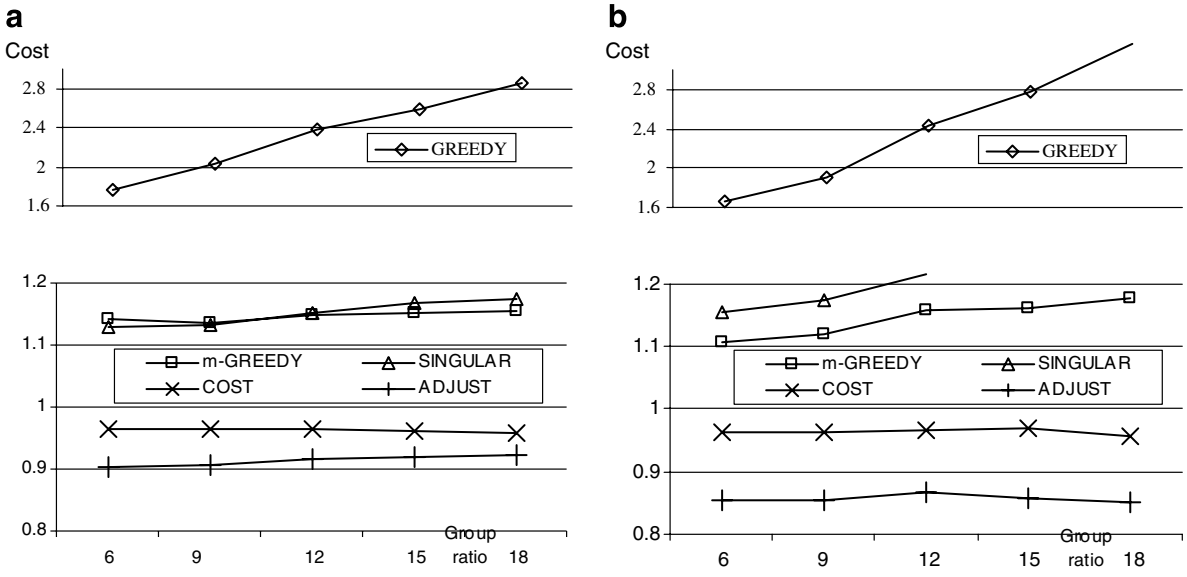

Fig. 6. Evaluation for cost performance on group ratio, $|S|/|R|$. $|R| = 24$. $|S| \in [6,9,12,15,18]$. $\Delta = \Delta_{\text{critical}}$. x-Axis indicates $|S|$ for fixed value of $|R|$. (a) No sources are also receivers. (b) All sources are also receivers.

critical-delta. Consider also the definition of *maximum-delta*, Δ_{max}, which is $\max_{s\in S,r\in R,c\in C}\{delay(s,c)+delay(c,r)\}$. Maximum-delta specifies the "boundary" where the results to be returned by anyone of the algorithms are no longer affected by the delay-bound parameter. In other words, maximum-delta is the upper-bound for the delay-bound range, beyond which any algorithm would be feasible. The range $[\Delta_{critical}, \Delta_{max}]$ specifies, for a problem instance on a given model the minimal delay-bound range of all possible solutions.

We also normalized the cost results of each algorithm for the results of the "reference" model which we chose as *SPAN/BASIC*. According to this, the cost results of each one of the other models are divided by the corresponding outcome obtained from the reference under the same measurement setup, and the indicated results on cost are relative performances to that of the reference. In our figures, the performance of the reference model appears as is 1.00 accordingly.

Figs. 5–8 show the results of our evaluations of the models tested on the group size, sources-to-receivers ratio, the delay-bound of the application and overlap of the source and receiver sets, respectively. A prominent finding of our measurements is the poor performance of *GREEDY* compared to the models tested. From our analysis, *GREEDY* does not consider core-trees and source-trees separately and constructs, for each receiver partition, a tree rooted at the core spanning all the sources in the group. The resulting tree combines the core trees and source trees and the data stream from a particular source is redundantly delivered to the other sources as well as to the receivers, adding on the delivery cost. Our figures depict *GREEDY*

