THE COMPLEXITY OF RP SELECTION IN MULTICAST CHANNELIZATION

Fei (Sophie) Che and Errol L. Lloyd [1]
University of Delaware
Newark, DE 19716

July 23, 2009

Abstract

Rendezvous point (RP) selection for multicast groups is the problem of selecting a node to serve as the RP-host for a multicast group. We consider rendezvous point selection in the context of channelization where groups have been established based on user preferences for a set of available flows. Thus, each of the flows associated with a group will arrive at the node that serves as the RP-host for that group, from which those flows will be multicast to the group subscribers. We study the simultaneous assignment of RP-hosts for a collection of multicast groups with the dual goals of a) not overloading any single node serving as a host; and, b) minimizing the total network traffic. Toward those ends we consider two versions of the problem. For Bounded Host Assignment, we give a polynomial time algorithm for finding an optimal assignment. For Host Traffic Constrained Assignment, we establish that the problem is NP-complete and then study approximation algorithms. Simulation results are provided for the latter problem comparing the effectiveness of the solutions produced by our algorithms with optimal solutions.

1 Introduction

The study of rendezvous point selection in this paper is based on taking as input a *collection* of multicast groups along with a set of network nodes that are potential hosts and a matrix where c_{ij} specifies the network cost of assigning group i to host j. When a host is assigned to a group this means that each flow received by that group is routed from its source to the group host which then multicasts those flows to the users subscribing to the group. The goal is to assign a host to each group in a way that minimizes the total cost of that assignment.

This problem arises in the context of *channelization* [AG01] wherein a specified number of multicast groups are formed from a set of flows, a set of users and a set of user preferences. The goal in channelization is to construct groups in a way that the network cost in terms of the total sizes (or numbers) of flows delivered to users is substantially less in comparison with using a single multicast group wherein all flows are delivered to all users. Note that channelization computes network costs so as to minimize a cost function involving the total bandwidth consumed and the amount of unwanted information received by receivers [AG01]. The channelization problem itself does not consider the specific cost of getting the flows to the users. This aspect, wherein costs are computed for each possible assignment of a group to a host and the goal is to assign groups to hosts in such a way that the total network cost is minimized, is the subject of this paper.

Specifically, we consider the simultaneous assignment of RP-hosts for a collection of multicast groups with the dual goals of a) not overloading any single node serving as a host; and, b) minimizing the total network traffic. Toward those ends we consider two versions of the problem. In the first, Bounded Host Assignment (BHA), there is a limit on the number of groups that can be assigned to any host. For this problem we give a polynomial time algorithm for finding an optimal assignment. For the second, Host Traffic Constrained Assignment ($HTCA$), there are host specific limits on the total flow bandwidth that a host can handle. We establish that $HTCA$ is NP-complete even if minimizing the total network cost is not a consideration, and then discuss approximation algorithms for $HTCA$ both with

[1]Email: {fei, elloyd}@cis.udel.edu. Prepared through collaborative participation in the Communications and Networks Consortium sponsored by the U. S. Army Research Laboratory under the Collaborative Technology Alliance Program, Cooperative Agreement DAAD19-01-2-0011. The U. S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation thereon.

and without considering the total network cost. Simulation results are provided for the latter problem comparing the effectiveness of the solutions produced by our algorithms with optimal solutions.

2 Background and Related Work

This section describes background and related work on rendezvous point selection and on channelization.

The concept of a rendezvous point (RP) as utilized in this paper appears in *CISCO* documentation on "Configuring a Rendezvous Point" [CI08] which says: *An RP acts as the meeting place for sources and receivers of multicast data. ... sources must send their traffic to the RP. This traffic is then forwarded to receivers down a shared distribution tree.* In [CI08], while general RP assignment methods are described, no specifics are given as to how to best select an RP.

Four categories of methods for RP selection[2] are described in [CZD95]. Simulation results in [CZD95] show that topology based and group based selections significantly outperform arbitrary and random selections.

