Computer Science and Artificial Intelligence Laboratory

Technical Report

MIT-CSAIL-TR-2003-009
AIM-2003-017

August 11, 2003

Near-Optimal Distributed Failure Circumscription

Jacob Beal

massachusetts institute of technology, cambridge, ma 02139 usa — www.csail.mit.edu

Abstract

Small failures should only disrupt a small part of a network. One way to do this is by marking the surrounding area as untrustworthy — *circumscribing* the failure. This can be done with a distributed algorithm using hierarchical clustering and neighbor relations, and the resulting circumscription is near-optimal for convex failures.

1 Introduction

Given a failure in a network, I want to be able to determine just how bad it is. One way of determining this is to *circumscribe* the failure region: mark a connected subset of nodes in the network graph such that the failure is entirely surrounded.

This is an important problem because it allows a distributed algorithm to contain the disruptions due to failure of a region of the network. Many services may be unaffected by a small failure, and should be able to continue running as before, while those few affected enter a recovery state. A large failure, on the other hand, may reasonably force the entire network into recovery mode.

In this paper, I introduce a mechanism for creating near-optimal circumscriptions based on neighbor relations in a hierarchical clustering, as well as an distributed algorithm implementing the hierarchical method.

2 Network Model

The network is an undirected graph where nodes are machines and edges are links between machines. Although applicable to any network, the ideas presented herein were designed in an amorphous computing model.[1] The particular model used features a high-diameter network embedded in two-dimensional space where nearby nodes are connected and distant nodes are not (e.g. a large ad-hoc wireless network). The method presented is therefor best tuned for that environment.

In this paper, however, terminology about distance, connectedness, or other "spatial" properties will refer only to the graph topology, and not to any spatial embedding associated with the graph.

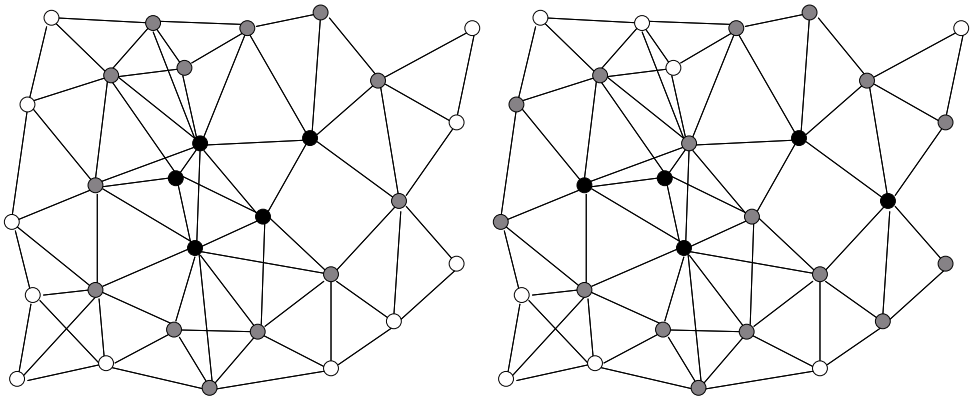
No geometric information, coordinates, or time synchronization is provided to machines in the network. Machines are, however, assumed to be partially synchronous: they have clocks with a relative drift of $|r| \leq \epsilon$, meaning that a fast clock can be at most $L = \frac{1+\epsilon}{1-\epsilon}$ times faster than a slow clock (this value is the *timing uncertainty*). This assumption allows use of the partially-synchronous perfect failure detector from [9], which detects a failed neighbor in constant time.

Finally, I assume a perfect communication model: messages on a link are delivered in order and without error within a bounded time. This assumption is stronger than necessary (what's really important is that the failure detector to produce correct answers), but simplifies analysis greatly at the cost of little generality.

