

A Decision Model for Minimizing Critical Component Costs While Achieving a Target Quality of Service

Christopher Garcia (cgarc001@odu.edu)

Engineering Management and Systems Engineering
Old Dominion University

Abstract. Systems are commonly modeled as discrete compositions of components. A useful distinction for analyzing many kinds of systems is to classify the component types into two sets: critical, and non-critical. In this context, a component type K is called *critical* if the system can have only one component of type K . That is, critical components can have no redundancy in the system.

As globalization increases, there is increasing pressure on businesses to deliver higher-quality services at lower costs. Furthermore, innovation and competitive pressure have resulted in increasingly complex services needed to meet customer demands. Many types of services are systems where all or most of the components are critical components. In this paper we examine a problem commonly encountered in the design of services: How to select critical service components in a way that minimizes cost, while ensuring that a desired service quality is achieved. We propose a decision model based on dynamic programming and reliability theory to solve this problem.

INTRODUCTION

Services play an important role in nearly every industry. Fundamentally, a *service* is simply a process carried out to that is intended to meet a particular request. Services are widely provided and employed by businesses – for instance, package delivery or outsourced secretarial functions. Services can be simple or complex, and they may be provided by humans, technology, or a combination of both. More complex services can typically be decomposed into a set of simpler functions or processes. To illustrate this consider, for example, a convention management service responsible for planning and supporting industrial conventions. In order to carry out the service of planning and supporting a convention, the management service will likely make arrangements with one or more hotels, foodservice providers, conference centers, equipment movers, and so on. Each of these is essentially an individual function or *component* of the overall convention management service.

The probability or percentage of time a service is able to meet a request is called its *quality of*

service (QOS). It is intuitively apparent that the reliability of the individual components within a service will have an effect on the QOS. A component's reliability may be increased by adding redundancy – that is, by adding duplicate or alternative components that can take over the function if it were to fail. In the convention management service, for instance, the probability of adequate lodging can be increased if more hotels are prearranged. However, not all types of components can have redundancy within a given service. These are referred to as *critical* component types. A critical component type K has two basic properties: 1) a component of type K is required by the service, and 2) the service cannot have more than one component of type K . In the convention management service example, a critical component might be the conference center (since conventions are typically confined to a single conference center).

Most often, the costly elements of a service are the critical components, and the non-critical component costs are relatively trivial by comparison. Furthermore, commercial services are commonly required to meet a minimum QOS. In this paper a decision model is given to

enable minimum-cost critical component selection while meeting a specified QOS. It is assumed that the reliabilities of each component under consideration are known, and we proceed in three steps. First, the problem is formally framed in terms of reliability theory. Second, the formal problem is converted into an equivalent canonical form suitable for the decision model. And finally, a decision model based on dynamic programming is formulated.

RELIABILITY THEORY

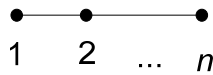
Reliability theory ([1], [2]) provides a framework to model the functioning of a system in terms of the functioning of its individual components. A system \mathbf{x} of n components is represented as a state vector $\mathbf{x} = (x_1, x_2, \dots, x_n)$. Here, $x_i \in \{0, 1\}$ represents the i^{th} component where $x_i = 1$ if the i^{th} component is functioning, and $x_i = 0$ if the i^{th} component has failed. It is assumed that the functioning of the system is determined entirely by the state vector \mathbf{x} . A *structure function* $\emptyset: \mathbf{x} \rightarrow \{0, 1\}$ is used to distinguish between

functioning and non-functioning state vectors as follows:

$$\emptyset(\mathbf{x}) = \begin{cases} 1, & \text{if the system functions in state } \mathbf{x} \\ 0, & \text{if the system fails in state } \mathbf{x} \end{cases}$$