Several natural RP selection algorithms and a range of performance metrics are considered in [FHM98]. The overall conclusion is that center based methods perform the best and that among those, using the center of the multicast tree is recommended.

In [MWM02] RP selection is studied in a videoconferencing setting where all participants are both sources and receivers. Here, the RP selection goal is to minimize a weighted sum of the costs of a unicast tree wherein group nodes forward packets to the RP and a *Steiner tree* wherein the packets are distributed to all members of the group.

Important aspects of the RP selection problems that we study relate to the underlying channelization problem and to the groups formed as a result of solving that channelization problem. Channelization was first studied as a mechanism for creating a collection of multicast groups in [AG01] and the description given here is based on their formulation, though with modified notation. An instance of the *channelization problem* consists of:

- t, a positive integer indicating an upper bound on the number of multicast groups to be formed
- a set F of flows $F_1, F_2, \cdots, F_m$ where the size (in bytes) of flow F_i is denoted by s_i
- a set of users $U_1, U_2, \cdots, U_p$
- for each user U_i, a set $P_i \subseteq F$ – these are the *preferences* of user U_i and specify the minimum set of flows that must be delivered to U_i

A solution to a channelization instance is a surjective mapping M from flows and users to $1, 2, \cdots, t$ such that for each user U_i, $P_i \subseteq \cup_{j \in M(U_i)} G_j$, where $G_j = \{F_k : j \in M(F_k)\}$. Relative to this mapping, if $j \in M(U_i)$ then user U_i is said to *subscribe* or *belong* to group j, and if $j \in M(F_i)$ then flow F_i is said to *belong* to group j. Thus, we can restate the requirement for the mapping M to be: for each user, each of the flows desired by that user must belong to at least one of the groups subscribed to by that user. Note that each user and each flow may be mapped to more than one group.

3 Problem Specification

In this section we formally define the two problems studied in this paper. For both problems we assume that there is an underlying channelization problem that has already been solved wherein groups have been created based on user requests for flows. We begin with the most basic RP assignment problem, where a node can serve as the RP for at most a single group:

Bounded RP-Host Assignment (BHA):

- Input: A set G of t groups, a set H of n *hosts*, and values c_{ij} for $1 \leq i \leq t$ and $1 \leq j \leq n$, where c_{ij} indicates the network cost that is incurred if host H_j serves as the RP for group G_i.
- Output: An injective mapping f from G to H such that $\sum_{i=1}^{t} c_{i,f(i)}$, the *total network cost*, is minimized.

We have several comments about this problem formulation. First, for the input, note that the costs c_{ij} can be calculated in a number of ways. Section 4 discusses the calculation of this cost in detail. Second, note that the set of hosts could possibly include all of the network nodes or could consist of a limited subset of them. Regardless, we assume that the number of hosts is at least equal to the number of groups. Third, for a given output, we say that group G_i is *assigned to* host $H_{f(i)}$.

[2] In several of the works cited in this section, the term "core" is used instead of RP. For clarity and consistency we will use the term RP in place of core in describing those papers.

Further extending the way in which hosts might constrain the groups for which they serve as the RP, we consider a host traffic constrained version of the problem. That is, the critical aspect for a host is not the number of groups for which it serves as RP, but rather the total amount of traffic passing through that host as a consequence of that host serving as an RP. In the following each host specifies an upper bound on the total traffic that it is willing to handle as an RP:

RP-Host Traffic Constrained Assignment (HTCA):

- A set G of t groups, a flow size f_i for each group, a set H of n *hosts*, traffic capacity bounds b_i for each host, and values c_{ij} for $1 \leq i \leq t$ and $1 \leq j \leq n$, where c_{ij} indicates the network cost that is incurred if host H_j serves as the RP for group G_i.
- Output: An injective mapping g from G to H for which the traffic bounds for each host are inviolate and $\sum_{i=1}^{t} c_{i,g(i)}$, the *total network cost*, is minimized.

