

Inferring Tree Topologies Using Flow Tests

S. Muthukrishnan*

Torsten Suel†

Radek Vingralek‡

1 Introduction

We consider the problem of discovering the structure of an unknown hierarchical network by means of measuring the maximum flow between the root and selected subsets of leaf nodes. More precisely, we are given a root node and a set of n leaf nodes, each identified by a unique label. The leaf nodes are the leaves of a capacitated hierarchical network. We do not have any additional information about the structure of the network, including the degrees of any internal nodes, the edge capacities, or which leaf nodes are in the same subtree.

Our goal is to infer the structure of the network (up to certain equivalences discussed below) by using only a simple test operation, in which we “switch on” a selected subset of the leaf nodes, causing these nodes to transmit data to the root at maximum speed, and then measure the total rate of data arriving at the root. We will refer to this operation as a *flow test*.

In this note, we derive upper and lower bounds for the number of flow tests required to determine the structure of a tree network. This result is not only of Discrete Math interest, but also of importance in hands-on systems engineering: (1) inferring disk structure in parallel databases, (2) inferring congestion in IP networks, and (3) inferring network structure in hierarchical bus architectures [5]. Due to space constraints, we motivate the description using (1) only.

The problem we consider arises in shared-disk parallel database systems (see, e.g., [2]). In this scenario, we are given a set of processors and a number of storage devices that are connected to the processors by a hierarchical system of buses consisting, e.g., of different levels of SCSI buses plus a system I/O-bus at the highest level. In order to efficiently schedule large parallel accesses to the data (e.g., scans in OLAP workloads), it is advantageous for the database system to know the structure of the bus system, so that any bottlenecks in the I/O architecture can be avoided by appropriate placement and scheduling decisions. However, there is currently no general way to inquire the complete structure of a multi-level hierarchy from the operating system. One

way to explore the structure is by testing the maximum data rate achievable by subsets of the disks, which corresponds to the flow test described above.

2 Preliminaries

The input to the problem consists of a root node r and n nodes that are labeled l_1 to l_n (in no particular order). The nodes $L = \{l_1, \dots, l_n\}$ are the leaf nodes of a tree T , whose structure is unknown. We assume that the root itself has degree 1, in order to model a limit on the total amount of flow coming out of the subnetworks. Each edge e of the tree has a real-valued capacity $c(e)$, and is directed from a child to a parent. A *flow test* is an operation that, given an arbitrary subset of the leaves, returns the value of the maximum flow between the set of leaves and the root. We assume the following three conditions on the tree T :

- (1) Every internal node has in-degree ≥ 2 and out-degree 1.
- (2) The edge capacities along any path from a leaf to the root are monotonically and strictly increasing.
- (3) For every internal node, the total incoming capacity is strictly larger than the outgoing capacity.

We justify these conditions by the observation that in general, there are many trees that are equivalent with respect to their flow routing properties, and that can thus not be distinguished at all by using our flow test operation. However, it can be shown that for every tree, there exists a unique equivalent (unordered) tree that satisfies the above conditions.

3 Upper Bounds

We now describe an algorithm for determining the structure of a tree, or, more precisely, the structure of the equivalent tree satisfying conditions (1) to (3). For a subset $S \subseteq L$ of leaves, we define the following terms: $f(S)$ denotes the maximum flow between S and the root. We say that S is *congesting* if $f(S) < \sum_{l \in S} f(\{l\})$. We say that S is *special* if S is congesting, $f(S) < f(S')$ for any S' with $S \subset S'$, and S does not contain any subset with these properties.

LEMMA 3.1. S is special iff it is the set of all leaves of a subtree of height 1 (i.e., a set of all siblings). Furthermore, the total outgoing capacity of this subtree is equal to $f(S)$.

*AT&T Labs – Research. muthu@research.att.com.

†Polytechnic University, Brooklyn. suel@photon.poly.edu.

‡Intertrust Technologies Inc. rvingral@intertrust.com.

