Revista Facultad de Ingeniería
ISSN: 0717-1072
facing@uta.cl
Universidad de Tarapacá
Chile

Coelho, Flávia S.; Sauvé, Jacques P.; Abbas, Cláudia B.

Availability evaluation of networks: an approach for n-tier client server architecture

Revista Facultad de Ingeniería, vol. 11, núm. 1, 2003, pp. 25-32
Universidad de Tarapacá
Arica, Chile

REVISTA FACULTAD DE INGENIERÍA, U.T.A. (CHILE), VOL 11 N°1, 2003, pp. 25-32

# AVAILABILITY EVALUATION OF NETWORKS: AN APPROACH FOR N-TIER CLIENT SERVER ARCHITECTURE

**Flávia S. Coelho**[1] **Jacques P. Sauvé**[2] **Cláudia B. Abbas**[3]

*Received november 19, 2002, accepted september 02, 2003*

***ABSTRACT***

*Published work on computer network reliability frequently uses availability as a performance measure. However, although several ways of defining availability have been proposed, none capture the overall level of service obtained by client hosts in a modern n-tier client/server architecture. We propose such a measure by calculating the fraction of client hosts receiving complete services from the network. We also extend a published, efficient heuristic method for calculating availability to take into account our new proposed measure. The end result is a procedure of polynomial complexity* $O(n_t^4)$*, where* $n_t$ *is the total number of components (hosts, links and interconnection equipment) in the network. Numerical results of applying the method to several networks are given.*

*Keywords: Evaluation of availability, network availability, dependability, n -tier client/server architecture.*

## INTRODUCTION

Computer networks must provide infrastructure that meets requirements imposed by the applications it must support. One of the main performance measures of interest is availability [1], [2], [3], [4], [5]. This measure is especially important in applications such as e-commerce, banking transaction systems, etc. where the mission-critical role of applications is evident.

Today, the main way of deploying applications is through the use of an n-tier client/server architecture [6]. Especially popular is the 3-tier architecture composed of the user services tier (client hosts), the middle tier (web servers, application servers, directory services servers, etc.) and the data tier (corporate database servers, mail servers, etc.). The middle tier is also frequently called the *business tie*r, since this is where *business application logic* mostly executes. Other than corporate servers accessed by all clients, certain other servers may exist: these are departmental servers, frequently used by workgroups for file storage and print services. It must be noted that whereas departmental servers are accessed by client hosts in a particular workgroup, corporate servers in the business and data tiers are accessed by all, or most, client hosts in the network.

In the context of such corporate networks, a particular client host obtains services from many servers. In order for a particular client host to obtain *full services* from the network, the following connectivity restrictions must be obeyed (all connectivity restrictions mentioned below are bi-directional):

1. The client host must have access to its departmental servers; this would be necessary for file sharing or other workgroup applications, for example;
2. The client host must have access to corporate servers from the middle tier; this would be necessary so that the main application business logic could execute;
3. Corporate servers in the middle tier must have access to each other; this is necessary so that, for example, a web server can locate a service through a directory server, create a business object on an application server and call its business methods;
4. Corporate servers in the middle tier must have access to corporate servers in the data tier; for example, this would be necessary whenever a business object executing in an application server (middle tier) needs persistence services from a database server (data tier).

The failure of hosts (clients or servers), interconnection equipment and communication links can all affect these connectivities and thus lower the availability of the client hosts' access to services. Several availability measures have been proposed in the literature: probability of network connectivity [7], [8], [9], [10],

[1] U. Católica de Brasília, Brasil, Phone: +55-610-356-9314 Fax: +55-83-356-3010, fcoelho@uch.br
[2] U. Federal de Campina Grande, Dept. de Sistemas e Computação, Brasil, Phone: +55-83-310-1120, Fax: +55-83-310-1124, jacques@dsc.ufcg.edu.br
[3] U. de Brasília , Dept. Engenharia Elétrica, Lab. Redes, Faculdade de Tecnologia, Brasil, Phone: +55-61-3072308 Fax: +55-61-274-6651, barenco@redes.unb.br

[11], [12], [13]; the probability of all operational devices accessing a particular device [8], [23]; the number of device pairs that can communicate [8], [10], [13], [15], [23]; and the number of devices that can communicate with a particular device [8]. *We claim that a new way of measuring availability must be foun*d. The issue must be examined from the user population point of view since this is what better reflects the effect failures have on the business. Since a network only exists to offer services to its user population, the effect of failures on the user population must be captured. Disconnecting a single user certainly not as important as disconnecting a database server affecting all users. What does it mean to say that "the network is 99.95% available"? We claim that availability should measure the fraction of the user population receiving full network services, on average.

