SCHEDULING DISASTER RECOVERY OPERATIONS IN INFORMATION TECHNOLOGY UNDER FISMA

by

Joshua Brashear

Submitted in Partial Fulfillment
of the Requirements for the Degree of
Master of Science in Mathematics
with Specialization in Industrial Mathematics

New Mexico Institute of Mining and Technology
Socorro, New Mexico
December, 2015

ABSTRACT

Disaster recovery planning for an information technology department can be treated as a resource constrained project scheduling problem. The focus of this paper is an event where information systems are disrupted en masse and must be restored based on the interdependencies between the components of the information systems. A finite number of technicians are available to be assigned to recovery tasks, where the set of systems a technician may restore is restricted by their authorizations which are treated as skills. The problem is observed from the context of an organization that follows the documentation requirements as well as cyber security and physical security requirements set by the United States Federal Information Security Management Act (FISMA). Data required for recovery planning will be available from security categorizations (SC), disaster recovery plans (DRP), information system contingency plans (ISCP), system security plans (SSP), and business impact analyses (BIA). This documentation, if well prepared, contains task precedence constraints and task durations that must be met when restoring information systems. Additional information required to augment the FISMA documentation to perform a large scale disaster recovery is discussed. This problem was expressed as a constraint program and a constraint solver was used to find feasible schedules for simulated data derived from PSPLib problems.

Keywords: project scheduling; disaster recovery; constraint programming; information technology; FISMA;

CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

AN - Activity Networks

AoN - Activity On Node

BCP - Business Continuity Planning

BIA - Business Impact Analysis

COOP - Continuity Of Operations Plan

CP - Constraint Program

DRP - Disaster Recovery Plan

DRSP - Disaster Response Scheduling Problem

FISMA - Federal Information Security Management Act

ISCP - Information System Contingency Plan

IT - Information Technology

MDP - Markovian Decision Process

MMPSP - Multi-Mode Project Scheduling Problem

MSPSP - Multi-Skill Project Scheduling Problem

MTD - Maximum Tolerable Downtime

NIST - National Institute Of Standards And Technology

RCPSP - Resource Constrained Project Scheduling Problem

RPO - Recovery Point Objective

RTO - Recovery Time Objective

SC - Security Categorization

SDRP - Sub Disaster Recovery Plans

SP - Special Publications

SSP - System Security Plans

This report is accepted on behalf of the faculty of the Institute by the following committee:

Brian Borchers, Advisor

I release this document to the New Mexico Institute of Mining and Technology.

Joshua Brashear Date

CHAPTER 1

INTRODUCTION

1.1 Problem Statement

The problem is to schedule the recovery of information systems after a disaster while respecting task precedence, information system security requirements in the form of task skills, and resource availability.

This paper concerns a common business scenario. An organization has a large information technology infrastructure. That organization creates readiness plans for disasters that may destroy or disrupt their information systems. The resulting plans protect their infrastructure and the business processes that depend upon it. These readiness plans determine how the organization will restore its information systems back to working condition if they are disrupted.

Disaster readiness plans require resources and staff to execute them. The amount of available resources and the number of available employees are finite. Information systems slated for recovery will be prioritized based on the interdependencies between information systems. A feasible schedule will assign the appropriate personnel to restore information systems in an order that respects precedence requirements, resource requirements, and the information system access restrictions which are implemented to meet U.S. federal compliance requirements for cyber security.

1.2 Background

Disaster preparedness is a best practice in both business and IT. Organizations prepare plans that aid in their readiness to respond to emergencies. These emergencies include natural disasters, social disasters, and breaches in cyber security. Disruptions cost the company the time to respond to the issue, and money caused by the downtime of a business process. Disaster planning reduces the impact of the disaster, where the impact is a measure of the detrimental effects of the disaster.

This paper will consider disaster recovery planning from the perspective of an organization that is required to meet cyber security objectives mandated by the United States federal government. These organizations are required to

perform contingency planning for information systems that will allow them to recover in the event of a disaster. Organizations that are affected by this mandate include federal executive agencies, and contractors to those agencies. Recovery plans and important system information are recorded in a collection of documents for legal compliance, termed a security package by Taylor [34]. Compliance with a law called FISMA will be discussed later. This paper is concerned with the event when computing resources are disrupted en masse and must be restored by the coordinated efforts of IT personnel.

In this paper, disaster recovery planning is considered from the understanding that many information systems may be disrupted simultaneously. To perform a successful large scale recovery using the contingency plans that are formed for individual information systems, all of the security packages for interdependent information systems must be consistent. A simple method for determining consistency with respect to temporal constraints will be given. System restorations can be prioritized by time and resource feasibility based on information system security controls. Multi-mode project scheduling is used to sequence events in the recovery in order to find a feasible IT disaster recovery plan involving multiple information systems. Suggestions for easing the difficulties in formulating consistent disaster recovery plans will be given. The data set used for this problem is simulated, using the types of data that can be found in security packages for FISMA compliance.

Under current FISMA regulations, affected IT departments are required to document interdependencies between information systems. The disaster plans created for FISMA are also formulated with the implicit assumption that either the necessary precedence requirements to recover an information system are satisfied, or that the problem is out of the current administrator's control and is left to other departments. This requires trust between departments, where administrators put part of the responsibility for the uptime of their own systems on parties who may not be aware that they provide essential services to those systems. There are also documentation and security issues that may arise, where systems may be declared to be circularly dependent on each other, or where the security controls in place may render an information system inaccessible to the administrators that are responsible for maintaining it.

This paper assumes that an organization implements security controls given in NIST Special Publications for compliance with FISMA. Disaster recovery plans are among those controls. By requiring these documents to follow known guidance, issues can be considered that may affect plans that are written for use in real organizations. This assumption also introduces some of the real world problems associated with compliance. In reality, the recovery plans that are designed for information systems are not ideal and can be problematic. A plan may contain incorrect data or be incomplete. One recovery plan can render another plan infeasible, or disagree with information that should be identical between two plans. The restoration priorities that are assigned to components of an information system can render the plan infeasible. The definitions that are used as the foundation of security controls such as disaster recovery policies can

be ambiguous or left open to interpretation by an organization. This project will propose a method for collecting data from security packages in an attempt to ensure that basic requirements for the feasibility of a disaster recovery plan are met. In this project, FISMA compliant disaster recovery plans will be treated as tasks. Those tasks will be assigned to personnel and scheduled to take place in an order that maintains temporal feasibility while attempting to minimize the impact of a disaster.

1.3 Disasters

The following sections present a brief overview of the connection between disaster recovery planning and the mathematical methods that support the creation of disaster recovery plans.

1.3.1 Disasters and Disaster Recovery

Much of the research in disaster recovery centers around natural disasters that affect the general population, such as fires and floods. A disaster response is an action that is taken to get an emergency situation under control. Paraphrasing the definition given by Altay et al., an emergency is any event, such as a disaster, where non-standard operating procedures must be implemented in order to recover from the effects of that event [1]. The goal of disaster response is an action to both to regain control and then to eliminate the adverse effects of the disaster. The effects of a disaster may be handled by dispatching emergency workers, or delivering food and water to survivors, etc. In IT, the effects of the disaster may be handled by dispatching computer technicians to restore the functionality of information systems. Applications of mathematics in disaster recovery include selecting plans to maximize some measure of general preparedness, scheduling workers to tend to hazards, and determining vehicle routes for supply delivery [13, 24, 28].

1.3.2 Disasters in Information Technology

Similar to the practices of local or state offices in performing natural disaster response planning, businesses may prepare to respond internally to disaster scenarios with Business Continuity Planning (BCP) and IT disaster recovery planning. The concerns of disaster recovery planning in IT are different from the concerns for natural disasters. In IT, recovery is focused on preserving the functionality of business processes and maintaining data integrity and availability. Disasters that may affect information systems include hazards that are unique to electronic devices. In addition to disruption through physical destruction, unique

hazards to information systems include things like malware or network based attacks.

Disaster plans in IT outline steps that need to be taken to restore information systems. An information system may be recovered by having a qualified professional visit the system and perform restorative actions on it. In the event of a disaster that requires an organization to relocate its resources, an organization may move to an alternate worksite and restore functionality there. In some cases, the information system is considered to be restored if the functionality that it provided has been resumed by a different system that may not resemble the original system.

Constraints may exist on the information systems to be restored, or on the personnel that are involved in the recovery. There are often dependencies between information systems such that one system cannot operate if another is disrupted. For example, the functions of servers and client systems often depend on other servers. The dependencies between information systems create precedence constraints that must be met during a recovery. There are also security considerations that affect the process of disaster recovery. It is often the case that the personnel that are authorized to maintain specific information systems are a small subset of the total personnel that perform information system administration tasks within the organization. Access restrictions to information systems mean that personnel within the organization are not completely interchangeable, and cannot be assigned to just any recovery task. The limited availability of personnel can affect the duration of the recovery process.

For examples of the scale of physical disasters that may result in disaster recovery plans being invoked, examples can be found in the damage to IT infrastructure caused by the 9/11 New York terrorist attacks and the 2005 hurricane Katrina. For an example of IT disasters that were not physically damaging, an example is the computer virus that was used to attack an upper estimate of thirty thousand computer systems at Saudi Aramco in 2012 [25]. The numbers of systems that may need to be recovered in large disaster can be in the thousands, and an appropriate number of personnel will be required to perform those tasks.

CHAPTER 2

REVIEW OF FEDERAL INFORMATION SYSTEMS

2.1 Laws and Risk Management Frameworks

In 2002, the E-Government act was passed in the United States, with a section titled the Federal Information Security Management Act (FISMA) [9]. FISMA creates minimum security requirements for information and computers in federal executive agencies and their contractors. The standards set by FISMA are published in a series of papers by the National Institute of Standards and Technology (NIST). These papers come in several categories, including NIST Federal Information Processing Standards (FIPS), and NIST series 800 Special Publications (SP 800). FIPS establish requirements that must be met by all federal executive agencies and contractors to those agencies. FIPS may also make other standards mandatory. For example, FIPS 200 also requires that security controls be implemented from SP 800-53 *Security and Privacy Controls for Federal Information Systems and Organizations* [20]. These minimum security requirements established by FISMA include activities such as planning disaster recovery strategies, documenting information system properties, and determining the relative importance of information systems. Other than SP 800-53, the series SP 800 papers are often flexible guides, rather than requirements, for implementing processes that satisfy FIPS requirements. They are flexible, because an organization may meet FISMA requirements in their own way rather than in the exact manner prescribed in the NIST SP guides.

United States executive agencies employ risk management frameworks to comply with FISMA. These include the NIST Risk Management Framework (NIST RMF), Department of Defense Risk Management Framework (DoD RMF), and the Office of the Director of National Intelligence's Intelligence Community Directive 503 (ICD 503) [34]. In following these frameworks, organizations produce policies. These policies enforce the implementation level details that affect business processes and security controls, which in turn affects disaster recovery planning and employee's access to information systems.

2.2 Federal Standards for United States Executive Agencies

Compliance with FISMA involves the creation of high level policies. Those policies must meet a set of general requirements. The content of a policy is not

fixed by FISMA or one of the risk management frameworks. Policies vary between organizations that are attempting to be compliant with the same requirements. This will lead to differences in the implementation level details that result from FISMA compliance. The fact that the implementation details will vary between organizations is important, because it means that organizations will use different approaches to security controls such as disaster planning. A disaster planning methodology for these organizations will either need to be designed for a specific organization as a one off type effort, or the methodology will need to be designed for use with a risk management framework. In this paper, the latter approach will be taken. This will have the effect that both the requirements and goals of recovery become more ambiguous. The methods in this paper are intended to apply to an organization that implements a compliance method that follows this list of NIST documents and FISMA related publications:

FIPS 199 Standards for security categorization of federal information and information systems

FIPS 200 Minimum Security Requirements for Federal Information and Information Systems

NIST SP 800-34 Contingency Planning Guide for Federal Information Systems

NIST SP 800-53 Security and Privacy Controls for Federal Information Systems and Organizations

NIST SP 800-18 Guide for Developing Security Plans for Federal Information Systems

FISMA Compliance Handbook A guide for managers implementing FISMA requirements

Information Systems are defined recursively as visualized in figure 2.1. This recursive property means that an information system may be composed of other information systems. Because of this, it is not obvious which information systems are singletons in the sense that they have no sub-components. This appears to have been defined this way for flexibility, since a single information system may be a composite of several discrete elements that may not function individually. A parent information system will have its own business impact analysis that has the potential to impose MTDs and RTOs on all of the child information systems of a large composite system. However, the actual performance time of the recovery will depend on the RTOs given in the BIAs of the child components.