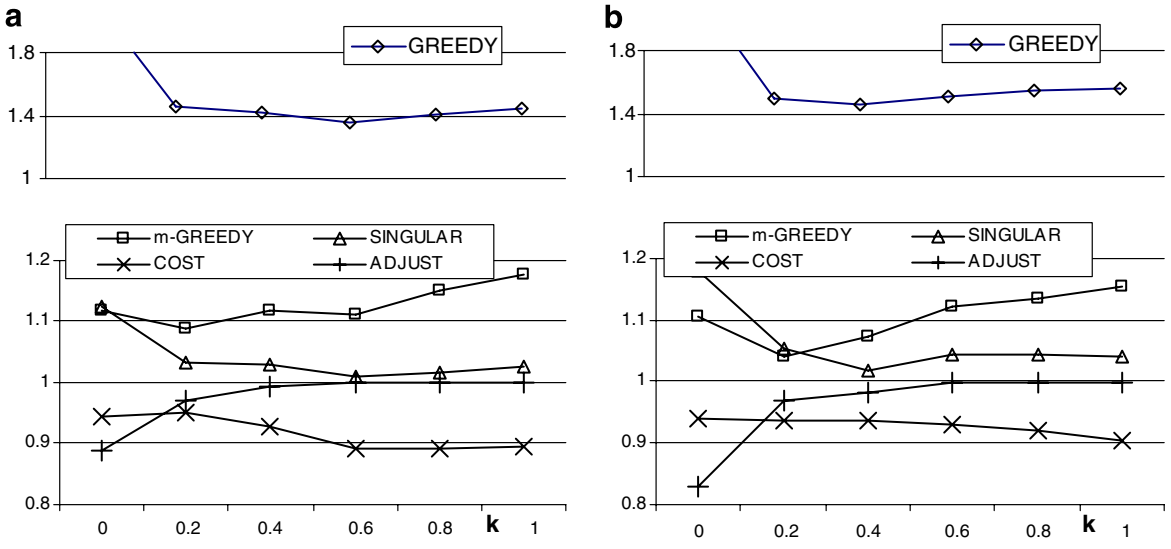

Fig. 7. Evaluation for cost performance on delay-bound of the application. $\Delta=\Delta_{critical}+kV$ where $V=\Delta_{max}-\Delta_{critical}$, $k\in[0, 0.2, 0.4, 0.6, 0.8, 1.0]$. $|R|=18$. $|S|=9$. (a) No sources are also receivers. (b) All sources are also receivers.

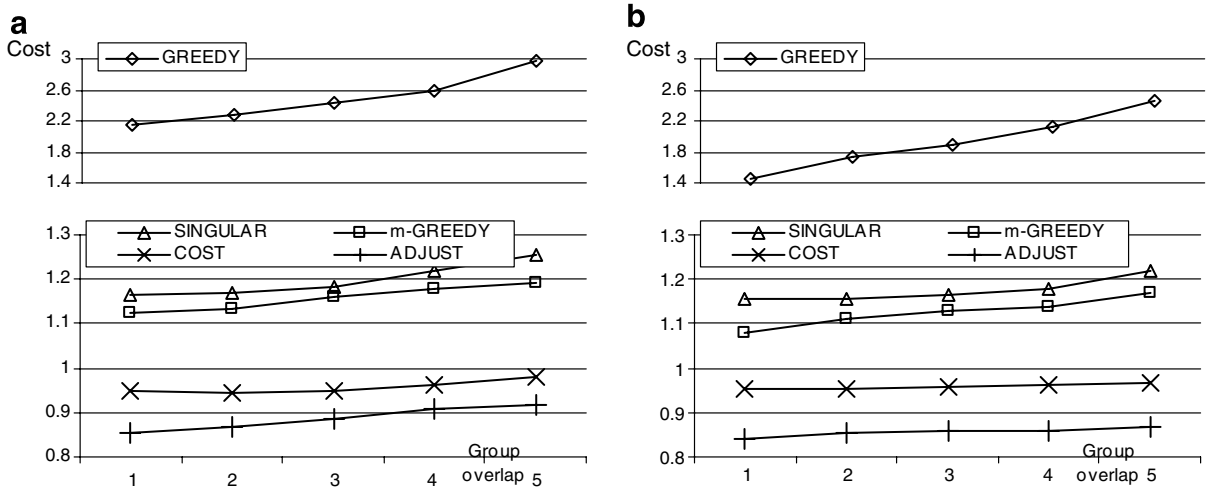

Fig. 8. Evaluation for cost performance on group overlap, $|S\cap R|$. $|R|=24$. $|S\cap R|\in[4,6,8,10,12]$. $\Delta=\Delta_{critical}$. x-Axis shows the range as [1–5] = [4,6,8,10,12]. (a) Half of the sources are also receivers. (b) All sources are also receivers.

at a separate scale, leaving the relatively close performance of the remaining models in a "magnified" range for a closer comparison. The difference in the performances of *SPAN* and *SINGULAR* attributes solely to their respective core selection algorithms since both models use the exact same tree construction module. Thus, *SPAN*'s higher cost-performance is a direct verification of our claim that non-singular solution space offers potential improvement on the efficiency of the solutions.