In the next section, we formally define a new measure of availability in computer networks based on the n-tier client/server architecture. After that, we present an efficient (polynomial complexity) heuristic method for calculating the new availability measure. Numerical results for particular topological configuration are present follow.

## A NEW MEASURE OF AVAILABILITY IN COMPUTER NETWORKS

We consider that the following network components are susceptible to failure:

- Hosts (clients, departmental servers, middle tier corporate servers and data tier corporate servers);
- Interconnection equipment (switches, routers, hubs, etc.);
- Individual links.

We use the following notation, shown in Table 1, in defining the new availability measure.

In order to define the new availability measure, we use several matrices described below. In this paper, we use the term *adjacent* in a graph-theoretic sense. Two components are *adjacent* if they are directly connected by a communication link. The components thus considered are hosts and interconnection equipment. The *Original Adjacency Matrix* between hosts and interconnection equipment is defined as follows. OAM = [$oam_{ij}$] is a matrix of order $(n_h + n_i) \times (n_h + n_i)$, where $1 \leq i, j \leq (n_h + n_{eq})$. The components are organized in the matrix in the following order: nc, nds, nmcs, ndcs, ni. A component of matrix is defined in the equation 1.

$$oam_{ij} = \begin{cases} 1, \text{ if component i is adjacent to component j} \\ 0, \text{ otherwise} \end{cases} \quad (1)$$

Observe that the components are numbered in the following order: clients, departmental servers, middle tier corporate servers, data tier corporate servers and interconnection equipment.

Table 1.- Notation

| ***Symbol*** | ***Meaning*** |
|---|---|
| $N_t$ | Total number of components $(n_h + n_l + n_i)$ |
| $N_c$ | Number of clients |
| $N_{ds}$ | Number of departamental servers |
| $N_{mcs}$ | Number of middle tier corporate servers |
| $N_{dcs}$ | Number of data tier corporate servers |
| $N_l$ | Number of individual communication links |
| $N_i$ | Number of interconnection equipaments |
| $N_h$ | Total number of hosts $(n_c + n_{ds} + n_{mcs} + n_{dcs})$ |
| $n_f$ | Number of failure states in the network |
| C | Client |
| Sd | Departamental server |
| Smc | Middle tier corporate server |
| Sdc | Data tier corporate server |
| L | Individual link |
| Eq | Interconnection equipment |
| $DS_i$ | Set of departmental servers for client i |
| DS | Set of all departmental servers |
| OAM | Original adjacency matrix |
| $SAM^s$ | Adjacency matrix for a particular failure state s |
| $DCM^i$ | Desired connectivity matrix for client I |
| $SCM^s$ | Actual connectivity matrix for a particular failure state |
| S | Particular failure state |

The *Desired Connectivity Matrix* (for a particular client) is the key to defining the new availability measure. It represents the partial connectivities required between host components so that a client may obtain full services from the network. Since each client host has particular needs, as far a connectivity is required, this matrix must be defined for each of the $n_c$ client hosts. This matrix is structured in the following order: nc, nds, nmcs, ndcs and includes only hosts, since only these are ultimately accessed to obtain network services. In this matrix, hosts are numbered in the following order: clients, departmental servers, middle tier corporate servers and data tier corporate servers.