3 Definitions

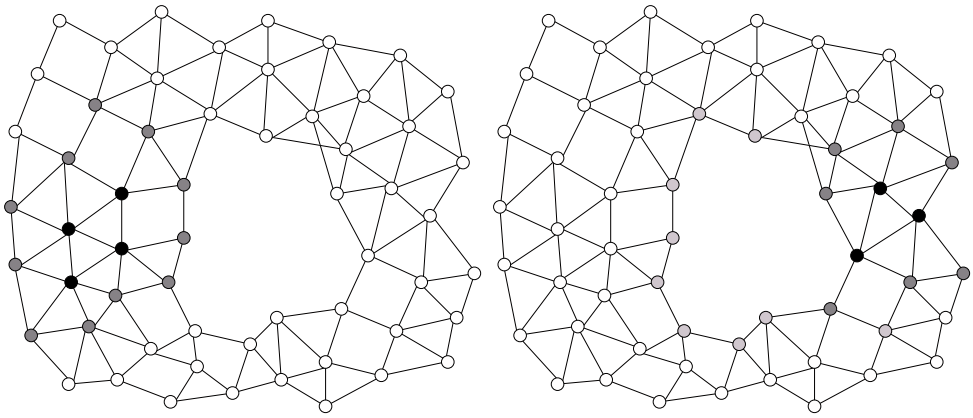
The failures considered here are **stopping failures**, in which a machine simply ceases to function. A **connected failure** is a connected set of machines which suffer simultaneous stopping failures. Machines neighboring this region on the graph detect the failure via a perfect failure detector: this set of machines is the **border** of the failure. The borders of several failures may intersect: in this case, the identities of the failures are necessarily blurred together. This is an **almost-connected failure** — a collection of connected failures such that the union of the failures and their borders forms a connected set. All of these failures are lumped together under the name **region failure**.

A connected set of non-failing machines which contains the border **circumscribes** a failure. If the border is connected, then it circumscribes the failure. If the border is disconnected, then more machines must be added in order to connect its components and form a circumscription — imagine a detour around a broken bridge. This detour might arbitrarily long, if the failure occurs at an important choke-point in the network. An **optimal circumscription** is a circumscription such that no other circumscription has a lower diameter. Note that if a failure partitions the network, then no circumscription exists: this is unsurprising,



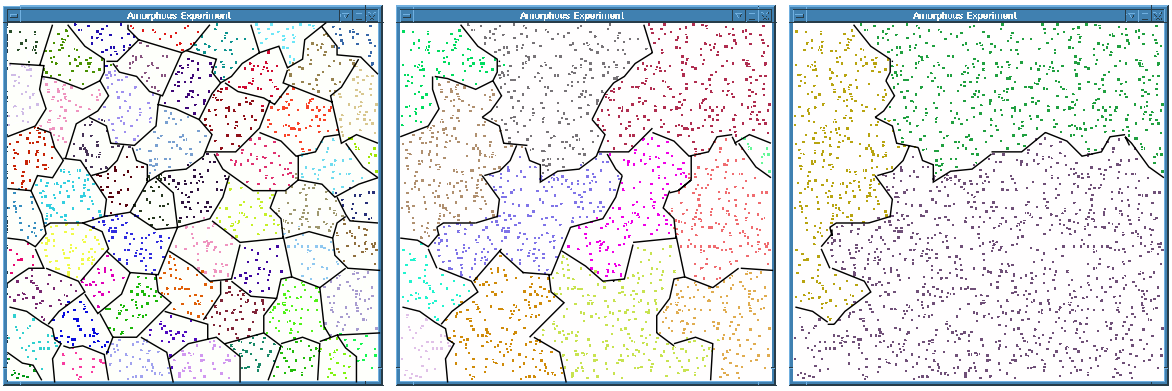
(a) Connected Failure (b) Almost-Connected Failure

Figure 1: There are two types of region failures: connected and almost-connected. Black nodes are failing machines and grey nodes are machines in the border of a failure. The left figure shows a region failure — a connected set of failing machines. The right figure shows an almost connected failure: connected failures with intersecting borders.



(a) Connected Border (b) Disconnected Border

Figure 2: Circumscribing a failure (black nodes) can require an arbitrarily large set when the border is not connected. If the border is connected (left) then the border (grey nodes) is itself a circumscription. If the border is not connected (right), then the graph topology may have changed enough to force the addition of many other nodes (light grey) to form a circumscription.