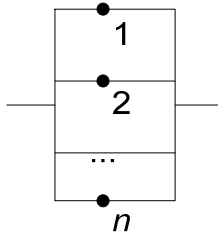
A more intuitive way to represent structure functions is by using diagrams, as shown in Figure 1. Circles are used to represent components, and lines that connect them specify different types of interdependencies. In particular, there are two prototypical system configurations that, when used in combination, can represent any system: *series* configurations and *parallel* configurations. A series configuration is depicted as a straight line connecting two or more components. In a series configuration, the entire system (or sub-system) fails if any component fails. A parallel configuration is depicted by an inward and outward forking of parallel lines, with each component (or sub-system) on its own line. A parallel system (or sub-system) will fail only if *all* of its components (or sub-systems) fail.

A Series Configuration



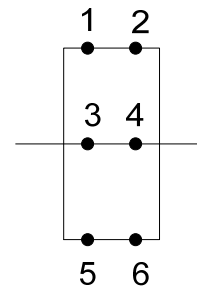
$$\emptyset(\mathbf{x}) = \min(x_1, x_2, \dots, x_n)$$

A Parallel Configuration



$$\emptyset(\mathbf{x}) = \max(x_1, x_2, \dots, x_n)$$

A Parallel System of 3 Series Sub-systems



$$\emptyset(\mathbf{x}) = \max \begin{cases} \min(x_1, x_2) \\ \min(x_3, x_4) \\ \min(x_5, x_6) \end{cases}$$

Figure 1: Structure function diagrams

Each component i has a probability p_i of functioning. Specifically, $p_i = P\{x_i = 1\}$, and the individual component states are assumed to be statistically independent. The component reliability vector \mathbf{p} , analogous to the state vector \mathbf{x} , is defined as $\mathbf{p} = (p_1, p_2, \dots, p_n)$. The *reliability* of the system is the probability that system is functioning: $P\{\emptyset(\mathbf{x}) = 1\}$. The reliability of the

system is determined by the individual component reliabilities and is denoted by the *reliability function* $r(\mathbf{p})$. The reliability function for series and parallel systems of n components, respectively, are shown below. Proofs are given in [2].

Series System

$$r(\mathbf{p}) = \prod_{i=1}^n p_i$$

Parallel System

$$r(\mathbf{p}) = 1 - \prod_{i=1}^n (1 - p_i)$$

PROBLEM REPRESENTATION

In the problem addressed by this paper, the objective is to select critical components at a total minimum cost while achieving a specified QOS. It is thus assumed that non-critical components are fixed, while critical components are decision variables. Services can very naturally be represented using the system diagram notation shown above. In this representation, the critical components are those that do not occur anywhere within any parallel sub-system. This is illustrated in Figure 2 below. Here, the white circles are critical components and the black circles are non-critical components.

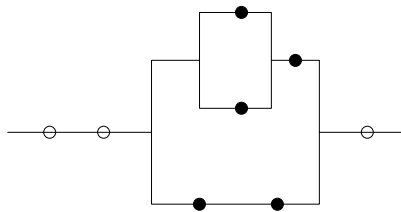


Figure 2: Critical (white) and non-critical (black) components

The QOS of a service is modeled as the system reliability. The component selection model we

develop below requires the problem to first be transformed into an equivalent problem in a canonical form. A system in this form is a series system consisting entirely of critical components. A problem is transformed into this form in two steps: 1) by collapsing each parallel sub-system into a “dummy” component of equivalent reliability, and then 2) combining each of these dummy components into a single critical component.