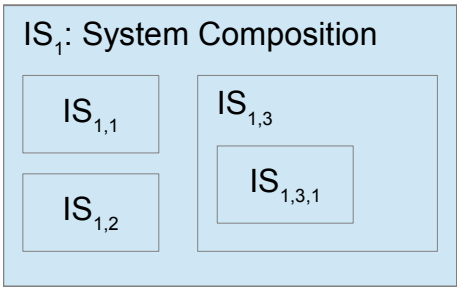


Figure 2.1: Nested System Composition

2.3 NIST Documents

Part of FISMA compliance requires that FIPS 199, FIPS 200, and NIST SP 800-53 be followed. Those documents will be discussed here, as well as others that are important. The output from these requirements is documented in a security package, whose relevant contents are diagrammed in figure 2.2.

Information systems are required by FIPS 199 to be categorized based on the potential security impact that a breach affecting that information system might have [31]. FIPS 199 defines three security objectives that frequently appear throughout other FISMA relevant NIST documents for information systems: confidentiality, integrity, and availability. Confidentiality protects data from unauthorized disclosure. Integrity ensures that data is not manipulated or falsified. Availability ensures that the information system is accessible.

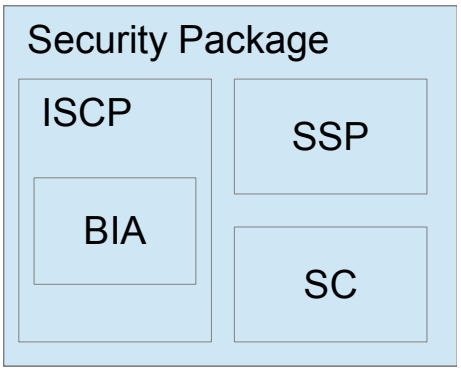


Figure 2.2: Composition of Security Package

When documenting the security objectives associated with an information system, each objective must be assigned a level. Levels describe the potential impact upon a security objective due to a breach in security on that information system. There are three levels that are required to be used, but an organization can add additional levels as needed. The required security category impact levels are low, moderate, and high.

A security categorization is made for an information system by associating impact levels with their respective security objectives. All information systems affected by FISMA must be given a security categorization. The security requirements for a system are determined by the highest level, or high water mark, that appears in the security categorization. One application of the security categorizations is to determine the priority for restoration of an information system. These security categorizations are documented as:

$$SC_{info\_system} = \{(confidentiality, level_1), (integrity, level_2), (availability, level_3)\}$$

A list of minimum security requirements in a number of categories related to information system management are established by FIPS 200 [20]. These security controls determine how the information system is accessed, the procedures for restoring that system after downtime, the data backup strategy applied to that system, the personnel that are allowed to access the system, and the organizational response to security incidents such as malware infections. The requirements in these categories are met by applying security controls found in SP 800-53 [14]. For example contingency planning for disaster recovery is a required security control, and it is outlined in SP 800-34 *Contingency Planning Guide for Federal Information Systems* [33].

The security controls that are implemented for an information system, must be documented in a system security plan (SSP). There should be an SSP for every information system. NIST SP 800-18 is a guide for making SSPs for information systems [32]. The SSP will be an important document for collecting the information needed to recover an information system. This document should have the system name, along with the contact information of the system owner and other important personnel. The list of services supported by the information system, and a list of expected user organizations should be given. A general description of the information system should be included in the SSP. But, most importantly, it is the document where important system interconnections are documented. This may not indicate which system is dependent on another, but it would be a good place to start for administrators to determine their information system interdependencies. Documenting precedences and dependencies is recommended by Taylor [34]. It will need to be assumed for this project, that the administrators for an organization have followed best practices and that precedence constraints are known.

The SSP will include a list of security controls for an information system. These security controls may be common controls or system specific controls. Common controls are security controls that can be reused and applied to many information systems. Since controls may include contingency plans and incident response plans, many systems may be covered by a common response plan in the event of a disaster. If there is a shortcoming in the security provided by common controls, this shortcoming may affect many systems. Some analysis may be needed to demonstrate that common controls, when taken together or across multiple systems, are free of negative interactions.

2.4 Contingency Planning

Disaster recovery planning for information systems is described in NIST SP 800-34 Contingency Planning Guide for Federal Information Systems [33]. This document describes several types of disaster response plans, including generic information system disaster recovery plans, called Information System Contingency Plans (ISCP). The other types of recovery plans described in the documents are business oriented Continuity of Operations Plans (COOP), and Disaster Recovery Plans (DRP). The size and effects of a disaster upon a business may vary, which leads to the implementation of contingency plans that are developed for different types of recovery operations. Preparation for server or client recovery is often referred to as disaster recovery planning in IT, which conflicts with the business process oriented use of DRP. The NIST documents refer to the IT oriented case as information system contingency planning (ISCP). DRP and COOP are more general, and will recover more than just IT infrastructure. DRP is an emergency response that requires relocation of resources to an alternate facility. COOP is an emergency response that might take place at an alternate facility. In the event of a disaster, COOP and DRP may implement ISCPs to restore

information systems. Recovery plans may be invoked as a response to an emergency. When a response plan is implemented, its goal should be to recover some information system from that disaster.

COOP
A continuity of operations plan (COOP) is designed to restore business or mission critical functions for an organization. COOP does not require that all systems be restored. It is only required that essential services be restored and maintained for up to 30 days.

DRP
A disaster recovery plan (DRP) is designed to restore the business functions housed in or provided by a facility. A DRP is used when the functions of a facility must be restored at an alternative location.

ISCP
An information system contingency plan (ISCP) is formed in order to respond to an information system disruption. These plans contain information that is used in the recovery of specific information systems.

This project is most concerned with finding a feasible order to execute ISCPs in order to recover many information systems, as well as assigning the resources required by the ISCPs. An ISCP is expected to contain essential information about an information system, such as time objectives, relevant recovery personnel, a BIA for that information system, and that information system's security categorization. The business impact analysis that is included in an ISCP for a computer system contains additional information that may not be included in the SSP.

2.4.1 Time Constraints for ISCPs

The information from the BIA may be used in constraints or objective functions when attempting to prioritize the restoration of information systems. The BIA contains: a list of business processes that depend on the information system, a list of resources that the system depends on, and a list of restoration time objectives. The following time objectives are appear in the document:

MTD
A maximum tolerable downtime (MTD) is an upper limit on the amount of time a business process can be offline. When this time limit is exceeded, it is expected that the system that has been disrupted will cause unacceptable harm to business operations. The business process may depend upon the functionality of information systems.

RPO

A recovery point objective (RPO) determines the maximum length of time between the performance of data backups. This is a way to determine a desired maximum amount of data that can be lost in a disaster. Ideally the amount of data lost would be zero, but there is a cost trade-off involved that becomes more expensive as the backup policy becomes more comprehensive.

RTO

A recovery time objective (RTO) is the estimated maximum allowable amount of time to complete a recovery plan for an information system once an ISCP is activated. It is usually expected that $RTO \leq MTD$.

A business process has an MTD, where the downtime of a business process depends on a set of information systems. In that set of information systems, each of those systems has an RTO. In some recovery planning methods a recovery time is included in the definition of the MTD, which includes tasks that must take place after the recovery. This additional recovery time may be called a Work Recovery Time (WRT), and is defined such that $RTO + WRT \leq MTD$. The NIST SP 800 documents do not require the use of WRTs in disaster recovery planning. An additional recovery time objective referred to as Recovery Time Actual (RTA) is a closer estimate of the actual time required to recover an information system. RTAs are not required by SP 800-34 to be included in an ISCP. However, SP 800-34 requires that a Testing Training and Exercise (TT&E) program is in place for ISCPs [33]. The TT&E program is used to ensure that employees are prepared to execute ISCPs for which they are responsible. This includes dry runs or even online tests of disaster recovery procedures. This document also indicates that during an actual recovery, event documentation needs to be kept detailing actions required to perform the recovery and any problems that come up. The event documentation is to be used to form reports that are used to update the ISCP. From the TT&E program, event documentation, or reports, it is expected that estimates can be made for RTA. In the event that an RTA can be derived, it would be preferable to use the RTA rather than an RTO which may be an untested upper estimate.

The ISCP assigns a team of personnel that will be responsible for carrying out the disaster recovery process for an information system. This ensures there are personnel assigned to the task of assessing the extent of the damage in the activation and notification phase of disaster recovery, which allows the assumption that it is known which systems have been disrupted. The ISCP is a plan for recovery that is independent of location. An ISCP may be invoked where the information system is located, or a replacement system may be set up at an alternate location as would occur during the execution of a DRP or COOP.

2.5 Business Impact Analysis

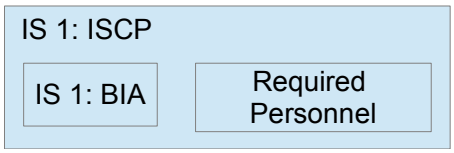


Figure 2.3: Information System Contingency Plan

The BIA indicates which business processes require which information systems. MTDs are given for the business processes associated with the information system under consideration. It also assigns RTOs to the components of the information system. Since the MTD is measured from the start of the disaster, if there are too many tasks with too few resources, the concept of MTDs could render all schedules infeasible from the outset. This document provides its own restoration priorities for the components. The priorities assigned to the information systems in the BIA are established based on the importance of the business processes that are supported by information systems, rather than on the order in which the information systems themselves can be restored. The components themselves will have dependencies on other information systems and it may not be possible to respect the restoration priorities given in the BIA. The BIA may contain information related to the cost of downtime for a business process. These costs may include items like penalties for the downtime of the information system exceeding the MTD of a business process, or costs that build up over time. However, just using the specification of the BIA, it is difficult to associate cost functions with specific information systems. If more specific cost information were required by internal policy, then it would be possible to form a cost based objective function for the recovery.

This document contains a list of resources that will be required to restore the information system. Examples of resources to include in the BIA are facilities, personnel, equipment, software, data files, system components, and vital records. The resources listed in the BIA include items that may not actually be required for restoration in many events. The only resources that matter for planning, are those that have a finite limit or must be shared between recovery tasks. Resources listed in the BIA may not be shared between multiple information systems and can be ignored. It will need to be determined and reported as part of the outage assessment, just which resource will be required for the execution of an ISCP. Some disasters may have the effect of physically destroying resources required for recovery tasks. In the event that equipment is destroyed during a disaster, the BIA also accounts for equipment replacement strategies. The estimated recovery time for a task may change depending on whether or not equipment has to be replaced, or borrowed, or serviced by a vendor. Which recovery time will be applied to an information system will need to be determined during the outage assessment.

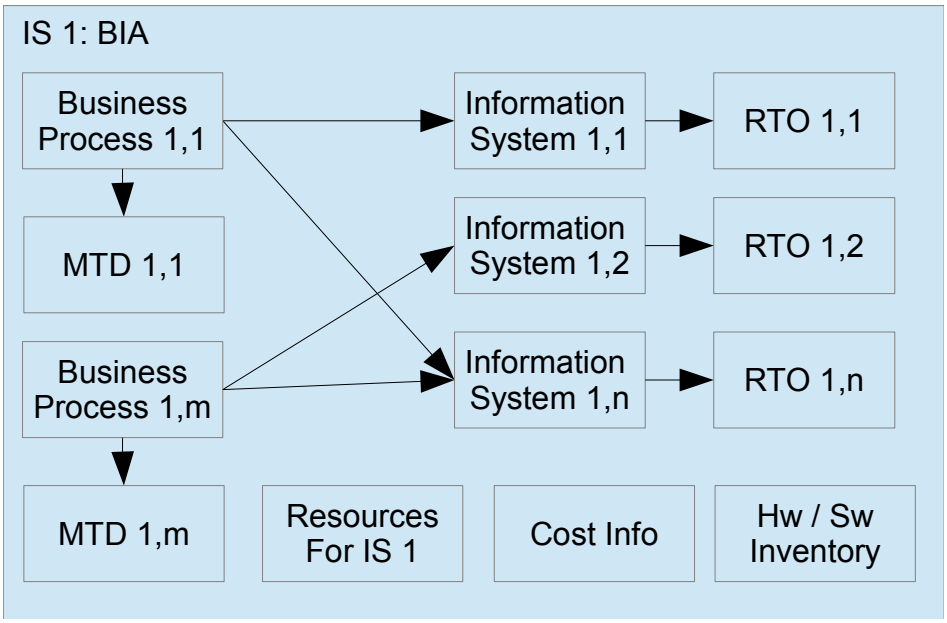


Figure 2.4: Business Impact Analysis

2.6 Relevance of FISMA to Scheduling Problems

FISMA provides a complete list of potential tasks that may be performed in the form of ISCPs. These come with estimated task durations and deadlines in the form of RTOs and RTAs. The documentation also keeps a list of resource demands for ISCPs. The BIA and SSP make loose references to data that could be used as precedence information. The shortcomings of poor precedence information is given in the chapter *Discussion*. During an outage assessment, a smaller list of tasks that must be performed in order to restore an organization are collected. With this information, the recovery can be modeled as a Multi-mode Resource Constrained Project Scheduling Problem.