Let $DCM^i = [dcm^i_{jk}]$ be a matrix of order $n_h \times n_h$, where $1 \le i \le n_c$, $1 \le j, k \le n_h$, the desired connectivity matrix for client i. The value of $dcm^i_{jk}$ can be 0 or 1, depending on the connectivity need between two hosts *j* and *k* in order to satisfy the needs of client i. When $dcm^i_{jk}$ is 0, hosts *j* and *k* need not be connected for client *i* to receive services from the network. In other words, should a failure disconnect hosts *j* and *k* (or should these hosts themselves fail), the availability of services for client *i* will not be affected. The case where $dcm^i_{jk}$ is 1 indicates the opposite situation where client *i* will no longer be obtaining full services from the network. It is also clear that, as far as client *i* is concerned, the availability of services for other clients is of no concern. The matrices $DCM^i$ are thus independent of one another.

Our approach is completely general, in the sense that any Desired Connectivity Matrix may be defined. The remainder of this paper uses a three-tier architecture as an example. For such an architecture, we may obtain the values of $[dcm^i_{jk}]$, as shown in the table 2.

In order to calculate the fraction of clients receiving full services from the network, we must compare the actual connectivities between hosts to the desired connectivities. Actual connectivities between hosts are subject to the effects of failure in the hosts themselves, in the interconnection equipment and in individual communication links. I order to factor in the effect of such failures, we consider the network's *failure state*s. The network can be in one of several failure states. A failure state indicates a set of failed components. That is, a particular failure state consists of one or more failed components. The sets {}, {1}, {2}, ..., {1, 2}, {1, 3}..., {1, 2, 3}, etc. where {i} means that component *i* has failed are examples of failure states.

We use an adjacency matrix for each failure state to represent the adjacencies between hosts and interconnection equipments representing the effects of failures for the given failure state. Let $SAM^s = [sam^s_{ij}]$ be the adjacency matrix for a failure state s. This matrix has the same structure as the Original Adjacency Matrix. The matrix is of order $(n_h + n_i) \times (n_h + n_i)$, where $1 \le s \le n_f$, $1 \le i,j \le (n_h + n_i)$, as shown in the equation 2.

$$sam_{ij} = \begin{cases} 1, \text{ if component i is adjacent to component j} \\ 0, \text{ otherwise} \end{cases} \quad (2)$$

Table 2.- Values of $[dcm^i_{jk}]$

| ***Value of*** $mcd^i_{jk}$ | ***Restrictions*** | ***Meaning*** |
|---|---|---|
| $mcd^i_{kj}$ | $j > k$, $1 \le j \le n_h$, $1 \le k \le n_h$ | Bi-directional connectivity is necessary |
| 1 | $j = k = i$ | Client i must be connected to itself (that is, client i must be operational) |
| 0 | $1 \le j \le n_c$, $j \le k \le n_h$, $j \ne I$ | The connectivity of other clients hosts is of no interest to client *i* |
| 0 | $j = i$, $1 < k \le n_c$ | Client *i* need not be connected to other clients |
| 1 | $j = i$, $n_c + 1 \le k \le n_c + n_{ds}$, $Sd_k \in DS_i$ | Client i must be connected to all its departmental servers, where $DS_i = \{k \mid Sd_k$ is a departmental server for client i$\}$ ($Sd_k$ is the k-th departmental server) |
| 1 | $j = i$, $n_c + n_{ds} + 1 \le k \le n_c + n_{ds} + n_{mcs}$ | Client *i* must be connected to all middle tier corporate servers |
| 0 | $j = i$, $n_c + n_{ds} + n_{mcs} + 1 \le k \le n_h$ | Client i need not be connected to data tier corporate servers |
| 0 | $n_c + 1 \le j \le n_c + n_{ds}$, $j < k \le n_h$ | Departmental servers need not be connected between themselves, or to corporate servers in the middle or data tiers |
| 1 | $n_c + n_{ds} + 1 \le k \le n_c + n_{ds} + n_{mcs}$ | Middle tier corporate servers must connected to each other and to data tier corporate servers |
| 0 | $n_c + n_{ds} + n_{mcs} + 1 \le j \le n_h$ | Data tier servers need not be connected to each other |

