A UNIFYING APPROACH TO PERFORMANCE
AND RELIABILITY OBJECTIVES

Yonatan LEVY and Patricia E. WIRTH

AT&T Bell Laboratories
Holmdel, NJ 07733, USA

It is common practice to specify system reliability requirements separate from performance objectives. In many distributed systems, however, failures cause degraded performance rather than unavailability. In this paper, performance and availability objectives are defined jointly, and their use for the determination of system capacity is demonstrated. The approach is to define several grades of service with corresponding availability levels. This approach enables the customer to explicitly define how frequently failures can be tolerated, where failures are defined in terms of the level of service degradation they cause. The analysis, however, is relatively simple since not all failure scenarios have to be considered, and existing methods of reliability and performance analysis can be applied. The use of the joint criteria to determine system capacity is demonstrated on multi-processor systems and a distributed communication network.

1. INTRODUCTION

The quality of the service provided to the users of a telecommunications system can be judged by a number of different customer perceived measures. These include such measures as the probability of successful service completion, the probability of being blocked, or the delay experienced in receiving service. The quality of the service observed by a user of the system can be degraded due to equipment failures (reliability) or to traffic congestion (performance). Although the user of the system may not be able to distinguish the cause of the degraded service, it is common practice to specify system reliability requirements in terms of availability separate from performance. Typically, a system may have reliability/availability objectives specified for component failure rates, component down-time per year, the system unavailability, etc., while the performance objectives for the system may include requirements on blocking during peak hours, component and end-to-end delay, response times, etc.

In many distributed systems and telecommunications networks the components are interdependent and the traffic originally destined for a failed component may be routed to another component(s) potentially causing performance problems for it. In this situation, a failure would cause degraded performance rather than total unavailability. The concept of "performability" was introduced by Meyer [1,2,3] to allow for the joint analysis of performance and reliability to satisfy an expected performance objective. In this approach the amount of cumulative performance (or effectiveness) of the observed system is evaluated during a specified time period sometimes called the "mission" time. This is done by identifying a performance variable and determining the model that represents the system and its environment. The model is considered to be solved when the distribution of the performance variable has been determined. For a large system, this approach can involve considerable computation since the performance must be evaluated over all failure scenarios and the number of system states can become intractable. For homogeneous systems (i.e., a single component type) with $n$ states, the complexities of obtaining the performability measure are polynomial in the number of states for a variety of algorithms (e.g., $O(n^4 m)$ where $m$ is a truncation parameter [4] and $O(n^3)$ [5]). For heterogeneous systems with $k$ component types an algorithm with complexity $O(kn^3)$ was recently proposed [6]. This type of performability analysis
has been applied to the design of computer systems [1,6-10] and telecommunication systems [11]. Another model that analyzes reliability in terms of customer-perceivable service effects was applied to cutoff calls in switching systems [12].

In this paper, performance and availability objectives are defined jointly for use in determining the amount of equipment (e.g., capacity for a fixed number of processors, or number of processors for a fixed loading) required to give customers an explicitly defined set of performance levels. Our approach is to define several grades of service with corresponding availability thresholds. This approach enables the customer to explicitly define how frequently failures can be tolerated, where failures are defined in terms of the level of performance degradation they cause. Such a joint definition of performance and availability objectives may much more clearly capture a user's perception of the system than would a measure which averages over all failure scenarios. For many new services, performance objectives for system response times are stated in terms of percentiles rather than expected values. Our joint definition of performance and reliability objectives is in this same spirit.

Using the joint objectives, the capacity of a system is derived by decomposing the model into an availability model and a series of performance models conditioned on the availability state. This analysis is similar to that used in some performability studies, and it is a good approximation since failures usually occur on a much larger time scale than that of arrivals and service completions, resulting in a nearly completely decomposable model as defined by Courtois [13]. As a result, existing methods and computer packages for performance and reliability analyses can be applied. Moreover, due to the way our criteria are defined, not all failure scenarios have to be considered in order to determine the required amount of equipment.

