Brief Announcement: A Randomized Algorithm for Label Assignment in Dynamic Networks

Meg Walraed-Sullivan, Radhika Niranjan Mysore,
Keith Marzullo, and Amin Vahdat

University of California, San Diego
La Jolla CA, USA
{mwalraed,radhika,marzullo,vahdat}@cs.ucsd.edu

In large-scale networking environments, such as data centers, a key difficulty is the assignment of labels to network elements. Labels can be assigned statically, e.g. MAC-addresses in traditional Layer 2 networks, or by a central authority as in DHCP in Layer 3 networks. On the other hand, networks requiring a dynamic solution often use a Consensus-based state machine approach. While designing Alias [2], a protocol for automatically assigning hierarchically meaningful addresses in data center networks, we encountered an instance of label assignment with entirely different requirements. In this case, the rules for labels depend on connectivity, and connectivity (and hence, labels) changes over time. Thus, neither static assignment nor a state machine approach is ideal.

In the context of large scale data center networks, practical constraints are important. A centralized solution introduces a single point of failure and necessitates either flooding or a separate out-of-band control network to establish communication between the centralized component and all network elements; this is problematic at the scale of a data center. We also require a solution that scales, is robust in the face of miswirings and transient startup conditions, and has low message overhead and convergence time. To this end, we specify the Label Selection Problem (LSP), which is the problem of practical label assignment in data center networks.

In LSP, we consider topologies made up of *chooser* processes connected to *decider* processes, as shown in Fig. 1. More formally, each chooser c has a set $c.deciders$ of deciders associated with it. This set can change over time. Each chooser c is connected to each decider in $c.deciders$ with a fair lossy link. Such links can drop messages, but if two processes p and q are connected by a fair lossy link and p sends m infinitely often to q, then q will receive m infinitely often. Both decider and chooser processes can *crash* in a failstop manner (thus going from *up* to *down*) and can *recover* (thus going from *down* to *up*) at any time. We assume that a process writes its state to stable storage before sending a set of messages. When a process recovers, it is restored to the state before sending the last set of message; duplicate messages may be sent upon recovery. So, we treat recovered processes as perhaps slow processes, and allow for duplicate messages.

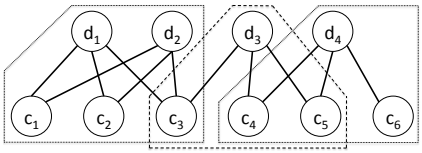

Fig. 1. *Sample Topology*

We require an assignment of labels to choosers such that any two choosers that are connected to the same decider have distinct labels. Figure 1 illustrates sets of choosers and the deciders they share. For instance, chooser c_3 shares deciders d_1 and d_2 with choosers c_1 and c_2 and shares d_3 with c_4 and c_5. Because of this, c_3 may not select the same label as any of choosers c_1, c_2, c_4 and c_5, but c_3 and c_6 are free to select the same label. We denote c's current choice of label with $c.me$: $c.me = \perp$ indicates that c has not chosen a label.

We specify LSP with two properties, **Progress:** For each chooser c, once c remains up, eventually $c.me \neq \perp$ and **Distinctness:** For each distinct pair of choosers c_1 and c_2, once c_1 and c_2 remain up and there is some decider that remains up and remains in $c_1.deciders \cap c_2.deciders$, eventually always $c_1.me \neq c_2.me$. A key difficulty in solving LSP is that a chooser can not necessarily know when its choice satisfies **Distinctness**. This is because its set $c.deciders$ can change over time, thereby introducing new conflicts.

To solve LSP, we introduce a simple, randomized Decider/Chooser Protocol (DCP). DCP is a Las Vegas type randomized algorithm: the labels that are computed always satisfy the problem specification, but the algorithm is only probabilistically fast. It is also a fully dynamic algorithm [3], in that it makes use of previous solutions to solve the problem more quickly than by recomputing from scratch.

We have applied DCP to a variety of problems in large scale networks. For instance, DCP can be used for handoff in wireless networks, with mobile devices functioning as choosers and APs as deciders; this exemplifies a topology with frequently changing links between choosers and deciders. Additionally, by applying DCP to a portion of Alias [2], we drastically reduce the amount of forwarding state needed in network elements. We also apply a modified version of DCP in Alias, in which the chooser is distributed across multiple nodes. Doing so allows for hierarchical address aggregation, which in turn reduces the forwarding state maintained by network elements.

Assigning labels to nodes is not a new problem. Perhaps closest to our problem is [1], which considers the issues of assigning labels to nodes in an anonymous network of unknown size. With DCP, we leverage the symmetry inherent in our network topology and use randomization to design a more practical algorithm.

References

1. Fraigniaud, P., Pelc. A., Peleg, D., Perennes, S.: Assigning labels in unknown anonymous networks. In: Proceedings of the 19th Annual Symposium on Distributed Computing, pp. 101–111. ACM (2000).
2. Walraed-Sullivan, M., Niranjan Mysore, R., Tewari, M., Zhang, Y., Marzullo, K., Vahdat, A.: Alias: Scalable, decentralized label assignment for data centers. In preparation for submission (2011).
3. Henzinger, M. R., King, V.: Randomized fully dynamic graph algorithms with polylogarithmic time per operation. J. ACM. 46, 502–516 (1999).