Brief Announcement: A Framework for Building Self-Stabilizing Overlay Networks

Andrew Berns
Department of Computer Science
The University of Iowa
Iowa City, Iowa 52242
adberns@cs.uiowa.edu

Sukumar Ghosh[*]
Department of Computer Science
The University of Iowa
Iowa City, Iowa 52242
ghosh@cs.uiowa.edu

Sriram V. Pemmaraju[†]
Department of Computer Science
The University of Iowa
Iowa City, Iowa 52242
sriram@cs.uiowa.edu

ABSTRACT

We describe a simple framework, called the *transitive closure framework* (TCF), for the self-stabilizing construction of *any* overlay network. The TCF is easy to reason about and algorithms derived from it stabilize within $O(\log n)$ more rounds than the optimal. As evidence of the power of this framework, we derive from the TCF a simple, self-stabilizing protocol for constructing Skip+ graphs in $O(\log n)$ rounds.

Categories and Subject Descriptors

C.2.4 [**Computer-Communication Networks**]: Distributed Systems

General Terms

Algorithms

Keywords

Overlay networks, self-stabilization

1. INTRODUCTION

Overlay networks are networks induced by logical or virtual links constructed over one or more underlying physical links. Many peer-to-peer systems (e.g., Chord and Tapestry) are overlay networks built atop the Internet. An ideal overlay network implementation should provide efficient routing, maintain small routing tables, and should quickly adapt to nodes joining, leaving, or failing. In order to achieve scalability, designers of overlay networks aim to obtain polylogarithmic or better bounds on the diameter (for efficient routing) and degree (for small routing tables). More recently, in an attempt to make overlay networks fault-tolerant, there has been research on trying to design self-stabilizing overlay networks (e.g., double-headed radix trees [1] and Skip+ graphs [2]).

We describe a simple framework for the self-stabilizing construction of *any* overlay network. This framework consists of three components: (i) a *detection* component, which nodes use to determine if there exists a fault in their 2-neighborhood, (ii) a *transitive closure* component, which nodes use to gather information about other nodes in the network and (iii) a *repair* component which nodes use to reset their neighborhoods. As evidence of the power of this framework, we describe a simple, self-stabilizing protocol for constructing a Skip+ graph in $O(\log n)$ rounds. In addition to being more efficient than the protocol of Jacob et al. [2], our protocol is simpler and far easier to reason about.

Algorithms for building overlay networks use a model of distributed computation that is quite different from traditional models such as $\mathcal{LOCAL}$ or $\mathcal{CONGEST}$ [4]. Nodes intentionally and repeatedly create or delete links to other nodes as they discover the existence or absence of other nodes and communication takes place on the evolving network. Since algorithms for building overlay networks use a non-traditional model of distributed computation, the question of how to measure the performance of such algorithms needs to be considered anew. Some measures that have been proposed in related work [3] include (a) worst case number of rounds needed to terminate or stabilize and (b) the maximum increase in the degree of a node during algorithm execution. Ideally, from the point of view of scalability, both measures should be sublinear, preferably polylogarithmic, in the number of nodes currently in the network.

Our framework leads us to identify a natural parameter of overlay networks called the *detection diameter* and show that this is a lower bound on the worst case running time of any self-stabilizing algorithm for building overlay networks. Our framework yields algorithms whose stabilization time is small, i.e., only $O(\log n)$ more than the detection diameter. While this is good news, the current versions of our algorithms do not provide any guarantee on node degrees during stabilization and these may grow to become linear. Research on self-stabilizing overlay networks is still in its infancy and as far as we know there are no self-stabilizing algorithms for building non-trivial overlay networks that obtain sublinear bounds on both measures mentioned above. For example, it is easy to construct examples for which the Jacob et al. protocol [2] will cause the node degrees to grow from a small constant to $\Omega(n)$ before they stabilize to $O(\log n)$. Viewed in this context, our framework shows that efficient stabilization is easy if one ignores the degree-growth constraint. Our framework also serves to clearly frame the difficulties in building overlay networks efficiently.

[*]Supported by NSF Grant CNS 0956780.
[†]Supported by NSF Grant CCF 0915543.

2. THE TRANSITIVE CLOSURE FRAMEWORK

The *Transitive Closure Framework* (TCF) is shown in Program 1. The general idea of the TCF is that any node that detects a fault (via the DETECT predicate in Line 1) initiates the transitive closure process (Line 6). This process ripples through the network quickly and soon enough the network becomes completely connected (Line 2). At this point, each node v has all the information it needs to "repair" its neighborhood and reset it to the correct neighborhood (Line 3). In Program 1, the actions needed to "repair" a node's neighborhood are encapsulated in the subroutine call to REPAIR(). The DETECT predicate and the REPAIR() subroutine need to be instantiated for specific overlay networks.

