Mixed-Integer Linear Models for Reliable System Design

Kelly M. Sullivan*

June 5, 2014

Abstract

System reliability can be increased by installing redundant components. We consider the problem of maximizing system reliability over a discretely constrained set of (independent) component installation decisions. For this class of problems, we develop a mixed integer linear program (MILP) that assumes availability of a binary decision diagram to represent the system's structure function. The model presented is valid for a general class of reliability systems. Moreover, whereas a majority of previous approaches utilize heuristic optimization approaches, our model can be solved exactly using existing MILP solvers. We present a computational study for the problem of allocating redundant components to serially connected k-out-of-n systems. A tightened formulation, capable of solving larger instances within seconds, ensues when each subsystem consists of identical components.

1 Introduction

One well-known strategy for increasing system reliability is to install redundant components. We consider the problem of maximizing system reliability over a discretely-constrained set of component installation decisions. We refer to this general problem as *redundancy allocation*.

While the majority of previous research in redundancy allocation focuses on highly-structured systems (e.g., independent, serially-connected, k-out-of-n subsystems), our approach is more general. In this paper,

*University of Arkansas, Department of Industrial Engineering, ksulliv@uark.edu

we present a new class of models for redundancy allocation in complex systems, i.e., systems that are not decomposable into nested k-out-of-n subsystems. The model we present is a mixed integer *linear* program (MILP), which derives from a linear programming (LP) model that computes system reliability. This model hinges upon representing the system as a binary decision diagram (or BDD, see [1] for origins). The existence of such a model is intriguing given that reliability computations are notoriously nonlinear.

Redundancy allocation has received considerable attention in the past (see [12] for a comprehensive survey of significant works prior to 2000 and [14] for a review of more recent research). With a few notable exceptions [10, 19], the majority of previous research (see, e.g., [9, 11, 15] and more recently, [21, 22]) focuses on optimal allocation of components to k-out-of-n subsystems that are connected in series. This problem is known to be NP-hard even in the special case of a series-parallel system with a single resource constraint [7].

Like [10] and [19], our approach is flexible in the sense that it applies across a broad class of systems that includes serially-connected k-out-of-n subsystems. Our approach is quite different from previously-posed approaches to solving generally structured reliability allocation problems, and we argue that it has significant advantages. The approaches in [10] and [19], for instance, makes use of an ad hoc implicit enumeration procedure that prunes subproblems based on system specific feasibility and reliability calculations. Contrarily, our approach poses redundancy allocation within the framework of a well-studied class of problems—the MILP—thus opening the door to a broad range of theory and algorithms that may lead to more efficient reliability optimization techniques.

Generality and flexibility are not the only potential advantages to our approach. Although our approach applies to systems that are more general than serially-connected k-out-of-n subsystems, we devote significant attention to this special case because of its importance in application. For this special case (hereafter referred to as the k-out-of-n redundancy allocation problem, or k/n-RAP), we believe our approach may yield significant algorithmic and theoretical advances. While a number of approaches have been developed

for solving k/n-RAP, ours is one of only a few known exact approaches.

Among other exact approaches for k/n-RAP ours stands out as being quite unique. Most notably, our approach is based upon solving a MILP, whereas most previous exact approaches are either rooted in dynamic programming [3, 16, 20], implicit enumeration to solve a nonlinear integer program [10, 17], or decomposition [6]. Our use of a linear model enables us to leverage commercial solvers, and the size of our model grows relatively slowly with respect to adding variables and constraints as compared to a dynamic programming approach. Moreover, our formulation extends easily to handle modified constraint and objective structures; such a modification is not possible (or at least not obviously so) for dynamic programming and implicit enumeration approaches. We are aware of one other MILP approach [4] for k/n-RAP, but this model provides only an approximate solution and relies on a log-transformation which may produce irrational data. Contrarily, the coefficients in our model are all rational provided that the component reliabilities are rational.

Our approach is in fact quite general, depending only on the assumptions that (i) the system of interest can be represented as a BDD, (ii) the system is coherent, and (iii) components are independent. The BDD (see [1] for origin) is a data structure that exploits logical relationships between variables to facilitate efficient representation and evaluation of binary-valued functions. Developing a BDD representation is straightforward for the problem classes that are most commonly studied in reliability, and there are established algorithms for doing so in complex systems as well. Binary decision diagrams have received some attention in the reliability literature (see, e.g., [2]), but to our knowledge, we are the first to use a BDD representation to give rise to an optimization model.

1.1 Problem Definition

We now provide a formal definition of our problem followed by a demonstration of how our problem specifies to k/n-RAP. To this end, we define C as the set of components that can be installed and let variables

$x_c,\ c \in C$, equal 1 (0) if component c is (is not) installed. Let $A_c,\ c \in C$, define Bernoulli random variables such that A_c equals 1 (0) if component c is functioning (failed). Define $p_c \equiv P(A_c = 1),\ c \in C$, as the reliability of component c. The following additional notation will be used throughout this paper.

DECISION VARIABLES AND SETS

C: set of components

x_c: whether (1) or not (0) component $c \in C$ is installed

X: set governing feasibility of x, $X \subseteq \{0,1\}^C$

SYSTEM PARAMETERS AND RANDOM VARIABLES

A_c: random variable indicating whether (1) or not (0) installed component $c \in C$ functions

A: random $|C|$-vector comprised of $\{A_c : c \in C\}$

p_c: reliability of component $c \in C$, $p_c = P(A_c = 1)$

ϕ: system structure function, $\phi : \{0,1\}^C \to \{0,1\}$

R: reliability function, $R : X \to [0,1]$

BINOMIAL AND NEGATIVE BINOMIAL PROBABILITY

$$b(k,K,p) = \binom{K}{k} p^k (1-p)^{K-k},\ 0 \le p \le 1,\ 0 \le k \le K$$

$$b^{-1}(k,K,p) = \binom{K-1}{k-1} p^k (1-p)^{K-k},\ 0 \le p \le 1,\ 1 \le k \le K$$

Now suppose $x \in X$ has been chosen and the A-variables have been realized, and let $a_c \equiv A_c x_c$ equal 1 if component c is installed and functioning and 0 otherwise. Given $a \in \{0,1\}^C$, define the *structure function* $\phi : \{0,1\}^C \to \{0,1\}$ such that $\phi(a)$ equals 1 if the system is functioning and 0 otherwise. We

consider the problem

$$\max_{x \in X} R(x) = P[\phi(A \cdot x) = 1], \tag{1}$$

where the operator, $\cdot$, represents an element-wise product and $X \subseteq \{0,1\}^C$ is a set governing feasibility of x. The assumptions implicit in our modeling approach are as follows:

Assumption 1. Components fail independently, i.e., the collection $\{A_c\}_{c \in C}$ are independent.

Assumption 2. The system is *coherent*, i.e., the following conditions hold:

1. ϕ is nondecreasing.

2. All components $c \in C$ are *relevant*. That is, for every $c \in C$, there exists $\bar{a} \in \{0,1\}^C$ with $\bar{a}_c = 0$ such that $\phi(\bar{a} + e_c) > \phi(\bar{a})$. (Note: e_c is the $|C|$-dimensional unit vector with a one in element c.)

Assumption 3. We have as input a BDD representation for the structure function ϕ.

Assumptions 1 and 2 are standard in reliability optimization. We now acquaint the reader with the BDD structure that underlies Assumption 3. A BDD is a connected, directed acyclic graph that is typically used as a computationally efficient means to represent and evaluate a function (such as the structure function ϕ) having multivariate binary input and univariate binary output. The BDD contains *vertices*, each of which is associated with an argument to the binary function. Directed *edges* are constructed to connect the vertices. One of the vertices is designated as the *root*—this is the unique vertex having no entering edges.

Edges are designated as either high or low, and exactly one edge of each type emanates each vertex. These edges may be incident upon another vertex, or they may enter one of two *terminals* (i.e., additional vertices that have no outgoing edges). We refer to the two terminals as s and f. Directed paths beginning at the root vertex and ending at s (f) reflect sufficient conditions to determine that the function evaluates to 1 (0). (We use s and f because we reliability structure functions indicate if a system (s)urvives or (f)ails.)

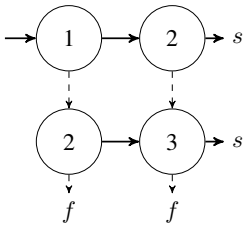

Figure 1: BDD illustration.

To illustrate, consider binary arguments $(a_1, a_2, a_3) \in \{0,1\}^3$ and the function

$$\phi(a_1, a_2, a_3) = \begin{cases} 1 & \text{if } a_1 + a_2 + a_3 \geq 2, \\ 0 & \text{otherwise,} \end{cases} \tag{2}$$

which is the structure function for a 2-out-of-3 system. We present in Figure 1 an illustration of one BDD corresponding to the function ϕ. Each vertex is labeled corresponding to its associated argument: The vertices labeled 1, 2, and 3 correspond respectively to arguments a_1, a_2, and a_3. The root vertex is the single vertex corresponding to a_1. High edges are drawn using solid arrows while low arcs are drawn using dashed arrows. To evaluate ϕ at $(a_1, a_2, a_3) = (0, 1, 1)$, one would trace a path as follows: Begin at the root vertex, labeled 1; because $a_1 = 0$, follow the *low* edge down to the vertex labeled 2; because $a_2 = 1$, traverse the *high* edge right to the vertex labeled 3; because $a_3 = 1$, traverse the *high* edge to terminal s. We refer to such a path as the *realized path* corresponding to input a. Because the realized path ends at s, we have identified that $\phi(a_1, a_2, a_3) = 1$. The reader may verify that, for all $a \in \{0,1\}^3$, $\phi(a) = 1$ if and only if the realized path ends at s; thus, the BDD is a valid representation of ϕ.