ADJUST outperforms *COST* for different group sizes (Fig. 5) and ratios of group members (Fig. 6) whenever the delay-bound is tight. Fig. 7 shows performances of the algorithms at varying values of the application delay-bound. At tight delay-bounds, *ADJUST* picks up the high performance of *SPAN* and further improves the results for cost efficiency. Domination count is a measure of the structure of the resulting multipoint paths – optimization on the domination count of the selected cores leads to less complicated trees. *ADJUST*'s performance advantage over *COST* at critical values of the application delay bound shows the significance of the path structure on the cost-efficiency of the path. However, *SPAN* tends to reduce the number of core selected when the delay-bound of the application is relaxed, leaving no room for *ADJUST* to find further improvements among the core trees as evident in Fig. 7. *COST*'s relatively steady performance over *SPAN* in all figures further reflects the effectiveness of the domination count as a criterion for measuring the transmission cost of the resulting solution.

ADJUST's performance improves when the source and receiver sets overlap under the same experimental setup (Fig. 8 and case (b) compared to (a) in all figures). This result is expected, since *SPAN* places the group members on the first selected core tree qualifying to serve/dominate each member. *ADJUST* further moves the members among the trees for their placement for cost efficiency, and the potential gain on the accurate placement of a member which is both a source and a receiver in the group is higher.

6. Conclusions

In this paper, we presented two core selection algorithms, *COST* and *ADJUST*. Our algorithms are the first core selection algorithms in the literature [12] processing on multiple metrics, cost and delay, for the optimization of the results for QoS orientation. *COST* mainly optimizes the cost distances to be traveled between the source–core and core–receiver pairs on the multicast trees, while ADJUST selects the cores based on the numbers of nodes they dominate and adjusting the set based on cost. Both algorithms consistently outperform their counterparts in the literature. *ADJUST* further improved the cost efficiency up to 20% compared to the algorithms proposed to date. *ADJUST* is particularly preferable for any group application demanding tight delay bounds while *COST* is at its best performance at relatively relaxed bounds when compared to existing the models. *COST* constructs the tress whereas *ADJUST* processes on a given set of trees to improve their performance. The two algorithms can be combined so that *COST* can be run first with *ADJUST* processing on its result to further improve the solution. *ADJUST* is designed to alter the tree structures only for better outcomes, whereas *COST* outperforms *SPAN* as demonstrated in the tests. Combining the two algorithms produces results only for better performances. Both *COST* and *ADJUST* operate for constrained solutions in the *SPAN* framework [10], which is distributed and asymmetric, on the information available to the node processing for core selection in this framework, and are applicable to Internet routing domains currently in operation.

Broadband group communication over the Internet is becoming ubiquitous over a wide range of services. These applications are usually delay-sensitive and demanding on network resources. The core-based architecture offers the significant advantage of partitioning the inter-domain route construction on QoS-demands of the applications into intra-domain problems by the placement of core nodes within the autonomous routing domains to coordinate transmission to receivers and/or to border routers for further transmission across ASs. *SPAN* and its extensions presented in this paper operate on local distance–vector information available at the routers, with no modification on their functionality. Our models can further enhance the support of inter-domain groups for participants across ASs through the construction and management of the intra-domain routes coordinated by the efficient placement of cores in each of the domains.

References

[1] A. Ballardie, Core Based Trees (CBT version 2) Multicast Routing – Protocol Specification, RFC 2198, September 1997.
[2] A. Ballardie, Core Based Trees (CBT) Multicast Routing Architecture, RFC 2201, September 1997.
[3] K.L. Calvert, E.W. Zegura, M.L. Donahoo, Core Selection Methods for Multicast Routing, IEEE ICCCN, Las Vegas, NV, USA, 1995.
[4] S.M. Chung, C.-H. Youn, Core selection algorithm for multicast routing under multiple QoS constraints, IEE Electronic Letters 36 (4) (2000) 378–379.
[5] S. Deering, D. Estrin, D. Farinacci, V. Jacobson, Protocol independent multicasting (PIM), dense mode protocol specification, Technical report, IETF-IDMR, 1994.
[6] F. Font, D. Mlynek, Applying clustering algorithms as core selection methods for multiple core trees, IEEE International Conference on Communalizations (2004).
[7] H. Frey, I. Stojmenovic, On delivery guarantees of face and combined greedy-face routing algorithms in Ad Hoc and sensor networks, MOBICOM, Los Angeles, CA, USA, 2006.
[8] M. Garey, D. Johnson, Computers and Intractability, W.H. Freeman, 2000.
[9] M. Kim, Y.-C. Bang, H-J. Lim, H. Choo, On efficient core selection for reducing multicast delay variation under delay constraints, IEICE Transactions on Communication E89-B (2006) 2385–2392.
[10] A. Karaman, H.S. Hassanein, Extended QoS-framework for Delay-constrained Group Communication, International Journal of Communication Systems, in press.