CHAPTER 3

SCHEDULING AND CONSTRAINT PROGRAMMING REVIEW

3.1 Scheduling Problems

Scheduling problems are concerned with forming sequences of tasks or activities, denoted by an index $i \in N = \{1 \dots n\}$, that satisfy a set of temporal constraints. Each task i has a start time S_i and the task takes a duration p_i to complete at its completion time $C_i = S_i + p_i$. A schedule S is a set of fixed start times $\{S_j | \ j \in N\}$. A task j may have precedence constraints such that $S_j \geq S_i + p_i$ for some i, that require some tasks to occur in a specific order. A schedule is feasible if its start times satisfy the set of constraints. In some cases, just finding a feasible schedule may be enough to meet our needs. In other cases, feasible schedules are chosen to optimize an objective function. The times will be discretized so that a task has positive integer duration, and tasks are scheduled to occur at integer valued times. The range of values that the start time of a task may take on gives the task four properties that are important. The smallest start time of a task is called an early start time (est) and the associated completion time that is at $est_i + p_i$ is the early completion time (ect). The largest start time of a task is called a late start time (lst), and the associated completion time that is at $lst_i + p_i$ is the late completion time (lct). A start time must occur between its early and late start times, $est_i \leq S_i \leq lst_i$. Common temporal constraints for the scheduling problem are given in table 3.1

The makespan is the duration of a schedule. The start of a schedule is marked by a dummy activity $S_0 = 0$ and the makespan is marked by the starting time of a final dummy activity S_{n+1}. A common objective is to minimize the makespan of the schedule. If there is cost information available, another common objective is to minimize the cost associated with a schedule. A schedule that minimizes the makespan will not necessarily minimize the cost, and vice versa.

The precedence constraints of a schedule may be represented as an activity network (AN), such as an activity on node (AoN) network. These networks are reviewed in [8, 23]. Using an AoN network, the precedence constraints form a digraph $G(V, E)$. Set V represent the tasks $i \in \{0, \dots, n+1\}$, where tasks 0 and $n+1$ are the dummy tasks for the start and end of the project. Set E contains arcs that represent precedence constraints where the weights of the arcs indicate the duration of the preceding task. If i precedes j, then arc

$S_i + p_i \leq t$	Deadline constraint, i finished before t
$S_i \geq t$	Release constraint, i cannot start until t
$S_i + p_i \leq S_j$	Precedence constraint, i finished before j starts
$S_i + p_i \leq S_j$ *or* $S_j + p_j \leq S_i$	Disjunctive constraint, i and j cannot overlap in time

Table 3.1: Common Scheduling Constraints

$(i,j) \in E = \{(i,j) |\ S_j \geq S_i + p_i,\ i,j \in V\}$. The properties of activity networks make it possible to determine if a scheduling problem is feasible. A scheduling problem, without release dates or deadlines, is feasible if there are no cycles in the precedence graph. Precedence constraints are given for all initial and terminal activities, so that all tasks occur between $S_0 = 0$ and $S_{n+1} = max_{i \in V}\{S_i + p_i\}$. An upper bound h called a horizon can be given on the duration of the project, where $h = \sum_{i=0}^{n} p_i$ and $S_{n+1} \leq h$. Scheduling problems are reviewed in [5, 23].

3.1.1 Resource Constrained Project Scheduling Problems (RCPSP)

If a task requires resources to be performed, resource constraints will be applied to a task. This brings us to the topic of Resource Constrained Project Scheduling Problems (RCPSP). Resource Constrained Project Scheduling problems are treated in [2, 5, 23]. Each task i requires a constant amount of available resources. These resources are assumed to be renewable, so that when a task is completed, the amount of resources it required become available for use again. Adding resources to the scheduling problem means that a feasible schedule must be both time and resource feasible. A task i requires an amount r_{ik}, called a demand, of resource r_k. Given a set of renewable resources $\mathcal{R} = \{r_k | k \in \{1 \ldots K\}\}$, the total resource requirements for a resource r_k at any time has an upper limit which is called the resource capacity R_k.

An active set $\mathcal{A}(S,t)$ is a convenient way of denoting which tasks are in progress at a time period t for a given schedule S, where $\mathcal{A}(S,t) = \{i \in V | S_i \leq t < S_i + p_i\}$. The usage of resource r_k at time t is given by $r_k(S,t) = \sum_{i \in \mathcal{A}(S,t)} r_{ik}$. Renewable resources limit the number of tasks that can be performed at the same time. There is an optimal solution to the RCPSP if there are no cycles in the AoN precedence graph and there is a feasible solution.

The Resource Constrained Project Scheduling Problem with minimum

makespan objective can be stated as:

$$\begin{aligned} & Min\ S_{n+1} \\ subject\ to\ & r_k(S,t) \leq R_k\ , k \in \{1 \dots K\}, t \in [0,h) \\ & S_i + p_i \leq S_j\ , for\ (i,j) \in E \\ & S_i \geq 0\ , i \in \{1 \dots n+1\} \\ & S_0 = 0 \end{aligned}$$

$r_k(S,t) \leq R_k$	Cumulative Renewable Resource Constraint
$r_k(S,t) \leq 1$	Disjunctive Cumulative Renewable Resource Constraint

Table 3.2: Common RCPSP Constraints

3.1.2 Multi-Mode Resource Constrained Project Scheduling Problems (MMPSP)

The RCPSP can be further specialized by allowing a task's resource requirements to be satisfied by more than one resource, which results in the Multiple-Mode Resource Constrained Project Scheduling Problem (MMPSP). The MMPSP is given a general review in the following papers [3, 11]. Methods for solving the MMPSP are discussed in the following [2, 23].

If tasks in a project may be satisfied by different combinations of resources, then those tasks may be performed in different modes. Each task i has an integer total number of modes M_i that the task may be performed in. In addition to solving the problem under temporal constraints, a feasible mode assignment $x \in \mathcal{N}^n$ needs to be made that also satisfies the resource constraints. The multimode problem also adds the potential for using non-renewable resources, where the resources that are used by a task are consumed and no longer available to other tasks. Task durations may depend on the mode x so that the task duration is now written as $p_i(x)$.

The set of renewable resources is now denoted $\mathcal{R}^\rho$ and the set of non-renewable resources is denoted $\mathcal{R}^\nu$. A renewable resource limits the number of tasks that can be executed in parallel which depend on that resource, but the resource is available at all times during the project. A non-renewable resource may be used up, and limits all tasks which depend on that resource for the entire duration of the project. The active set and resource usage also depend on the mode of the project, where

$$\mathcal{A}(S,t,x) = \{i \in V | S_i \leq t < S_i + p_i(x)\}$$

$$r_k(S,t,x) = \sum_{i \in \mathcal{A}(S,t,x)} r_{ik}(x)$$

The definition of resource usage changes with the mode assignment x, so that

$$r_{ik}(x) = \sum_{m \in M_i} r_{ikm} x_{im}$$

$$x_{im} = \begin{cases} 1 & \text{if task } i \text{ executed in mode } m \\ 0 & \text{otherwise} \end{cases}$$

The MMPSP can be stated as: [2, 23]

$$\begin{aligned} & Min\ f(S) \\ subject\ to\ & r_k(S,t,x) \leq R_k\ , for\ r_k \in \mathcal{R} = \mathcal{R}^\rho \cup \mathcal{R}^\nu, t \in [0,h) \\ & \sum_{i=1}^{n} r_{ik}(x) \leq R_k\ , for\ k \in \mathcal{R}^\nu \\ & S_i + p_i(x) \leq S_j\ , for\ (i,j) \in E \\ & S_i \geq 0\ , i \in \{1 \ldots n+1\} \\ & S_0 = 0 \end{aligned}$$

$r_k(S,t,x) \leq R_k, k \in \mathcal{R}^\rho$	Mode Dependent Cumulative Resource Constraint
$\sum_{i=1}^{n} r_{ik}(x) \leq R_k, k \in \mathcal{R}^\nu$	Non-Renewable Resource Constraint

Table 3.3: Common MMPSP Constraints

3.1.3 Multi-Skill Resource Constrained Project Scheduling Problems (MSPSP)

One version of the MMPSP that is relevant to this project, is the Multi-Skill RCPSP (MSPSP). This problem is treated in [2, 7, 15, 17, 28].

In one variation of the MMPSP, employees are the resources that are used by each task. The skill constraints for a task are met by assigning employees that have those skills to the task. Employees can only be assigned to a single task during a time period. Employees are treated as a disjunctive renewable resource, where a resource r_k is disjunctive if the capacity $R_k = 1$. In this problem, no other

resources are considered except employees. For this Multi-Skill example, task durations do not vary with the mode of the task.

Employees with the same skills are interchangeable. Any employee with some skills required by a task may be assigned to that task. Each assignment of employees to a task, represents a mode of execution for that task. Since employees are interchangeable, the number of modes for this problem is usually large. Enumerating all of the modes in order to solve the problem can be unreasonably difficult.

A set of skills that will be considered in the project are given as a set $\mathcal{L} = \{L_1, \ldots, L_p\}$. The skill requirements for a task are satisfied by assigning resources that possess those skills to a task. An employee that does not have a skill to contribute to a task should not be assigned to a task. The number of resources required to satisfy a skill requirement is b_{ij} for task i and skill L_j. An employee has a 1 or 0 value of the skill. Let l_{jk} represent the whether or not an employee k has skill j. Instead of allowing there to be a single mode for every possible assignment of employees to tasks, it is easier to represent the mode x as a $n \times k$ matrix, where $x_{ik} = 1$ if employee $k \in \{1 \ldots K\}$ is assigned to task i, and $x_{ik} = 0$ otherwise. Then the skill constraint is a sum, such that for task i and skill j, $\sum_{s=1}^{p} \sum_{k=1}^{K} x_{ik} l_{sk} \geq b_{ij}$.

This formulation of the problem below is simplified from the version presented in [2]:

$$
\begin{aligned}
& Min\ f(S) \\
subject\ to\ & r_k(S,t,x) \leq 1\ , for\ k \in \{0,K\}, t \in [0,h) \\
& S_i + p_i \leq S_j\ , for\ (i,j) \in E \\
& \sum_{s=1}^{p} \sum_{k=1}^{K} x_{ik} l_{sk} \geq b_{ij}\ , for\ i \in \{1 \ldots n\}, j \in \{1 \ldots p\} \\
& S_i \geq 0\ , i \in \{1 \ldots n+1\} \\
& S_0 = 0
\end{aligned}
$$

$\sum_{s=1}^{p} \sum_{k=1}^{K} x_{ik} l_{sk} \geq b_{ij}$	Skill/Assignment Requirement Constraint
$r_k(S,t,x) \leq 1$	Disjunctive Cumulative Resource (Employee) Constraint

Table 3.4: Common MSPSP Constraints

3.2 Solving Scheduling Problems

The problem considered by this paper is an MMPSP that has skill requirements similar to an MSPSP. Approaches to solving the MMPSP include enumerative and heuristic methods. The following books contain common methods for

solving the MMPSP [2, 23]. For this project, Constraint Programming will be used to search for solutions to a problem in disaster recovery scheduling.

3.2.1 Constraint Programming

Constraint Programming (CP) is a method that can be used to solve, among other things, combinatorial optimization problems. It comes from research in Artificial Intelligence (AI) and Operations Research (OR). The purpose of CP is to determine if a problem has a solution, given some constraints and variables with associated domains. A solver is used to determine if a problem has a solution. The solver may do this by performing search and constraint propagation. A brief introduction to these concepts will be given in the following sections. Introductions to CP can be found in [12, 18]. More thorough coverage can be found in [29].

3.2.2 Variables and Constraints

A variable x_i is given an associated domain d_i from which it might take values. A shorthand notation $[a..b]$ is used to represent a set of integer values $\{a, a+1, \ldots, b\}$. A partial assignment is made when only some of the variables of interest have been given values, and an assignment is made when all variables have been given values. A primitive constraint c_j on variables $x_1, \ldots, x_k$ maps to true or false. Constraints may be formed from the usual combinations of mathematical operations and relations on integers, such as $+$, $\times$, $=$, $<$, $\leq$, etc. An example primitive constraint is $x_i \leq x_j + x_k$. Primitive constraints are true if they hold under some assignment of values to the variables from their respective domains, otherwise the constraint is false. A constraint is satisfiable if it has an assignment for which it holds.

Complex constraints may be made by combining multiple constraints. Primitive constraints may be connected by logical operations to form new constraints, such as by conjunction $\wedge$, disjunction $\vee$, negation $\neg$, implication $\rightarrow$, and equivalence $\equiv$. Another complex constraints might describe set membership where $P(x)$ is true if and only if $x \in \mathcal{S}$. A constraint might describe sequences or whether a list of variables is sorted. One common constraint is $AllDifferent(x_1, \ldots, x_k)$, which is true only if $x_i \neq x_j$ for $i, j \in \{1, \ldots, k\}$. More complicated constraints like the $AllDifferent$ constraint are called global constraints. It is often better to model a problem using available global constraints, rather than attempting to model them as primitives. A summary of global constraints, including $AllDifferent$, is given in [27].