Generating BDDs for other types of non-complex systems is fairly straightforward, as shall become apparent in the following section. BDDs can also be generated for complex systems in a systematic manner; however, there is no guarantee that the size of the BDD will be polynomially bounded. In the following, we develop definitions, notation, and theory to characterize a system's BDD.

1.2 BDD Notation and Properties

Define V as the set of internal vertices in the BDD. Each vertex $v \in V$ is associated with a (not necessarily unique) component $c(v) \in C$. For simplicity of exposition, we also define auxiliary vertices s and f corresponding to the terminals of the BDD.

For $v \in V$, let $h(v)$ ($\ell(v)$) denote the destination of the high (low) edge emanating v, where $h(v) \in V \cup \{s\}$ and $\ell(v) \in V \cup \{f\}$. Letting v^* denote the root vertex, we assume there is a directed path through the BDD from vertex v^* to every other vertex $v \in V$; if this is not the case, vertex v can be removed. Because $[V, c(\cdot), h(\cdot), \ell(\cdot)]$ defines a BDD, the following property must hold:

Property 1. The edges defined by $h(\cdot)$ and $\ell(\cdot)$ create no directed cycles. (We may therefore assume there exists an ordering $<$ on $V \cup \{s, f\}$ such that $u < \ell(u)$ and $u < h(u)$ for all $u \in V$.)

If there exists a directed path from $v \in V$ to $v' \in V$, we say v' is a *descendant* of v. One may always construct a BDD in such a way that it is *reduced* (see [5]), i.e., such that the following property is satisfied.

Property 2. If $v' \in V$ is a descendant of $v \in V$, then $c(v') \neq c(v)$.

Property 1 implies that there exists a directed path from each non-terminal vertex to either s or f. Furthermore, we may remove irrelevant vertices (i.e., vertices from which there is a directed path to either s or f, but not both) because these vertices have no effect in evaluating ϕ. Hence, we assume our BDD satisfies the following property:

Property 3. For each $v \in V$, there exist directed paths from (i) v^* to v, (ii) v to s, and (iii) v to f.

To illustrate, suppose the vertices in Figure 1 are numbered $V = \{1, 2, 3, 4\}$, where vertices 1, 2, 3, and 4 are respectively in the upper-left, upper-right, lower-left, and lower-right positions. Table 1 specifies the values of $c(v)$, $h(v)$, and $\ell(v)$ for each $v \in V$. Vertex 1 is the root in this BDD, which is clear because there are no edges incident upon this vertex. Inspection reveals that the BDD is acyclic (i.e., Property 1 is

Table 1: Notation for BDD example.

v	$c(v)$	$h(v)$	$\ell(v)$
$1(v^*)$	1	2	3
2	2	s	4
3	2	4	f
4	3	s	f

satisfied), no directed path connects vertices corresponding to the same component (Property 2 is satisfied), and directed paths exist from $v^* = 1$ to s and f through every vertex (Property 3 is satisfied).

1.3 Specification to k-out-of-n Redundancy Allocation

In this section, we provide a formal definition of k/n-RAP and demonstrate one way to construct a BDD for such a system. We consider a system consisting of m independent subsystems. Subsystem $i \in \{1, \ldots, m\}$ is a k_i-out-of-n_i system, where k_i (a positive integer) is a constant and n_i, the number of independent components in subsystem i, is to be decided. Components having reliability r_i can be installed in subsystem $i \in \{1, \ldots, m\}$ at a cost of $d_i > 0$ each, and a total budget of $B > 0$ is available for installing components in the various subsystems. At least L_i and at most $U_i \geq L_i$ components must be installed in subsystem $i \in \{1, \ldots, m\}$. (If no such bounds are available, then $U_i = \lfloor B/d_i \rfloor$ and $L_i = k_i$, $\forall i \in \{1, \ldots, m\}$ will suffice.) We seek to identify $n_i \in \{L_i, \ldots, U_i\}$, the number of components that should be installed in each subsystem $i \in \{1, \ldots, m\}$ in order to maximize the system's reliability. This problem may be formulated intuitively as

$$\max \left\{ \prod_{i=1}^{m} \left(\sum_{k=k_i}^{n_i} b(k, n_i, r_i) \right) \middle| \sum_{i=1}^{m} d_i n_i \leq B; n_i \in \{L_i, \ldots, U_i\},\ \forall i \in \{1, \ldots, m\} \right\}, \tag{3}$$

where each term in the product corresponds to the (cumulative binomial) probability that at least k_i of the n_i components in subsystem i are functioning.

We may define k/n-RAP as an instance of the general redundancy allocation problem as follows. For

each subsystem $i \in \{1, \ldots, m\}$, define the component set $C_i = \{[i,j] : \ j \in \{1, \ldots, U_i\}\}$, and let $C = \cup_{i=1}^{m} C_i$. When discussing k/n-RAP, we may refer to a component as either $c \in C$, $[i,j] \in C$, or both. Moreover, it will be convenient to modify notation referring to a particular component $c = [i,j] \in C$ by changing occurences of "subscripted c" into "subscripted ij." For instance, we will use x_{ij} instead of x_c to represent whether or not component $c = [i,j] \in C$ has been installed.

Define $p_c = r_i$ as the reliability of component $c = [i,j] \in C$. The structure function $\phi : \{0,1\}^C \to \{0,1\}$ for this system is given by

$$\phi(a) = \begin{cases} 1 & \text{if } \sum_{[i,j] \in C_i} a_{ij} \geq k_i, \ \forall i \in \{1, \ldots, m\} \\ 0 & \text{otherwise,} \end{cases} \tag{4}$$

and the feasible set of component installations is

$$X = \left\{ x \in \{0,1\}^C \ \middle| \ \sum_{i=1}^{m} \sum_{[i,j] \in C_i} d_i x_{ij} \leq B; \ L_i \leq \sum_{[i,j] \in C_i} x_{ij} \leq U_i, \ \forall i \in \{1, \ldots, m\} \right\}. \tag{5}$$

We construct a BDD for k/n-RAP by first constructing a BDD to represent each subsystem. Define the vertex set for subsystem $i \in \{1, \ldots, m\}$ as

$$V_i = \left\{ [i,j]^t \ \middle| \ j \in \{1, \ldots, U_i\}; \ t \in \{\max\{1, j - k_i + 1\}, \ldots, \min\{j, U_i - k_i + 1\}\} \right\}. \tag{6}$$

In this definition, there may be multiple vertices associated with component $[i,j]$: We distinguish among these vertices using index t. Note, however, that there is a unique vertex $[i,1]^1$ corresponding to $j = 1$ for each $i \in \{1, \ldots, m\}$. Henceforth, we refer to this vertex as $v_i^* = [i,1]^1$. For $v = [i,j]^t \in V_i$, define

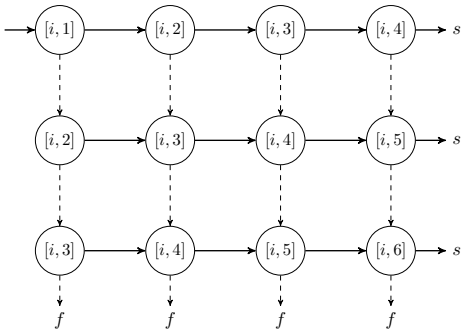

Figure 2: BDD for a 4-out-of-6 system.

$c(v) = [i, j]$ and create high and low edges emanating $v \in V_i$ as follows:

$$h(v) = \begin{cases} [i, j+1]^t & \text{if } t > j - k_i + 1 \\ s & \text{otherwise} \end{cases}$$

$$\ell(v) = \begin{cases} [i, j+1]^{t+1} & \text{if } t < U_i - k_i + 1 \\ f & \text{otherwise} \end{cases}$$

The resulting BDD represents subsystem i using $k_i(U_i - k_i + 1)$ vertices and twice as many edges.

We now illustrate in Figure 2 such a BDD for a subsystem i with $k_i = 4$ and $U_i = 6$. In this figure, vertices v are labeled by $c(v) = [i, j]$; that is, there are three vertices corresponding to components $[i, 3]$ and $[i, 4]$; two vertices corresponding to components $[i, 2]$ and $[i, 5]$; and one vertex corresponding to components $[i, 1]$ and $[i, 6]$. For vertex $v = [i, j]^t \in V_i$, index t corresponds to the row. Thus, $[i, 1]^1$, $[i, 2]^1$, $[i, 3]^1$, and $[i, 4]^1$ define the vertices in the first row; $[i, 2]^2$, $[i, 3]^2$, $[i, 4]^2$, and $[i, 5]^2$ define the second row of vertices; and $[i, 3]^3$, $[i, 4]^3$, $[i, 5]^3$, and $[i, 6]^3$ comprise the third row. In this example, vertex $v_i^* = [i, 1]^1$ is the root vertex. High edges are constructed from each vertex in V_i into the vertex immediately to the right (if one exists) or into s (otherwise). Low edges are constructed from each vertex into the vertex immediately below (if one exists) or into f (otherwise). Observe that any path from the root vertex into s corresponds to a sequence of events that results in at least k_i components functioning (i.e., because there are k_i columns) and

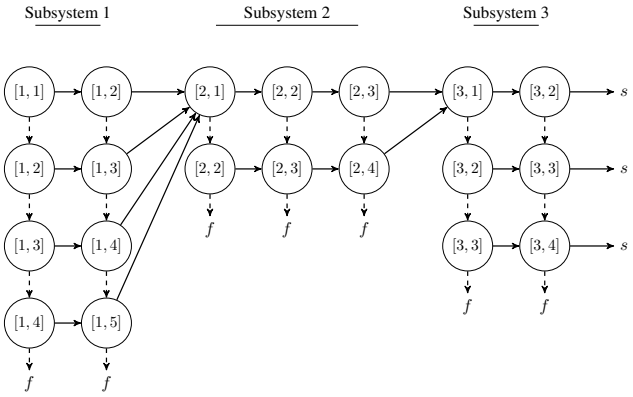

Figure 3: System-level BDD for three serially-connected k-out-of-n subsystems.

any path from the root into f corresponds to a sequence of events that results in fewer than k_i components functioning (because there are $U_i - k_i + 1$ rows).