4 Channelization Considerations

The fundamental question addressed in this paper is how to assign groups to hosts so as to minimize network costs. A key question then is: *What is the network cost of assigning a group to a particular host?* While making this determination is somewhat orthogonal to the main question that we study, it remains a critical factor in the overall cost of the solution that we produce. In this section we discuss how we determine this cost. At the outset we note that the algorithms and solutions we give in subsequent sections are *not* dependent on the use of these particular costs.

The value that we utilize in this paper as the cost of assigning a group G_i to a host H_j is based on the cost of transmitting these two sets of data: 1) for each flow in G_i, the information provided by that flow is transmitted from the source of the flow to H_j; and 2) those flows are then transmitted using multicast to the users who are subscribing to G_i.

In this paper our goal is to have the cost associated with assigning G_i to H_j reflect as closely as possible the total network traffic created by that assignment. Accordingly, we sum the costs for the two components itemized above. We compute those two costs as follows:

1. We take the cost of transmitting a flow F_k to H_j to be $s_k * len_{kj}$, where s_k is the size of F_k in bytes (as defined in section 2.2) and len_{kj} is the length of the shortest path in the network from the source of F_k to H_j. Note that the shortest paths between H_j and all of the flows from all of the groups can be computed with a single run of Dijkstra's shortest path algorithm.

2. The cost of transmitting the flows from H_j to the users depends on the multicast tree that is utilized. Such trees are most often based on Steiner trees rooted at H_j. Since finding optimal Steiner trees is NP-hard, the use of a heuristic is required. Note that the main algorithmic results of this paper are independent of which heuristic is utilized. However, for definiteness we choose to use, for group G_i, the Steiner tree that results from merging the shortest paths (in terms of the number of links) from each user in G_i to the prospective host H_j. Thus, the cost of transmitting the flows from H_j to the users in G_i is the combined size of those flows multiplied by the number of links in the constructed Steiner tree.

Theorem 4.1 *The total cost of assigning each group G_i to each host H_j can be determined in time $O(e+v \log v+ e')$ where there are e links and v nodes and e' is the sum of the sizes of the computed Steiner trees.*

We conclude this section by noting that there are a multitude of other possibilities for the network cost.

5 BHA in Polynomial Time

This section describes a polynomial time algorithm for solving the BHA problem. The approach we take is to construct a complete bipartite graph, where one set of nodes corresponds to the groups and one set of nodes corresponds to hosts. The edge between a group and a host is weighted with the network cost if that host is the RP for that group. The problem then becomes one of finding a *minimum cost maximum matching*[3] for the constructed graph. The details are given in Algorithm 1.

In the remainder of this section we analyze Algorithm 1. The following theorem follows directly from the statement of the algorithm:

[3]In minimum cost maximum matching, the objective is to find a 1:1 matching with as many matches as possible, and among all such matchings, to find one that minimizes the sum of the costs of the edges in the matching.

Algorithm 1 – OPTIMAL BHA ALGORITHM

Input: An instance $\langle G, H, c_{ij} \rangle$ of BHA.
Output: An injective mapping f from G to H.

1: Construct a complete bipartite graph $BG = \langle V_g, V_h, E \rangle$ where there is one vertex in V_g for each group in G, and one vertex in V_h for each host in H. The edge between the vertices corresponding to group G_i and host H_j has weight c_{ij};
2: Compute a *min cost maximum matching* f of BG;
3: **return**f;

Theorem 5.1 *The running time of Algorithm 1 is* $O(t* n + T_{mwm})$ *where* $O(T_{mwm})$ *is the running time of an algorithm for minimum weight maximum matching.*

The present best running time for minimum weight maximum matching is $O(n(m + n \log n))$ [G90].