In Section 2 we formally define the joint performance and reliability criteria. In Section 3 we provide a discussion of the decomposability on which the analysis is based, as well as an algorithm for capacity determination. The application of our method to a multiprocessor system and a communication network is demonstrated in Section 4 where it is shown how to efficiently determine the capacity of the systems.

2. DEFINITION OF PERFORMANCE/AVAILABILITY CRITERIA

2.1 Illustrative Example: Three Service Grades

Before the level of redundancy required for a system or network of systems can be determined, the availability requirements and performance criteria for service under all conditions, including failures, have to be specified. We propose a framework for specifying performance/availability criteria which takes advantage of the decomposability of the system. An illustrative example framework consists of three performance levels or service grades:

- **Good Service** ($S_G$) - The service level expected most of the time, possibly even under certain types of failures. This performance level can be thought of as that which customers consider to be optimal for the service being provided.

- **Marginal Service** ($S_M$) - The service level that is acceptable only under failure conditions. This performance level can be thought of as that which customers may find tolerable for brief periods of time, but which is intolerable if frequently encountered.

- **Unacceptable Service** ($S_U$) - The service level that may occur only under severe failure conditions. This performance level can be thought of as that which customers find intolerable whenever encountered.

An example of these service grades might be a packet switched "echoplex" application in which good service might be described as occurring if 95% of the echo delays are less than 100 msec, marginal service - as occurring if the 95th percentile of the delay distribution is between 100 msec. and 500 msec, and unacceptable service could be defined to occur whenever delays are above the
The marginal service level. Clearly all these definitions apply over some observation period, which is how telecommunications systems are usually measured.

The availability criteria then specify the minimum probability threshold for the occurrence of Good Service, $P_G$, and the maximum probability threshold for the occurrence of Unacceptable Service, $P_U$. These probability thresholds must be satisfied uniformly over time. In the echoplex example we might specify $P_G$ to be 90% meaning that failures that cause the system to provide service worse than the good service level ($S_G$) should occur no more frequently than 10% of the time. Additionally, we might specify $P_U$ to be 1% meaning that failures severe enough to cause the system to provide service worse than the marginal service level ($S_M$) should occur no more frequently than 1% of the time.

When these criteria are applied to a given system architecture and equipment configuration and failure rates, two capacities can be determined based on the Good Service and Marginal Service levels:

- $C_G$ - The maximum load for which, under a given system configuration, failures cause a service level worse than $S_G$ with probability not exceeding $1 - P_G$.
- $C_M$ - The maximum load for which, under a given system configuration, failures cause a service level worse than $S_M$ with probability not exceeding $P_U$.

The limiting capacity of the system ($C$) is then defined to be the smaller of the two:

$$C = \min\{C_G, C_M\}.$$  

2.2 Generalized Framework

The performance/availability framework illustrated in the previous section can be generalized to accommodate multiple grades of service. We let $S_1$ signify the good service level and $S_L$ signify the unacceptable service level. The $L$ well ordered levels of performance are given by

$$S_1 \prec S_2 \prec \ldots \prec S_L.$$  

A set of availability thresholds

$$F_1 \ll F_2 \ll \ldots \ll F_{L-1}$$  

is then specified. In the previous example, $F_1 = P_G$, and $F_2 = 1 - P_U$. These thresholds can be interpreted to mean that

$$\text{Prob}\{\text{Performance level is } S_\ell \text{ or better} \} \geq F_\ell.$$  

We then define the sub-capacities to be

- $C_\ell$ - the maximum load for which, under a given system configuration, failures cause a service level worse than $S_\ell$ with probability not exceeding $1 - F_\ell$.

In this generalized framework, the system capacity is the minimum of the $L - 1$ capacity levels determined from the reliability thresholds. That is,

$$C = \min\{C_\ell; \ell = 1, 2, \ldots, L - 1\}.$$  

3. ANALYSIS

The state of a system subject to failures can be represented by a pair $(X, Y)$, where $X$ is the congestion state of the system (e.g., number of jobs, virtual waiting time), and $Y$ is the availability state of the system (e.g., number of failed components, time elapsed since failure). Clearly, $X$ and $Y$ can each be multi-dimensional. Some of the research in this area have explored the joint
analysis of \((X,Y)\). These models are highly specialized, intractable, and can be solved only for small systems [10]. Moreover, the analysis often yields a measure which represents some performance measure of the system averaged over all failure scenarios [7,8,10].