(a) Level 1 (b) Level 2 (c) Level 3

Figure 3: An example of an appropriate hierarchy: this shows the middle three levels of a five-level hierarchy produced by the PNHierarchy algorithm [3] on 2000 particles. Each cluster is a different color, with thick black lines showing the approximate boundaries.

since circumscription aids consistency and availability, thereby degrading partition tolerance, as per Brewer's conjecture.[4][5]

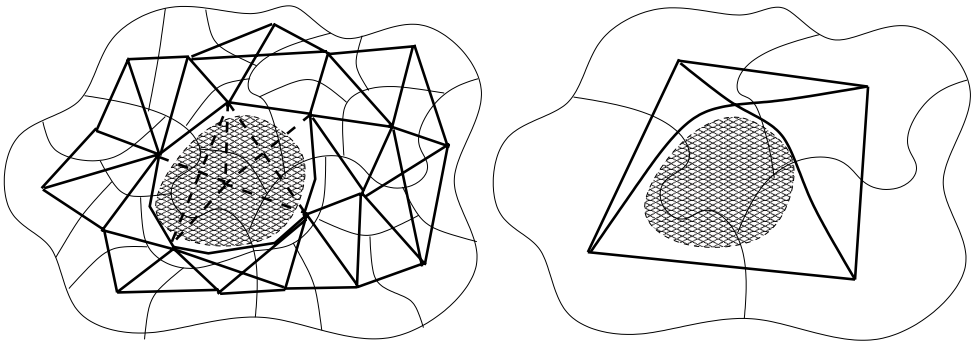
The goal, then, is to find a circumscription no more than a constant factor worse than optimal for region failures.

4 Circumscription Via Hierarchy

One more component is needed, an appropriate hierarchical clustering with neighbor relations. A hierarchical clustering organizes the network into a tree topology: walking up the tree proceeds from a leaf cluster containing a single machine through exponentially larger clusters to the root cluster, which contains every machine in the network. To be an appropriate hierarchical clustering for my purposes, there are four additional requirements:

- All root-to-leaf paths must be the same distance — i.e. the clustering can be organized into "levels" with the leaves at level zero and the root at level n.
- The hierarchy has $O(\log diam)$ levels (this will usually be guaranteed by the exponentially larger clusters)
- The distance between any two machines in an ith level cluster is bounded by some maximum distance d_i which scales exponentially (i.e. $d_i/d_{i-1} = b$ for every level). The at the top level of the network (the root of the tree) d_i must be at least equal to the diameter and no more than a constant factor greater (e.g. if d_i is powers of two, the root should be bounded by the next power of two from the diameter).
- Every cluster maintains a set of **neighbors** — same-level clusters nearby in the network, regardless of location in the hierarchy. Two level i clusters are required to be neighbors if any two members are within $3d_i$ hops of one another. The subset of neighbors within d_i are **tight neighbors**. The **neighborhood** of a cluster is the union of the cluster and its neighbors.

Given a hierarchy of this type, there is a surprisingly simple method of circumscribing a region failure. The neighbor relationships offer approximations of the network topology at progressively rougher levels of refinement — at a high enough level, the description is rough enough that the failure hardly changes it. This turns out to be a good enough approximation of the underlying network to generate a near-optimal circumscription.



(a) Lost Connections (b) No Lost Connections

Figure 4: Circumscription can be found by testing for invalid neighbor relationships. The left illustration shows a network divided into small regions with neighbor relations (thick lines). A failure (shaded area) breaks some neighbor relationships (dotted lines). At a higher level of the hierarchy, however, the neighbor relationships still hold (right), so circumscription can be guaranteed.

The procedure is simple: consider the "border clusters" — clusters intersecting the failure or its border. If every border cluster in layer i is still connected to all of its neighbors (or else can prove that all neighbors disconnected from it are definitely dead), then the union of the layer i border clusters and their neighbors circumscribes the failure. If we select the lowest layer for which this is true, then it is within a constant factor of optimal for convex failures.

I will now proceed to prove these claims:

First, some variables to work with:

- F is a region failure that does not partition the network.
- B_F is the border of failure F
- C_{Bi} is a set of border clusters — level i clusters intersecting F or the border of F. If a cluster C is split by the failure such that it exceeds the maximum radius, then it is analyzed as a set of clusters C' with the same name and neighbor relationships.
- $N(C)$ is the set of neighbor clusters of cluster C

The final preliminary is one more definition, **provably dead**, which asserts when it is safe to consider a cluster completely destroyed:

Definition 1: Provably Dead *Following a failure F, a set of clusters D is* **provably dead** *if four conditions hold:*

1. *Every cluster in D is a tight neighbor of every other cluster in D*
2. *Following the failure, there is some connected component which intersects every tight neighbor of D which is not a member of D.*
3. *No tight neighbor of D is still a neighbor following the failure.*

Essentially, these conditions say that a provably dead region is one where every tight neighbor can verify with every other tight neighbor that no connection with the region exists any more. The first condition ensures that the diameter of the neighbor-graph is small; the second requires that the tight neighbors be able to communicate, and the third states that the provably dead region must be in fact dead.

