J. Nig. Ass. Math. Phys
Vol. 6 (2002)

ON DISTANCE BETWEEN NODES IN PARTITIONED NETWORKS

Nwanze, E. D. and Abada, G. O
Department of Computer Science, University of Benin, Benin City

ABSTRACT

In this paper we describe network partition from the viewpoint of recursive trees which result as computer networks are exposed to series of degradations. Our attempt here is to establish that the mean and variance of the distance between nodes over the set (n-1)! recursive trees also occurs in partitioned networks. The operational approach adopted is a model-theoretic analogue of the notion of random recursive trees as they apply to computations that occur forever.

1. INTRODUCTION

The proliferation of distributed computing systems has given rise to widespread investigations into the problem of how best to model behaviours within such systems. In the development of models for these systems, innovative progress has been made by a number of authors [1-14]. However, the failure of applications for such systems has imposed high demands on reliability, availability and performance. Here we discuss network partition as a meta-theoretic analogue of the resulting random tree that is recursive and finite within an active network distribution.

Theoretically, a *partitioned network* may be thought of as a tree in which the edges have a common origin (the root node) and every node corresponds to a state or an event in the network. The root node is a possible source of a message within the network. *Partition relation* is a term in set theory used to describe patterns that result in partitioning. Partition relations in networks are well explained in the literature [6-9]. However, this paper focuses on the resulting tree structure from partitioned networks. A tree is a connected graph that has no cycles. The existence of a well-ordering makes it possible to define modified partitioned relations in a network (N) by selecting from $(N)^r$ a subset $[N]^r$ consisting of r consecutive nodes of N.

Suppose that N is of well-order type under the relation $<\cdot$ which we may write as

$$N=\{n_0,..,\hat{n}_x\}<\cdot \quad \cdots \quad \cdots \quad \cdots \quad \cdots \quad \cdots \quad \cdots \cdots \cdots \cdots \cdots \quad (1)$$

The notation $\hat{n}$ means that n_x is excluded. The notation $[N]$ Denotes the set consisting of the nodes of the resulting tree.

A tree $T_n \equiv [N]^r$ with n nodes labelled 1, 2, ..., n is a *recursive tree* if n=1 or if $n \geq 2$ and T_n is obtained by joining the n^{th} node to one node of some recursive tree T_{n-1}. According to Moon [7] there are (n-1)! recursive trees T_n and a tree with n labelled nodes is recursive if and only if the labels of the nodes in the path from the first node to the k^{th} node of the tree form an increasing sub-sequence of {1, 2, ..., k} for k=1, 2, ..., n. In a tree the distance between nodes *i* and j is considered as the number D_{ij} of edges in the path P joining *i* and *j*. A *path* is a sequence of states which represent the nodes in the tree

The distance between nodes in trees is well-discussed [6-9,14]; however, our interest is its application in partitioned networks. And as such, in the next section, we discuss the theoretical background to the issue of computing the distance between nodes in partitioned networks. However, a major contribution to this paper associates this distance between these nodes with the complexity of the network and tries to configure a computational model that will address the problems of network failures. A proof of our major assertion in section 3 establishes the connection. Further, the discussions that follow in the other sections expose the nature of applications that suffer from problems associated with partitioned networks. Accordingly, the average distance between two nodes in a random recursive tree is approximately *2log n* where n is large. With respect to the set of n^{n-2} ordinary trees with n labelled nodes the corresponding result was found to be $(\pi n)^{1/2}$ [8], thereby establishing the point that recursive trees are shorter and bushier than ordinary trees on the average.

2. THEORETICAL BACKGROUND

The main thrust of this paper is to establish that the mean and variance of D_{ij} over the set (n-1)! recursive trees T_n also occurs in a partitioned networks. The proof of assertions in the next section and their implications in other sections will corroborate our basic discussion on recursive trees in this section.

Let P(i,j;d) denote the probability that $D_{ij} = d$ in a random recursive tree T_n. Assume that $1 \leq i < j \leq n$ and $0 \leq d \leq n-1$ then it follows that

$$P(i,j;d) = 0 \quad \text{if } d \geq n$$

or $$P(i,j;d) = 1 \quad \text{if } d \leq n \text{ or } \max(i, j) > n \quad \ldots\ldots \quad \ldots\ldots \quad (2)$$

Again, if i<j in a subtree T_{j-1} determined by the first j-1 nodes of a recursive tree T_n and suppose the j^{th} node of T_n is joined to node x of T_{j-1}, then the distance D_{ij} equals i plus the distance between node i and x in T_{j-1}. The node x could be any one of the j-1 nodes of the tree T_{j-1} and these possibilities are equally likely.