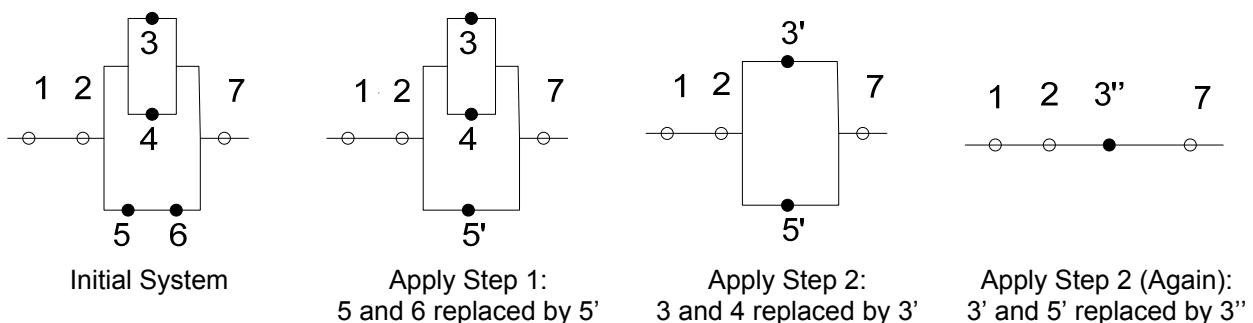
Collapsing Each Parallel Sub-system

Parallel sub-systems are “collapsed” into equivalent dummy components of equal reliability. This is accomplished by applying two alternating steps:

1. Replace each innermost series sub-system $x' = (x_s, x_t, \dots, x_u)$ with a dummy component c of equivalent reliability p_c . From the series system reliability given above, it follows that $p_c = \prod_{i=s}^u p_i$.
2. Replace each innermost parallel sub-system $x' = (x_s, x_t, \dots, x_u)$ with a dummy component c of equivalent reliability p_c . From the parallel system reliability given above, it follows that $p_c = 1 - \prod_{i=s}^u (1 - p_i)$.

These steps are applied repeatedly until the system that remains is a series system. An example of this process being carried out is given in Figure 3 below.

Figure 3: Collapsing parallel sub-systems. White circles represent critical components.



Combining the Dummy Components

Once a series system is obtained, the next step is to combine together the dummy components into a single critical component \mathbf{Q} , which will be called the *base component*. In order to do this, we rely on the following lemma:

Lemma 1: A series system $x = (x_1, x_2, \dots, x_n)$ can have its components rearranged into any order and have the same reliability.

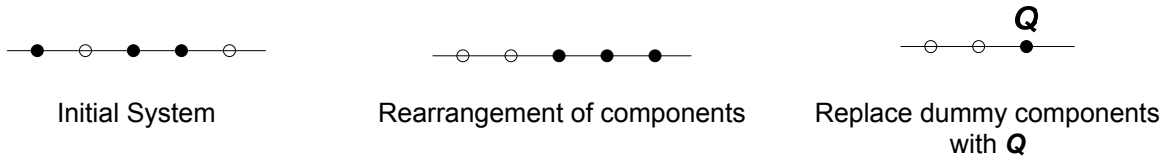
Proof: Let $\mathbf{p} = (p_1, p_2, \dots, p_n)$ be the component reliability vector corresponding to x . Hence, $r(\mathbf{p}) = \prod_{i=1}^n p_i$ as given above. Let \mathbf{p}' be a vector consisting of values p_1, p_2, \dots, p_n in some other order than in \mathbf{p} . Then by the commutative property of multiplication it immediately follows that $r(\mathbf{p}') = r(\mathbf{p})$. ■

At this point we have an n -component series system consisting of 1) k critical components and 2) $n - k$ dummy components. To construct \mathbf{Q} we first reorder the system so that components $1, \dots, k$ are the critical components, and components $k + 1, \dots, n$ are the dummy components. We construct \mathbf{Q} as a component with a probability of functioning $p_Q = \prod_{i=k+1}^n p_i$. Then it follows that

$$r(\mathbf{p}) = \prod_{i=1}^n p_i = \prod_{i=1}^k p_i \prod_{i=k+1}^n p_i = \left[\prod_{i=1}^k p_i \right] p_Q$$

Thus, by replacing components $k + 1, \dots, n$ with \mathbf{Q} we obtain a series system of equivalent reliability that consists solely of critical components. This is shown in Figure 4 below:

Figure 4: Combining the dummy components. White circles are critical components



COMPONENT SELECTION MODEL

In order to apply the component selection model we assume we have a series system of n distinct critical components (as obtained by the preceding step). For each component i in this system we have a set of sources S_i from which to select the i^{th} component. Each component $k_j \in S_i$ has an associated cost c_{ij} and reliability p_{ij} . Additionally, we assume a required minimum QOS of R .