We now represent actual connectivities between network hosts, for a particular failure state using an Actual Connectivity Matrix. This matrix is obtained by evaluating the effect of failures for that particular state and how these failures disconnect hosts from one another. Let $SCM^s = [scm^s_{ij}]$, where $1 \leq s \leq n_f$, $1 \leq ij \leq n_h$. $SCM^s$ is of order $n_h$ x $n_h$ and is structured similarly to the Desired Connectivity Matrix. The values $scm^s_{ij}$ are obtained,as shown in the equation 3.

$$scm^s_{ij} = \begin{cases} 1, \text{ if host i is connected to host j} \\ 0, \text{ otherwise} \end{cases} \quad (3)$$

With these 4 matrices in hand (OAM, $SAM^s$, $DCM^i$, $SCM^s$), we are now ready to define the new availability measure for a network based on an n-tier client/server architecture. We consider all possible failure states. In a network with $n_t$ components susceptible to failure, there are $2^{n_t}$ possible failure states. The failure states are denoted by $S_k$, where $\leq 1 k \leq 2^n_t$ $S_k$ represents the k-th failure state, where $S_1 = \phi$, $S_2 = \{1\}$, $S_3 = \{2\}$ and so on. Thus, $S_1$ corresponds to the state without failure and $S_{2^{nt}}$ corresponds to the state where all components have failed.

State $S_k$ occurs with probability $P(S_k)$, as shown in the equation 4.

$$P(S_k) = \prod^{n_t}_{i=1} p_i \,(q_i/p_i)^{T_i(S_k)} \quad (4)$$

Where $pi$ $(1 \leq i \leq n_t)$ is the probability of component $i$ being operational, $q_i = 1 - p_i$, and the value of $T_i(S_k)$ is shown in the equation 5.

$$T_i(S_k) = \begin{cases} 0, \text{ if component i is operational in state } S_k \\ 1, \text{ otherwise.} \end{cases} \quad (5)$$

Finally, the new network availability measure, representing the fraction of client hosts receiving full network services is given by equation 6.

$$A = \sum^{2^{n_c}}_{k=1} P(S_k)A(S_k) \quad (6)$$

Where $A(Sk)$ is the fraction of clients receiving full services in failure state $S_k$, as shown in the equation 7.

$$A = [\sum^{n_c}_{i=1} A_i(S_k)]/n_c \quad (7)$$

Where $A_i(S_k)$ indicates whether client i is receiving full network services in failure state $S_k$ and is given by equation 8.

$$A_i(S_k) = \begin{cases} 1, \text{ if } DCM^i \text{ AND } SCM^{S_k} = DCM^i \\ 0, \text{ otherwise.} \end{cases} \quad (8)$$

In this last equation, *AND* is a boolean operation and must be realized cell by cell to compare the Desired Connectivity Matrix for client i ($DCM^i$) and the Actual Connectivity Matrix for failure state $S_k$ ($SCM^{S_k}$). Thus, $DCM^i$ *AND* $SCM^{S_k} = DCM^i$, if and only if $dcm^i_{jt}$ *AND* $scm^{S_k} = dcm^i_{jt}$ $\forall 1 \leq jt \leq n_h$. In this case, connectivities for client $i$ are satisfied and it is receiving full services from the network in failure state $S_k$.

Evaluating network availability by considering all $2^{n_t}$ possible failure states results in an algorithm with exponential complexity and thus imposes serious restrictions on the size of the network considered. [13] presents a detailed study concerning the difficulty of calculating traditional availability measures.

In the next section, we present a heuristic method for evaluating network availability using the measure introduced above. Only the most probable failure states are considered and the algorithm has polynomial complexity.

## EFFICIENT EVALUATION METHOD FOR THE AVAILABILITY MEASURE

The straightforward evaluation of the new availability measure, A, has computational complexity $O(2^{n_t})$, since all failure states must be considered. We seek a heuristic method which would allow its approximate calculation in polynomial time. Such a method can be found in [14], [15] and is based on enumerating the *most probable* failure states.