By implication, if $1 \leq i \leq n$ and $1 \leq d \leq n-1$, then we state as follows:

Lemma 1

$$P(i,j;d) = \frac{1}{j-1}\{P(1,i;d-1) + \cdots + P(2,i;d-1) + \cdots + P(j-1,i,d-1)!$$

Having established lemma (1), we try to determine the expected value of D_{ij}.

Let E(i,j) be the expected value of D_{ij} over the (n-1)! recursive tree T_n. If $1 \le 1, j \le n$ then

Theorem 1

$E(i,j) = h_i + h_{j-1} + 1/i$

$$where\ h_1 = 1\ and\ h_k = \frac{1}{2} + \frac{1}{2} + \cdots + \frac{1}{k}\ for\ k \ge 2.$$

As postulated in (6), we now provide the following corollaries which are consequences in this theory

Corollary 1

If $1 < j \le n$ then $E(i,j) = 1 + ½ + \ldots + 1/(j-1) = \log j + O(1)$

Corollary 2

If $1 \le i < j \le n$ then $E(i,j) = E(1,i) + E(i,j) + 2(1/i-1)$ ... ...

If $1 \le i < j \le n$ and $n > 1$ let A(i,n) denote the average distance between node i and all the other nodes of a random recursive tree T_n; then we can provide the following

Corollary 3

$$A(i,n) = \frac{n}{n-1}\{E(i,n+1) - 1\}$$

If $n \ge 2$ let B(n) denote the average distance between paths of distinct nodes in a random recursive network tree T_n; then the following declarations can be made

Corollary 4

$B(n) = 2 \log n + O(1)$

The variance of D_{ij} is established with the theorem that follows:

If $1 \le i < j \le n$ then

Theorem 2

$V(i,j) = E(i,j) - 3H_i - H_{j-1} + 2 - 4$

From theorem 1 and following the proof of theorem (2) in Moon [8] we can establish that

$E(t-1,t) = 2E(1-t) + (t-1)^{-1} - 2$ for $t \ge 2$. Hence

$$\sum_{t=2}^{i} E(t-1,t)/t = E^2(1,i+1) - H_i - 2h_i - \frac{1}{i} \quad \ldots\ldots\ldots\ldots\ldots\ldots (3)$$

The following corollary is a direct consequence of theorem 2 establishing variance:

$$V(1,j) = \left\{1 + \frac{1}{2} + ..+ 1/(j-1)\right\} - \left\{1 + \frac{1}{2^2} + ..+ 1/(j-1)^2\right\} = Log\ j + 0(1) \qquad (4)$$

3.0 COMPUTATIONAL MODEL

Suppose the recursive network tree T_n is denoted by a configuration S=(N, L) where N is the set of processors or nodes and L is the set of physical communication channels. And suppose also that each node and edge is associated with an "active" or "failed" state denoted by a boolean variable. These individual states are combined to form a diagnostic state vector, which epitomises the active state of the entire configuration graph denoted by $S=(nb_1, nb_2, \ldots nb_N, lb_{1,2}, lb_{2,3}, \ldots lb_{N-1,N})$.

From works of Machle, et al in [5] and based on the perspective discussed above, diagnosability of nodes and links can be defined as follows:

A node n_j is diagnosable from n_j if the there exist a path in the configuration graph $P_{ij} = < n_i l_{i,k}, n_k, l_{k,r}, \ldots n_j, l_{m,j} >$ where n_i, n_k, etc. are operation nodes.

We assume that a communication link $l_{j,k}$ is diagnosable from node n_i, if nodes n_j and n_k are diagnosable from n_i. Each node can diagnose up to k nodes in some protocol and broadcast their status to the other nodes in the configuration. There is the additional requirement that the communication subsystem does not fail, in that diagnostic messages can be sent and acknowledged with fixed time intervals if the node being checked is functional. Since each node has an independent communication processor, node failure can be distinguished from link failure. In most cases, confirmed node failure results in its entire outgoing links being faulty. For the purpose of this paper we shall concentrate on node failure which partitions networks and constrains the network operating system not to transmit tasks to failed nodes or over non-operational links.