A solver is used to determine if there is an assignment of values to $x = (x_1, \ldots, x_n)$ such that all of the constraints are satisfied. A solver may search for solutions to the problem by assigning values to variables and then shaving

down the domains for the remaining variables until each variable has taken on a single value. At this point, the solver has arrived at a potential solution for the problem, which then needs to be evaluated to determine if it is feasible. If the solution to the problem is not unique, the solver may arrive at different solutions by choosing to cut down the domains in a different way, and assigning different values to the variables. From this point on $\leftarrow$ will be used to denote assignment, as in the computer programming sense.

3.2.3 Consistency

Consistency checks are performed in CP to determine if the domains of variables contain values that are inconsistent with their constraints. A feasible solution can not be formed from values of variables that are inconsistent, so those values can be removed from the domains of their associated variables. This pruning of the domains is constraint propagation, which is one of the core features of CP. Different methods are used, such as node, arc, bounds, or other more complex checks. The checks that are used are often polynomial time methods. Finding domains that are consistent for all of the constraints does not usually result in a solution to the problem. Having consistent domains indicates that there might be a feasible solution, but it does not guarantee it.

Node consistency is achieved when the domain of a single variable is consistent with the constraints that only involve that single variable. Given a constraint $X \geq 5$, for $d = [0..10]$, the values from $[0..4]$ can be removed from the domain, and after reassignment $d \leftarrow [5..10]$ the domain d is consistent with the constraint. If there are no values in a domain that are consistent, then $d = \emptyset$ and there is no solution to the problem.

Arc consistency is achieved when the domain of two variables are consistent with the constraints that involve only those two variables. Given constraints $X_i + X_j \leq 5$ and $d_i = [2..5], d_j = [0..5]$, the values $[4..5]$ can be removed from domain d_j, and after reassignment $d_j \leftarrow [0..3]$ the two domains become consistent with the constraints.

Each time values are pruned from a domain for one variable, all of the other constraints involving that variable need to be checked again for consistency, which means checking the domains of all of the variables involved in those constraints. Since simply finding consistent domains does not usually result in finding a feasible solution, a guess and check type process called search is used to assign a value to a variable and then check to see if that leads to inconsistencies.

3.2.4 Search

Search is used in an attempt to determine if a feasible solution exists to a problem. A variable is unbound if its value is still unknown and may be any one

of multiple values from its domain. An unbound variable is selected and a value is assigned to that variable from its domain. After the variable is set, propagation occurs to determine if the constraints are still satisfiable after the assignment has been made. After propagation, if there is a domain $d_i = \emptyset$ then the value that was assigned to the variable cannot lead to a feasible solution and the search process undoes the previous decision and assigns a different value to the variable. If there are no inconsistencies after propagation, then the next variable is chosen and assigned a value. This process continues until a feasible solution is found, or until the search process has proven that there are no solutions.

This describes backtracking, which is a common search process. A tree search is used. Nodes in the tree represent states which describe which variables have been assigned variables and the domains of all variables. A value is selected and assigned to the variable at the current search depth, and the tree search follows a branch to a new state. Propagation occurs, and if all the resulting domains are consistent, the process is repeated. A potential solution is found when the search reaches a leaf node of the tree, which corresponds to the event that all of the variables are assigned values. The assignment at this node is evaluated to determine if it is a feasible solution. If it is, then the search process can stop. If it is not a feasible solution, then the search backtracks up the tree and then chooses a different value for the variable at that search depth. Backtracking can be used to make a complete solver, which if given sufficient time is guaranteed to reject all infeasible solutions and find all feasible solutions if they exist. The problems treated by CP are often NP-Complete, which means that the length of time required by the solver to locate a feasible solution or determine that there are no solutions grows exponentially with the number of variables used to model the problem. Heuristic search methods are often used to approach these problems, where the method is not guaranteed to find good solutions, but may work well enough for the user's needs in practice.

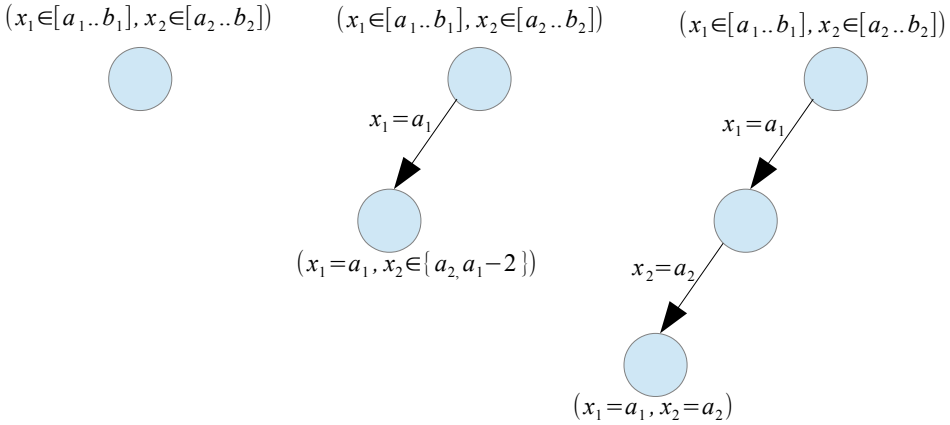
If an objective function is made that associates a real number with an assignment, an optimization process can be performed. The objective is evaluated at each feasible solution and its value is recorded for that assignment. After finding a feasible solution, the search may continue for other feasible solutions. At the end of the process, the assignment with the best objective value is selected. This makes it possible to find optimal solutions through backtracking if they exist.

As an example for backtracking, suppose we have a problem for $x = (x_1, x_2)$ with initial domains $d_1 = [a_1..b_1], d_2 = [a_2..b_2]$. A solution is required that satisfies the following constraints:

$$x_1 + x_2 \geq k$$
$$x_2 \leq x_1 - 2$$

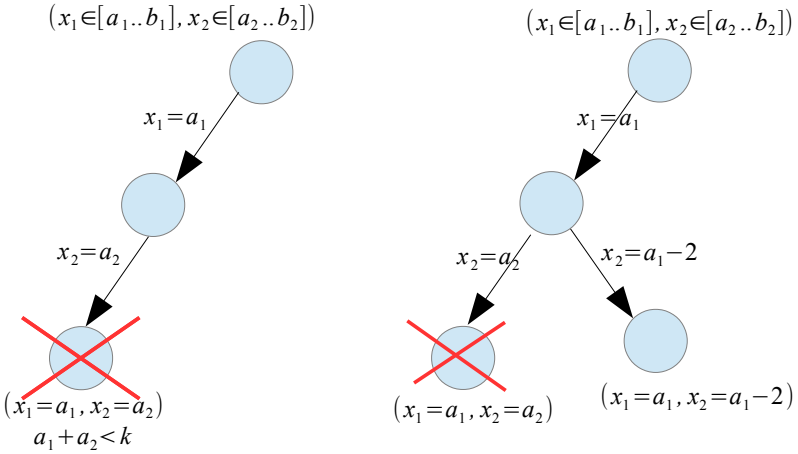
The solver may choose a value $x_1 = a_1$ from d_1. Suppose propagation occurs because of the second constraint, and the domain for x_2 is updated to

Figure 3.1: Descent to Leaf Node



$d_2 = \{a_2, a_1 - 2\}$. After propagation is complete, the solver assigns a value to x_2, which brings the solver to the leaf node at $x = (a_1, a_2)$. If this assignment was a feasible solution, the solver could stop the search at this point and return the assignment as a solution. This is the final assignment in Figure 3.1. Suppose that the assignment is not feasible, because it violates the first constraint, and $x_1 + x_2 = a_1 + a_2 < k$. The solver rejects the solution at this leaf node, and backtracks to the previous node, except a_2 is removed from the set of possible values that the x_2 can take. Then the next value for x_2 is chosen, where $x_2 = a_1 - 2$. This is the final assignment in Figure 3.2.

Figure 3.2: Backtrack



3.2.5 Cumulative Constraints and Scheduling

In scheduling problems, variables may represent start times, task durations, employee assignments, etc. Most of the constraints given in Table 3.1 may be expressed as primitives. In order to express the constraints of the RCPSP and MMPSP, complex constraints are needed. In addition to the simple constraints such as precedence $c \leftarrow (S_j \geq S_i + p_i)$, cumulative resource constraints are be used. This constraint ensures that the resource demands of tasks in progress at the same time do not exceed the capacity of the resource. This has the same effect as the constraints on $r(S,t)$ in the RCPSP. The difference, is that in implementation the cumulative constraint is supported by an algorithm that allows a solver to remove inconsistent values from the domains for the start times S_i in a way that preserves solutions if they exist.

A cumulative resource constraint requires a set of tasks $T' = \{i_1, \ldots, i_m\}$ with their associated resource demands $r' = \{r_{i_1 k}, \ldots, r_{i_m k}\}$, and the capacity of the resource they share R_k. The cumulative constraint may be written as $Cumulative(T', r', R_k)$. This constraint will take into account the durations of the tasks, their start times, and will adjust their domains.

A simple review of the cumulative constraint is given in [18]. Algorithms for implementing cumulative constraints are given by [19, 30, 36]. One method for implementing a cumulative constraint is called edge finding. For the method given by Vilim, the constraint keeps track of a lower bound on the early completion time of a set of tasks and the resource usage that results from those tasks. The tasks in the set do not necessarily have to be performed at the same time. It groups the tasks according to the latest completion time of a task. As it adds tasks to the group, at some point a task newly added to the group may cause the resource capacity to be exceeded. This indicates that the recently added task must occur later in time, and early start times for that task are pruned until the resource conflict is resolved.

This manner of edge finding described here is implemented using energy based reasoning, as described by [30, 36]. The energy of a task i is its resource demand multiplied by its duration, $e_i = r_i p_i$. The energy of a set Θ is $e_\Theta = \sum_{i \in \Theta} e_i$. Using this energy based rule, if a set of tasks Θ is already grouped together then another task i can be checked to see if its start time needs to be changed by adding it to the group. Let the capacity of the resource be R. If the energy of the combined set is $e_{\Theta \cup \{i\}} > R(lct_{\theta \cup \{i\}} - ect_\Theta)$ then this tells us something about the completion time of all tasks in the set versus the completion time of the task i, $C_j < C_i$ for all $j \in \Theta$. This allows the start times of individual tasks to be adjusted by comparing them to groups of tasks. If i satisfies the inequality such that $C_j < C_i$ for all $j \in \Theta$, then the early start time of i can be increased to $est_i = max_{\omega \subseteq \Theta}(est_\omega + \frac{\lceil e_\omega - (R - r_i)(lct_\omega - est_\omega) \rceil}{R})$. A similar rule exists which is used to adjust the end times of tasks.

To demonstrate the cumulative constraint, suppose there are three tasks that require the same resource with a capacity of $R = 3$:

Task	est_i	lst_i	p_i	r
Task 1	0	7	3	1
Task 2	0	7	3	2
Task 3	0	7	3	3

The solver may begin the search by selecting Task 1 and fixing its start time at $S_1 = 0$. Since Task 3 demands the full capacity of the resource, the cumulative constraint will prune the domain for the start time of Task 3 by pushing its early start time up to $est_3 = 3$ to prevent a resource conflict. The resource demand of Task 2 will conflict with the demand of Task 3, but neither of their domains will be pruned since they can be scheduled to occur at different times. The search may then choose Task 2, whose resource demand is low enough that the start time of Task 2 can be fixed at $S_2 = 0$, and Task 2 may occur at the same time as Task 1.

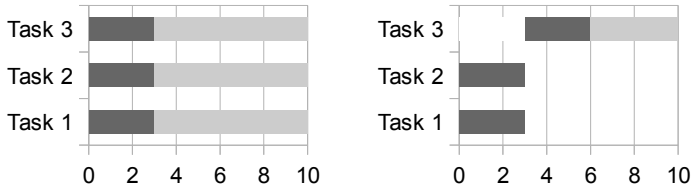


Figure 3.3: Cumulative Constraint: Before and After Propagation

To demonstrate the disjunctive cumulative constraint, suppose there are two tasks that require the same disjunctive resource:

Task	est_i	lst_i	p_i	r
Task 1	0	3	3	1
Task 2	0	7	3	1

The solver may begin the search by selecting Task 1 and fixing its start time at $S_1 = 0$. The cumulative constraint will then prune the domain for the start time of Task 2 by pushing its early start time up to $est_2 = 3$. The solver could have also started by fixing the start time of Task 2 at $S_2 = 0$ instead, and then pruning the domain for the start time of Task 1 by pushing the early start up to $est_1 = 3$. This would have the effect of fixing the start time of Task 1 at time $S_1 = 3$ without the solver having to choose Task 1 and set its start time during search.

