Contents lists available at ScienceDirect

Operations Research Letters

journal homepage: www.elsevier.com/locate/orl

A distributed dual ascent algorithm for the Hop-constrained Steiner Tree Problem

Marcelo Santos[a], Lúcia M.A. Drummond[a,*], Eduardo Uchoa[b]

[a] Department of Computer Science, Fluminense Federal University, Brazil

[b] Department of Production Engineering, Fluminense Federal University, Brazil

ARTICLE INFO

Article history:
Received 27 October 2008
Accepted 11 August 2009
Available online 17 September 2009

Keywords:
Distributed algorithms
Multicast routing
Steiner tree

ABSTRACT

The Hop-constrained Steiner Tree Problem is often used to model applications of multicast routing with QoS requirements. This paper introduces a distributed heuristic for the problem based on the application of dual ascent over a graph transformation introduced by Gouveia et al. The proposed algorithm is shown to yield significantly better solutions than the previously known algorithms.

1. Introduction

Several emerging network applications, such as web conferencing and video streaming, require the distribution of large quantities of data to a subgroup of nodes of a network, what is called multicast. The problem of finding the best set of communication paths for a multicast session is known as multicast routing and is often modelled as a Steiner Tree Problem (STP) [1,2]; where one looks for a least costly tree connecting the required nodes. However, only minimizing the connection costs is often not enough. In many applications, there are also quality of service requirements limiting the maximum communication delays.

Suppose that most of the traffic in the multicast session comes from a single source node, so quality of service requirements are imposed on the paths from that node. This leads to the so-called *Depth-constrained Steiner Tree Problem* (DSTP), defined as follows. Let $G = (V, A)$ be a directed graph, where V denotes the set of nodes in the network and A is the set of arcs associated to the communication links, $R \subseteq V$ be the set of required nodes in the multicast, and $s \in R$ be the source node. For each arc $a \in A$ there are two positive numbers, arc cost c_a and arc delay d_a. Bidirectional links are represented by pairs of opposite arcs in A, costs and delays over each such pairs of arcs may be symmetric or not. There is also a positive number H defining the maximum delay limit. The problem is finding a set of arcs A' with minimum cost such that the induced subgraph $G' = (V', A')$ is a directed tree containing a path with total delay not exceeding H from s to every other node in R. One may simplify the problem by replacing path delay by path length. This corresponds to assuming that all arc delays d_a can be made equal to 1 and setting H as the maximum number of arcs allowed in a path. This problem is known in the literature as the *Hop-constrained Steiner Tree Problem* (HSTP), each arc in a path is viewed as a "hop" in the communication. The recent work by Gouveia et al. [3] have shown that the HSTP can be transformed into a STP defined over a suitable defined layered graph, obtaining by stacking $H + 1$ copies of the original graph. This transformation is quite practical. It is shown that specializing known STP (centralized) algorithms to work on the layered graph yields results that are much better than those obtained by previous HSTP algorithms.

Most of the proposed algorithms for solving the STP and its generalizations are centralized. A centralized algorithm assumes that a single processor has full knowledge about the topology and the current state of the whole network. However, in the case of a large wide area network, the overhead to collect, store and update this information can be prohibitive. In this context, there is a need for algorithms that can be distributed over the processors corresponding to the nodes in the network, and where each node only knows about its immediate neighborhood. In this paper we propose a new distributed heuristic (LDAH) for the HSTP, combining the distributed Dual Ascent algorithm for the STP introduced in [4] with the graph transformation given in [3]. This graph transformation is particularly interesting for the distributed case because all the $H + 1$ copies of a node can be represented as processes running in the physical processor corresponding to the node in the original graph.

Distributed heuristics for the more generical DSTP, such as those by Feng and Yum [5], Guo and Matta [6] and Jia [7], can be

* Corresponding address: Rua Passo da Pátria, 156, Bl.E, Niterói RJ, 24240-220, Brazil.

E-mail address: lucia@ic.uff.br (L.M.A. Drummond).

applied on the HSTP. In this paper, Jia's algorithm will be called DSPH, since it is an adaptation of the distributed Shortest Path Heuristic (SPH) [8] for the Depth constraint. The DSPH starts with a tree containing only the source node. At each step, it selects the cheapest path, respecting the depth constraint from the source, that connects the current partial tree to a still non-connected required node. That path is added to the tree. The algorithm finishes when all required nodes are connected. Our computational experiments compare the LDAH with the recently enhanced (and corrected) version of the DSPH by Huang and Lee [9]. The newly proposed algorithm usually gives substantially better solutions. Moreover, unlike DSPH, LDAH also provides a good lower bound on the value of an optimal solution.