A distinctive provision of this paper is to model a computation that is defined and constrained by distance and time. A transition function F(d,k) is suggested at every node that has the ability to diagnose each neighourhood node for failures in its reconfiguration. The transition function would bring about reduction in the difficulties that are associated with partial failures. Also reconfiguration information can be sent using this function which communicates suggested repair protocol by displaying required modifications to the faulty processes. It should be noted that we are in a situation where it is unlikely that any proposed solution to partition problem will be unique. And the best choice may well not only depend upon the particular features of the network, but also on functional constraints not fully represented in the network system. Let us continue with the computational model.

Given the recursive tree T_n, a node is an *element* of the tree. A path is well-ordered subset P of the tree such that if $j \in P$ and $i<j$ then $i \in P$. In other words, a path is a partial order or computation on the elements of the tree. A path P is extensible if there is a node j in the tree $P = \{i: i<j\}$ if it represents a

total order on the network. A path has a last node j if P={i: i< j=n). Note that a path has last node if and only if it is defined W_α for a successor ordinal α (a variant of the well-order type introduced in section 1) [5].

Given $T=T_n$, a subtree of T is a subset $T' \subset T$ with the induced order such that $\forall x \in T$, $\forall y \in T(y<x)$, $y \in T'$ [Kunen]. The maximal consistent configuration of a set of computations C denoted by MCC(C) is the maximal subtree of T such that for every node j in MCC(C), every configuration step s in C, and every consequence of the computation in j(s)=T or j(s)=F otherwise. If j cannot compute s, and none of its ancestor nodes (in the ordering <) has failed to compute any step in C, then j is a failure node. By implication j is partitioned. In particular, if j is last node of a path in a maximal consistent configuration of a Recursive tree, then any extension of j will fail. Node j+1 is a partitioned node. A path in MCC(C) whose extension (if any) are all failure nodes is called maximal path of MCC(C). It is clear that a given C computation is possibly in MCC(C) that is defined on the entire tree.

The following lemma ensures that a partitioned node cannot exist at a path that is defined to a limit ordinal. Put differently, a path does not exist across a network partition.

Lemma 2:

Let C be a set of computations, then every maximal path MCC(C) has a last element.

Proof:

Let P be a maximal path of MCC(C) for such a C. If P is not extensible, then the total transitions represented by P is the last element of P. So, we may assume P to be extensible. Let k be such that P={i:i<k}, and the $k \subseteq W_\alpha$. Suppose α is a limit ordinal. By definition, $k=\cup_{\beta<\alpha} k \mid W_\alpha$. Then k is consistent with C since for every $\beta<\alpha$, $k \mid W_\alpha$ is consist with C. This contradicts the assumption that P is a maximal path of MCC(C). Thus, α is a successive ordinal, and $k \mid W_{\alpha-1}$ is the last element of P.QED.

Corollary 5:

If $k \subseteq W_\alpha$ is a failure node, then α is either o or a successor ordinal.

Corollary 6:

Let C be a set of computations not containing a null value (not an empty set), then C is unsatisfiable if and only if every maximal path in MCC(C) extends to failure node. A direct consequence of this is that active nodes are partitioned from failed and faulty node.

The corollaries are easy consequences of the closure lemma. Corollary 5 indicates that a failure node is not part of an active partition and so cannot appear at the limit. Consequently, whenever a node failure occurs in a distributed system, it always cuts the network into partitions.

Corollary 6 serves a role in our work by trying to prove that active partitions cannot extend computations to partitions outside it, except when such partitions are recovered to active state.

4. NETWORK APPLICATIONS

4.1. PARTITIONED NETWORK

The most disruptive of communication failures are partition failures, which fragment the network into isolated sub-networks called partitions. The failure of applications envisaged for distributed networks imposes high demands on reliability, availability and performance as mentioned earlier. Unless partitioned networks are detected and recognised by all affected processors in a distributed system, independent and uncoordinated updates may be applied to different copies of data, thereby compromising the correctness of data.

Variations in the execution time of network transactions are solely with data dependencies. Depending on whether the complexity of the system increases, the data dependency will increase, resulting in an increased variance in the execution time of tasks at each node. To illustrate, let us consider a simple example of configuration of arity a and distance d. The total number of nodes in the tree is given by $(a^d-1)(a-1)$. In its classical case of simplicity, worst-case and average-case execution times can be characterized as $k(a^d-1)(a-1)$ and $kv(d)$, respectively.

In the above case, k is the average time to expand and evaluate a single node as represented in computing B(n) in section 1, and v(d) is the multiplier for the number of nodes examined in the average case. Assuming d=10, a=3, v(d)= 30, the ratio of the average case to the worst case is 3:29524 ($\cong$ 1:1000).