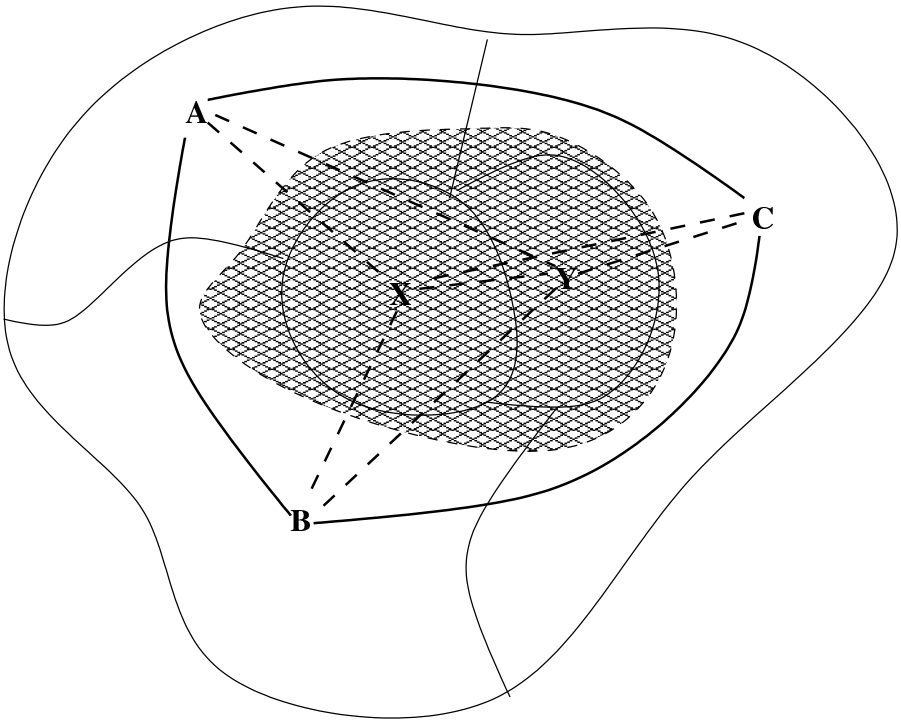


Figure 5: Nodes X and Y, in the shaded failure region, are provably dead because they are tight neighbors of each other and all of their other tight neighbors (A, B, and C) can communicate and confirm that nobody can talk to X and Y

Theorem 2: *Following a failure F, let i be a level of hierarchy in which, for every member of C_{Bi}, all of its pre-failure tight neighbors are either still neighbors or else provably dead. Then the union of neighborhoods of border clusters, $C_{Bi} \cup N(C_{Bi})$, contains a connected component which circumscribes the failure F.*
Proof: Assume this is false. Then every border cluster is neighbor to all of its pre-failure tight neighbors that are not provably dead, but the union of neighborhoods does not have a connected component containing the border of F.

First, note that a cluster is not necessarily connected (in practice, they often will be, but the definition does not require it, and some clustering methods will produce disconnected clusters). The neighborhood of a cluster, $A \cup N(A)$, however, has a connected component containing the cluster A. This is because every cluster within d_i of a node in A is a neighbor of A, and the maximum distance between nodes in A is d_i — thus, there is a path between any two nodes in A along which all nodes are, by definition, in the neighborhood of A.

Similarly, if clusters A and B are still neighbors, then the union of their neighborhoods, $A \cup N(A) \cup B \cup N(B)$ contains a connected component containing A and B. Each cluster is in a connected component of its neighborhood, and A and B are neighbors, so there must be a path from A to B which is no more than $3d_i$ in distance. Every node on this path is a member of A's neighborhood, so A and B must be in the same connected component of the union of their neighborhoods.