The BDDs for each subsystem can now be combined to form a single BDD for the entire system. To this end, we define $V = \cup_{i=1}^{m} V_i$ and designate $v_1^* = [1,1]^1$ as the system-level root vertex. We then modify the values of some $h(v)$ for some vertices $v \in V$ in order to connect the sub-BDDs. For $v = [i,j]^t \in V \setminus V_m$ such that $h(v) = s$, we change the value of $h(v)$ to $[i+1,1]^1$. Figure 3 displays the system-level BDD for a system with $m = 3$ subsystems defined by $(k_1, k_2, k_3) = (2, 3, 2)$ and $(U_1, U_2, U_3) = (5, 4, 4)$. Properties 1–3 are all satisfied for this sytem, and in addition, each path from the root vertex into s corresponds to a (perhaps incomplete) set of component function/fail events that is sufficient for the system to function. In the following section, we demonstrate how this BDD (and BDDs for more general systems) gives rise to a MILP for the corresponding redundancy allocation problem.

Before proceeding with the model development, we discuss extension of the version of k/n-RAP discussed in this section. In particular, we note that the BDD for k/n-RAP is a representation of the system's structure function, which does not depend on either d_i or $r_i,\ i \in \{1, \ldots, m\}$. Under this observation, extension to an "asymmetric" version of k/n-RAP (i.e., reliabilities $p_c,\ c \in C_i$ and costs $d_c,\ c \in C_i$ may not all be equal) is trivial. In fact, the BDD for asymmetric k/n-RAP is identical to the BDD prescribed

for the "symmetric" version of k/n-RAP described in this section. Our general model never assumes (see Section 1.1) equal component reliabilities, so this extension is handled with ease. For asymmetric k/n-RAP, we may therefore obtain a valid MILP formulation by simply replacing d_i with d_{ij} in (5) and redefining $p_c = r_{ij}$ for $c = [i, j] \in C$. Moreover, our model derives from an LP that computes system reliability when $x \in X$ is fixed; therefore alternative definitions of X (so long as they are linearly constrained) produce a valid MILP for the resulting reliability maximization problem. Thus, our approach is easily extended to model other well-known reliability optimization problems such as the problem of component selection (see, e.g,. [8]). In the following section, we develop the BDD-based MILP followed by two illustrative examples.

2 Model Development

The BDD representation for ϕ leads to a recursive approach for representing $\phi(A \cdot x)$. For $v \in V$, define

$$\Phi(v, x) = \begin{cases} A_{c(v)}\Phi(h(v), x) + (1 - A_{c(v)})\Phi(\ell(v), x), & \text{if } x_{c(v)} = 1, \\ \Phi(\ell(v), x), & \text{if } x_{c(v)} = 0, \end{cases} \tag{7}$$

where $\Phi(s, x) \equiv 1$ and $\Phi(f, x) \equiv 0$. When $x \in X$ is fixed, $\Phi(v, x)$ is a 0-1 random variable indicating whether or not the subsystem represented by the BDD rooted at vertex $v \in V$ functions. Thus, for fixed $x \in X$, $\Phi(v^*, x)$ is a random variable that equals 1 if the realized path through the BDD ends at s (and 0 otherwise); hence, $\phi(A \cdot x)$ is equivalent to $\Phi(v^*, x)$. The following lemma will be useful in establishing our model:

Lemma 1. For any realization of A, $\Phi(h(v), x) \geq \Phi(\ell(v), x),\ \forall x \in \{0, 1\}^C,\ \forall v \in V$.

Proof. Let x and A be fixed binary $|C|$-vectors, and suppose there exists $v \in V$ such that $\Phi(h(v), x) < \Phi(\ell(v), x)$. Because $\Phi(\cdot, x)$ is binary, this implies that $\Phi(h(v), x) = 0$ and $\Phi(\ell(v), x) = 1$. Let $V_1 \subset V$ denote any set of vertices, including v^* but not including v, such that the vertices of V_1 form a path from

v^* to v (such a path exists due to Property 3). We may partition the vertices in V_1 into $V_1^H \cup V_1^L$, where $u \in V_1^H$ if u's high edge is on the path to v and $u \in V_1^L$ if u's low edge is on the path to v. Note that all descendants of v must be elements of $V \setminus (V_1 \cup \{v^*\})$ by Property 2. For each $c \in C$, define $\bar{a}_c$ as follows: If $c = c(v)$ or $c = c(u)$ for some $u \in V_1^L$, then $\bar{a}_c = 0$; if $c = c(u)$ for some $u \in V_1^H$, then $\bar{a}_c = 1$; otherwise, $\bar{a}_c = A_c x_c$. Using this solution, we see that $\phi(\bar{a} + e_{c(v)}) = 0$ and $\phi(\bar{a}) = 1$, contradicting the assumption that ϕ is nondecreasing. □

We may use the recursion (7) to establish a new form for the objective function in Equation (1). To this end, define $\alpha(v, x) = P[\Phi(v, x) = 1] = \mathbb{E}[\Phi(v, x)]$ for $v \in V$ and observe that $R(x) = \alpha(v^*, x)$. The following corollary follows from taking the expectation of both sides of the inequality in Lemma 1.

Corollary 1. $\alpha(h(v), x) \geq \alpha(\ell(v), x),\ \forall x \in \{0,1\}^C,\ \forall v \in V.$

Having established the relevant theory, we now state our MILP equivalent of Model (1). To this end, we associate flow variables with each edge in the BDD: For $u \in V$, define y_u and z_u, respectively, as the flow on $(u, h(u))$ and $(u, \ell(u))$. Flow along u's high edge corresponds to probability that component $c(u)$ functions while flow along u's low edge corresponds to probability that $c(u)$ fails. For $u \in V$, let b_u equal 1 if $u = v^*$ and let $b_u = 0$ otherwise. The following model sends one unit of flow from vertex v^* and attempts to maximize flow entering node s:

$$\max_{x \in X,\ y, z \geq 0} \sum_{v:h(v)=s} y_v, \tag{8a}$$

$$\text{s.t.} \sum_{v:h(v)=u} y_v + \sum_{v:\ell(v)=u} z_v + b_u = y_u + z_u,\ \forall u \in V, \qquad : \pi_u \tag{8b}$$

$$y_u \leq p_{c(u)} \left(\sum_{v:h(v)=u} y_v + \sum_{v:\ell(v)=u} z_v + b_u \right),\ \forall u \in V, \qquad : \beta_u \tag{8c}$$

$$y_u \leq p_{c(u)} x_{c(u)},\ \forall u \in V, \qquad : \delta_u \tag{8d}$$

Constraints (8b) enforce flow balance while Constraints (8c) bound the flow on each vertex u's high edge as a proportion of all flow incoming into u. Constraints of the form (8c) have been used previously in a reliability-based optimization problem [18], but to our knowledge they have never been applied to any system other than a basic series or parallel system. Constraints (8d) force each component $c \in C$ to act as if it were failed whenever c is not installed. We now establish the model's validity.

Theorem 1. For fixed $x \in X$, Model (8) has an optimal objective value of $R(x) = \alpha(v^*, x)$.

Proof. Associating dual variables π, β, and δ with Constraints (8b)–(8d), the LP dual of (8) is given as

$$\min_{\pi \text{ urs, } \delta,\beta \geq 0} \pi_{v^*} + p_{c(v^*)}\beta_{v^*} + \sum_{u \in V} p_{c(u)} x_{c(u)} \delta_u, \tag{9a}$$

$$\text{s.t. } \pi_u - \pi_{h(u)} + \beta_u - p_{c(h(u))}\beta_{h(u)} + \delta_u \geq 0, \ \forall u \in V, \qquad : y_u \tag{9b}$$

$$\pi_u - \pi_{\ell(u)} - p_{c(\ell(u))}\beta_{\ell(u)} \geq 0, \ \forall u \in V, \qquad : z_u \tag{9c}$$

where $\pi_f \equiv \beta_f \equiv \beta_s \equiv 0$, $\pi_s \equiv 1$, and (arbitrarily) $p_{c(f)} \equiv p_{c(s)} \equiv 1$ in order to ensure the dual constraints appear correctly for $u \in V$ such that $h(u) = s$ or $\ell(u) = f$. In Model (8), note that if $x_{c(u)} = 1$, then Constraint (8d) is implied by (8c); thus, we may fix $\delta_u = 0$ for all such u in Model (9). As a result, either $x_{c(u)} = 0$ or $\delta_u = 0$ for all $u \in V$, and we may simplify the objective of (9) as $\pi_{v^*} + p_{c(v^*)}\beta_{v^*}$.

We now prove existence of an optimal solution for Model (9) in which $\pi_u + p_{c(u)}\beta_u = \alpha(u, x), \ \forall u \in V$. We complete this proof inductively using $u = s$ and $u = f$ as base cases. If $u = s$, we see that $\pi_s + p_{c(s)}\beta_s = 1 \equiv \alpha(s, x)$ because $\pi_s \equiv 0$ and $\beta_s \equiv 0$. If $u = f$, then $\pi_f + p_{c(f)}\beta_f = 0 \equiv \alpha(f, x)$ because $\pi_f \equiv \beta_f \equiv 0$. This completes the proof of the base case.