The component selection model we propose is based on dynamic programming [4]. Dynamic programming is an optimization method that can be used to solve multi-stage decision problems, where the optimal decision at a given stage can in general be made from an optimally-chosen preceding stage. This property enables dynamic

programming decision models to be specified as a set of recursive equations.

We begin with an intuitive observation: for a sub-system of components $i + 1, \dots, n$ from our critical component system, there is an associated cost c and reliability p . When we choose the i^{th} component having cost c_i and reliability p_i we get a new sub-system consisting of components i, \dots, n with a cost of $c_i + c$ and a reliability of $p_i p$. This insight lets us devise a recursive way to represent the problem, where the stages represent the critical components, and the decision to be made at each stage i is to select the i^{th} component out of a set of sources S_i .

Let $s_j = (c_j, p_j)$ denote a possible sub-system j consisting of components i, \dots, n where c_j denotes the cost and p_j denotes the reliability.

When there are many different sources available for each component, there are many possible sub-systems s_j . Consider, for example, two possible candidate sub-systems $s_1 = (10, 0.99)$ and $s_2 = (20, 0.65)$. Clearly, it is never optimal to choose s_2 when s_1 can instead be chosen at a lower cost and higher reliability. We express this relationship by saying that s_1 *dominates* s_2 , denoted as $s_1 > s_2$. In general, $s_j > s_k$ whenever

$c_j < c_k$ and $p_j \geq p_k$ or $c_j \leq c_k$ and $p_j > p_k$. A *dominating sub-system* at a given stage is one that is not dominated by any other sub-system. Because non-dominating sub-systems are never optimal, they can be eliminated at each stage from further consideration.

We can now define our decision model as follows:

S_i	The set of sources from which to choose component i
c_{ij}	The cost of component i chosen from source j
p_{ij}	The reliability of component i chosen from source j
R	The minimum required QOS
$f_{i,p}(j)$	The minimum cost of a dominating subsystem i, \dots, n having reliability p where component i is chosen from source j

Table 1: Nomenclature

Model:

$$(1) f_{n,p}(j) = \min_{c_{nj}} \{p = p_{nj} \text{ and } p_{nj} \geq R\}$$

$$(2) f_{i,p}(j) = \min_{j \in S_i} \{c_{ij} + f_{i+1,p'}(k) \mid p = p_{ij}p' \text{ and } p \geq R\}$$

Equation (1) specifies the basis step – that is, how to select the last component n . Equation (2) specifies the inductive step – how to select each component $1, \dots, n-1$ given the sub-system options that immediately follow them. Thus, this is a “work backwards” formulation – we begin by building sub-systems starting at n and then progress backwards one stage at a time, until a final optimal system can be chosen at stage 1.

An Example Problem

A small example may be helpful in demonstrating how to apply this model. We assume we are given the series system shown in Figure 5, and we are given the component choices found in Table 2. Furthermore, we assume we must meet a target QOS of $R = 0.9$.

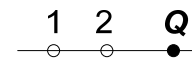


Figure 5: Example system

Component i	Source j	Cost c_{ij}	Reliability p_{ij}
1	1	9	0.972
1	2	10	0.98
2	1	4	0.944
2	2	7	0.901
2	3	3	0.98
2	4	2	0.97
Q	1	12	0.98

Table 2: Component choices

In this example we have three components. **Q** is the combination of non-critical components, derived as shown in the previous section. Because non-critical components are assumed to be fixed, there will always be only one choice for **Q**.