Our criteria lead naturally to a decomposition, where the process \(Y\) is analyzed as an availability model, and a series of performance models \(\{X|Y=y\}\) is then used to analyze the performance under a given availability state. This approach is also justified by the fact that changes in the congestion state \(X\) occur at a much faster rate (order of seconds or milliseconds) than changes in the availability state \(Y\) (order of hours, days, or even months). This means that the process \((X,Y)\) is nearly completely decomposable in the sense that the transition matrix has the form \(Q_0 + \epsilon Q_1\), where \(Q_0\) is a block triangular matrix accounting for transitions in \(X\) for a fixed \(Y\), and \(\epsilon\) is the ratio of transition rates between and within levels of \(Y\). For such processes, the errors in steady-state probabilities incurred by decomposition are on the order of \(\epsilon\) [13]. For typical systems, \(\epsilon\) is on the order of \(10^{-3}\) to \(10^{-6}\), which is a good enough accuracy considering that most of the parameters are actually estimates subject to statistical errors. Hence, in what follows, only steady-state measures of the decomposed model will be used.

The basic idea is to order the availability states according to the steady-state capacity of the model \(\{X|Y=y\}\). For each service level \(\ell\), we identify the set of availability states that satisfies the availability objective \(F_\ell\), and then determine the capacity \(C_\ell\) by the state with the lowest capacity in this set. The transient effects can be ignored since transitions into failure states occur more often from 'better' states than from 'worse' states, and thus using the steady-state capacity is on the conservative side.

The actual algorithm to determine the capacity is as follows:

**Step 0:** Decide on a method for computing the probability \(P(y)\) of availability state \(y\). Also, define the capacity and decide on a method to compute the capacity \(C_\ell(y)\) for each availability state \(y\) and service level \(\ell\). No actual computations are involved in this step.

**Step 1:** Compute the cumulative sum of probabilities \(\Sigma P(y)\) in order of decreasing capacity. For some systems, this order is obvious without actually computing the capacity, while for other systems only a partial order is evident, and some capacity computation is needed. For each service level \(\ell; \ell=1,2,...,L-1\), (i.e., except the last one) identify the critical state \(y_\ell\), which is the first one to cause \(\Sigma P(y)\) to exceed \(F_\ell\).

**Step 2:** For each service level \(\ell; \ell=1,2,...,L-1\), compute the capacity \(C_\ell = C_\ell(y_\ell)\). The system capacity is then given by \(C = \min\{C_\ell; \ell=1,2,...,L-1\}\).

As demonstrated by the examples in the next section, many failure states that occur with small probabilities are not considered at all as they are included in the unacceptable service level \(L\). Other methods of determining capacity based on performability require computing over the whole range of failure scenarios.

### 4. APPLICATIONS

The joint service/availability criteria and their use in determining system capacity is demonstrated on two systems: a multiprocessor system, where the performance measure is response time, and a communication network, where the performance measure is the completion rate.

#### 4.1 A Multiprocessor System

The multiprocessor system has \(N\) processors, and each one is assumed to have availability of \(p\). The processing power of \(n\) processor is \(\phi(n)\), where \(\phi(1) = 1, \phi(n-1) \leq \phi(n) \leq n\), and \(\phi(n)/n\) decreases with \(n\). This concavity reflects the increase in communication overhead. Jobs arrive at a rate of \(\lambda\) and require an average processing time of \(\bar{x}\) on a single processor. Thus, the load on the
system is expressed as \( a = \lambda \bar{x} \). The main performance measure is a function of the response time. For example, the mean response time, assuming ideal processor sharing among \( k \) available processors, is given by \( R(a,k) = \frac{\bar{x}}{\phi(k) - a} \). The number of available processors is a random variable, \( K \), and the availability function \( A_N(k) = \Pr(K \geq k) \), which denotes the probability of at least \( k \) processors available, can be calculated using a standard reliability model. The performance grades are defined in terms of the response time. For example, \( S_G: R \leq R_G, S_M: R \leq R_M \), and \( S_U: R > R_M \). \( S_G \) should occur with probability of at least \( P_G \), and \( S_U \) should occur with probability not exceeding \( P_U \).