The inductive part of the proof will be completed by considering the vertices $u \in V$ in reverse-topological order ($v > u$ if v is topologically ordered after u), which is justified by Property 1. We show that, if π_v, β_v, and δ_v are fixed such that $\pi_v + p_{c(v)}\beta_v = \alpha(v, x)$ for all $v \in V, \ v > u$, then the resulting version of Model (9) has an optimal solution in which $\pi_u + p_{c(u)}\beta_u = \alpha(u, x)$. To this end, let u be any

vertex, and suppose π_v, β_v and δ_v are fixed for all $v > u$ such that $\pi_v + p_{c(v)}\beta_v = \alpha(v, x)$. We refer to Model (9) as the "level-u model" when variables are fixed in this manner. Because $h(u) > u$ and $\ell(u) > u$, Constraints (9b) and (9c) corresponding to vertex u in the level-u model reduce to

$$\pi_u + \beta_u + \delta_u \geq \alpha(h(u), x), \tag{10a}$$

$$\pi_u \geq \alpha(\ell(u), x). \tag{10b}$$

These constraints now amount to constant lower bounds on $\pi_u + \beta_u + \delta_u$ and π_u. The other constraints in which π_u, β_u, and δ_u appear may be summarized comprehensively as

$$\pi_u + p_{c(u)}\beta_u \leq \pi_v + \beta_v + \delta_v, \ \forall v \in V, h(v) = u, \tag{11a}$$

$$\pi_u + p_{c(u)}\beta_u \leq \pi_v, \ \forall v \in V, \ell(v) = u. \tag{11b}$$

From Systems (10) and (11), it follows that minimizing $\pi_u + p_{c(u)}\beta_u$ subject to Constraints (10) results in the least restriction imposed on $\pi_{v'}$ and $\beta_{v'}$, $v' < u$ and therefore admits an optimal solution to the level-u model because the objective coefficients on π_u, β_u and δ_u are zero. (If $u = v^*$, minimizing $\pi_u + p_{c(u)}\beta_u$ subject to (10) still admits an optimal solution because the objective function terms involving v^* are exactly $\pi_{v^*} + p_{c(v^*)}\beta_{v^*}$.) Thus, there exists an optimal solution to the level-u model such that π_u, β_u, and δ_u solve

$$\min\{\pi_u + p_{c(u)}\beta_u : \text{Constraints (10)}\}. \tag{12}$$

By Corollary 1, we see that $\alpha(h(u), x) \geq \alpha(\ell(u), x)$; hence, inspection of (12) reveals existence of a level-u optimal solution with $\pi_u = \alpha(\ell(u), x)$. We characterize the β- and δ-components of the solution to (12) in two cases: (i) $x_{c(u)} = 1$ and (ii) $x_{c(u)} = 0$. In case (i), $\delta_u = 0$ and it is therefore necessary to set $\beta_u = \alpha(h(u), x) - \alpha(\ell(u), x)$. In this case, $\pi_u + p_{c(u)}\beta_u = p_{c(u)}\alpha(h(u), x) + (1 - p_{c(u)})\alpha(\ell(u), x) \equiv$

$\alpha(u,x)$. In case (ii), we may complete the solution to (12)—at a reduced objective function value—via setting $\delta_u = \alpha(h(u),x) - \alpha(\ell(u),x)$; in this case, $\pi_u + p_{c(u)}\beta_u = \alpha(\ell(u),x) \equiv \alpha(u,x)$.

Applying the result inductively, we may construct an optimal solution to (9) in which $\pi_u + p_{c(u)}\beta_u = \alpha(u,x),\ \forall u \in V$; hence, the optimal objective value for (9) is $\alpha(v^*,x) \equiv R(x)$. The result follows by applying the strong duality relationship of Models (8) and (9). □

Theorem 1 establishes that MILP (8) is a valid formulation of (1). We now prove a result that provides additional context for the optimal y- and z-values.

Theorem 2. Assume $x \in X$ is fixed and let $V(1) = \{u \in V : x_{c(u)} = 1\}$ and $V(0) = V \setminus V(1)$. The resulting version of Model (8) has an optimal solution in which Constraints (8c) are tight for all $u \in V(1)$ and Constraints (8d) are tight for all $u \in V(0)$. Moreover, the solution satisfying these conditions is unique.

Proof. In the proof of Theorem 1, we established existence of an optimal solution to the dual Model (9) in which all constraints are tight and all variables equal zero with the possible exception of $\pi_u,\ u \in V$, $\beta_u,\ u \in V(1)$, and $\delta_u,\ u \in V(0)$. The KKT conditions for linear programs imply that any feasible solution to the primal Model (8) satisfying complementary slackness conditions with this dual solution must be optimal for Model (8). Under the dual solution found in the proof of Theorem 1, the complementary slackness conditions are a subset of the conditions

$$y_u = p_{c(u)} \left(\sum_{v:h(v)=u} y_v + \sum_{v:\ell(v)=u} z_v + b_u \right),\ \forall u \in V(1), \tag{13a}$$

$$y_u = p_{c(u)} x_{c(u)} = 0,\ \forall u \in V(0). \tag{13b}$$

Moreover, because (8c) dominates (8d) for $u \in V(1)$ and vice-versa for $u \in V(0)$, a primal solution (y,z) is feasible whenever (13) is satisfied along with nonnegativity and Constraints (8b); hence any nonnegative (y,z) satisfying (13) and (8b) must be optimal for (8).

We construct an equivalent system of $2|V|$ equations by subtracting (13a) from (8b) for each $u \in V(1)$ and (13b) from (8b) for each $u \in V(0)$. The resulting system is given as

$$y_u = \begin{cases} p_{c(u)} \left(\sum_{v:h(v)=u} y_v + \sum_{v:\ell(v)=u} z_v + b_u \right) & \text{if } u \in V(1) \\ 0 & \text{if } u \in V(0), \end{cases} \tag{14a}$$

$$z_u = \begin{cases} (1 - p_{c(u)}) \left(\sum_{v:h(v)=u} y_v + \sum_{v:\ell(v)=u} z_v + b_u \right) & \text{if } u \in V(1) \\ \sum_{v:h(v)=u} y_v + \sum_{v:\ell(v)=u} z_v + b_u & \text{if } u \in V(0). \end{cases} \tag{14b}$$

Note that, if $y_v \geq 0$ and $z_v \geq 0$ are respectively fixed for all $v \in V$ such that $h(v) = u$ and $\ell(v) = u$, then y_u and z_u have a unique, nonnegative solution under (14) because $0 \leq p_{c(u)} \leq 1$ and $b_u \geq 0$. Hence, we may fix the values of y_u and z_u for all $u \in V$ by considering the vertices in topological order (invoking Property 1). By construction, this solution satisfies the KKT conditions and is therefore optimal for Model (8).

To establish the uniqueness of the above solution, let $u_1 < u_2 < \ldots < u_{|V|}$ denote a topological ordering of the vertices. Arrange the $2|V|$ equations of System (14) by first taking (14a) and (14b) corresponding to u_1, then (14a) and (14b) corresponding to u_2, and so on, leaving (14a) and (14b) corresponding to $u_{|V|}$ as the last two equations. Similary, arrange the $2|V|$ variables in topological order (i.e., $y_{u_1}, z_{u_1}, y_{u_2}, z_{u_2}, \ldots, y_{u_{|V|}}, z_{u_{|V|}}$). The resulting system is lower triangular with nonzero diagonal coefficients and therefore the solution established above is unique. □

Theorem 2 establishes an important interpretation of y and z. For fixed $x \in X$ and a realization of the random vector A, one may trace a unique "BDD path" to determine the value of $\phi(A \cdot x)$. Under random A, the BDD path is random as well. The previous result establishes that, at optimality, y_u and z_u respectively represent the probability that edges $(u, h(u))$ and $(u, \ell(u))$ are traversed by the realized BDD path.

The results in this section imply that k/n-RAP can be linearized using the BDDs described in Section 1.3. We now illustrate for a special case of k/n-RAP.

Table 2: Data for symmetric k/n-RAP example system.

Subsystem, i	d_i	r_i	k_i	L_i	U_i
1	4	0.75	1	1	5
2	4.5	0.8	1	1	5
3	6	0.9	1	1	4

Example 1 (Series-Parallel Redundancy Allocation). Consider an instance of k/n-RAP with $B = 33$ and $m = 3$, and subsystems as defined in Table 2. Noting that $k_1 = k_2 = k_3 = 1$, the system consists of three serially connected parallel subsystems. The BDD for this instance is depicted in Figure 4(a). Each edge is labeled with the associated flow variable, and one unit of flow is depicted as entering the root vertex to represent supply. The remaining data for Model (8) are given by fixing X as in Equation (5) and associating reliabilities $p_c = r_i$ with each component $c = [i, j] \in C$.

Solving Model (8) over this system produces the optimal solution depicted in Figure 4(b). The optimal system reliability is $R = 0.93555$, obtained by installing three components in subsystem 1 and two components each in subsystems 2 and 3. In Figure 4(b), each edge is labeled with its corresponding flow, and vertices $v \in V$ are shaded if the corresponding component $[i, j] = c(v)$ is installed (i.e., if $x_{ij} = 1$).

The previous example illustrates some potential advantages of our approach. Adapting Model (8) to the series-parallel redundancy allocation produces a valid MILP that requires only $|C|$ binary variables (with the remaining variables continuous). Moreover, in adapting the model to a more general k/n-RAP model (i.e., by generating Model (8) from the BDD defined in Section 1.3), the number of binary variables remains constant. Furthermore, extending the model to the case where components in the same subsystem can have different reliabilities and/or costs is trivial via rewriting the budget constraint in Equation (5) as $\sum_{c \in C} d_c x_c \leq B$, and by defining $p_c = r_c,\ c \in C$, as the reliability of component c.