Thus, to fail, C_{Bi} must be split into at least two connected components X and Y such that no cluster in X is a tight neighbor of any cluster in Y. However, before the failure, $X \cup Y \cup F$ had a connected component containing the failure and its border ($F \cup B_F$). Since a cluster in X cannot be a tight neighbor of a cluster in Y, there must have been some set of clusters G completely contained in F (i.e. not part of C_{Bi}) which connected X and Y. Thus we have a contradiction since either G is empty and X and Y must contain tight neighbors, or else both X and Y contain some cluster which had a tight neighbor in G that is either provably dead (requiring clusters intersecting X and Y to be connected and therefor tight neighbors) or else missing. □

A complementary relationship also holds, which is vital for producing a distributed algorithm implementing circumscription: if any component is missing a neighbor, then all components are missing neighbors.

Corollary 3: *Following a failure F, let i be a level where some member of C_{Bi} is no longer related to a pre-failure tight neighbor which is not provably dead. Then every cluster in C_{Bi} is related by a chain of neighbor relations to a cluster missing a non-provably dead neighbor.*
Proof: For almost all cases, a stronger condition holds, that every border node is part of a connected component of C_{Bi} which intersects a cluster with a missing neighbor. As above, if a component of C_{Bi} which intersects the boundary does not intersect every cluster in C_{Bi}, then there must be a missing neighbor (by pre-failure connectedness).

That leaves only the case of divided clusters, where some connected component X intersects every cluster in C_{Bi} while another component Y intersects only some, but X and Y are not connected because all of the components in Y are split by the failure from components in X.

In this case, however, some cluster in Y must still have a neighbor relation with some tight neighbor Z that is not in C_{Bi} (otherwise the network is partitioned) and either X is missing its neighbor connection to Z or else X and Y are related by a neighbor relation through Z. □

Now that conditions for circumscription have been established, I show that there is a level at which they are guaranteed to hold:

Theorem 4: *Following a failure F, let $d(B_F)$ be the maximum distance between any two machines in the border B_F following the failure, and $d'(F \cup B_F)$ be the maximum distance between any two failing or border machines, before the failure. Then F is circumscribed by $C_{Bi} \cup N(C_{Bi})$ for every level i where $d_i \geq \max(d(B_F), d'(F \cup B_F))$*
Proof: If $d_i \geq d'(F \cup B_F)$ then all clusters in the border clusters C_{Bi} and all clusters completely contained in the failure F are tight neighbors before the failure. Every machine in every border cluster is within d_i of some machine in the border B_F, a relation which holds following the failure as well. Then by the assumption that $d_i \geq d(B_F)$, the distance between any two border clusters following the failure is at most $3d_i$, implying they must still be neighbors. Thus, $C_{Bi} \cup N(C_{Bi})$ contains a connected subset which contains every point in the border B_F, circumscribing F. □

Corollary 5: *Under the above conditions, any cluster contained entirely within F is provably dead following the failure.*
Proof: Let D be the set of clusters completely contained in F:

Condition 1 (D a complete graph of tight neighbors): Since $d_i \geq d'(F \cup B_F)$, every cluster in F is a tight neighbor of every other cluster in F.

Condition 2 (Tight neighbors in a connected component): For any machine in F, every tight neighbor of its cluster is within d_i of the border B_F. The border is contained in $C_{Bi} \cup N(C_{Bi})$, which, by circumscription, must contain a connected component which intersects every group within d_i of the border.

Condition 3 (Deadness): Every member of D is completely in F, thus completely dead.

Thus, any cluster contained entirely in F is provably dead following the failure. □

Corollary 6: *Under the above conditions, for any member of C_{Bi}, every pre-failure tight neighbor is either still a neighbor or else provably dead.*
Proof: Consider a border cluster $C \in C_{Bi}$ and a cluster N which is a tight neighbor before the failure.