The mathematical model used is as follows: a computer network is represented by a non-directed graph $G = (V, E)$ with $n_v = n_h + n_i$ vertices (one vertex for each host and interconnection equipment) and $n_l$ edges (one for each bi-directional communication link). $V$ is a finite set of vertices $V = \{v_1, v_2, ..., vn_v\}$ and E is a finite set of edges $E = \{e_1, e_2, ..., en_l\}$, where each edge is identified by a pair of vertices, $e_k = (v_a, v_b)$, where $1 \leq k \leq n_i$ and $1 \leq a,b \leq n_v$ and $v_a, v_b \in V$. Proper loops, $a = (v_a, v_a)$, are permitted. Specifically, we have, $V = \{C_1, ..., Cn_c, Sd_1, ..., Sdn_{ds}, Smc_1, ..., Smcn_{mcs}, Sdc_1, ..., Sdcn_{dcs}, Eq_1, ..., Eqn_i\}$ and $E = \{e_1, ..., en_l\}$.

Vertices and edges may be in one of two states: operational or failed. Intermediate states, such as one in which component performance is degraded, are not considered. To each component (vertex or edge) $i$ is associated a probability $p_i$ of being in the operational state. $q_i = 1 - p_i$ is the probability of this same component being in the failed state. Observe that $p_i$ can be obtained by (MTTF)/(MTTF + MTTR), where MTTF is Mean Time To Failure and MTTR is Mean Time To Repair. These states change at random and are independent of one another. Finally, the components are ordered such that $R_1 \ge R_2 \ge ... \ge R_{n_t}$, where $R_i = q_i/p_i$.

To determine the most probable network failure states, the states are generated in decreasing order of probability according to a failure state generation algorithm given in [14], [15]. For the $n_m$ most probable failure states, $S_k$, where $1 \le k \le n_m$, we have $P(S_1) \ge P(S_2) \ge ... \ge P(S_{n_m})$. The value of $n_m$ may be chosen in several ways: as a fixed value, as a fraction of all states, as a function of the total number of components, or in such a way as to obtain a desired level of precision while calculating network availability.

Since only most probable states are considered, it would be useful to obtain upper ($A_{sup}$) and lower ($A_{inf}$) bounds for the availability measure, A. [15] show how to do this as follows. Let $M$ be an arbitrary performance measure, and let $M(S_k)$ be its value when the network is in failure state $S_k$. This measure must be such as to obey the following relation given by equation 9.

$$M(S_{2^{n_t}}) \le M(S_k) \le M(S_1), \text{ where } 1 \le k \le 2^{n_t} \quad (9)$$

In other words, network performance is best in state $S_1$, when the network is fully operational and is worse in state $S_{2^{n_t}}$, when all components have failed. We can thus obtain an upper bound by using the measure $M(S_1)$ for all states not considered in the enumeration and we can obtain a lower bound by using the measure $M(S_{2^{n_t}})$ for all states not considered in the enumeration.

For the proposed availability measure, A, we have the expressions shown in the equations (10) and (11).

$$A_{inf} = \Sigma^{n_m}_{k=1} P(S_k)A(S_k) + (1 - \Sigma^{n_m}_{k=1} P(S_k))A(S_{2^{n_t}}) \quad (10)$$

$$A_{sup} = \Sigma^{n_m}_{k=1} P(S_k)A(S_k) + (1 - \Sigma^{n_m}_{k=1} P(S_k))A(S_1) \quad (11)$$

Since in state $S_{2^{n_t}}$ all components have failed, we have $A(S_{2^{n_t}}) = 0$. Further, in state $S_1$ no component has failed and we have $A(S_1) = 1$. Therefore, we have the equations 12 and 13.

$$A_{inf} = \Sigma^{n_m}_{k=1} P(S_k)A(S_k) \quad (12)$$

$$A_{sup} = \Sigma^{n_m}_{k=1} P(S_k)A(S_k) + (1 - \Sigma^{n_m}_{k=1} P(S_k)) \quad (13)$$

We are ready to give the algorithm for calculating the upper and lower bounds for network availability.

***Algorithm for Evaluation Method for the Network Availability Measure***

**Purpose**: calculate upper and lower bounds for the availability of a network based on a n-tier client/server architecture, that is, the fraction of client hosts receiving full network services.