In the following example, obtained from [13, p. 192], we further illustrate the flexibility of our approach

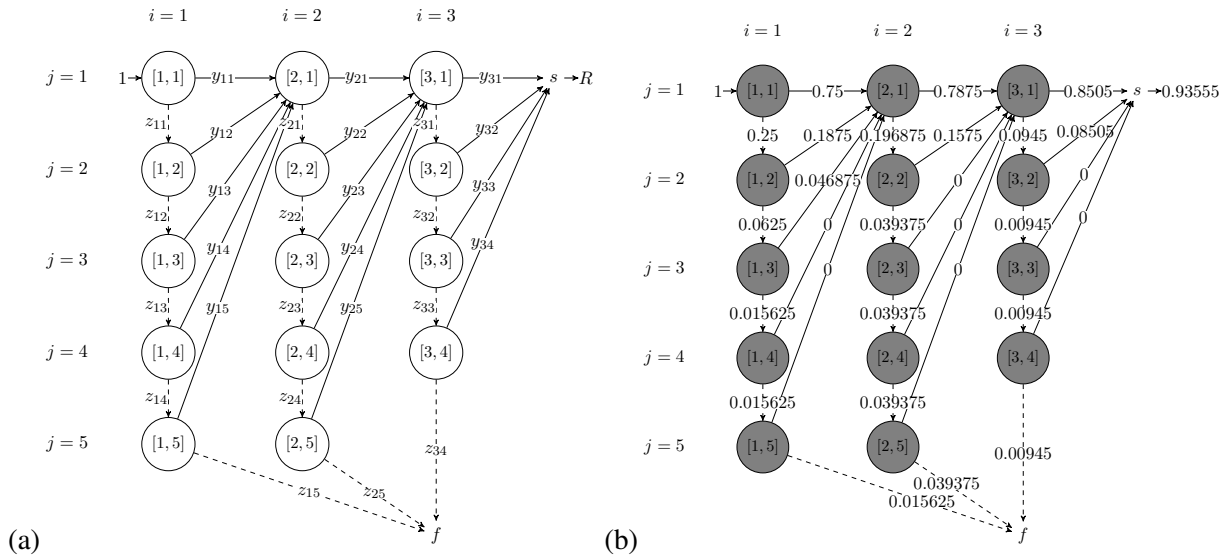

Figure 4: (a) BDD-based MILP for example instance. (b) Optimal solution for example instance.

by (i) considering a complex system and (ii) using an alternate definition of X. In this example, X defines a variable space in which ten components, each having different relabilities, are assigned to ten different positions in the system's block diagram.

Example 2 (Component Assignment in a Complex System [13, p. 192]). Consider the system depicted in Figure 5, in which labels $1, \ldots, 10$ represent positions in which components must be installed. Available for installation are components $1, \ldots, 10$, one of which must be installed in each position. Assuming the reliabilities of components $1, \ldots, 10$ are 0.98, 0.95, 0.94, 0.93, 0.88, 0.80, 0.77, 0.69, 0.66, and 0.64, to which position should each component be assigned to maximize system reliability? (Note: The system functions so long as there exists a path from left to right through Figure 5 using only functioning components.)

Define $C = \{[i, j] : i \in \{1, \ldots, 10\},\ j \in \{1, \ldots, 10\}\}$, where $x_{ij} = 1$ indicates that compo-

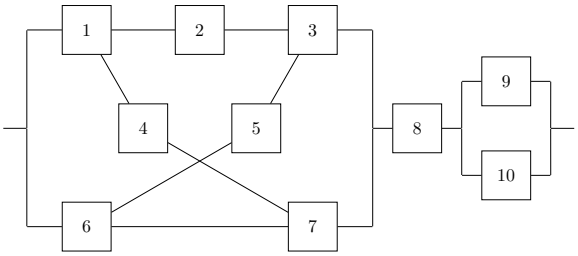

Figure 5: Block diagram for complex system assignment problem.

nent j is assigned to position i. In this problem, feasibility of x is defined by

$$X = \left\{ x \in \{0,1\}^{10\times 10} \;\middle|\; \begin{array}{l} \sum_{i=1}^{10} x_{ij} = 1,\ \forall j \in \{1,\ldots,10\} \\ \sum_{j=1}^{10} x_{ij} = 1,\ \forall i \in \{1,\ldots,10\} \end{array} \right\}. \tag{15}$$

We present in Figure 6 a BDD for this system. This BDD has a nested structure. Each position $i \in \{1,\ldots,10\}$ is represented by one or more "blocks" (labeled i^1, i^2, etc.) in Figure 6. These blocks are connected together by solid and dashed edges. One may verify that all paths ending at s (f) in the block-level BDD corresponding to a combination of events in which the system functions (fails).

Let K_i denote the number of blocks associated with position i. For each $i \in \{1,\ldots,10\}$ and $k \in \{1,\ldots,K_i\}$, the sub-BDD for block i^k is defined over vertices $V(i^k) = \{[i^k, j] : j \in \{1,\ldots,10\}\}$ using edges given as follows: For $j \in \{1,\ldots,9\}$, define $h([i^k,j]) = s$ and $\ell([i^k,j]) = [i^k, j+1]$; for $j = 10$, define $h([i^k,j]) = s$ and $\ell([i^k,j]) = f$.

Note that this sub-BDD resembles one of the parallel subsystems from Figure 4. In order for a component at position i to function, at least one of $[i,j],\ j \in \{1,\ldots,10\}$ must function. In this case, however, we happen to know (due to the definition of X) that only one such component is installed in position i.

The system-level BDD is now constructing by combining the sub-BDDs for each block. The

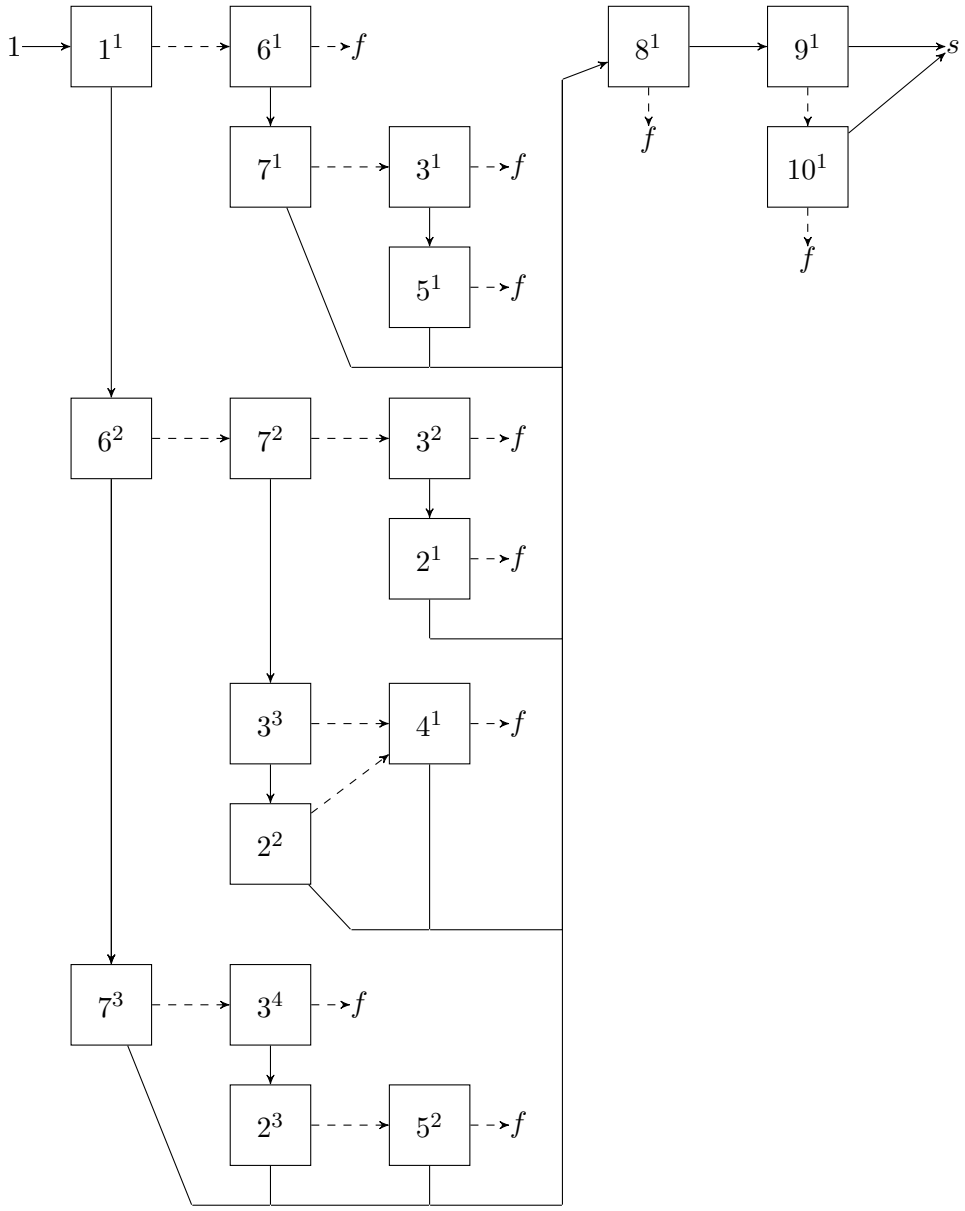

Figure 6: BDD for complex component assignment instance.

vertex set for this BDD is given by $V = \cup_{i\in\{1,\ldots,10\},k\in\{1,\ldots,K_i\}} V(i^k)$, where vertex $[i^k, j]$ corresponds to component $c([i^k, j]) \equiv [i, j] \in C$. Edges are then redirected using the edges in Figure 6 as a key. For each dashed edge in the diagram (say, beginning at block i^k and ending at block $\bar{i}^{\bar{k}}$), adjust edge destinations as follows: For $j \in V(i^k)$ such that $\ell([i^k, j]) = f$, set $\ell([i^k, j]) = \bar{v}^*$, where $\bar{v}^*$ is the root vertex of the sub-BDD corresponding to block $\bar{i}^{\bar{k}}$. If the dashed edge emanating block i^k terminates at f, no change is required.