Assume this is false: then there is some pair C and N for which N is not provably dead and not a neighbor of C following the failure. If N is completely contained in F, then it is provably dead. Therefor there must be some element of N outside of F. Before the failure, there was a path from C to N less than d_i in distance. This path must intersect the failure F to be changed by it, and therefor there must be elements of both clusters N and C within d_i of the border B_F both before and after the failure. By assumption, the maximum distance between elements in B_F is at most d_i following the failure, so there must be a path between N and C of at most length $3d_i$ following the failure, which would imply that N and C are still neighbors. □

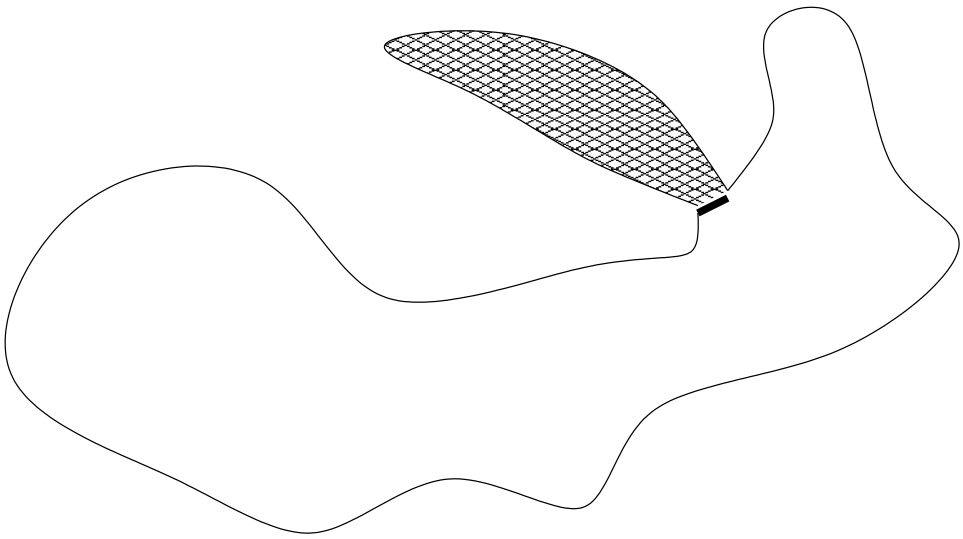


Figure 6: An example of a non-convex failure: a blobby network with a long thin appendage has the entire appendage fail. The optimal circumscription (thick black line) covers only the border between the failing appendage and the main body. The circumscription which will actually be found, however, will be much larger than optimal because clusters inside the failure must be provably dead.

Finally, the hierarchy method of circumscription is competitive with the optimum circumscription for a large class of failures. These **convex failures** are failures for which the diameter of the border is at least as large as the diameter of the failure — to be precise, $d(B_F) \geq d'(F \cup B_F)$.

This is because clusters completely contained in the failure need to be provably dead, even if the diameter of the optimal circumscription happens to be tiny. Consider, for example, the failure of a large region connected to the graph by a single small choke-point: the optimal circumscription is the tiny neck, and proving the region past the neck is dead requires long-distance neighbor relations.

Theorem 7: *For a convex failure F, let i be the minimum level for which $d_i \geq d(B_F) \geq d((F \cup B_F)$. The diameter of the circumscription component of $C_{Bi} \cup N(C_{Bi})$ is $11b$-competitive with the diameter of an optimal circumscription (e.g. 22-competitive if d_i is powers of 2).*

Proof: Every element in C_{Bi} is within d_i of B_F, so every element in $N(C_{Bi})$ is within $5d_i$ of B_F (Neighbors are within $3d_i$ and have elements at most d_i further away). The distance between elements in the border is at most $d(B_F) \leq d_i$, so the whole diameter of the circumscription component of $C_{Bi} \cup N(C_{Bi})$ is at most $11d_i$. Since, by assumption, d_i is the smallest d_j greater than $d(B_F)$ and since d_j values scale exponentially by ratio b, $d_i \leq b \cdot d(B_F)$ and therefor the diameter is bounded by $11b \cdot d(B_F)$.

An optimal circumscription has diameter of at least $d(B_F)$, because the connected subset must contain the entire border.

Thus hierarchy circumscription is $11b$-competitive with optimal for convex failures. □

This isn't a very nice bound, but it is within a constant of optimal, and could be improved markedly by post-processing of the circumscription set. It is, however, a fair estimate of how far information must travel to guarantee success.

5 Distributed Circumscription Algorithm

The hierarchy method of circumscription lends itself easily to a distributed implementation. The key property is Corollary 3, which mean that no cluster can prematurely decide it has found the circumscription.