For stage 3, we apply Equation (1) giving $f_{3,0.98}(1) = 12$. Working backwards, we determine the possible sub-systems at stage 2 by applying Equation (2) to compute each possible sub-system cost and reliability. We note

that for stage $i = 2$ the only possible sub-system for stage 3 is the **Q**, and thus the only value of $p' = 0.98$. Thus, for component 2 from source 1 we obtain:

$$(a) p = p_{21}p' = (0.944)(0.98) = 0.925512$$

$$(b) f_{2,0.925512}(1) = c_{21} + f_{3,0.98}(1) = 4 + 12 = 16$$

Performing these computations on all possible choices for component 2 we have:

Source j	p	$f_{2,p}(j)$
1	$p = p_{21}p' = (0.944)(0.98) = 0.925512$	$f_{2,0.925512}(1) = 16$
2	$p = p_{22}p' = (0.901)(0.98) = 0.88298$	N/A, since $p < 0.9$
3	$p = p_{23}p' = (0.98)(0.98) = 0.9604$	$f_{2,0.9604}(1) = 15$
4	$p = p_{24}p' = (0.97)(0.98) = 0.9312$	$f_{2,0.9312}(1) = 14$

Table 3: Stage 2 computations

For stage 2, selecting source 2 results in a reliability of $0.88 < 0.9$, so this option is eliminated. Similarly, the sub-system obtained by choosing source 1 is dominated by those

obtained from sources 3 and 4, so it also is eliminated. The choices that remain are sources 3 and 4. We now perform similar computations for stage 1:

Source j	$f_{2,p'}(k)$	p	$f_{1,p}(j)$
1	$f_{2,0.9604}(3) = 15$	$p = p_{11}p' = (0.972)(0.9604) = 0.93$	$f_{1,0.93}(1) = c_{11} + f_{2,0.9604}(3) = 9 + 15 = 24$
1	$f_{2,0.9312}(4) = 14$	$p = p_{11}p' = (0.972)(0.9312) = 0.9$	$f_{1,0.9}(1) = c_{11} + f_{2,0.9312}(4) = 23 *$
2	$f_{2,0.9604}(3) = 15$	$p = p_{12}p' = (0.98)(0.9604) = 0.94$	$f_{1,0.94}(2) = c_{12} + f_{2,0.9604}(3) = 25$
2	$f_{2,0.9312}(4) = 14$	$p = p_{12}p' = (0.98)(0.9312) = 0.91$	$f_{1,0.91}(2) = c_{12} + f_{2,0.9312}(4) = 24$

Table 4: Stage 1 computations

Thus, the minimum cost required to provide the service at a QOS of 0.9 is 23 (denoted by * in

Table 4). This service can be obtained by selecting component 1 from source 1, component 2 from source 4, and Q .

CONCLUSIONS

In this paper we have shown how to select the critical components for a service in a way that achieves a target quality of service at a minimum cost. Services can be represented using reliability theory. We have shown how to represent services in this manner and transform them into a form that can be used in a dynamic programming decision model. We have shown such a model for determining how to select the critical components and demonstrated how to apply it.

REFERENCES

1. Gertsbakh, I.B. (1989), *Statistical Reliability Theory*, Marcel Dekker
2. Ross, S. M. (2007), *Introduction to Probability Models*, Ninth Edition pp. 571-617, Academic Press
3. Hines, W.W., Montgomery, D.C, Goldsman, D.M, Borror, C.M. (2003), *Probability and Statistics in Engineering*, Fourth Edition, Wiley
4. Bellman, R. (1957), *Dynamic Programming*, Princeton University Press
5. Cormen, T.H., Leiserson, C.E, Rivest, R.L., Stein C. (2001), *Introduction to Algorithms*, Second Edition, MIT Press/McGraw-Hill
6. Denardo, E.V. (2003), *Dynamic Programming*, Dover