The execution time of the variance V(i,j) associated with the computation of the distance D_{ij} of the node configuration problem associated network systems can be reduced through abstractions and pruning of successive computations.

Suppose the nodes of a recursive tree T_n in a network are susceptible to partitioning and that if an unpartitioned node y is joined to a partitioned node, say, x where x<y, then the information will broadcast from x to y with probability p where $0<p<1$. If $1 \le i \le j \le n$ and $0 \le d \le j-1$, let q(i,j;d) denote the probability that the path from the 1st node to the j^{th} node of Tn contains the i^{th} node and Dij = d. It is easy to see that

$$q(i,j;d) = \frac{1}{j-1}\{q(i,j;d-1)+(i,i+1;d-1)+\cdots+q(i,j-1;d-1)\}$$

where $d \ge 1$. If the i^{th} node of T_n is partitioned and Q(i,j) denotes the probability that the j^{th} node is also partitioned from the i^{th} node, either directly or indirectly,

$$Q(i,j)=\sum_{d=1}^{j-1} P^d q(i,j;d)=\frac{p}{j-1}\{Q(i,j)+Q(i,i+1)+\cdots+Q(i,j-1)\} \quad \cdots\cdots\cdots\cdots\cdots\cdots (6)$$

then it follows that

It is easy to show that by implication

$$Q(i,j) = P\frac{r(j)}{r(i)}\cdot\frac{r(p+j-1)}{r(p+i)} \tag{7}$$

By definition Q(i,i,)=1. Thus if R(i,n) denotes the expected total number of nodes of T_n in the partitioned network from the i^{th} node, including the i^{th} node itself, then where $1 \le i \le n$, by (6) and (7)

$$R(i,n) = \frac{1}{r(p+1)}\cdot\frac{r(p+n)}{r(n)} \approx \frac{1}{r(p+1)}n^p \tag{8}$$

We note also that

$$R(1,n) = \sum_{j=1}^{n} Q(i,j)Q(i,n+1) = \frac{n}{p}Q(i,n+1) = \frac{r(i)}{r(p+1)}\cdot\frac{r(p+n)}{r(n)} \tag{9}$$

As $n \to \infty$

If the 1^{st} node of one of the n^{n-2} ordinary tree with n labelled node in a network is partitioned and the rest nodes also affected, then based on the assumption above and the basic proof shown in Moon [8], on infections and spreads of infection in recursive trees, this will eventually result in a total of approximately $(1-p)^{n-2}$ partitions on the average when n is large. This means that the partitioning of the root node of a computer network will completely partition the network since the node represents the server in a network in client-server architecture.

4.2 PARTITIONED OPERATIONS

Again, suppose a recursive tree T_n represents a distributed database system in a Network and the first node represents the producer. By storing copies of shared data on processors where they are frequently accessed, the need for expensive remote read accesses is decreased while performance and availability is improved. By storing copies of critical data on processors with independent failure modes, the probability that at least one copy of the data will be accessible increases. In theory, data replication or more harshly, partitioning of some nodes on a network makes it possible to provide arbitrarily high data unavailability. There is a concern about how reliable and secure transactions are in such networks with the objective to present each user with virtual unified system in which files may be accessed without concern. When communication fails between sites containing copies of the same logical data item, mutual consistency between copies becomes difficult to ensure.

Further, suppose each person in the network has to contribute his data to the immediate supplier. Suppose that each such person also has to pass on to his immediate supplier the fraction p of all franchise data received from those it supplies directly or indirectly; the producer, of course keeps all data that come to him

If $1 \leq i \leq n$, let N(i,n) denote the expected net data returns to the i^{th} person with respect to the data collected from the n-1 persons other than the producer. A slight modification of the argument in the proceeding paragraph can be used to show that

$$N(i,n) = \frac{1}{p^r(p+1)} \cdot \frac{r(p+n)}{r(n)} - \frac{1}{p}$$

and

$$N(i,n) = \frac{(1-p)}{p} \cdot \frac{r(i)}{r(p+1)} \cdot \frac{r(p+n)}{r(n)} \cdot \frac{1}{p}$$

where $1 < i \leq$

We note that N(n,n) = -1 for all values of p. If p = 1 then N(1,n) = n-1 and N(i,n) = -1 for $1 < i \leq n$; if $P \to 0$ then