Program 1 Transitive Closure Program

```
Program for process $u$
Variables: neighborhood $N(u)$, Boolean $detect_u$

do forever
1. $detect_u = \texttt{DETECT} \vee detect_u$
2. if $detect_u \wedge \forall v \in N(u) : (detect_v \wedge (\{N(v) \cup v\} = \{N(u) \cup u\}))$ then
3.     $N(u) = \texttt{REPAIR}(N(u) \cup u)$
4.     $detect_u = false$
5. else if $detect_u \vee (\bigvee_{v \in N(u)} detect_v)$ then
6.     $N(u) = N(u) \cup \{\bigcup_{v \in N(u)} N(v)\}$ //transitive closure
7.     $detect_u = true$
   fi
od
```

To reason about TCF we need some definitions. Let V be a set of nodes with distinct IDs given by the function $id : V \to \mathbb{Z}^+$; each node $v \in V$ may also have an associated random bit string given by the function $rs : V \to \{0,1\}^*$. Given the triple $\lambda = (V, id, rs)$, we assume that there is a unique overlay network $G_\lambda = (V, E_\lambda)$. G_λ may be directed and we will use $N_\lambda(v)$ to denote the neighborhood of v in G_λ. Sometimes it will be convenient to call G_λ the *ideal* network. Each node $v \in V$ maintains a set of neighbors and the value of this set at the end of a round $t \in \mathbb{Z}^+$ is denoted by $N^t(v)$. Let $E^t = \{(v,w) \mid w \in N^t(v)\}$ and let $G^t = (V, E^t)$ denote the directed graph induced by current sets of node neighborhoods. Note that these directed edges can be converted to undirected overlay edges in one step. A node $v \in V$ is said to be *faulty* after round t if $N^t(v) \neq N_\lambda(v)$. The system is said to be *faulty* after round t if some node in V is faulty after round t. As is standard, we assume that node IDs and random bit strings are incorruptible. We assume the locally shared model of communication (rather than the message-passing model) and in this model, each node $v \in V$ can read its own state (i.e., set of neighbors) and the state of all of its current neighbors. As a result, each node v knows its current *2-neighborhood* $N_2^t(v) := \cup_{u \in N^t(v)} N^t(u)$ and more importantly, v also knows the subgraph $G^t[N_2^t(v)]$ induced by its 2-neighborhood. Node v can check, via local computations, if this "local" view of G^t matches its view of how the nodes in $N_2^t(v)$ should be connected in the the ideal network. This check is denoted by the predicate DETECT in Program 1 and a node for which this check fails is called a *detector*.

3. OUR RESULTS

Algorithms derived from the TCF are initiated by detectors and therefore the efficiency of these algorithms depends, to some extent, on the distribution of detectors in the network. For certain networks (e.g., SKIP graphs) the presence of even a single detector is not guaranteed when the network is faulty - and this is in fact the primary motivation of Jacob et al. [2] for defining SKIP+ graphs. To formalize the notion of the distribution of detectors, we define the *detector diameter* $D(n)$ of any family $\mathcal{F}$ of overlay networks as follows. For a fixed $\lambda = (V, id, rs)$ let $G_\lambda = (V, E_\lambda)$ be an n-node member of $\mathcal{F}$. Let $G = (V, E)$ be a weakly connected network on V, representing a particular state of the system and let $D \subseteq V$ be the set of detectors in G. The *detector diameter of G with respect to G_λ*, denoted $D_{G_\lambda}(G)$, is the maximum hop distance in G between any node in V and a closest detector. This means that if the initial state of the system is network G, then some node v in G is $D_{G_\lambda}(G)$ hops from the closest detector and thus the TCF algorithm initiated by detectors requires $D_{G_\lambda}(G)$ rounds to reach v. The detector diameter $D(n)$ of $\mathcal{F}$ is the maximum of $D_{G_\lambda}(G)$ over all n-node members G_λ of $\mathcal{F}$ and all weakly connected networks G with node set $V(G_\lambda)$. It is worth noting that if random strings are indeed used to define the family $\mathcal{F}$ of overlay networks, then $D(n)$ is a random variable. Our main result is this.

THEOREM 3.1. *Any self-stabilizing algorithm for building overlay networks in $\mathcal{F}$ takes at least $D(n)$ rounds stabilize. The TCF yields an algorithm whose stabilization time if $O(D(n) + \log n)$.*

As evidence of the power of this framework, we show the following result for SKIP+ graphs.

THEOREM 3.2. *The detector diameter for the family of* SKIP+ *graphs is $O(\log n)$. Thus the TCF yields an algorithm for building* SKIP+ *graphs in $O(\log n)$ time.*

In a sense, TCF is a theoretical exercise because it ignores the node-degree constraint. However, it clearly points out that one possible approach for achieving scalable self-stabilizing overlay network algorithms is to seek memory-constrained transitive-closure-like algorithms.

4. REFERENCES

[1] J. Aspnes and Y. Wu. O($\log n$)-time overlay network construction from graphs with out-degree 1. In *OPODIS*, pages 286–300, 2007.

[2] R. Jacob, A. Richa, C. Scheideler, S. Schmid, and H. Täubig. A distributed polylogarithmic time algorithm for self-stabilizing skip graphs. In *PODC '09: Proceedings of the 28th ACM symposium on Principles of distributed computing*, pages 131–140, New York, NY, USA, 2009. ACM.

[3] M. Onus, A. W. Richa, and C. Scheideler. Linearization: Locally self-stabilizing sorting in graphs. In *ALENEX*. SIAM, 2007.

[4] D. Peleg. *Distributed computing: a locality-sensitive approach.* Society for Industrial and Applied Mathematics, 2000.