Since the response time with \( k \) available processors is stochastically smaller than with \( j < k \) processors, the ordering of availability states is trivial in this case. Therefore, for any response time measure, the critical states for capacity determination are given by

\[
k_G = \max\{k; A_N(k) \geq P_G\} \quad \text{and} \quad k_M = \max\{k; A_N(k) \geq 1 - P_U\}.
\]

The capacity of \( N \) processors is given by

\[
C(N) = \min\{C_G(N), C_M(N)\},
\]

where

\[
C_\ell = \max\{a; R(a,k_\ell) \leq R_\ell\}; \quad \text{for} \quad \ell = G \text{ or } M.
\]

Hence, finding \( k_\ell \) involves a pure availability model, while the determination of \( C_\ell \) involves a pure performance model. For the mean response time measure,

\[
C_\ell = \phi(k_\ell) - \frac{\bar{x}}{R_\ell}; \quad \text{for} \quad \ell = G \text{ or } M.
\]

Similarly, for a given load \( a \), one can find the level of redundancy required by determining the minimum number of processors, \( N(a) = \min\{N; C(N) \geq a\} \), that satisfies the criteria.

4.2 A Communication Network

We next consider a communication network providing an enhanced service (e.g., voice recognition or announcements). For efficient routing and processing, \( N \) units that provide the service are distributed to switches in the network. Calls attempting to access a failed or congested unit are rerouted to one central pool of units. More precisely, we consider a topology where \( N - K \) sites overflow into one central pool. Each site has a single PU (processing unit) and \( n \) channels for providing the service. The central pool has \( K \) PUs and \( nK \) channels, and calls may arrive directly there for service. Calls overflow from the sites to the central pool either when all channels are busy or when the PU fails. If all channels in the central pool are busy or unavailable, the call is blocked. For the analysis here, we assume that each site has capacity of \( c_1 \) and is engineered such that the level of overflow due to congestion is negligible. A model with engineered overflows for this system is available, however, and is based on the same principles.

For this system, the main performance measure is the level of blocking; that is, the fraction of calls that cannot access a channel. The joint performance/availability criteria are defined with respect to the probability of blocking, \( B \), in a way similar to the previous example, so that the performance grades \( B_G \) and \( B_M \) are defined and the availability thresholds \( P_G \) and \( P_U \) are specified.

An availability state is denoted by \((i,j)\) if \( i \) sites and \( j \) PUs in the central pool are unavailable. Again, any model can be used to derive the probability of each state. Assuming independence of the PUs and unavailability of \( q_1 \) for each PU, then the probability of availability state \((i,j)\) is

\[
P(i,j) = q_{N-K,i} q_{K,j}, \quad \text{where} \quad q_{Ni} = \binom{N}{i} q_1^i (1-q_1)^{N-i}.
\]
For each site, as long as \( 1 - q_1 \geq P_G \), the site capacity is given by \( c_1 = B^{-1}(n, B_G) \), where \( B^{-1} \) is the inverse of the Erlang blocking function. Under failure, the calls of the failed site are rerouted to the central pool, and therefore experience the same service grade as other calls arriving to the pool. However, any failure scenario affects a given site only if this particular site has failed, while the central pool is affected by any failure. Hence, satisfying the performance/availability criteria for the central pool is sufficient, and we now focus on the central-pool view.