$$N(i,n) = \frac{1}{i} + \frac{1}{i+1} + \cdots + \frac{1}{n-1}$$

for $1 \leq i < n$. Furthermore, if N(i,n) = 0 then

$1 \approx (1-p)^{1/p} n$ when $0 < p < 1$ by Stirlings formula

So if $1 \leq d \leq n-1$ and f(d, n) denotes the expected number of nodes in a recursive tree T_n such that $D_{ij} = d$ and then it is easy to see that

$$f(d,n) = \sum \frac{1}{a_1 a_2 \cdots a_d}$$

where the sum is over all sets of integers $\{a_1, a_2 \ldots a_d\}$ such that $0 < a_1 < \ldots < a_d < n$. It follows from then that if d = 0 then

$$f(d,n) \approx \frac{(\log n)^d}{d!}$$

CONCLUDING REMARKS

Consider for example, an airline reservation system implemented by a distributed database over a network that splits into two partitions when the communication network fails. If, at the time of failure, all the nodes have one seat remaining, for say Nigerian Airways flight 202, reservations could be made in both partitions. This would violate correctness. Who gets the last seat? Typically, the processors themselves cannot discern the cause or extent of a partition failure. At best, a processor may be able to identify the other processors in its partition, it will not be able to distinguish between the case to which those processors are simply isolated from it and the case in which those processors are down. In terms of operations, the transaction processing system that detects a fault will ensure that both updates are done, or that neither update is done, regardless of client or server machine crashes and regardless of other clients attempting to update the object. Transactions are useful primarily for ensuring reliability and consistency in the presence of concurrency and failures. A transaction either *runs to completion* or *commits* or a failure occurs and it *aborts*. Our modelled computation ensures that a partitioned node cannot exist along a path defined to a limit.

The primary claim of this paper is that in many common situations networks partitions impair a lot of on-line transactions and diminish data consistency as well as integrity. Restricting attention to the problem of mending partitions exposes the danger inherent in certain critical transactions that may be necessary. In this paper, therefore, we have provided a configuration model with stabilisation properties that ensures that any faulty node is detected. This also ensures availability of the network resources to the system. A possible reconfiguration system is prescribed using transition function that restores computations that would have failed, ensuring reliability. The maximal consistency theory espoused ensures stability, performance and consistency.

REFERENCES

[1] Bernstein, P. A and Goodman N. "Concurrency control in Distributed Database systems." ACM computing surveys 13 (2) (1981),185-221.

[2] Chung, J., Liu, J. and Lin, K. Scheduling Periodic Jobs that allow imprecise results. IEEE Transactions Computing 39, 9(1990), 1156-1174.

[3] Gulati, S., Iyangar, S. S. and Barhen, J. "The Pebble-Crunching Model for Fault-tolerant Load Balancing in Hypercube Ensembles". The Computer Journal, 33, 3(1990), 204-214.

[4] Kunen, K. Set Theory: An Introduction to Independent Proofs. In Studies in Logic and Foundation of Mathematics, North Holland, New York, (1980).

[5] Machle, E., Moritzen, K. and Wirl, K. "A Graph Model for Diagnosis and Reconfiguration and its Application to Fault-tolerant Multiprocessor system." IEEE Conference on Fault-tolerant Computing. (1986), 292-297.

[6] Meir, A and Moon, J.W. "The Distance Between Points in Random Trees." J. Comb The 8 (1970), 99-103

[7] Moon, J. W. "A Problem on Random Trees". J. Comb Th., 10(1971), 201- 5

[8] Moon, J. W. "The Spread of Blight in a Random Tree." Jaarverslag Die Suid-Afrikaanse Wirkundoveraniging 13(1970), 83-8

[9] Moon, J.W. Counting Labelled Trees. Canadian Mathematical Congress, Montreal (1970)

[10] Moss, J, Griffeth, and Graham, M. "Abstractions in concurrency control and recovery management (revised)." Coins Technical Report. University of Massachusetts at Amhest (1986), 86–70.

[11] Reed, D. P. "Implementing Atomic Actions on Decentralized Data." ACM Transactions on Computer System. (1983), 3-23.

[12] Tenabaum, A. S., van Renesse, R., van Straveren, G., Sharp, G. J., Mullender, S. J., Jansen, J. and Rossum, G. "Experiences with the Amoeba Distributed Operating System." Comm. Of the ACM, 33, 12 (1990), 46-63.

[13] Upfal, Eli. "An O(log N) Deterministic Packet-Routing Scheme." Journal of the ACM, 39, 1 (1992), 55-70.

[14] Wilson. R. J. Introduction to Graph Theory. Oliver and Boyd, Edinburgh (1972)