**Input**: number of clients, number of departmental servers, number of middle tier corporate servers, number of data tier corporate servers, total number of components, type of each component and its probability of failure (in increasing order of probability), list of departmental servers and the clients it serves, list of adjacencies between hosts and interconnection equipment.

**Output**: upper and lower bounds for the network availability by considering the nm most probable network failure states.

```
EVALUATE NETWORK AVAILABILITY {
obtain_clients_for_deparmental_servers();
obtain_original_adjacency_matrix();
obtain_desired_adjacency_matrix_for_clients();
S1=ϕ;
calculate_probability_of_first_failure_state();
execute_algorithm_ORDER() [15]; /* gives nm most probable states and their probability */
lower_bound = 0;
cumulative_state_probability = 0;
from i = 1 to nm do {
        generate_adjacency_matrix_for_i-th_failure_state( );
        execute_transitive_closure_algorithm ( ); (Note)
        generate_connectivity_matrix_for_i-th_network_failure_state( );
        clients_ok=calculate_number_of_clients_with_full_network_service_in_state_i( );
        lower_bound=lower_bound+(clients_ok/number_clients) * probability_state[i];
        cumulative_state_probability=cumulative_state_probability + probability_state[i];
        }
        upper_bound=lower_bound+(1-cumulative_state_probability);
}
```

(Note): Given the adjacency matrix X of graph $G = (V, E)$, the *transitive closure* algorithm generates the adjacency matrix $X^+$ of the transitive closure of G. The *transitive closure* of $G$ is the graph $G^+ = (V, E^+)$, where an edge (i, j) is in $E^+$, if and only if there is a path from $i$ to j, that is, there exists a sequence of vertices $v0, ..., vt$ with t > 0, i = $v0$ and $(vr, vr+1) .E +$ for all r<$t$ and $vt$=$j$ [16]. The graph $G+$ yields connectivity between hosts.

We may now obtain the computational complexity of every step in the algorithm, as shown in table 3.

Table 3.- Computational Complexity of Proposed Algorithm

| ***Step*** | ***Computational Complexity*** |
|---|---|
| obtain_clients_for_departamental_servers | $O(n_{ds})$ |
| obtain_original_adjacency_matrix | $O[(n_h + n_i)^2]$ |
| obtain_desired_adjacency_matrix_for_clients | $O(n_c n_h^2)$ |
| calculate_probability_of_first_failure_state | $O(n_t)$ |
| execute_algorithm_ORDER [14, 15] | $O(n_t^2 n_m + n_t n_m \log n_m)$ |
| generate_adjacency_matrix_for_i-th_failure_state for most probable states | $O[n_m(n_h + n_i)^2]$ |
| execute_transitive_closure_algorithm | $O[n_m(n_h + n_i)^3]$ |
| generate_connectivity_matrix_for_i-ththth_network_failure_stateformostprobable states | $O[n_m(n_h + n_i)^2]$ |
| calculate_number_of_clients_with_full_network_service_in_state_i_for_most_probable_states | $O(n_m n_c n_h^2)$ |

Since the value of $n_t$ is larger than the values of $n_{ds}$, $n_c$, $n_h$, $n_i$, the computational complexity of the algorithm is $O[n_m n_t^3 + n_m n_t \log n_m]$. Since $n_m$ is usually chosen as a linear function of $n_t$, we have complexity $O(n_t^4)$. Thus, our heuristic algorithm has polynomial computational complexity and may be used to calculate the availability of much larger networks than would be the case with a standard exponential complexity algorithm.

## EXPERIMENTAL RESULTS

The heuristic method was used to evaluate network availability for several 3-tier configuration topologies. The number of probable states, $n_m$, was set at $100n_t$, a value which was found to yield high precision, that is a small difference between upper and lower bounds for availability. The algorithm was coded in C and run on a Pentium II machine clocked at 300 Mhz, with 192 Mbytes of RAM memory, and under Linux with GNU C compiler. The size of topological configurations varied between 40 and 145 components with the component probability of failure varying between 99.9% and 99.99%. Observe that typical network components have availability close to 99.95% [17], [18], [19], [20], [21]. Full experimental results are given in [22]. Some highlights are given below.