[11] A. Karaman, H.S. Hassanein, DCMC – delay-constrained multipoint communication with multiple sources, IEEE Symposium on Computers and Communications (ISCC) (2003).
[12] A. Karaman, H.S. Hassanein, Core-selection algorithms in multicast routing – comparative and complexity analysis, Computer Communications 29 (8) (2006) 998–1014.
[13] H.-C. Lin, Z.-H. Lin, Selection of candidate cores for core-based multicast routing architectures, IEEE International Conference on Communalizations (2002) 23.
[14] J.A. Sanchez, P.M. Ruiz, I. Stojmenovic, Energy efficient geographic multicast routing for sensor and actuator networks, Journal of Computer Communications, in press.
[15] H.F. Salama, Multicast routing for real-time communication on high-speed networks, Ph.D., Thesis, Department of Electrical and Computer Engineering, North Caroline State University, 1996.
[16] C. Shields, J.J. Garcia-Luna-Acevez, The ordered core based tree protocol, IEEE INFOCOM (1997).
[17] S.B. Shukla, E.B. Boyer, J.E. Klinker, Multicast tree construction in network topologies with asymmetric link loads, TR NPS-EC-94-012, Naval Postgraduate School, September 1994.
[18] P.R. Sheu, S.T. Chen, A fast and efficient heuristic algorithm for the delay and delay variation-bound multicast tree problem, Computer Communication 25 (8) (2002) 825–833.
[19] H. Takahashi, A. Matsuyama, An approximate solution for the steiner problem in graphs, Math. Jap. 24 (1980) 573–577.
[20] D.G. Thaler, C.V. Ravishankar, Distributed center-location algorithms, IEEE Journal on Selected Areas in Communication 15 (3) (1997) 291–303.
[21] B.M. Waxman, Routing of multipoint connections, IEEE Journal of Selected Areas in Communication (1988) 1617–1622.
[22] L. Wei, D. Estrin, The trade-offs of multicast trees and algorithms, IEEE ICCCN, San Francisco, CA, USA, 1994.
[23] D. Zappala, A. Fabbri, V. Lo, An evaluation of multicast trees with multiple active cores, Journal of Telecommunication Systems, Kluwer, March 2002, pp. 461–479.

Ayse Karaman is holding a B.Sc. and an M.B.A. from Middle East Technical University in Ankara, an M.Sc. from McMaster University in Hamilton ON and a Ph.D. from Queen's University in Kingston ON. Her academic research primarily lies in the area of combinatorial algorithms. Her current work focuses on design and analysis of routing algorithms and protocols for traffic engineering and performance efficiency in communication networks. Her background includes wast industrial experience at development of large-scale systems at project leader/manager level. Dr. Karaman is currently teaching Çankaya University in Ankara.

Hossam Hassanein is a leading researcher in the School of Computing at Queen's University in the areas of broadband, wireless and variable topology networks architecture, protocols, control and performance evaluation. Before joining Queen's University in 1999, he worked at the department of Mathematics and Computer Science at Kuwait University (1993–1999) and the department of Electrical and Computer Engineering at the University of Waterloo (1991–1993). Dr. Hassanein obtained his Ph.D. in Computing Science from the University of Alberta in 1990. He is the founder and director of the Telecommunication Research (TR) Lab http://www.cs.queensu.ca/~trl in the School of Computing at Queen's. Dr. Hassanein has more than 250 publications in reputable journals, conferences and workshops in the areas of computer networks and performance evaluation. Dr. Hassanein has organized and served on the program committee of a number international conferences and workshops. He is a senior member of the IEEE and serves as the Secretary of the IEEE Communication Society Technical Committee on Ad hoc and Sensor Networks (TC AHSN). Dr. Hassanein is the recipient of Communications and Information Technology Ontario (CITO) Champions of Innovation Research award in 2003.