This algorithm assumes three pieces of data are stored at every machine: a list of clusters the machine

belongs to, a list of neighbors and tight neighbors for each cluster, and a list of tight neighbors for each tight neighbor of each cluster.

Upon failure, every machine in the border detects the failure and begins hunting (bottom up) for the minimum level that will produce a circumscription. At level i, the border machines transmit a wakeup to everything within d_i hops, activating all machines which are members of C_{Bi} (the clusters which intersect the failure but not the border test for and detect their failed members on this wakeup call).

Every machine in C_{Bi} tests for missing neighbors. News of missing neighbors is propagated by gossip through the neighbor graph, and machines which can be eliminated as provably dead are marked as such. Thus, by Theorem 2 and Corollary 3, eventually every machine agrees whether there are missing neighbors which are not provably dead, even if the machines are in disconnected components. If there are missing neighbors, the border machines increment level i and try again. If there are not, then the appropriate level is found and all machines in $C_{Bi} \cup N(C_{Bi})$ mark themselves as members of the circumscription.

By Theorem 4, this algorithm will eventually find a level agreed upon by all machines for which $C_{Bi} \cup N(C_{Bi})$ is a valid circumscription, and by Theorem 7 if the failure is convex this will be competitive with an optimal circumscription for that level. Time to converge is dependent on the time to detect missing neighbors and communicate this information via gossip, which takes no more than the final diameter at each level, for a logarithmic number of levels, giving $O(n \log n)$ where n is $\max(d(B_F), d'(F \cup B_F))$.

6 Contributions

I have described a simple distributed algorithm which finds a circumscription for any stopping failure that does not partition a network. This circumscription is near optimal for convex failures.

Circumscribing a failure is a powerful tool for exception handling in distributed algorithms, as it bounds the region in which exception handling needs to take place. Optimal circumscription marks only a region directly proportional to the severity of the failure, so small failures allow most of the network to continue running the algorithm without any interruption. One obvious application is in distributed atomic storage, such as the Persistent Nodes in [2].

In a larger scope, it is interesting to consider what relation there may be between the space-limited failures dealt with by circumscription and the time-limited failures considered by Khazan.[6] I conjecture that there may be a general property unifying the two, much like the dynamic finger and working set properties for data access. Since distance in hops is equivalent to time lag when links are homogeneous, it is tempting to think there is a relativistic principle bounding the necessary effects of failures in a large network.

If this is, indeed, the case, then it should be possible to formalize lower bounds on the space-time interval of damage incurred by a failure, allowing explicit analysis of design tradeoffs between data availability and consistency. Moreover, I expect that, as indicated by this system, it will be possible to asymptotically approach such a lower bound.

References

[1] H. Abelson, D. Allen, D. Coore, C. Hanson, G. Homsy, T. Knight, R. Nagpal, E. Rauch, G. Sussman and R. Weiss. Amorphous Computing. AI Memo 1665, August 1999.

[2] Jacob Beal. Persistent Nodes for Reliable Memory in Geographically Local Networks. MIT AI Memo 2003-011.

[3] Jacob Beal. A Robust Amorphous Hierarchy from Persistent Nodes. MIT AI Memo 2003-012.

[4] Eric Brewer. Towards Robust Distributed Systems. (Invited Talk) Principles of Distributed Computing, Portland, Oregon, July 2000.

[5] Seth Gilbert and Nancy Lynch. Brewer's Conjecture and the Feasibility of Consistent, Available, Partition-Tolerant Web Services. Sigact News, 33(2), June 2002.

[6] Roger Khazan. Formal Design and Analysis of a New Virtually Synchronous Group Communication Service for Wide Area Networks. PhD Thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA, May 2002.

[7] J. Kleinberg. Detecting a Network Failure. Proc. 41st IEEE Symposium on Foundations of Computer Science, 2000.

[8] L. Kleinrock and J. Silvester. Optimum transmission radii for packet radio networks or why six is a magic number. *Proc Natl. Telecomm. Conf.* pp 4.3.1-4.3.5, 1978

[9] Nancy Lynch. Distributed Algorithms. Morgan Kaufmann Publishers, Inc, San Francisco, California, 1996. Chapter 8, pages 199-234.