In some schedules, there may be a need for optional tasks. This occurs in the MSPSP, where the mode of the task may be used to indicate whether or not it is assigned to an employee. There are methods for scheduling optional tasks, given in [37]. Instead of keeping a tally of which employees are assigned to which task, an optional task may be created for each employee that may be assigned to the task, which occurs at the same time as the actual task. Then, the optionality of those tasks is constrained so that the skill constraint is met. Optional task

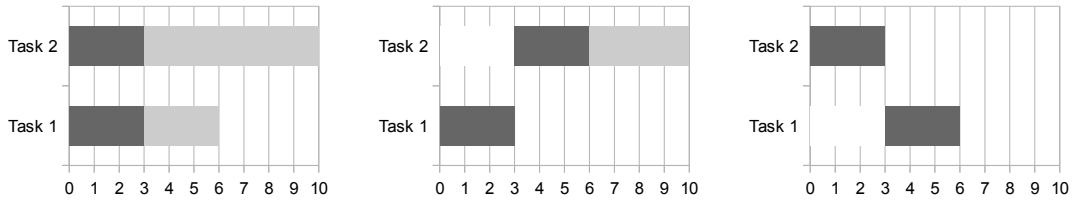


Figure 3.4: Disjunctive Cumulative Constraint: Before and After Propagation

scheduling works by allowing the resource demand of a task to vary with the mode, so that the demand of task i on resource j is

$$r_{ij}(x) = \begin{cases} r_{ij} & \text{if x = 1 (task performed)} \\ 0 & \text{if x = 0 (task not performed)} \end{cases}$$

The tasks are subject to the cumulative constraint, but only the performed tasks count against the resource capacity at a given time. Constraint propagation may render some tasks unperformed. If an optional task of unknown performance becomes inconsistent, it is set to unperformed. The cumulative constraint on optional tasks can be given a set of integer variables that indicate whether or not their associated task is performed $x' = \{x_{i_1}, \ldots, x_{i_m}\}$, so that the cumulative constraint for optional tasks might be written $Cumulative(T', r', x', R_k)$. This notation is non-standard, but it is descriptive.

3.2.6 Bounds

The objective value at a feasible solution can be used as a bound on the optimal objective value. Other bounds on the optimal solution can be determined through a problem specific method. In the case of objective minimization, the objective value forms an upper bound on the value of the optimal solution and likewise a lower bound may be found. If the difference between the upper bound and the lower bound is small enough, the search may quit as it is close enough to an optimal solution for the user. Bounds may also be used to guide search in cases where the lower bound of a parent node may be no less than the lower bounds of its children. In this case a node with the least lower bound may be explored in order to push up the lower bounds found during the search process, so that the program can guarantee to the user that it found a solution within some distance of the optimal solution. During the search process, it is common for the lower bound and the objective value to approach each other quickly early on and then slowly draw closer as the search is allowed to progress, as in Figure 3.5. In order to avoid the computation time that is required for small improvements in the solution, the user can set a limit β on the bound so that the search process

stops if $\frac{objective\ value - lower\ bound}{lower\ bound} \leq \beta$. For example, if the user desires the process to stop if a solution is determined to be within 5% of the optimal solution then the user can set $\beta = 0.05$. All of the following lower bounds discussed are bounds on the makespan of a project.

Lower bounds that are relevant to the MMPSP are given by [15, 35]. One simple lower bound on the makespan can be found by dividing the horizon of the project by the number of workers available. Other methods can be found using an energy based approach. The energy based approach is the same as the process used in edge finding to estimate the early completion time of a project for the cumulative constraint. The energy of a task i is its resource demand multiplied by its duration, $e_i = r_i p_i$. A lower bound can be put on the duration of the time for a task to complete by subdividing the task into a sequence of unit duration tasks that contain a portion of the energy assigned to the original task. Energy is allocated to each subtask by dividing the total task energy by the capacity of the resource $\frac{e_i}{R_i}$. The new subtasks are allowed to be executed in parallel. If the capacity of the resource is large enough to accommodate them, several subtasks may be scheduled to occur in parallel.

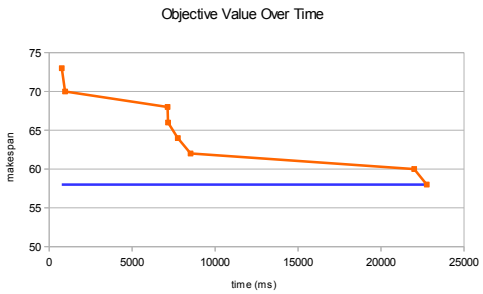


Figure 3.5: Objective Value Approaching Fixed Lower Bound

Putting a lower bound on the duration of a single task might not make sense when its duration is fixed, but the same reasoning is used to put a lower bound on the duration of a set of tasks. The energy of a set of tasks Θ is the sum of the energies of the tasks, $e_\Theta = \sum_{i \in \Theta} e_i$. The set of tasks that require the same resource are subdivided into unit duration tasks with their portion of energy requirements, and the precedence constraints between tasks are temporarily ignored. A lower bound on the duration of a group of tasks is $\lceil \frac{e_\Theta}{R_i} \rceil$. The early start time of a set of tasks Θ is $est_\Theta = min_{i \in \Theta}\{est_i\}$. The lower bound on the early completion time of set Θ is $ECT_{LB}(\Theta) = est_\Theta + \lceil \frac{e_\Theta}{R_i} \rceil$. To get a lower bound on the project duration, this lower bound can be found for each cumulative resource. The largest of these lower bounds is still a lower bound on the duration of the project, and this can be used to estimate how close the current solution is to optimality.

Lower bounds can be found from disjunctive resources like employees by summing the durations of the tasks that resource is required for. If an employee is assigned to a set of tasks Θ, then a lower bound on the completion time of the project is the total duration of tasks that the employee must perform, beginning from the earliest start time of the tasks that they have to handle $ECT_{LB}(\Theta) = est_\Theta + \sum_{i \in \Theta} p_i$. This is just another form of finding lower bounds through energy based reasoning, but for disjunctive constraints.

For the MMPSP with simple temporal constraints, it must be possible to model the precedence constraints as an acyclic digraph for feasibility. An acyclic digraph is one of the special cases where it is reasonable to find the longest path in a graph. In general, finding the longest path is NP-complete. The durations of the tasks act as distances, and the precedence constraints are the arcs between tasks. The longest path is a lower bound on the duration of the project due to the precedence constraints that must be respected. This lower bound can be found by the Bellman-Ford algorithm using the dummy node for the start of the project as the starting point.

A lower bound on the makespan of the project may be found by sifting through the domains of the task start variables at different search depths and finding the largest early completion for those tasks. When a variable is scheduled, its start time S_i is set to be equal to its early start time est_i. The largest early start time is $est_{max} = max\{est_1, \ldots, est_n\}$. A set of tasks Θ_l that may affect the completion time of the project is made where $\Theta_l = \{i \mid lct_i > est_{max}\}$. In order to find the largest early completion time, the set of early completion times that exceed est_{max} is found by looking at the set of tasks contained in Θ_l. A new set is made $\Theta_e = \{i \mid est_i > est_{max},\ i \in \Theta_l\}$. The sets can be recorded at each level of a tree search so that only a subset of the full list of variables needs to be checked at each decision. When the solver backtracks, the set of late completion times can be restored to its previous state.

CHAPTER 4

PREVIOUS APPROACHES TO OPTIMIZATION IN DISASTER RECOVERY AND INFORMATION TECHNOLOGY

Models in disaster recovery may look at the problem from a broad range of perspectives. The mathematical approach used to solve a disaster recovery problem will vary with the intended recovery efforts. The final result may be a simple assignment problem, a traveling salesman problem, a project scheduling problem and so on. The disaster recovery problem is often split up into phases, where the number and purpose of phases involved vary depending on the phase definitions that are used. Distinguishing between phases of a disaster recovery problem creates opportunities to optimize the disaster recovery process at different points.

The four phase approach referenced by Altay et al. in recovery planning for natural disasters is similar to the approach used by the US federal government for containing computer security incidents on information systems [1, 4]. The disaster recovery problem may be approached as a mitigation, preparedness, response, or recovery issue [1]. This four phase approach will be adopted for the following sections. The mitigation phase eliminates hazards before there is a problem to lessen the potential impact of a disaster. The preparedness phase distributes information to potentially affected parties and trains emergency personnel to respond to a disaster. The response phase is the immediate reaction after the declaration of a disaster in order to get the effects of the disaster under control. The recovery phase is the longterm effort to bring the situation back to normal. The efforts from one phase may fit into another phase, and the distinctions between phases can become difficult.

4.1 Mitigation and Preparedness Phases

In IT, a common preparation for a disaster is backup planning. Backup planning contains tasks that fit into both mitigation and preparedness. These plans exist to prevent data loss in the event of a disaster. Duplicate copies of important information are kept, and may be used to restore a system in the event that it is disrupted. Decision theory may be used to select and compose backup strategies to prepare for data loss [21]. The plans are formed by serial or parallel combinations of storage technologies which provide comprehensive

backup strategies for different applications. The purpose of combining storage methods in this way is to minimize the RTO, RPO, and cost associated with a strategy.

An approach to optimizing disaster recovery can be made by composing disaster recovery plans (DRP) out of smaller sub disaster recovery plans (SDRP). One goal is to solve a sub-plan selection problem, in order to form a DRP [24, 38]. The term DRP was already used above, from NIST SP 800 documents, to describe a recovery plan that requires relocation of computing resources. In this case, a disaster recovery plan is any plan that is invoked during a disaster. An SDRP is a plan to perform some restorative action in the event of a disaster and a DRP can be treated as set of sub disaster recovery plans (SDRP) which have some associated costs and resource requirements.

In solving the sub-plan selection problem, it is assumed that an organization has already created a list of SDRPs to perform recovery actions as part of its operations. The organization is assumed to have some desired features in mind for their DRP, such as minimal cost or time to complete. The organization's needs are reflected in the objective function or constraints used in the model. These desired attributes may come from a BIA in the form of time objectives such as MTD and RTO. Bryson et al. approached the sub-plan selection problem by maximizing what was termed the overall protective value of the DRP [24]. The protective value was determined from the costs and resources used by the SDRPs, an estimate of the reliability of an SDRP, and the likelihood that a disaster would be covered by an SDRP.

Difficulties may come from the organization that will be implementing the plan. In practice, the organization may not have enough previously established SDRPs to adequately cover the disasters that they would like to be prepared for. The resources listed for an SDRP may be a best guess since the SDRP planners may not have an adequate understanding of the costs or resources required to implement that SDRP. Attempting to solve the sub-plan selection problem may highlight the shortcomings in the existing SDRPs or reveal missing SDRPs, and help the organization to formulate more comprehensive plans.

The DRPs and SDRPs from the previous research papers are analogous to ISCPs, DRPs, and COOP from NIST documentation. The issue of having an inadequate number of SDRPs can be sidestepped by narrowing our focus to organizations that meet FISMA requirements. An organization that meets these requirements must have contingency plans for every important system. However, the same human induced problems regarding estimation of a plan's cost and resource usage persist under the assumption of FISMA compliance. Poor estimates in the disaster recovery plan could lead to an infeasible plan being deemed feasible.

4.2 Response and Recovery Phases

Decision theory may be applied to the response and recovery phases of disaster planning, by individually managing resources and personnel to handle problems [28, 40]. Zeng et al. use a Markovian Decision Process (MDP) to represent the behaviors of actors during a disaster [40]. Actors are individuals that have been assigned roles and responsibilities during a disaster recovery. These actors may have responsibilities other than disaster response, such as reporters or public relations workers. This process will lead the actors to make decisions during a recovery. Zeng et al. proposed to represent the MDP using a language extension of DT-GOLOG, which is an implementation of a decision theoretic logic programming language. This method appears to have no implementation to date.

Rolland et al., treated the disaster recovery process as a generalized multiple resource constrained project scheduling problem (MRCPSP), which they called a disaster response scheduling problem (DRSP) [28]. This is an assignment problem with a scheduling component. Emergency response teams are assigned to tasks, where each task requires some resources to complete. Tasks may have precedence relations, so that one task may be required to be completed before another. Teams can be assigned to any task, but a mismatch cost will be incurred if they are assigned to a task that they are not suited for. In this case, resources are assigned to tasks and teams are scheduled to complete those tasks in order to facilitate a disaster recovery. The objective is to minimize the costs incurred during disaster recovery by scheduling the best available teams to handle disasters in a reasonable order.