1. As the number of most probable failure states considered ($n_m$) approaches the total number of network components ($n_t$), the upper and lower bounds for availability rapidly converge. As an example, consider the network where all links and clients have 99.95% availability and all interconnection equipments and servers have 99.98% availability. Fig. 1 shows the upper and lower availability bounds for several values of $n_m$, showing convergence. We used a value of $n_m = 100n_t$ for most experiments. An alternative would be to use as many failure states as would be required to produce a given difference between upper and lower bounds. For example, a difference of 0.00005 would be adequate for components with availability close to 0.999 since that would yield a precision of 4 decimal places;

2. Adding clients without otherwise affecting the topology of the network does not affect availability. This is to be expected since the measure of availability used is the *fraction* of clients receiving full service, thus effectively factoring in the number of clients. This result gives positive evidence that we are indeed evaluating availability correctly and that we are capturing how topological aspects of the network affect availability;

3. Adding servers (departmental and corporate, in any tier) generally lowers availability, as expected;

4. Raising the availability of individual components raises the overall network availability, also as

expected. This is further evidence that the algorithm is implemented satisfactorily;

5. Adding alternative paths in the topology raises the overall network availability, also as expected. This is further evidence that the algorithm is implemented satisfactorily;

6. We found it reasonably easily to achieve 99.95% network availability. Higher values require the use of alternative paths in the topology;

7. A comparison between a *collapsed backbone* topology and a hierarchical topology, consisting of core, distribution and access levels indicates that a hierarchical topology has better availability, as long as single points of failure are avoided. Otherwise, a collapsed backbone yields better availability;

8. Experimental results agree with the computational complexity of the method, that is $O[n_m n_t^3 + n_m n_t log\ n_m]$. For fixed $n_m$, the number of operations grows as $O(n_t^3)$. Doubling the total number of components, from 40 to 80, for example, should multiply execution time by 8, as indeed happens (3.35 seconds and 28.9 seconds, respectively).

## CONCLUSIONS

When we started researching the current problem, we were investigating network design methodologies. Network design deals with finding appropriate network solutions (architectures, topologies, equipment, protocols, etc.) in order to meet certain design constraints such as cost and performance. Among performance goals are those dealing with delay, throughput and availability. Although techniques dealing with delay and throughput are well-solved, we found that the same could not be said about availability. In particular, we found that availability measures suggested in the literature did not address the problem from the "business" point of view. In other words, what should it mean for a network to be 99.97% available?

Our first contribution is thus to offer a new measure of availability that has direct significance to the business: the fraction of clients receiving full network services. We claim that such a measure is easily understandable by network operators, network administrators, mid-level and top-level management and is thus preferable to other measures used to date. The measure has been formally defined and an efficient, polynomial-time, heuristic method was produced for calculating the new measure. This is our second contribution. Experimental results have confirmed the expected behavior of the availability measure as various network parameters change.

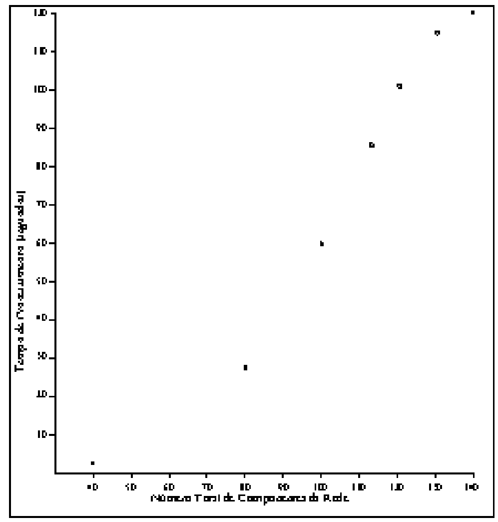

Fig. 1. – Execution time versus total number of network components

Further work includes the introduction of the heuristic method in a general network design tool. So far, we can only analyze a given network. It would be beneficial to find ways of *designing* networks to meet given availability criteria. For example, it is possible to factor our method in the work of [23], [24]. The heuristic method is especially suited to this task since the enumeration of most probable failure states indicates where the most critical failure points are located (this information is part of the failure state definition). Another direction would be to take into account mechanisms that inherently improve availability, such a the Spanning Tree Protocol [25], dynamic routing [26], HSRP (Hot Standby Routing Protocol) [18] and DLD (Deterministic Load Distribution) [19].