For every solid edge in the diagram, beginning at block i^k and ending at block $\bar{i}^{\bar{k}}$, make a similar adjustment as follows: For $j \in V(i^k)$ such that $h([i^k, j]) = s$, set $h([i^k, j]) = \bar{v}^*$, where $\bar{v}^*$ is the root vertex of the sub-BDD corresponding to block $\bar{i}^{\bar{k}}$. If the solid edge emanating block i^k terminates at s, no change is required.

This procedure yields a BDD representation of the system described in this example, thereby enabling the use of Model (8). Solving the resulting model in CPLEX yields an optimal solution of $x_{18} = x_{27} = x_{36} = x_{49} = x_{5,10} = x_{61} = x_{73} = x_{82} = x_{94} = x_{10,5} = 1$ (and all other x-variables equal to zero), with a reliability of 0.935775.

Our solution to Example 2, obtained by solving Model (8) in CPLEX, matches the heuristic solution in [13], thus verifying our approach. Furthermore, because CPLEX solved the model within seconds, we are encouraged about the possibility of solving larger instances involving complex systems.

3 Strengthening the Symmetric Model

Preliminary investigation indicates that the continuous relaxations of Model (8) can be weak. In this section, we demonstrate how one may exploit subsystem symmetry within an instance of k/n-RAP to generate a tightened formulation of the BDD-based MILP. Recall that $C_i = \{[i, j] : j \in \{1, \ldots, U_i\}\}$ represents the set of components that can potentially be included in subsystem $i \in \{1, \ldots, m\}$. In the symmetric version

of k/n-RAP, the cost d_c and reliability p_c of a component $c \in C_i$ are replaced by d_i and r_i, which are common across all components in subsystem i. Because all components $[i,j] \in C_i$ are equivalent, we may break symmetry by adding constraints

$$x_{ij} \geq x_{i,j+1},\ \forall i \in \{1,\ldots,m\},\ j \in \{1,\ldots,U_i - 1\}, \tag{16}$$

We define $\bar{X}$ as the set that results from adding Constraints (16) to X. Thus, for symmetric k/n-RAP, Model (8) remains valid upon replacing X with $\bar{X}$. Under Constraints (16), we may strengthen Constraints (8d) as

$$y_u \leq Q_i b^{-1}(j - t + 1, j, r_i) x_{ij},\ \forall u = [i,j]^t \in V, \tag{17}$$

where $Q_i = \prod_{i'=1}^{i-1} \left[\sum_{k=k_{i'}}^{U_i} b(k, U_i, r_i)\right]$. In (17), the coefficient on x_{ij} is the product of two factors: (i) Q_i, an upper bound on the combined reliability of subsystems $1,\ldots,i-1$, and (ii) the (negative binomial) probability that, when considering components $[i,1],[i,2],\ldots,[i,j]$ in order, component $[i,j]$ is the $(j - t + 1)$-th functioning component. Note that each of these components must be installed if $[i,j]$ is installed, by Constraints (16). We establish the validity of (17) after proving the following result.

Theorem 3. Let $x = \hat{x} \in \bar{X}$ be fixed and define $U_i^* \geq 1,\ i \in \{1,\ldots,m\}$, as the largest value of j such that $\hat{x}_{ij} = 1$ (i.e., so that $\hat{x}_{i1} = \hat{x}_{i2} = \cdots = \hat{x}_{i,U_i^*} = 1$ and $\hat{x}_{i,U_i^*+1} = \hat{x}_{i,U_i^*+2} = \cdots = \hat{x}_{i,U_i} = 0$). There exists an optimal solution to the resulting version of Model (8) in which

$$y_u = \begin{cases} b^{-1}(j - t + 1, j, r_i)(y_{v_i^*} + z_{v_i^*}), & u = [i,j]^t \in V,\ 1 \leq j \leq U_i^* \\ 0, & u = [i,j]^t \in V,\ U_i^* < j \leq U_i, \end{cases} \tag{18a}$$

$$z_u = \begin{cases} b^{-1}(t, j, 1 - r_i)(y_{v_i^*} + z_{v_i^*}), & u = [i,j]^t \in V,\ 1 \leq j \leq U_i^* \\ b(t - j + U_i^*, U_i^*, 1 - r_i)(y_{v_i^*} + z_{v_i^*}), & u = [i,j]^t \in V,\ U_i^* < j \leq U_i, \end{cases} \tag{18b}$$

where $b(k, K, p) \equiv 0$ if $k < 0$.

Proof. Theorem 2 implies the existence of an optimal solution in which Constraints (8c) are tight for all $u = [i, j]^t \in V,\ j \leq U_i^*$, and Constraints (8d) are tight for all $u = [i, j]^t \in V,\ U_i^* > U_i$. We establish, by induction on j for each $i \in \{1, \ldots, m\}$, that this solution satisfies Conditions (18).

The base case establishes the result for all vertices $u = [i, j]^t \in V_i$ with $j = 1$. Note that this case corresponds to the unique vertex $v_i^* = [i, 1]^1$. Because $\hat{x}_{v_i^*} = 1$, Theorem 2 implies that $y_{v_i^*}$ is equal to r_i multiplied by the flow into vertex v_i^* which, by Constraint (8b), is equal to $y_{v_i^*} + z_{v_i^*}$. Thus,

$$y_{v_i^*} = r_i(y_{v_i^*} + z_{v_i^*}) = b^{-1}(1, 1, r_i)(y_{v_i^*} + z_{v_i^*}), \tag{19}$$

which equals to $b^{-1}(j - t + 1, j, r_i)(y_{v_i^*} + z_{v_i^*})$ for $j = t = 1$. Moreover, we have that

$$z_{v_i^*} = \frac{1 - r_i}{r_i} y_{v_i^*} = (1 - r_i)(y_{v_i^*} + z_{v_i^*}) = b^{-1}(1, 1, 1 - r_i)(y_{v_i^*} + z_{v_i^*}), \tag{20}$$

which equals to $b^{-1}(t, j, 1 - r_i)(y_{v_i^*} + z_{v_i^*})$ when $j = t = 1$. This completes the proof of the base case.

For the inductive step, we assume (18a) and (18b) hold for all $u = [i, j]^t \in V_i$ with $j < j'$, where $j' \in \{2, \ldots, U_i\}$. We then use this assumption to establish that (18a) and (18b) hold for all $u = [i, j]^t \in V_i$ such that $j = j'$. We prove this step in two cases: (i) $j' \leq U_i^*$ and (ii) $U_i^* < j' \leq U_i$. In the each of these cases, the proof involves adding binomial and negative binomial probabilities. The identities

$$b(k, K, p) = b^{-1}(k, K, p) + b^{-1}(K - k, K, 1 - p), \quad k \in \{0, \ldots, K\},\ 0 \leq p \leq 1, \tag{21a}$$

$$b^{-1}(k, K, p) = pb(k - 1, K - 1, p), \quad k \in \{1, \ldots, K\},\ 0 \leq p \leq 1, \tag{21b}$$

are used without proof. (These identities are direct from the formulas for b^{-1} and b.)

In case (i), $\hat{x}_{ij'} = 1$ and by Theorem 2, Constraint (8c) must be tight for all $u = [i, j]^t$ with $j = j'$. Let

$u = [i, j]^t$ be a vertex with $j = j'$. Vertex u's incoming edges can be classified into three subcases:

(a) If $t = 1$, the only incoming edge is a high edge from vertex $[i, j' - 1]^t$.

(b) If $t = j'$, the only incoming edge is a low edge from vertex $[i, j' - 1]^{t-1}$.

(c) If $1 < t < j'$, there are two incoming edges, a high edge from vertex $[i, j' - 1]^t$ and a low edge from vertex $[i, j' - 1]^{t-1}$.

In subcase (i:a), Theorem 2 implies for $u = [i, j']^1$ and $v = [i, j' - 1]^1$ that

$$y_u = r_i y_v = r_i b^{-1}(j' - 1, j' - 1, r_i)(y_{v_i^*} + z_{v_i^*}) = b^{-1}(j', j', r_i)(y_{v_i^*} + z_{v_i^*}), \tag{22}$$

which equals to $b^{-1}(j' - t + 1, j', r_i)(y_{v_i^*} + z_{v_i^*})$ because $t = 1$. Similarly,

$$z_u = (1 - r_i)y_v = r_i^{j'-1}(1 - r_i)(y_{v_i^*} + z_{v_i^*}) = b^{-1}(1, j', 1 - r_i)(y_{v_i^*} + z_{v_i^*}), \tag{23}$$

which equals to $b^{-1}(t, j', 1 - r_i)(y_{v_i^*} + z_{v_i^*})$ because $t = 1$.

In subcase (i:b), Theorem 2 implies for $u = [i, j']^{j'}$ and $v = [i, j' - 1]^{j'-1}$ that

$$y_u = r_i z_v = r_i b^{-1}(j' - 1, j' - 1, 1 - r_i)(y_{v_i^*} + z_{v_i^*}) = b^{-1}(1, j', r_i)(y_{v_i^*} + z_{v_i^*}), \tag{24}$$

which is equal to $b^{-1}(j' - t + 1, j', r_i)$ because $t = j'$. Similarly,

$$z_u = (1 - r_i)z_v = (1 - r_i)^{j'}(y_{v_i^*} + z_{v_i^*}) = b^{-1}(j', j', 1 - r_i)(y_{v_i^*} + z_{v_i^*}), \tag{25}$$

which equals $b^{-1}(t, j', 1 - r_i)(y_{v_i^*} + z_{v_i^*})$ because $t = j'$.