In the context of disaster recovery in an IT department, it may be possible to treat the problem as Rolland et al. have done, by assuming that personnel are able to operate different systems with differing skill levels. This would not be unusual in an IT department, where for example, a Windows administrator may occasionally be called upon to perform Linux administration tasks. If an IT department collects metrics from all tasks that are assigned to its employees, there may be sufficient data available to build reasonable penalties associated with assigning personnel to tasks that they are not suited for.

However, unlike the case in natural disaster recovery, it is not likely to be possible to assign IT personnel to all activities. It is often the case that personnel in an IT department are granted or denied access to components of information systems based on training plans, clearance levels, their work assignments, the location of their work areas, and other company authorizations. It may be assumed that personnel are only granted access to systems that they are qualified to handle. Under the assumption of FISMA compliance, employee authorizations and training are recorded. This information is may be queried from company servers. Under these assumptions, any workers with access to a system are equally qualified to restore the system. In this way, workers do not have to be assessed for relative measures of competence because the organization grants or denies administrative permissions based on the performance measures used in their training program.

Ideally, only qualified workers are granted access to an information system. This assumption may be problematic in reality, because workers are often granted rights to systems they are not qualified to handle. This problem may be partially resolved by considering the extent of the employee's usual work areas. An employee can be assigned to restore systems to which they have access within their usual work areas. The onus of responsibility falls to the organization to maintain best practices in system administration and to keep their authorizations and training data up to date.

CHAPTER 5

DISASTER RECOVERY IN INFORMATION TECHNOLOGY AS A PROJECT SCHEDULING PROBLEM

5.1 Recovery Problem

The recovery problem as a Multi-Mode Resource Constrained Scheduling Problem has much in common with the MSPSP given earlier. Each task is associated with a single skill, and no other task requires that skill. The skill represents whether or not the employee is eligible to perform the task. A pool of employees that possess that single skill will be formed for each task during preprocessing. An employee has a unit skill value, so that the required skill level for the task is the number of employees from the pool that must be assigned to the task. An employee is a disjunctive renewable resource, in that they can only perform one task at a time. Once assigned to a task, that employee must carry it through to completion. This recovery problem differs from the MSPSP, in that each task may also require renewable resources that are not employees. To separate the set of resource that represent employees from other renewable resources, a new set $\mathcal{R}^e$ is used for the set of employees. An employee is represented by an integer id number $i \in \mathcal{R}^e$. The resource requirement of a task for employees is the skill requirement b_{ij} for task i and skill j, which is satisfied if the mode for the task assigns the number of employees required for a task.

5.2 Summary of Method

A short summary of the proposed workflow for producing a disaster recovery schedule is presented first. Before a disaster occurs, the security packages for information systems are collected, reviewed for obvious conflicts, and stored in an accessible format. Obvious conflicts would be things like bad precedence constraints, overly large resource requirements, or cyclic dependencies between tasks. The data will be used to make an organizational precedence graph that represents the information systems for the entire organization. The problem will be stored in preparation for a disaster.

After a disaster occurs, an outage assessment is used to mark which systems were disrupted in the stored organizational precedence graph. This com-

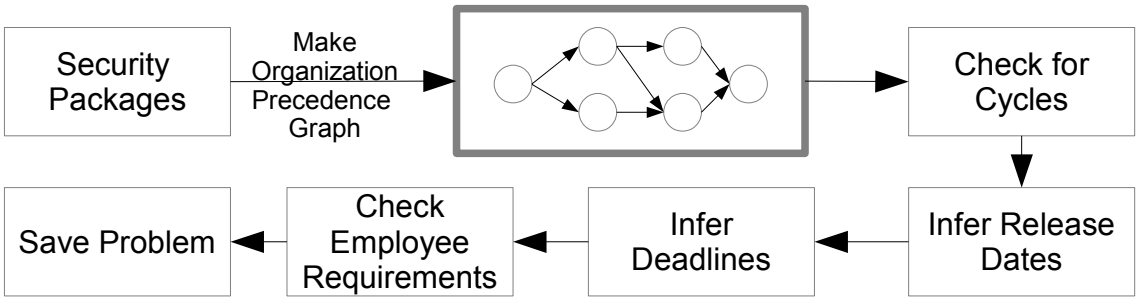


Figure 5.1: Process Summary: Before Disaster

bined information is used to produce a new precedence graph which only represents disrupted systems. After the information regarding the disaster is processed, a search is made for feasible recovery schedules by passing the problem to a solver.

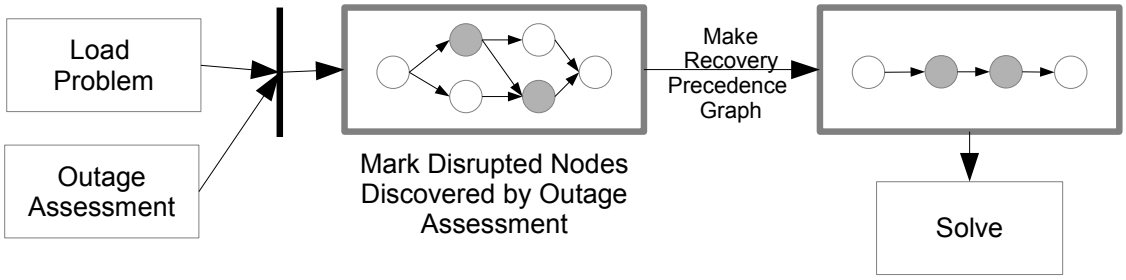


Figure 5.2: Process Summary: After Disaster

5.3 Problem Assumptions

In this paper, it is assumed that the planning organization is following all relevant NIST guidance. Just assuming FISMA compliance is not enough to form a feasible scheduling problem. Some loose requirements already present in FISMA need to be strengthened. In this case, we must make assumptions about the data that is provided by the security categorization, BIA, SSP, ISCP, the availability of this data, and the accuracy of the data. If there are details that are required in order to move forward with the project, but are not guaranteed by NIST SP documents or FISMA, they will be noted.

In addition to the interconnection agreements from the SSP and some precedence requirements from the ISCP, the precedence constraints between information systems are known. The dependencies between information systems is critical since these determine which information systems must be restored first.

Due to the ambiguity introduced by the nested systems-of-systems concept associated FISMA compliance, it is assumed that only the precedence and duration requirements that come from the security package for a singleton information system, a system which has no sub-components with their own security packages, can be respected. It is assumed that any cyclic dependencies in the graph that defines the precedence constraints within the organization will be resolved before the recovery plan is needed.

A finite number of personnel have limited access to buildings or computers. The only personnel that will be considered for restoring an information system will be those listed in the ISCP. Among those administrators listed in the ISCP, it is possible that only a subset of them have access to the information system. In order to determine if an administrator has access to the information system, the access controls for that system will need to be consulted. It is assumed that the access controls are documented and accessible in order to decide whether or not an administrator has access to an information system. This process will include verifying that the administrator has sufficient permissions, physical access, etc, to ensure that they are able to perform the restoration task from start to finish without interruption.

The BIA is used to provide a list of renewable resources that an information system requires in order to be restored. No information system requires more resources than are available. The BIA also gives priorities for information systems, which are demands that indicate which information systems should be restored in which order. These priorities are formed with business processes in mind, which means that they can represent an infeasible restoration order. It is assumed that the restoration order given in the BIA cannot be respected. Also, due to the systems-of-systems concept, allowing nesting of systems, it cannot be assumed that the RTOs or MTDs cannot be respected for all information systems. It is assumed that the only RTOs and MTDs that will be respected are associated with information systems that are singletons, which do not sub-components with their own security packages.

A summary of problem assumptions:

- The recovery task for an information system is given by the ISCP for that system.
- An access list containing the list of administrators that may restore a system is available for all components of the information systems
- For every component that needs to be restored, there is an administrator with access to the system. If there is a system that requires restoration, and there is no administrator for it, then the schedule is infeasible.
- There are adequate resources available for all restoration tasks.
- The resources required by a single task cannot exceed the availability of any resource.

- It must be possible to restore the dependencies of a component before the restoration of the component itself.
- Task durations have been estimated. An RTO/MTD is estimated for all components of an information system to be restored.
- There is a complete list of available administrators.
- Administrators are not reassigned tasks based on task escalation or due to the absence of another administrator.
- The scope of the restoration process is known in advance due to disaster planning and outage assessments. The systems that have been affected by the disaster are known.
- Once an administrator begins a task, it continues to completion. There is no preemption.
- Self-referencing or circular dependencies among information system have been resolved before attempting to schedule a recovery. If they have not, then there is no feasible schedule.

Concerns that cannot be met or guaranteed:

- The RTO/MTD may not be respected for information systems that are composed of other information systems
- Restoration priorities given by the BIA may not be respected
- Minimum or maximum response times by employees may not be respected

5.4 Data Collection

The set of information systems that are singletons will be collected. These are information systems that are not composed of other systems that have their own security packages. A precedence graph will be formed from the set of collected security packages. The resources that are associated with the information system will be recorded. ISCPs will be checked for potential individuals or groups that will be assigned to the recovery. Then, information system access controls will be consulted to determine if each individual in the list of administrators has access to the information system. Only the list of administrators that have proper access will be retained. The personnel that have access to the information system will be added to a set from which administrators will be chosen as they are assigned tasks. One of these sets will exist for each recovery task represented by an ISCP. If this set is empty, a warning should be issued since a task with unassigned personnel may render the schedule infeasible if it does not occur automatically.

After the graph has been created, it needs to be checked for errors that would render any disaster recovery plans derived from it infeasible. The graph needs to be checked for cycles. If there are cycles, then the problem is infeasible. Cycles will be reported to the user, and they will need to resolve them. This graph will represent the entire organization at a time when it is completely functional and there is no disaster. At this point, it does not make sense to look for release dates and deadlines, because the time constraints that can be inferred from the graph will change after the outage assessment that marks disrupted nodes.

Using the outage assessment data, nodes in the larger precedence graph for the company will be marked as disrupted. These nodes will be added to a smaller graph. Using the original graph, precedence requirements will be determined between the disrupted nodes. This can be done using a depth first search and noting that if there is a path from a disrupted node A to a disrupted node B, then B depends on A. This smaller graph may have redundant arcs. At this point, the task duration information needs to be associated with the disrupted nodes. If there is disaster recovery testing/training/exercise (TT&E) data for an information system, or records kept after an actual recovery, then any RTAs will be associated with the information system as the task duration. Otherwise, any RTOs will be associated with the information system as its task duration. If we wish to attempt to respect the MTDs associated with a business process that is supported by this information system, this would be the time to set the MTD as a deadline or due date for the task. However, in the event of a disaster that stretches personnel thin enough, treating the MTDs as deadlines may render the schedule infeasible from the outset.

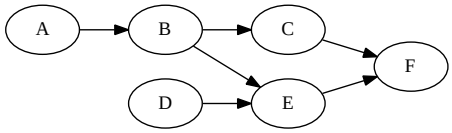


Figure 5.3: Organizational Precedence Graph

The transitive reduction of this graph can be found, in order to remove the redundant arcs if needed. This new smaller graph, is the one that should be used to form the scheduling problem that is used to schedule employees to perform disaster recovery tasks.

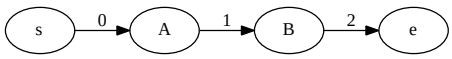


Figure 5.4: Initial Graph

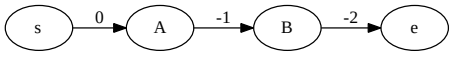


Figure 5.5: Finding Release Dates

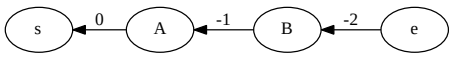


Figure 5.6: Finding Deadlines

Some preprocessing can be applied to reduce the sizes of the domains for the task start variables before the solver begins searching for solutions. From the graph of disrupted nodes, an inferred release date est_i and a minimum time delay d_i from the end of the task to the end of the project can be determined for each task. Since the graph used for scheduling is a directed acyclic graph, it is possible to find longest paths between nodes in $O(|E||V|)$. First, negate the weights of all the arcs in the recovery graph so that the Bellman-Ford algorithm can be used to find longest paths between nodes. Using the start node as a source, the negative of the inferred release date for each task will be found, such that

$S_i \geq est_i$. This is the smallest start time for each task. After this, reverse the direction of all the arcs in the graph. Using the Bellman-Ford algorithm with the end node as a source, this will find the negative of the minimum amount of time that each task must precede the end of the project, d_i. This will produce a constraint on the latest time that a task may be completed, so that $lct_i \leq S_{n+1} - d_i$. The initial domain for a task start variable for task i is $est_i \leq S_i \leq horizon - (d_i + p_i)$.

This documentation will also include security categorizations, resource requirements, and access controls for its components. The resource requirements can be used to establish resource constraints. The access controls will be used to establish constraints on technician assignments.