Next we discuss the correctness of the algorithm. We begin by noting that since the graph constructed in step 1 is a *complete* bipartite graph, and since we assume that $|G| \leq |H|$, it follows that the matching computed in step 2 is of cardinality $|G|$. Thus, every group is matched with a host. Furthermore, since the graph constructed in step 1 contains a edge between every group and every host, it is possible for any group to be matched with any host, and more generally, any matching between the groups and the hosts is possible. Since step 2 finds the minimum cost matching of cardinality $|G|$, this assignment of groups to hosts is, as desired, of minimum total cost. Thus, we have:

Theorem 5.2 *Algorithm 1 computes an injective mapping from G to H such that the total network cost is minimized.*

In the remainder of this section we relax the constraint that each host can be the RP for a single group by allowing each host to specify the maximum number of groups for which it is willing to serve as the RP. Accordingly we associate a non-negative integer d_j with host H_j indicating that H_j is willing to serve as the RP for up to d_j groups. To solve this relaxed version of problem, we extend Algorithm 1 as follows:

1. For each host H_j, create h_j nodes in V_h. We say that each of these nodes *mirrors* H_j.
2. The edge between the vertex corresponding to group G_i and any vertex that mirrors host H_j has weight c_{ij}.

The correctness of the extended algorithm follows from the construction and the proof of the correctness of Algorithm 1. The running time of the extended algorithm is based on the number of vertices in V_h and V_g:

Theorem 5.3 *The running time of extended Algorithm 1 is* $O(t * d' + T_{mwm})$ *where* $d' = \sum_{j=1}^{n} d_j$ *and* $O(T_{mwm})$ *is the running time of an algorithm for minimum weight maximum matching.*

6 HTCA Complexity

In this section we consider the RP selection problem when there is a bound for each potential host in H of the total traffic that may be handled by that host. Such constraints are related to quality of service issues in not overloading a particular host.

We begin by noting that associated with each group there is a set of flows provided by that group and a set of users who subscribe to the group. Note that if there is a bound on the traffic handled by a host and if each host can only be assigned a single group, then the problem can be handled by the algorithm of the prior section by simply not including a (group, host) edge in the bipartite graph for a host that cannot handle a particular group because the total size of the flows for that group is larger than the traffic bound for that host.

Thus, we assume that there is no apriori bound on the number of groups that may be handled by any given RP-host. Rather, there is only a constraint on the total size of the flows handled by the RP-host.

6.1 The complexity of HTCA

In this section we consider complexity issues associated with HTCA. We begin by establishing:

Theorem 6.1 *RP-host traffic constrained assignment (HTCA) is strongly NP-complete.*

The proof is based on a reduction from the *3-Partition* problem [GJ79]. The details are omitted here due to space constraints.

Note that because $HTCA$ is strongly NP-complete, there is no possibility of a pseudo-polynomial algorithm when the number of hosts is not fixed. Also, by doing the

reduction from *Partition* instead of 3-Partition, we can show that the host assignment problem is NP-complete even there are just two hosts.

6.2 Approximation algorithms for $HTCA$

In this section we consider approximation approaches for two versions of $HTCA$ where we do (do not) consider the issue of minimizing the network cost as well as meeting the host capacity bounds.

6.2.1 $HTCA$ without network costs

We begin by considering $HTCA$ when minimizing the network cost is not an issue. That is, the goal is to assign groups to hosts so that a maximum number of groups are assigned to hosts and so that all of the host capacity bounds are respected. We will refer to this as MAG-$HTCA$ (Maximize Assigned Groups). Relevant to MAG-$HCTA$ is the *Multiple Knapsack Problem (MKP)*:

- Instance: A pair (B, S) where B is a set of m knapsacks and S is set of n items. Associated with each knapsack is a capacity and associated with each item is a *size* and a *profit*.
- Question: Find a subset U of S of maximum profit such that the U has a feasible packing in B.

It is straight-forward to change an instance of MAG-$HTCA$ into an instance of MKP by letting hosts become knapsacks, letting host capacity constraints become knapsack capacities, and letting groups be items with sizes equal to flow sizes and with all profits being 1. Note that the profit of 1 reflects the assignment of the group to a host. Since there is a PTAS for MKP [CK2000], we have:

Theorem 6.2 *There is a PTAS (polynomial time approximation scheme) for MAG-HTCA.*

Thus, given any $\epsilon > 0$, there is a polynomial time algorithm that produces a solution to MAG-$HTCA$ that is within a $1+\epsilon$ factor of the maximum number of groups assigned to hosts.