In subcase (i:c), we have for $u = [i, j']^t$, $v = [i, j' - 1]^t$, and $v' = [i, j' - 1]^{t-1}$ that

$$y_u = r_i(y_v + z_{v'}), \tag{26a}$$

$$= r_i \left[b^{-1}(j' - t, j' - 1, r_i) + b^{-1}(t - 1, j' - 1, 1 - r_i) \right] (y_{v_i^*} + z_{v_i^*}), \tag{26b}$$

$$= b^{-1}(j' - t + 1, j', r_i)(y_{v_i^*} + z_{v_i^*}), \tag{26c}$$

where (26c) follows due to successive application of identities (21a) and (21b). This establishes the result for y_u. Similarly,

$$z_u = (1 - r_i)(y_v + z_{v'}), \tag{27a}$$

$$= (1 - r_i) \left[b^{-1}(j' - t, j' - 1, r_i) + b^{-1}(t - 1, j' - 1, 1 - r_i) \right] (y_{v_i^*} + z_{v_i^*}), \tag{27b}$$

$$= b^{-1}(t, j', 1 - r_i)(y_{v_i^*} + z_{v_i^*}), \tag{27c}$$

establishing the result for z_u.

In case (ii), Theorem 2 implies $y_v = 0$ for $v = [i, j']^t$ when $j' > U_i^*$. Thus, z_v carries all of the flow entering vertex v. To prove case (ii), we need only establish that the flow entering vertex v is $b(t - j' + U_i^*, U_i^*, 1 - r_i)$. To this end, we consider subcases (a) through (c) as in case (i). In subcase (ii:a), we have for $u = [i, j']^1$ and $v = [i, j' - 1]^1$ that

$$z_u = y_v, \tag{28a}$$

$$= \begin{cases} b^{-1}(U_i^*, U_i^*, r_i)(y_{v_i^*} + z_{v_i^*}), & j' = U_i^* + 1 \\ 0, & j' > U_i^* + 1, \end{cases} \tag{28b}$$

$$= b(1 - j' + U_i^*, U_i^*, 1 - r_i)(y_{v_i^*} + z_{v_i^*}),\ j \geq U_i^* + 1, \tag{28c}$$

which establishes the result because $t = 1$.

In subcase (ii:b), we have for $u = [i, j']^{j'}$ and $v = [i, j' - 1]^{j'-1}$ that

$$z_u = z_v, \tag{29a}$$

$$= \begin{cases} b^{-1}(U_i^*, U_i^*, 1 - r_i)(y_{v_i^*} + z_{v_i^*}), & j' = U_i^* + 1 \\ b(U_i^*, U_i^*, 1 - r_i)(y_{v_i^*} + z_{v_i^*}), & j' > U_i^* + 1, \end{cases} \tag{29b}$$

$$= b(U_i^*, U_i^*, 1 - r_i)(y_{v_i^*} + z_{v_i^*}),\ j \geq U_i^* + 1, \tag{29c}$$

which establishes the result because $t = j'$.

In subcase (ii:c), Theorem 2 implies for $u = [i, j']^t$, $v = [i, j' - 1]^t$, and $v' = [i, j' - 1]^{t-1}$ that

$$z_u = y_v + z_{v'}, \tag{30a}$$

$$= \begin{cases} \left[b^{-1}(U_i^* - t + 1, U_i^*, r_i) + b^{-1}(t - 1, U_i^*, 1 - r_i)\right](y_{v_i^*} + z_{v_i^*}), & j' = U_i^* + 1 \\ b(t - j' + U_i^*, U_i^*, 1 - r_i)(y_{v_i^*} + z_{v_i^*}), & j' > U_i^* + 1, \end{cases} \tag{30b}$$

$$= \begin{cases} b(t - 1, U_i^*, 1 - r_i)(y_{v_i^*} + z_{v_i^*}), & j' = U_i^* + 1 \\ b(t - j' + U_i^*, U_i^*, 1 - r_i)(y_{v_i^*} + z_{v_i^*}), & j' > U_i^* + 1, \end{cases} \tag{30c}$$

$$= b(t - j' + U_i^*, U_i^*, 1 - r_i)(y_{v_i^*} + z_{v_i^*}),\ j \geq U_i^* + 1, \tag{30d}$$

where (30c) follows due to (21a). This establishes the result for subcase (ii:c) and completes the proof. □

Corollary 2. Consider an instance of the Symmetric k/n-RAP defined by subsystems $i \in \{1, \ldots, m\}$, components $C = \bigcup_{i=1}^m C_i$, vertices $V = \bigcup_{i=1}^m V_i$, high edges $h(v),\ v \in V$, and low edges $\ell(v),\ v \in V$, as defined in Section 1.3. There exists an optimal solution to the version of Model (8) in which $x \in \bar{X}$ and inequalities (17) are satisfied for all $v \in V$.

Proof. Due to Theorem 3, there exists an optimal solution to this version of Model (8) in which $x \in \bar{X}$,

and y and z satisfy Equations (18a) and (18b) with respect to x. Because $x_{i1} = x_{i2} = \cdots = x_{i,U_i^*} = 1$ and $x_{i,U_i^*+1} = x_{i,U_i^*+2} = \cdots = x_{i,U_i} = 0$, it follows that $y_v = b^{-1}(j - t + 1, j, r_i)(y_{v_i^*} + z_{v_i^*})x_{ij},\ \forall v = [i, j]^t \in V$. Inequality (17) is therefore valid because Q_i is an upper bound on $(y_{v_i^*} + z_{v_i^*})$. □

In the following section, we present computational results that demonstrate that (17) significantly strengthens the symmetric k/n-RAP formulation and increases the size of problems that can be solved.

4 Computational Results

In this section, we present computational results for k/n-RAP. We perform two sets of experiments. In the first set, we solve asymmetric k/n-RAP instances in which each subsystem's components are not assumed to be identical. In the second set of experiments, we assume symmetric k/n-RAP instances (i.e., in which the components in each subsystem are identical). The improved symmetric k/n-RAP formulation, which is capable of solving much larger instances, may lead to ideas for developing improved solution methods for more general redundancy allocation problems.

4.1 Asymmetric k/n Computational Results

In what follows, we describe the procedure used to generate asymmetric instances. We specify the following data for each instance: m, the number of subsystems; and B, the budget for installing components. Due to the way in which component costs are generated (see below), the expected cost of installing exactly one component (chosen at random) in each subsystem is $3m$; we therefore consider values of B such that $B \geq 3m$ (for $k = 1$) and larger values of B for $k > 1$.

For each subsystem $i \in \{1, \ldots, m\}$, we set $L_i = 1$ and $U_i = 7$. We generate the BDD for each instance as described in Section 1.3. For $i \in \{1, \ldots, m\}$, costs $d_c,\ c \in C_i$, are generated uniformly and independently in the interval $(1 + 2(i - 1)/(m - 1), 3 + 2(i - 1)/(m - 1))$. Reliabilities $p_c, c \in C_i$, are generated uniformly and independently in the interval $(0.75+0.1(i-1)/(m-1), 0.85+0.1(i-1)/(m-1))$.

In each experiment set, we report results averaged across five instances generated as described in the previous paragraph. We report $\mathbf{T}$, $\mathbf{R}$, and $\mathbf{U}$, where $\mathbf{T}$ is the CPLEX solution time in seconds (or percent optimality gap after one hour), $\mathbf{R}$ is the best known system reliability, and $\mathbf{U}$ is the number of instances (out of five) that did not solve within an hour. In our experiments, we use the CPLEX 12.5 solver accessed via the Concert Technology C++ libraries. A Dell PowerEdge R900 machine with 4 Intel Xeon X7350 Processor cores (with 2.93 GHz and 32 GB of RAM) was used to perform our experiments.

Table 3 summarizes the result of three sets of experiments. In the first two sets (labeled $k = 1$ and $k = 3$), the parameters k_i for each subsystem $i \in \{1, \ldots, m\}$ take on the same value (either 1 or 3). In the last set (labeled $k \in \{1, 2, 3\}$), the value of k_i is given as $(i \bmod 3)$ for $i \in \{1, \ldots, m\}$, so that some subsystems require at least three components to function while others require only one or two.

The results in Table 3 suggest that the BDD-based MILP regularly produces optimal or near optimal solutions in the $k = 1$ and $k \in \{1, 2, 3\}$ sets when m is small or B is large, although the latter case may be misleading due to the fact that the resulting reliabilities are close to 1. In the remaining test cases (including all experiments in the $k = 3$ set), significant optimality gaps are common after one hour. This suggests that the LP relaxation of Model (8) can be weak in general (i.e., in the absence of subsystem symmetry). In the following section, we demonstrate that the tightened formulation of the symmetric k/n-RAP model described in Section 3 can be solved quite efficiently. Due to the improvement that results from the tightened symmetric model (as illustrated in the following section), we are optimistic about the possibility of solving asymmetric k/n-RAP instances more efficiently via pursuing tightened formulations of a more general version of Model (8).