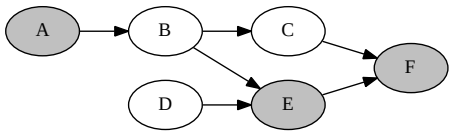


Figure 5.7: Disrupted Organizational Precedence Graph

The access restrictions, work areas, group memberships etc. will be condensed into a task eligibility matrix M, where an entry M_{ij} is 1 if employee i has the necessary rights to perform task j. The number of tasks that an employee is eligible to perform should be much smaller than the number of tasks in the project. This matrix is expected to be sparse.

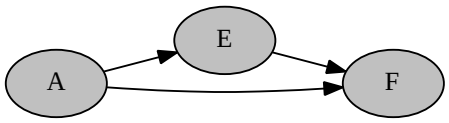


Figure 5.8: Recovery Precedence Graph

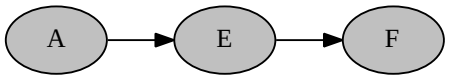


Figure 5.9: Transitively Reduced Recovery Precedence Graph

5.5 Determining Access Requirements

The access requirements to an information system my vary wildly depending on the system. Based on the requirements of FISMA, it should be possible to consult any number of databases or documents to determine if an administrator has access to the components of an information system. A non-exhaustive list of access requirements that must be met in order to access an information system are given below:

- Has physical access, if required, to the area/facility/building/room/datacentre, etc.
- Has an account on the network
- Has permissions to access the system, including user/group, local administrator, or access granted by active directory or locally set user group access dependent upon NIS or LDAP, etc.
- Has other computing/electronic authorizations granted by management.
- Has authorizations granted by training, etc.
- Has authorization based on security clearance level, etc.

Since these access requirements to an information system vary, a general process cannot be given in this paper for checking them. However, it can be assumed that the organization has established a comprehensive process for determining whether an administrator has access to a given information system, and whether they have the necessary permissions to perform a restoration task on that system. By comparing the complete list of administrative permissions to the list of users or groups that are expected to be capable of restoring the system, only those administrators that have all of the required permissions to the system should be added to a set of workers that are capable of performing restorative actions. It is from this authorizations list that workers will be selected from the list and later assigned to tasks in a disaster recovery schedule. The list of users that are expected to be perform a task will be given in the ISCP. In the previous section, users with proper permissions were enumerated in a 0-1 matrix. However, there is additional requirement that a user does not just have proper permissions, they must also be listed as an authorized employee in the ISCP. A second 0-1 matrix M of size $m \times n$ can be formed for m restoration tasks and n administrators, where an entry M_{ij} is 1 if administrator j is both listed as an employee that may handle a task and has the necessary permissions to perform task i.

5.6 Problem Formulation

After the recovery graph has been made, that data is provided to the solver. In this implementation, the information is saved to a file to be read in by the solver. The solver used is the Google OR-Tools constraint programming API [10]. The file format used adds some fields to the RCP format used with the benchmark resource constrained scheduling library, PSPLib. Test problems from PSPLib were changed to contain employees, skills, and skill requirements using the NetworkX package [22]. In a real world scenario, the data that is required to form a project scheduling problem would be queried from a database. But for this project, the data must be simulated. For simulated project scheduling data, the Project Scheduling Problem Library (PSPLib) is a source for small test problems [16, 26]. The PSPLib tool is still available, but only runs on older platforms, making it difficult to generate new test problems. However, some problems are available for download.

5.6.1 Implementation

Single mode PSPLib problems were changed to represent MMPSPs. A PSPLib problem was read in from an RCP file by a Python script. The number of available employees is provided, along with the maximum and minimum number of employees required by any task in the project. For each task, a random subset of employees is chosen that may perform the recovery task, and a random skill requirement is set between the minimum and maximum set size.

The NetworkX package is used to check for cycles, and to apply the Bellman-Ford algorithm for preprocessing the start times of the tasks. The minimum start time for a task is associated with the task in the output file, as well as the minimum delay from the end of the task to the end of the project.

The output from this operation is an MMPSP where the modes correspond to the employee assignments, and there are at least as many employees available as are required to perform the task. These files represent the organization after the outage assessment has occurred A list of problems to be solved is collected together into a file, dubbed a manifest, which tells the solver which MMPSP problems to solve. After each file was read in, the solver was allowed to run for a specified time limit before moving on to the next problem. Results were dumped to log files, which form the data for the results section.

5.6.2 Constraint Program

A method to model an MMPSP was suggested by one of the authors of Google OR-Tools. Instead of associating numeric modes that describe resource consumption with tasks, the modes are represented by optional tasks that consume the mode dependent resources required by the original task. These optional tasks are then constrained to occur at the same time as the original task. Proceeding in this manner, each employee that is eligible to perform a disaster recovery task is given an optional task that is synchronized in time to the original task. If there are ten employees that are eligible to perform a task, then ten optional tasks are made in addition to the original task. This is one possible method for modeling an MMPSP using OR-Tools.

For each task i, a fixed duration interval variable T_i is made, which represents all time information of the task. The task interval represents the start time S_i, the completion time C_i, the early start time est_i, late start time lst_i, early completion time ect_i, late completion time lct_i, and duration $p_i = RTO_i$ of the task. The start time S_i is allowed to be in the interval $[est_i, h - (d_i + p_i)]$, where est_i was found in preprocessing by the Bellman-Ford algorithm to be the least start time for task i. The task interval variable implicitly sets constraints on minimum and maximum start time of the task, so that these do not need to be added. If all of the domains for the start variables are set above zero, then the dummy task at S_0 does not need to be added to the project.

After all of the tasks have been read in, precedence constraints are made between the variables. If an arc (i, j) is given in the precedence graph, then a constraint is made such that $S_j \geq S_i + p_i$. The makespan is represented by a maximum function provided by the solver API. The makespan is set as $S_{n+1} = max_{i \in V}\{S_i\}$. It is at this point that a lower bound for the makespan is set, where $S_{n+1} \geq dist_{max}(0, n+1)$ in the precedence graph.

For each task i that demands a non-employee resource k, a resource demand r_{ik} is made. For each non-employee cumulative resource k, an integer

resource capacity R_k is made. For each resource k a cumulative constraint is made that involves all tasks that use that resource

$$\begin{aligned} Cumulative(&\{S_i \mid \textit{Task i requires resource k}\}, \\ &\{r_{ik} \mid \textit{Task i requires resource k}\}, \\ &R_k) \end{aligned}$$

For each task i, there is a set of employees that may perform that task $\mathcal{L}_i$. This is the set of employees that have the skill L_i. The employees are tracked by their numeric ids, so that $\mathcal{L}_i = \{j \mid \textit{employee j may perform task i}\}$. An optional task interval T_{ij} is made for each employee id j in $\mathcal{L}_i$. A constraint is placed on the start time of S_{ij} of task interval T_{ij}, so that $S_{ij} = S_i$. The duration of T_{ij} is p_i. Whether or not an optional task interval T_{ij} is performed is set by a boolean valued variable x_{ij}. Task i is performed by employee j if $x_{ij} = 1$. The skill requirement b_i is the number of employees that are required for task i. This requirement is met by setting a constraint that $\sum_{j \in \mathcal{L}_i} x_{ij} = b_i$. For each employee j involved in the recovery, an integer resource capacity $R_j = 1$ is made. For each employee j a cumulative constraint is made that involves all optional tasks that may use that employee.

$$\begin{aligned} Cumulative(&\{S_{ij} \mid j \in \mathcal{L}_i for i \in V\}, \\ &\{1\}, \\ &R_j = 1) \end{aligned}$$

A constraint program is given below which incorporates all of the above, to represent a disaster recover problem as an MMPSP.

$$
\begin{aligned}
& Min\ S_{n+1} \\
subject\ to\ & Cumulative(\{S_i \mid Task\ i\ requires\ resource\ k\}, \\
& \{r_{ik} \mid Task\ i\ requires\ resource\ k\}, \\
& R_k)\ for\ k \in \mathcal{R}^\rho \\
& Cumulative(\{S_{ij} \mid j\ \in \mathcal{L}_i for i \in V\}, \\
& \{1\}, \\
& R_j = 1)\ for\ j \in \mathcal{R}^e \\
& \sum_{j \in \mathcal{L}_i} x_{ij} = b_i\ for\ i \in V \\
& S_{ij} = S_i,\ for\ j \in \mathcal{L}_i\ and\ i \in V \\
& S_i + p_i \leq S_j\ , for\ (i,j) \in E \\
& S_i \geq S_{i,min}\ , for\ i \in V \\
& S_0 = 0
\end{aligned}
$$

5.7 Objective Functions

5.7.1 Makespan

A common objective is to minimize the duration of a project, where the duration is called the makespan. A constraint solver can perform optimization like this by putting constraints on the makespan during the search process. When the problem is first being formed, the makespan will be assumed to be between 0 and a large number. If a solution is found that is feasible, a constraint will be added that will put a maximum on the next makespan that might be found, C'_{max}, such that $C'_{max} \leq C_{max} - \delta$. The step size δ controls whether or not the solver will continue searching for solutions that are not at least δ better than the previous solution. For example, if the programmer only cares if each step is at least a fraction of the horizon H better, then they can set $\delta = \lceil \alpha H \rceil$, $0 < \alpha < 1$. The search process can be sped up by using step sizes at the cost of losing better solutions. If a lower bound C_{LB} is placed on the objective function at each node in the search, the solver can be made to backtrack at a node in the tree if the lower bound is not some step size better the previous objective value C_{prev}, such that $C_{LB} \leq C_{prev} - \delta$. In the case where the objective value is an integer, increasing the step size above 1 may discard any guarantee that a solution can be found for a problem that has at least one solution.

One formulation of a makespan objective is to say that $C_{max} = S_{n+1}$. This will judge schedules with the same makespan as being equally desirable,

regardless of the scheduling time of the tasks between the start and end. Another way to handle it might be to minimize $C = \sum_{i=1}^{n} C_i$, which is the summation of the completion times of all tasks, which will not only reduce the makepsan but it will prioritize all tasks to occur as soon as possible.

For this project, there may be issues with minimizing the makespan under the assumption of FISMA compliance. The RTO is meant to be set as an upperbound on the task duration before the task affects the MTD of the business process it supports. This means that scheduling using RTOs as task durations is attempting to minimize an upper estimate of the makespan of the project. An implicit assumption is made, that the RTO is a reasonable estimate of the task duration. The outcome from scheduling like this will be affected by how poorly the RTOs reflect the actual task durations. This is why the RTAs need to be derived from TT&E data if it is available.

Another way to model a makespan objective, is as sum of weighted completion times, $C = \sum_{i=1}^{n} w_i C_i$. This would allow the schedulers to prioritize systems with larger weights to be scheduled earlier. As for scheduling with FISMA, there is a problem in determining the weights that might be applied to a system. It would be possible to simply use the security categorizations as the weights if they were defined numerically. Recall that the security categorization is the tuple that is assigned to every information system that determines if it is a high, moderate, or low priority system. Each system has a level that rates its confidentiality, integrity, and availability between high and low. The scheduler would need to determine which category and level they think needs to be weighted more highly. The high water mark could be used instead, so that each system would only have a single weight to choose from. The trouble with this, is that there is still no guarantee that higher priority systems are not dependent upon lower priority systems which could lead to a less desirable recovery.

5.7.2 Lateness

Lateness and tardiness are relevant if tasks are assigned due dates. A task should be scheduled before its due date. A due date differs from a deadline, because the schedule is not treated as infeasible if the task is scheduled beyond this point [23]. The lateness of task i is the amount of time that a task's completion time C_i exceeds its due date d_i, so that $L_i = C_i - d_i$. Another concept related to lateness is tardiness, T_i where the tardiness is $T_i = max(C_i - d_i, 0)$.

One objective, is to minimize the total lateness of the schedule, $L = \sum_{i=1}^{n} C_i - d_i$. This will prioritize schedules that finish before their due dates, but it allows for schedules that may exceed them. Under FISMA, lateness might be modeled by how much the completion time of a task exceeds its MTD if that task is started too late, $L_i = C_i - MTD_i$. It does not make sense to treat the MTD as a deadline, because the MTD of a business process cannot be respected. If it is only important for a schedule to make sure that tasks occur before their due dates, then the sum of tardiness can be minimized instead.

5.7.3 Cost

In simple cases, a cost based objective function is just a weighted sum of completion times, $Cost = \sum_{i=1}^{n} w_i C_i$. The weights w_i represent rates with units of $\frac{cost}{time}$. A better schedule prioritizes systems with higher cost rates. In other cases, costs may be functions of the completion time, so that the cost function is given by $Cost = \sum_{i=1}^{n} f_i(C_i)$.

The cost of disaster recovery is difficult to calculate using only the assumption of FISMA compliance. Costs are really associated with the downtime of business processes by the BIA. A business process is affected by many information systems, so that a cost for the downtime of a business process isn't associated with a single information system. Sometimes there may be costs associated with specific information systems. However, it appears for the most part, there will not be a simple way to associate costs with individual information systems.