Using this lemma, the problem of discovering the tree structure can be reduced to finding subsets that are special (which we can efficiently check):

LEMMA 3.2. *For any set S of leaves, $n - |S| + 1$ flow tests suffice to check whether S is special, assuming we have already checked all subsets of S .*

We now give a high-level description of our algorithm for discovering the tree structure. We first measure the outgoing capacity of each leaf by performing a flow test on each singleton. Then we start enumerating subsets of leaves of increasing cardinality. Once a special set S is discovered, we replace it by a single node with outgoing capacity $f(S)$, and then include this new node in the generation of subsets, starting again with untested sets of smallest cardinality. We can show:

THEOREM 3.1. *There exists an algorithm for discovering tree structure that uses at most $\sum_{i=1}^{d+1} \binom{2n-2}{i}$ flow tests, where d is the maximum in-degree of any node.*

For constant d , this bound is $O(n^{d+1})$. If d is known, then the bound improves to $\sum_{i=1}^d \binom{2n-2}{i} = O(n^d)$, which is almost tight as shown in the next section. Slightly better and more complicated bounds can be derived in terms of the degrees d_i of the individual nodes. Improvements can also be obtained by considering the *congestion-degree* of a node, which we define as 1 plus the size of the largest set of incoming edges whose joint capacity does not exceed the outgoing capacity of the node.

4 Lower Bounds

We can also prove lower bounds for the number of flow tests that are needed to determine the tree structure. These bounds are almost tight both in terms of the maximum in-degree d and the maximum congestion-degree. For simplicity, we sketch only the case of $d = \sqrt{n}$. Consider a tree with n leaves and in-degree $d = \sqrt{n}$. Assume that each leaf has an outgoing capacity of 1. On the next higher level, each node has outgoing capacity $d - 1$, while the edge entering the root has capacity d . Now consider the flow $f(S)$ arriving at the root during a flow test on a set S of leaves:

- if $|S| < d$, then $f(S) = |S|$,
- if $|S| > d$, then $f(S) = d$, and
- if $|S| = d$, then $f(S) = d - 1$ iff S consists exactly of all d leaves that share a subtree. Otherwise, $f(S) = d$.

We can now prove the lower bound based on an adversary-style argument. We allow the algorithm to

have knowledge about the in-degree d and the entire structure of the tree, except that the algorithm does not know which sets of d leaves are grouped together in the subtrees. According to the above arguments, any flow tests on sets S with $|S| \neq d$ are useless as the result is already known. Hence, the only useful flow tests are those with $|S| = d$, which essentially answer the question of whether the leaves in S are in the same subtree.

A simple argument shows that for any sequence of less than $\binom{n-1}{d-1}$ flow tests, we can construct a tree such that no test is performed on a set S consisting of all leaves of a subtree. We can extend this argument to show that up to $\binom{n-d-1}{d-1}$ additional tests may be required to find the second such set, $\binom{n-2d-1}{d-1}$ for the third, and so on, resulting in the following bound:

THEOREM 4.1. *Up to $\sum_{i=0}^{\frac{n}{d}-1} \binom{n-id-1}{d-1}$ flow tests are required to infer the structure of a tree.*

Note that this bound is $\Omega(n^d)$ for constant d . Also, note that in the above construction, the maximum congestion-degree is equal to the maximum in-degree, and hence the bound also applies to this measure.

5 Discussion and Conclusions

We discussed the problem of discovering the structure of a tree network by performing flow tests. (See [3] for a different problem in this genre.) We have also obtained some results on how to schedule data transmissions on a discovered tree network under different models of network behavior; details of this part of the work will appear in the full version. See also [1, 4] for some related work on this problem.

References

- [1] E. Coffman, M. Garey, D. Johnson, and A. LaPaugh. Scheduling File Transfers. *SIAM J. on Computing*, 14(3):744–780, 1985.
- [2] D. DeWitt and J. Gray. Parallel Database Systems: the Future of High Performance Database Systems. *Communications of the ACM*, 35(6), 1992.
- [3] T. Hagerup, J. Katajainen, N. Nishimura, and P. Ragde. Characterizing Multiterminal Flow Networks and Computing Flows in Networks of Small Treewidth. *J. of Computer and System Sciences*, 57(3):366–375, 1998.
- [4] D. Kagaris, G. Pantziou, S. Tragoudas and C. Zaroliagis. Transmissions in Networks with Capacities and Delays. *Networks*, 33, 1999.
- [5] F. Meyer auf der Heide, H. Racke and M. Westermann. Data management in hierarchical bus networks. *Proc. ACM Symp. Parallel Algorithms and Architectures*, 2000.