6.2.2 $HTCA$ with network costs

Here we consider the assignment of groups to hosts so that the host capacity bounds are respected and so that the network cost is minimized. This is a difficult dual optimization problem since $HTCA$ is NP-hard even without trying to minimize the network cost. In this section we consider four approaches to $HTCA$ with network costs:

1. **Reduction to Generalized Assignment**: In the Generalized Assignment Problem (GAP) [ST93], the input is a set of independent jobs and a set of unrelated parallel machines where job i requires time p_{ij} when processed on machine j and incurs a cost of c_{ij}. Further, each machine has a maximum processing time, and the goal is to assign jobs to machines so that for each machine, the total time of its assigned jobs does not exceed its maximum processing time and such that the total incurred cost is minimized. The reduction of $HTCA$ with network costs to GAP is accomplished by mapping groups to jobs, hosts to tasks, flow sizes to times, host traffic constraints to maximum processing times, and network costs to costs. Note that this reduction is approximation preserving, so that any approximation algorithm for GAP will apply to $HTCA$ with network costs. Indeed, an approximation algorithm for GAP is given in [ST93]. That algorithm takes as input a value C, and returns an assignment for which the total cost does not exceed C and in which the maximum processing times are not exceeded by more than a factor of 2. Unfortunately, it is not clear how to utilize this for $HTCA$. In $HTCA$, the host bounds are inviolate, hence the only possible way to apply the algorithm of [ST93] is to set the host capacity bounds to half of their actual values. But, even then there is an issue in how to set C. There is no apriori value for C and no apparent way of estimating that value.

2. **Greedy assignment on network cost**: In this approach, for each group and each host we calculate a *relative assignment cost (RAC)* which is the network cost of assigning that group to that host multiplied by the percentage of the host capacity used by that group. Then, assignments of groups to hosts are done in increasing order of RAC values. The details are in Algorithm 2. It is easy to see that the running time Algorithm 2 is

Algorithm 2 – GREEDY RAC ASSIGNMENT

```
1: for each group i and host j do
2:     RAC_ij ← c_ij × (f_i/b_j);
3: Sort the t * n RAC values into non-decreasing order;
4: for each RAC_ij in the sorted list do
5:     if group i has not yet been assigned
6: and residual-capacity_j ≥ f_i then
7:         Assign G_i to H_j;
```

$O(t * n \log tn)$ since sorting dominates. Details omitted due to limited space we can also show:

Theorem 6.3 *The worst case approximation ratio of Algorithm 2 is unbounded.*

3. **Geographic assignment:** This method of assigning groups to hosts is related to the geographic locations of the flows and users associated with each group. Specifically, for each group and each host we calculate the Euclidean distance from the *Geographic centroid* [H95] of that group to that host. Then, we assign groups to hosts in increasing order by distnace. The details are given in Algorithm 3. Since the lower bound for computing a convex hull in two-dimensions is $O(n \log n)$ [Yao81], we can show that the running time of Algorithm 3 is $O(Max(t * n \log tn, (m+p)\log(m+p)))$. Details omitted due to space constraints we can also show:

Theorem 6.4 *The worst case approximation ratio of Algorithm 3 is unbounded.*

Algorithm 3 – GEOGRAPHIC ASSIGNMENT

1: **for** each group i **do**
2: Compute the convex hull containing all of the flow sources and users in group i;
3: Find the geographic centroid gc_i of that convex hull;
4: **for** each group i and host j **do**
5: $dist_{ij} \leftarrow distance(gc_i, H_j)$;
6: Sort the $t * n$ dist values into non-decreasing order;
7: **for** each $dist_{ij}$ in the sorted list **do**
8: **if** group i has *not* yet been assigned
9: and *residual-capacity*$_j \geq f_i$ **then**
10: Assign G_i to H_j;

4. **Using 0-1 Integer Programming (0-1 IP)**: Here, we express $HTCA$ with network costs as a 0-1 Integer Programming problem [GJ79]. This is done as follows: Let x_{ij} be a 0-1 variable indicating whether or not group G_i is assigned to host H_j. Then:

Minimize $\sum_{i=1}^{t} \sum_{j=1}^{n} c_{ij} * x_{ij}$

subject to for each group i and host j,

$$\sum_{j=1}^{n} x_{ij} \leq 1 \text{ and } \sum_{i=1}^{t} f_i * x_{ij} \leq b_j,\ x_{ij} \in \{0,1\}$$

Unfortunately, 0-1 IP is NP-hard [GJ79]. But, there are efficient solvers such as $CPLEX$, $GLPK$, and $MINTO$ etc. Those solvers will often find an optimal solution fairly quickly, though in the worst case the running time is exponential.

7 Simulation Results

In this section we describe simulation results for the problem $HTCA$ with network costs. Recall that in subsection 6.2.2 we describe four approaches to $HTCA$ with network costs. In this section we compare the performance of the last three methods: the greedy algorithm on RAC (referred to as *RAC*), the algorithm using Geographic centroid (referred to as *Geocenter*) and 0-1 Integer Programming.

Our experiments are based on channelization instances generated by running the channelization algorithm, *Flow Based Merge (FBM)* [AG01]. That algorithm works as follows: initially assign each flow to a different multicast group. Then iteratively merge two groups that least increase the network cost (considering only the total *bandwidth* delivering flows to groups and then to corresponding users). Once we have generated a channelization solution, then RAC, Geocenter and a 0-1 IP solver $CPLEX$ [CPL] are run on the groups of that channelization solution. Some details of the experimental setup follow.

We begin by randomly generating an instance of channelization. Here, the flows and users are generated with regard to the following parameters:

- high rate *or* low rate – Each flow, once created, is assigned either *high* flow rate r_H with probability $Prob_r$ or *low* flow rate r_L with probability $1 - Prob_r$.
- user preference – Each flow is either *popular* with probability $Prob_p$ or *unpopular* with probability $1 - Prob_p$. Note that the popularity of a flow is uncorrelated to its rate. A user will add a popular flow to its preferences with probability $Prob_{pop}$ and an unpopular flow with probability $Prob_{unp}$. Usually, $Prob_{pop} >> Prob_{unp}$.

The network cost of assigning a group G_i to a host H_j is $c_{ij} = \sum_{k:F_k \in G_i} s_k * len_{kj} + f_i * |T_s|$ where s_k is the size of F_k in bytes, len_{kj} is the length of the shortest path in the network from the flow source F_k to H_j, f_i is the sum of sizes of all of the flows in G_i and $|T_s|$ is the number of links in the Steiner tree rooted at H_j with users as either interior nodes or leaves. It follows that the final network cost of assigning groups to hosts is $\sum_{i=1,j=g(i)}^{t} c_{ij}$ where $g(i)$ is the mapping from G_i to a host.

In Figure 1, we plot results of 30 different instances with the following parameters: For each instance, 400 network nodes are randomly distributed in a square area.

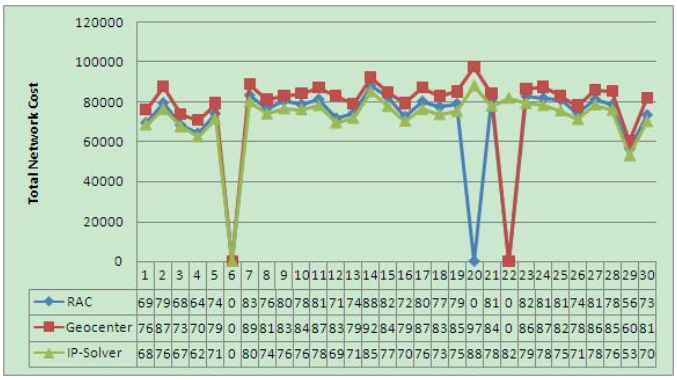

Figure 1: Cost comparison on different approaches

Then, 120 flows, 200 users and 20 hosts are randomly selected from those nodes. Associated with flows and users are their flow rates and user preferences generated with $r_H = 10, r_L = 1, Prob_r = 0.3, Prob_p = 0.4, Prob_{pop} = 0.6, Prob_{unp} = 0.1$. Also, associated with each host is a capacity randomly generated in the range $[60, 80)$. The FBM algorithm runs on this instance to produce an assignment of flows and users to 30 groups. RAC, Geocenter and IP solver are then applied to map groups to those hosts. The x-axis of Figure 1 shows the experiment number, the y-axis shows the total network cost and the table below shows the cost values (in thousands).