Attempts to decrease computational complexity of proposed algorithm and errors bounds are being studied.

As a final note, observe that the method proposed is not only applicable to n-tier architectures, although its applicability for such architectures shows that modern networks can be adequately handled by the method.

## REFERENCES

[1] Bulletproofing; "Networks for High-Availability Computing", White Paper.

www.3com.com/technology/tech-net/white-papers/500693.htm, 1998.

[2] V. Bonaventura; "Service Availability of Communication Networks", Proceedings 1980 National Telecommunication Conference, 1980.

[3] J. Buchanan, E. Weal; "High-Availability Networks", www.3com.com/nsc/netage/q1_1999/p24.html, 1999.

[4] High System Availability. Product Bulletin. Cisco Systems, 1999.

[5] Prerequisites for High-Availability Infrastructures. White paper. Cisco Systems, 2000.

[6] J. Edwards; "3-Tier Server/Client at Work", John Wiley & Sons, 1999.

[7] K. K. Aggarwal, J. S. Gupta, K. B. Misra; "A Simple Method for Reliability Evaluation of a Communication Systems", IEEE Transactions on Communications, Vol. 23, 1975.

[8] M. O. Ball; "Computing Network Reliability", Operations Research, Vol. 27, 1979.

[9] M. Ball, R. M. Van Slyke; "Backtracking Algorithms for Network Reliability Analysis", Annals of Discrete Mathematics, Vol.1, 1977.

[10] P. Kubat; "Estimation of Reliability for Communication Computer Networks - Simulation/Analytic Approach", IEEE Transactions on Communications, Vol. 37, 1989.

[11] D. Torrieri; "Calculation of Node-Pair Reliability in Large Networks with Unreliable Nodes", IEEE Transactions on Reliability, Vol. 43, 1994.

[12] R. Van Slyke, H. Frank; "Network Reliability Analysis: Part I". Networks, Vol. 1, 1972.

[13] M. O. Ball; "Complexity of Network Reliability Computations", Networks, Vol. 10, 1980.

[14] V. O. K. Li, J. A. Silvester; "An Approach for Studying the Performance of Networks with Unreliable Components", Technical Report CSI-02-01. University Southern California, 1983.

[15] V. O. K. Li, J. A. Silvester; "Performance Analysis of Networks with Unreliable Components", IEEE Transactions on Communications, Vol. 32, 1984.

[16] S. Even; "Graph Algorithms", Rockville, 1979.

[17] Link Builder MSH Multi-Services Hubs; Product Data Sheet;

www.3com.com/products/dsheets/400153.html, 2000.

[18] HSRP Support; Product Bulletin #768; www.cisco.com/warp/public/732/100/768-pb.htm, 1999.

[19] Deterministic Load Distribution; Product Bulletin#271;www.cisco.com/warp/public/732/102/271-pb.htm, 1999.

[20] High System Availability; Product Bulletin; Cisco Systems, 1999.

[21] Prerequisites for High-Availability Infrastructures; White paper; Cisco Systems, 2000.

[22] F. Coelho; "Avaliação de Disponibilidade de Redes de Computadores Baseadas na Arquitetura Cliente/Servidor em n Camadas", Master's Thesis, Universidade Federal da Paraíba, 2000.

[23] S. Pierre, A. Elgibaoui; "A Tabu-Search Approach for Designing Computer-Network Topologies with Unreliable Components", IEEE Transactions on Reliability, Vol. 46, 1997.

[24] A. Balakrishnan, T. L. Magnanti, P. Mirchandani; "Designing Hierarchical Survivable Networks", Operations Research, Vol.46, 1998.

[25] R. Perlman; "Interconnections"; Addison-Wesley, 1992.

[26] M. Ford, H. K. Lew, S. Spanier, T. Stevenson; "Internetworking Technologies Handbook"; Cisco Press, 1997.