4.2 Symmetric k/n Computational Results

Symmetric instances are generated exactly as asymmetric instances, with the following exception: After the values of d_c and p_c are randomly generated for the first component in subsystem i, the values of $d_{c'}$ and

Table 3: Asymmetric k/n-RAP computational results.

k	m	B	**T**	**U**	**R**
1	6	18	1.45	0	0.6219
		30	98.9	0	0.9012
		42	1230	0	0.9775
		54	1.18%	4	0.9944
	8	24	17.1	0	0.5103
		40	5.94%	5	0.8714
		56	2.73%	5	0.9689
		72	0.69%	5	0.9923
	10	30	451	0	0.4365
		50	16.4%	5	0.8393
		70	3.86%	5	0.9614
		90	0.97%	5	0.9901
3	6	54	72.40%	5	0.2655
		66	18.7%	5	0.6048
		78	2.36%	3	0.8201
		90	0.37%	2	0.9202
	8	72	333%	5	0.1668
		88	66.1%	5	0.5148
		104	18.9%	5	0.7692
		120	4.38%	4	0.8986
	10	90	710%	5	0.1018
		110	108%	5	0.4295
		130	32.4%	5	0.7068
		150	9.57%	5	0.8695
1/2/3	6	36	121	0	0.3556
		48	727	0	0.7286
		60	1090	0	0.9117
		72	779	0	0.9700
	8	48	25.6%	5	0.4133
		64	19.4%	5	0.7593
		80	4.82%	5	0.9252
		96	1.40%	5	0.9731
	10	60	1290%	5	0.3040
		80	34.6%	5	0.7073
		100	9.18%	5	0.8985
		120	2.54%	5	0.9652

$p_{c'},\ c' \in C_i \setminus \{c\}$, are fixed equal to d_c and p_c, respectively. As before, the reported values $\mathbf{T}$ (solution time in seconds, or optimality gap percentage after one hour) and $\mathbf{R}$ (best known reliability) are averaged across five randomly generated instances. Values $\mathbf{U}$ represent the number of instances (out of five) that fail to solve to optimality within one hour. Results are reported for each profile using two formulations, the general formulation (without (16)–(17)) and the tightened formulation (with (16)–(17)).

The results in Table 4 suggest that behavior of the general BDD-based MILP (first column of results) is similar with respect to asymmetric and symmetric instances. CPLEX produces an optimal solution within one hour using this formulation in only three instances, which all correspond to the profile $k = 3$, $m = 6$, $B = 90$. The tightened formulation described in Section 3 (second column of results) exhibits much faster convergence. CPLEX is unable to identify an optimal solution in only one profile ($k = 3$, $m = 25$, $B = 275$) in which the optimal reliability is apparently much smaller than what would be encountered in practice.

Interestingly, the general formulation consistently produces high quality solutions as evidenced by the fact that the $\mathbf{R}$-columns are identical within a tolerance of 0.00004. However, the general formulation struggles to produce quality bounds. For example, in the profile corresponding to $k = 1$, $m = 66$, $B = 42$, CPLEX is unable to prove an upper bound within 7% in one hour even though the *a priori* upper bound $R = 1$ is within 9%.

5 Conclusions and Future Work

In this paper, we present a general mixed integer linear programming framework for modeling combinatorial optimization problems in reliability. We prove that the formulation is valid for a very general class of redundancy allocation problems that assumes availability of a BDD representation of the system structure function. Our approach provides a flexible exact approach for solving problems of this type. Computational results indicate the relaxations of the general formulation can be weak; however, we demonstrate that it is possible to strengthen this formulation significantly for structured problem instances. For the classical

Table 4: Symmetric k/n-RAP computational results.

			Without (16)–(17)			With (16)–(17)		
k	m	B	**T**	**U**	**R**	**T**	**U**	**R**
1	6	30	11.3%	5	0.7257	0.064	0	0.7257
		42	7.09%	5	0.9182	0.063	0	0.9182
		54	2.00%	5	0.9778	0.049	0	0.9778
	8	40	47.1%	5	0.6559	0.112	0	0.6559
		56	10.2%	5	0.9062	0.092	0	0.9062
		72	2.66%	5	0.9738	0.066	0	0.9738
	10	50	65.5%	5	0.5998	0.149	0	0.5998
		70	13.1%	5	0.8841	0.108	0	0.8841
		90	0.036%	5	0.9650	0.096	0	0.9650
	25	125	234%	5	0.3041	5.17	0	0.3041
		175	34.5%	5	0.7451	0.938	0	0.7451
		225	8.12%	5	0.9251	0.493	0	0.9251
3	6	66	127%	5	0.3906	0.134	0	0.3906
		78	33.7%	5	0.6887	0.094	0	0.6887
		90	7.71%	2	0.8634	0.083	0	0.8634
	8	88	352%	5	0.2034	0.296	0	0.2034
		104	67.3%	5	0.5543	0.243	0	0.5543
		120	9.01%	5	0.8010	0.184	0	0.8010
	10	110	213%	5	0.2131	0.536	0	0.2131
		130	34.8%	5	0.5714	0.325	0	0.5714
		150	19.0%	5	0.8076	0.298	0	0.8076
	25	275	876%	5	0.0111	87.8%	5	0.0111
		325	393%	5	0.1920	65.3	0	0.1921
		375	72.9%	5	0.5495	5.19	0	0.5495
1/2/3	6	48	65.4%	5	0.4750	0.107	0	0.4750
		60	19.2%	5	0.8057	0.083	0	0.8057
		72	4.73%	5	0.9337	0.086	0	0.9337
	8	64	82.9%	5	0.5126	0.159	0	0.5126
		80	211%	5	0.8087	0.115	0	0.8087
		96	6.17%	5	0.9330	0.125	0	0.9330
	10	80	157%	5	0.3925	0.269	0	0.3925
		100	36.0%	5	0.7277	0.223	0	0.7277
		120	9.67%	5	0.9030	0.160	0	0.9030
	25	200	982%	5	0.1101	175	0	0.1101
		250	108%	5	0.4878	2.84	0	0.4878
		300	24.2%	5	0.7940	1.92	0	0.7940

"symmetric" version of the redundancy allocation problem, the improved formulation solves larger instances with relative ease. Moreover, employing the general formulation on symmetric instances with known optimal solutions suggests that partial exploration of the MILP's branch-and-bound tree is sufficient to produce near-optimal solutions most of time but unable to prove optimality.

Future work may seek to improve upon the BDD-based MILP formulations and solution approaches presented in this paper, either for the general redundancy allocation problem or for other special cases. Based on the results presented in Section 4, the ability to generate strong inequalities could significantly enhance the capability of the approach presented herein. Noting that a given system may have many possible BDD representations, an interesting follow-on investigation may seek to characterize efficient strategies for generating the BDD and solving the resulting MILP.

References

[1] S. B. Akers. Binary decision diagrams. *IEEE Transactions on Computers*, C-27:509–516, 1978.

[2] J. D. Andrews and S. J. Dunnett. Event-tree analysis using binary decision diagrams. *IEEE Transactions on Reliability*, 49:230–238, 2000.

[3] R. Bellman and S. Dreyfus. Dynamic programming and the reliability of multicomponent devices. *Operations Research*, 6:200–206, 1958.

[4] A. Billionnet. Redundancy allocation for series-parallel systems using integer linear programming. *IEEE Transactions on Reliability*, 57:507–516, 2008.

[5] R. E. Bryant. Symbolic boolean manipulation with ordered binary decision diagrams. *ACM Computing Surveys*, 24:293–318, 1992.

[6] D. Cao, A. Murat, and R. B. Chinnam. Efficient exact optimization of multi-objective redundancy allocation problems in series-parallel systems. *Reliability Engineering & System Safety*, 111:154–163, 2013.

[7] M.-S. Chern. On the computational complexity of reliability redundancy allocation in a series system. *Operations Research Letters*, 11:309–315, 1992.

[8] D. W. Coit and A. E. Smith. Reliability optimization of series-parallel systems using a genetic algorithm. *IEEE Transactions on Reliability*, 45:254–266, 1996.

[9] D. W. Coit and A. E. Smith. Solving the redundancy allocation problem using a combined neural network/genetic algorithm approach. *Computers & Operations Research*, 23:515–526, 1996.

[10] C. Ha and W. Kuo. Reliability redundancy allocation: An improved realization for nonconvex nonlinear programming problems. *European Journal of Operational Research*, 171:24–38, 2006.

[11] S. Kulturel-Konak, A. E. Smith, and D. W. Coit. Efficiently solving the redundancy allocation problem using tabu search. *IIE Transactions*, 35:515–526, 2003.

[12] W. Kuo and V. R. Prasad. An annotated overview of system-reliability optimization. *IEEE Transactions on Reliability*, 49:176–187, 2000.

[13] W. Kuo, V. R. Prasad, F. A. Tillman, and C.-L. Hwang. *Optimal Reliability Design: Fundamentals and Applications*. Cambridge University Press, 2001.

[14] W. Kuo and R. Wan. Recent advances in optimal reliability allocation. *Computational Intelligence in Reliability Engineering (SCI)*, 39:1–36, 2007.

[15] Y.-C. Liang and A. Smith. An ant colony optimization algorithm for the redundancy allocation problem (RAP). *IEEE Transactions on Reliability*, 53:417–423, 2004.

[16] K. B. Misra. Dynamic programming formulation of the redundancy allocation problem. *International Journal of Mathematical Education in Science and Technology*, 2:207–215, 1971.

[17] K. B. Misra and U. Sharma. An efficient algorithm to solve integer-programming problems arising in system-reliability design. *IEEE Transactions on Reliability*, 40:81–91, 1991.

[18] J. R. O'Hanley, M. P. Scaparra, and S. García. Probability chains: A general linearization technique for modeling reliability in facility location and related problems. *European Journal of Operational Research*, 230:63–75, 2013.

[19] V. R. Prasad and W. Kuo. Reliability optimization of coherent systems. *IEEE Transactions on Reliability*, 49:323–330, 2000.

[20] A. Yalaoui, E. Châtelet, and C. Chu. A new dynamic programming method for reliability & redundancy allocation in a parallel-series system. *IEEE Transactions on Reliability*, 54:254–261, 2005.

[21] W.-C. Yeh and T.-J. Hsieh. Solving reliability redundancy allocation problems using an artificial bee colony algorithm. *Computers & Operations Research*, 38:1465–1473, 2011.

[22] L. Zia and D. W. Coit. Redundancy allocation for series-parallel systems using a column generation approach. *IEEE Transactions on Reliability*, 59:706–717, 2010.