From the Figure 1, the two primary observations are that 1) RAC performs nearly as well as optimal and 2) that Geocenter does a bit worse than RAC. Note also that Figure 1 shows cases for both RAC and Geocenter where not all groups can be assigned to some host.

In the figure we only display results for 30 groups and 20 hosts due to space constraints. Results for other combinations of groups and hosts show the same trends as in this figure.

8 Conclusions

In this paper we studied RP selection problems based on an underlying channelization problem where groups have been established. Two versions of the problem were considered. For BHA, we gave a polynomial time algorithm based on the use of *minimum cost maximum matching*. For $HTCA$, we first established its NP-completeness and then gave approximation algorithms for two versions of $HTCA$ where we do (or do not) consider the issue of minimizing the network cost. For $HTCA$ without network costs (MAG-$HTCA$), we show that it is reducible to MKP and thus there exist a PTAS for MAG-$HTCA$. While for $HTCA$ with network costs, we show that it is reducible to GAP. Beyond that, we present two greedy algorithms, RAC and Geocenter, and also apply 0-1 IP to the problem. We compare the performance of the four approaches for the problem $HTCA$ with network costs and the results show that RAC is a good approximation. Improvements to the results given here seem to depend on improvements to approximation results to the MKP or to the GAP.

Acknowledgement: We thank S.S. Ravi of the University at Albany for his insights on this paper.

Disclaimer: The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Laboratory or the U. S. Government.

References

[AG01] M. Adler, Z. Ge, J. F. Kurose, D. Towsley and S. Zabele. "Channelization Problem In Large Scale Data Dissemination", ICNP, 2001, 100–109.

[Yao81] A. C. Yao. "A Lower Bound to Finding Convex Hulls", J. ACM 28, Vol. 28, 1981, 780–787.

[H95] R. Honsberger. "Episodes in Nineteenth and Twentieth Century Euclidean Geometry", Math. Assoc. Amer., 1995.

[CI08] *http://www.cisco.com/en/US/docs/ios/solutions_docs/ip_multicast/White_papers/rps.html*

[CK2000] C. Chekuri and S. Khanna. "A PTAS for the Multiple Knapsack Problem", Proc. 11th ACM-SIAM Symp. on Discrete Algorithms (SODA), 2000, 213–222.

[CPL] *http://www.ilog.com/products/cplex/*

[CZD95] K. L. Calvert, E. W. Zegura, and M. J. Donahoo. "Core selection methods for multicast routing", Fourth Inter. Conf. on Computer Communications and Networks (ICCCN '95), 1995, 638–642.

[FHM98] E. Fleury, Y. Huang, and P. K. McKinley. "On the performance and feasibility of multicast core selection heuristics", Proc. 7th Inter. Conf. on Computer Communications and Networks, 1998, 296–303.

[G90] H. N. Gabow. "Data structures for weighted matching and nearest common ancestors with linking", Proc. 1st ACM-SIAM Symp. on Discrete Algorithms (SODA), 1990, 434–443.

[GJ79] M. R. Garey and D. S. Johnson. "Computers and Intractability: A Guide to the Theory of NP-completeness", W. H. Freeman and Co., 1979.

[MWM02] J. F. Macq, L.A. Wolsey, B. Macq. "A rendezvous point selection algorithm for videoconferencing applications", Proc. IEEE Inter. Conf. on Multimedia and Expo (ICME), 2002, 689–692.

[ST93] D. B. Shmoys and E. Tardos. "An approximation algorithm for the generalized assignment problem", Mathematical Programming, Vol. 62, 1993, 461-474.