2. Sequential dual ascent heuristic

Wong's Dual Ascent (DA) [10] is a primal–dual algorithm for the standard STP, without depth or hop constraints. In this algorithm, each arc $a \in A$ has its non-negative *reduced cost* $\bar{c}_a$, a value that is initialized with the original arc cost. Reduced costs may be only decreased, arcs with zero reduced cost are called *saturated*. Those arcs induce the *saturation graph* $G_S = (V, A_r(\bar{c}))$, where $A_r(\bar{c}) = \{a \in A : \bar{c}_a = 0\}$. As all original costs are positive, this graph starts with no arcs. Most DA operations are defined over the current saturation graph. Let C be the set of nodes of a strongly connected component of G_S, i.e., a maximal subgraph containing a directed cycle going through any pair of nodes in it. This set is a *root component* if (i) C contains a required node, (ii) C does not contain s, and (iii) there is no required node $t \notin C$ reaching C by a path in G_S. Given a root component C, define $W(C) \supseteq C$ as the set of nodes reaching C by paths in G_S and let $\delta^-(W)$ be the directed cut in G consisting of the arcs entering W. The DA algorithm follows:

```
Wong's Dual Ascent
  $LB \leftarrow 0$;
  $\bar{c}_a \leftarrow c_a$, for all $a \in A$;
  While (root components exist in $G_S$) {
    Choose a root component $C$;
    $W \leftarrow W(C)$;
    $\Delta \leftarrow \min_{a \in \delta^-(W)} \bar{c}_a$;
    $\bar{c}_a \leftarrow \bar{c}_a - \Delta$, for all $a \in \delta^-(W)$;
    $LB \leftarrow LB + \Delta$;
  }
Output: $LB$ and $\bar{c}$
```

At first, each required node other than s corresponds to a root component. In each round, a root component C is chosen and the reduced costs of all arcs entering $W(C)$ are decreased by Δ, the smallest such reduced cost. The partial lower bound is increased by the same amount. At least one arc is saturated in each round. Some saturations reduce the number of root components, until there are no root components anymore. At this point, G_S contains at least one directed path from s to every other terminal, i.e., it contains Steiner trees. In order to obtain a good heuristic solution, one should complement the DA with an additional "cleaning" step, removing arcs from G_S until only a single Steiner Tree remains. A very effective cleaning procedure [11] consists in applying the SPH algorithm over the restricted graph G_S. The overall method, called *Dual Ascent Heuristic* (DAH), then obtains a solution and a lower bound guaranteeing its quality.

3. Distributed dual ascent heuristic

A distributed version of the DAH was presented in [4], its agents are associated with the *fragments*. A fragment is defined as a set $W(t)$ formed by all nodes reaching a non-source required node t by a path of saturated arcs. The required node t identifying a fragment $W(t)$ is also known as its *leader*. Let C be the strongly connected component containing t, $W(t) = W(C)$. A node can operate on behalf of several fragments, because it can belong to different fragments.

By definition, all nodes in a fragment are connected to its leader by saturated arcs. Among such arcs, the algorithm keeps in each fragment a tree used for intra-fragment message exchange, *convergecast messages* from nodes in the fragment to the leader and *broadcast messages* from the leader to the fragment's nodes. A fragment *growing round* consists of finding the minimum reduced cost of an entering arc and subtracting this value from the reduced cost of all entering arcs. The first operation is performed using a convergecast, the result is then broadcast. When decreasing the reduced costs of fragment-entering arcs, some new arcs become saturated, the corresponding nodes must be included into the fragment and the fragment's tree is updated. The fragment leader keeps the partial lower bound due to the fragment growing rounds. As fragments grow, their nodes may start to overlap. When fragments have a common entering arc, only one of these fragments may grow through this arc and decrease its reduced cost. This situation represents a classical mutual exclusion problem, where the shared resource is the reduced cost of the common arc. In order to resolve this conflict, only the fragment with the largest identification number will continue growing, while the other fragments are suspended. A fragment finishes its execution when it is reached from the source node. The fragment leader then broadcasts a termination message containing also the partial lower bound so that the source node can calculate the global lower bound. When the source node reaches all terminals, it initiates a broadcast to terminate the DA phase. After that, the distributed SPH algorithm [8] is used on the restricted graph containing only saturated arcs, then producing a Steiner tree.

The distributed DAH has worst case complexities of $O(|R| \cdot |V|^2)$ messages, $O(|R| \cdot |V|^2)$ global time (maximum message sequence), and $O(|R|)$ local time. An alternative to perform the DAH would be electing a leader node to locally solve the problem and then broadcast the solution. It would take $O(|V| \cdot |E|)$ messages and $O(|V|)$ time to concentrate all the relevant information about G in the leader, the local time complexity of the sequential DA algorithm is $O(|E| \cdot \min\{|E|, |R| \cdot |V|\})$. Considering such complexities and the memory demand at the leader node, the fully distributed approach proposed is an appealing alternative for multicast routing.

4. Modelling the Hop-constrained Steiner problem over a layered graph

We use a transformation similar to the one in Gouveia et al. [3] in order to transform a HSTP into a standard STP over a layered graph $G_L = (V_L, A_L)$, defined as follows:

- $V_L = \{(i, h) : 0 \leq h \leq H, i \in V\}$,
- $A_L = A_0 \cup A_1$, where:

$$A_0 = \{((i, h), (j, h + 1)) : (i, j) \in A, 0 \leq h \leq H - 1\}$$
$$A_1 = \{((i, h), (i, h + 1)) : i \in V, 0 \leq h \leq H - 1\}.$$

The cost of the arcs in A_0 is equal to the costs of the corresponding arcs in the original graph, arcs in A_1 receive zero cost. The new source is defined as vertex $(s, 0)$, the other required nodes are $\{(t, H) : t \in R\}$. Fig. 1(a) shows an example of an original HSTP instance with $H = 2$, required nodes are represented as squares, non-required nodes as circles, node 0 is the source, costs appear near the corresponding arcs. It can be seen that the optimal unconstrained STP solution is not feasible for the HSTP, since required node 5 would be connected to 0 by a path with