The capacity available at the central pool for local traffic at service level \( \ell \) and availability state \((i,j)\) is given by

\[
C_{\ell}(i,j) = B^{-1}(n(K-j), B_{\ell}) - ic_1, \quad \text{for} \quad \ell = G, M.
\]

If \( C_{\ell}(i,j) < 0 \), then this failure scenario cannot be handled at service level \( \ell \). We are now ready to use the algorithm, and the availability states are ordered by decreasing capacity at the central pool. Summing the state probabilities in this order, the capacity \( C_G \) (\( C_M \)) is determined by the critical failure scenario that causes the summation to exceed \( P_G (1 - P_U) \). Usually in this system \( C_M \) turns out to be the limiting capacity. If the capacity turns out to be negative, it means that \( K \) (the number of PUs in the central pool) should be increased. Hence, one can determine, for a given number of sites, what is the minimum number of PUs required in the central pool and the total load that can be handled, or, for any load, how many sites and PUs are needed.

We now illustrate the algorithm with a numerical example. The system parameters are \( n = 72 \) and \( q_1 = .0023 \), and the performance/availability criteria are set at \( B_G = .01, B_M = .1, P_G = .95 \), and \( P_U = .0001 \). Since \( B_G = .01 \), \( c_1 = 58 \text{ Erls} \). The results of the algorithm for \( K = 2 \) are shown in Table 1. The system availability states are ordered by decreasing capacity, and the table shows the capacities (which are independent of \( N \)), the cumulative sum of state probabilities for \( N = 10 \) and \( 30 \) with the critical states indicated by (G) and (M), and the largest \( N \) for which this is the critical state.

**Table 1. Capacity Determination for \( K = 2 \)**

<table>
<thead>
<tr>
<th>Avail. State ((i,j))</th>
<th>Capacity</th>
<th>( \Sigma P(i,j) )</th>
<th>Largest ( N )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( C_G )</td>
<td>( C_M )</td>
<td>( N = 10 )</td>
<td>( N = 30 )</td>
</tr>
<tr>
<td>(0,0)</td>
<td>125.0</td>
<td>( .9772(G) )</td>
<td>( .9336 )</td>
</tr>
<tr>
<td>(1,0)</td>
<td>67.8</td>
<td>94.5</td>
<td>( .9953 )</td>
</tr>
<tr>
<td>(0,1)</td>
<td>58</td>
<td>73.4</td>
<td>( .9998 )</td>
</tr>
<tr>
<td>(2,0)</td>
<td>*</td>
<td>36.5</td>
<td>( .99991(M) )</td>
</tr>
<tr>
<td>(1,1)</td>
<td>*</td>
<td>15.4</td>
<td>*</td>
</tr>
<tr>
<td>(0,2)</td>
<td>*</td>
<td>0</td>
<td>*</td>
</tr>
</tbody>
</table>

* Not necessary

Consider first the "good service" capacity. All the units in the network are in service with probability \( q_{N0} \). As long as \( q_{N0} \geq P_G \), the capacity is determined by full availability, so \( C_G = B^{-1}(72K, .01) \). For \( P_G = .95 \), up to 22 units (\( N \leq 22 \)), and any \( K \), the full-availability capacity can be used. If \( q_{N0} < P_G \), the case of one site failure has to be included in the "good service" category. The probability of good service then becomes \( q_{N0} + q_{K0}q_{N-K,1} \) which is well above .95 for all practical values of \( N \), and the corresponding capacity is \( C_G = B^{-1}(72K, .01) - c_1 \).
The marginal-level capacity available to local demand at the central pool under scenario \((i,j)\) is
\[ \text{CM}(i,j) = B^{-1}(72(k-j)+1) - ic_1. \]
Next, additional state probabilities are calculated and summed in decreasing-capacity order until \(\Sigma P(i,j) > 1-P_U = .9999\). The last column in Table 1 shows the largest \(N\) for which a particular state is the one that determines the capacity. This is actually the only information required out of the availability model. If the performance criterion is changed, or if the model is altered to account for overflows, this information can still be used to identify the critical states. The capacities, obviously, would be different. For this example, at most five scenarios have to be considered, and for \(N > 39\) more than 2 units are required at the central pool. For \(K = 3\), at most ten scenarios (out of \(4(N-2)\)) have to be evaluated, and up to 170 sites can be supported.

REFERENCES