In a schedule where simple monetary penalties are only applied at a given rate if a task exceeds its due date, the cost function may be modeled by a weighted sum of tardinesses, $Cost = \sum_{i=1}^{n} w_i max(C_i - d_i, 0)$. In other cases, the cost may be a sum of functions of tardiness $Cost = \sum_{i=1}^{n} f_i(T_i)$. These tardiness costs are relevant to FISMA, because stiff monetary penalties may be associated with important information systems that exceed their MTD. However, these costs may not be simple rates, and instead may be penalties that are applied if the tardiness exceeds a given sequence of thresholds, in $\frac{cost}{day}$. Not all information systems will come with penalties, and a schedule that minimizes penalties might not be a very good schedule from other points of view like makespan, or even lateness.

CHAPTER 6

DISCUSSION

6.1 Results

The following results were found from solving PSPLib problems after adding skill requirements to the tasks. The sets of problems are grouped by the number of tasks in the schedule, based on what was available in the RCP file format. For each problem, many MMPSPs were formed by the method given in section 5.6.1. These were formed by adding skill requirements to each task. The number of available employees was capped at 10. An MMPSP was formed from each problem for every combination of available employees $a \in \{1, \ldots, 10\}$ and maximum employees $m \in \{1, \ldots, a\}$. The step size was set based on the horizon H, at $\delta = (0.01)H$. The time limit for the solver to search for solutions was capped at 30 seconds. For each problem, if a feasible solution was found then the properties of the solutions were recorded. If no solution was found, it only contributed to the number of unsolved problems.

The original PSPLib problems were generated based on parameters that the authors used to estimate the difficulty in solving those problems. The kinds of parameters that are used to predict the complexity of scheduling problems are discussed in [6, 16]. Unfortunately, the way that the problems were converted into MMPSPs did not take these difficulty measures into account. That makes these results difficult to interpret, as the predictors of complexity were changed when the employee skill requirements were added to the tasks.

Tasks	**Problems**	**%DiffMax**	**%DiffAvg**	**Feasible**	**%Feasible**
30	2640	69.28	22.14	1199	45.42
60	2640	68.64	19.15	835	31.63
120	3300	63.79	16.96	440	13.33

Tasks	**AvgNumSoln**	**MaxNumSoln**	**Unsolved**	**%Unsolved**
30	2.01	8	1441	54.58
60	1.86	7	1805	68.37
120	1.39	4	2860	86.67

Table 6.1: Summary of Results for Disaster Recovery MMPSPs

The default search process used by OR-Tools is a depth first left to right tree search. The search process does not change the values of the start times for variables near the top of the tree very often. Because of this search method, the lower bounds on the early completion time of the project do not increase very quickly. In all problems from this data set, the initial lower bound that was found before the search started based on the energy, longest path, and work load per employee was better than the lower bound based on the largest early completion time found during search for all problems where feasible solutions were found. If the structure of the problem does not yield an optimal solution near this lower bound, the estimate of how close the objective value is to the lower bound will not be very good.

6.2 FISMA and NIST SP 800 series

The following is a discussion of the documentation provided by NIST which forms the basic requirements for FISMA compliance, as well as the suggestions that guide other practices.

6.2.1 Business Impact Analysis

The requirements that are set for the Business Impact Analysis have some issues that affect scheduling. The problems dealt with here are the durations of tasks, deadlines, costs of downtime. The definition of an RTO is meant to be an upper bound on the amount of the time a task may take to complete. If the task exceeds its RTO, it is already late. However, the RTO is the only estimate of task duration that is given if there is no TT&E data. This means that any schedule that is produced by using the RTOs as task durations while minimizing the makespan is finding some schedule that is attempting to minimize a worst case scenario. If the RTOs are large and overestimate the recovery time, then the schedule that is produced may be poor.

The definition of MTD and its relationship to RTOs used in NIST SP 800-34 is vague. The document states that exceeding the RTO will impact the MTD, but it is not clear if that means the MTD will be exceeded. If a Work Recovery Time (WRT) is used, where $RTO + WRT \leq MTD$, the duration of additional tasks that must be performed after the IT department has restored an information system may be represented. If the RTO is surpassed, then the additional tasks corresponding to the WRT may complete after the MTD. If staying below the MTD is a primary concern, then it would be important to also take into account the office tasks that fill the WRT and impact the MTD. None of the SSP, ISCP, BIA, require non-IT tasks to be represented. Any schedule that is produced without taking office tasks and their precedences into account may be poor if the office tasks can significantly impact the outcome of the disaster. This might be the case if the penalties for exceeding the MTD are large, and minimizing cost is

the objective of the recovery. The objective functions that can be formed from the data required by FISMA to be in the BIA may not be very good measures of the actual properties of the recovery of business processes because the disruption of a business process depends on factors that are not information systems.

One of the problems with FISMA compliance, is that it does not give a good way to associate a single information system with the disruption of a business process. Due to the fact that an information system may be a collection of information systems, one parent information system that is associated with a business process may contain collection of child information systems. Without having clear impacts on business processes or other information systems stated in the BIA for each child information system, it would be difficult to infer which child information system might be able to disrupt the parent system and in turn, the business process. This makes it difficult to determine which information systems' completion times may contribute to the cost or lateness and tardiness with respect to the MTD of a business process. Downtime propagation may be one way to model this situation, and this is treated in [39]. These authors give a method that would be applicable to assessing the feasibility of conflicting requirements imposed on nested information systems. However, their method uses more information than is required to be kept under FISMA requirements and could not be met in the general case. This method would require organizational policies that imposed better documentation.

6.2.2 Precedence Issues

To be able to form a schedule from recovery tasks, it is important that the precedence constraints and task durations be predictable. However, the ISCPs allow that a disrupted information system may be replaced by a different system in a different location as long as the original functionality that was provided by system is restored. There may be recovery modes with different precedence constraints, security controls, and required employees than the original task. The simple 0-1 matrix that records if employees are eligible to perform tasks may itself be mode dependent for a task and the precedence constraints for a task may be mode dependent. This means that the organizational precedence graph that was formed during the planning stage before the disaster may be invalid if the precedence constraints of an information system change after the occurrence of a disaster.

One of the concepts that was ignored in this paper is the recovery of information systems that have dependencies that involve or are affected by networking requirements. If one information system depends upon another, and it needs to communicate with that system over the network, then some communication pathway needs to exist between them. Since computer networks can route information along different pathways there may be more than one way to satisfy the precedence requirement that one information system is able to communicate with another. The tasks for recovery will need to be subdivided in

a way that the precedence requirements are predictable and can be known before a disaster occurs. This leads to a problem with implicit dependencies that may exist and are unknown to the owners of an information system. The introduction of computer networking into the disaster recovery process also introduces the issue that converting the PSPLib problems into MMPSPs may not be a very good reflection of the kinds of problems that would arise in a networked environment. In order to study these kinds of problems, either real world data would be required in the form of cases studies, or simulated data would need to be made based on computer network analysis.

NIST contingency planning documents do not require that tasks which can be run in parallel be noted as such in the ISCP or BIA. They also do not require for the steps in the ISCP to be segmented into sections with their own time estimates. Consider the following example of why this can cause the method used in this paper to fall short. On the client side of IT, a single technician may perform many setup tasks in parallel, while finishing tasks may be executed in sequence. One example where this can occur is in setting up clients using disk imaging technologies. The technician may insert a CD into the client computer that directs the client to automatically set itself up by retrieving the operating system from a server on the network. The technician may begin the setup process for many clients in parallel, and then wait for them to finish. A finishing step in the process may require the technician to retrieve a user's personal data from the network and direct their programs to use that data. Retrieving a user's personal data may take the technician's full attention, so that only one finishing task can be performed by that technician at a time. In this case, the technician may start many setup tasks and a single finishing task in parallel. Without information regarding which tasks may occur in parallel, and which must occur in sequence, we may only safely assume that the tasks recorded in accordance with NIST guidance must be performed in sequence. This may lead to an inefficient recovery plan when attempting to schedule client recovery tasks.

The BIA only requires that interdependencies between information systems be recorded. If some elements of an information system are not large enough to warrant having their own ISCPs with their own documentation of precedence constraints it is possible for a perfectly valid cyclic dependency to be formed between two information systems, which the method in this paper cannot deal with. Consider the following example, suppose there are two information systems A and B. There are two services on A, labeled A_1 and A_2, where B depends on A_1, and A_2 depends on B. By denoting this at the information system level, a cyclic dependency is formed between A and B, which renders the scheduling problem infeasible. This problem can be solved representing information system A as two separate subsystems, A_1 and A_2. This results in a modeling issue that can only be solved through human intervention.

6.2.3 Contingency Plan

The testing, exercise, and recording requirements that are associated with the ISCP could be used to derive an RTA for a task duration. The RTA is a closer estimate to the actual duration of a task than an RTO. RTOs may not be very good estimates of the maximum downtime of an information system. In the usual case with an organization composed mostly of office workers, the IT department is likely to have significant experience with restoring client computers. If the IT department keeps data on client recovery they should be able to provide good estimates on the RTAs for clients. This would mean that it might be possible to apply stochastic optimization methods to client machines which are likely outnumber servers and network equipment. It is unclear to this author if client machines are covered by common controls such that the actual precedence requirements of individual client machines are documented. Recall that controls are the source of contingency plans. In this author's experience, it is more likely the case that IT support personnel have a general idea of what their customers may require in an area in the sense of precedence requirements, but that a contingency plan that could be used for scheduling likely does not exist for each client computer. In the server side case, it is expected that there will not be nearly enough data points for this portion of the recovery to be treated with stochastic optimization. This volume of data that would be required to apply stochastic optimization to the server side/network equipment side of the problem would need to come from case studies across organizations.

A list of administrators should be provided in the ISCP that have the ability to perform restorative actions associated with an information system. This list of administrators may name specific administrators, or administrative groups that should perform the task. If the document names an individual, that individual's access to the system needs to be verified. For example, it would not be uncommon for an administrator to be granted permission to perform remote administrative tasks, but not be granted physical access to the facility or datacentre where the server itself resides. In the event of a disaster, the usual administrator may not have the ability to perform the necessary restorative actions. During the disaster planning process it will be necessary to enumerate all possible permissions that are required for an administrator to perform restorative actions, including the unusual permissions that would only be required in the event of a disaster. An example of unusual permissions might be the authorization for an employee to retrieve data backups from an offsite facility.

The assumption that a 0-1 matrix can be made that represents whether or not an administrator that is listed in the ISCP has the permissions to perform a recovery task may be difficult to accept in practice. The information for controls can be kept in anything from databases to handwritten entries on paper forms. Collecting this information from scattered sources and expecting it to be up to date or correct for the use of scheduling may be a tall order as the absence of a single permission can disallow an administrator access to an information system. This has the potential to cause what was found to be a feasible schedule according to the data on hand to be infeasible in reality. A second issue that should be

considered, is the fact that most administrative personnel that are listed in the ISCP are likely to be specified as groups of employees, rather than by name as individuals. The solution space of the problem grows quickly with the number of employees that are available to perform tasks. By specifying groups to handle tasks, rather than smaller lists of individuals that are likely to handle those tasks, the complexity of the scheduling process is much larger than it needs to be.

A rescheduling method would be needed for this kind of disaster recovery planning to become useful. Discoveries may be made during the recovery process that will render the current schedule infeasible. These discoveries may include mode varying precedence requirements, insufficient user permissions, invalid task durations or resource requirements. Problems are expected to arise during disaster recovery and the organization needs to be able to respond to that kind of change. A good place to start may be with the rescheduling methods for RCPSPs that are discussed in [2].

6.3 Constraint Programming Solver

Google OR-Tools was used as the solver in this project. At the time of this writing, both the solver and its documentation are incomplete. The only method for scheduling that is included by default is a depth first search with backtracking. For reasonably sized problems the first choice of variable is fixed for the duration of the search. This means that the lower bound that is calculated at the root node tends to be the lower bound which is used to determine how close a solution is to optimality. This search process may be modified through the design of custom decision builders and search monitors, but that requires a fair amount of knowledge of the internal workings of OR-Tools.

In Google OR-Tools, there is a method available to implement local heuristic search, which may speed up the discovery of solutions. This would be critical for disaster response, as the scheduler cannot be expected to take an excessive amount of time to find a schedule when the organization is pressed for time. A popular heuristic method for the MMPSP and MSPSP is Tabu search. Heuristics for solving these problems are given in [3, 7, 15, 28].

6.4 Issues Modeling Employee Schedules

This method of scheduling is not entirely realistic. It ignores both weekends and periods before and after work hours. The tasks are allowed to be scheduled at any time of day and run to completion. During disaster recovery, even if a company began operating twenty-four hours a day until it completed the recovery, there would still need to be adequate constraints in place to keep employees from working too many hours in a day or keep them from being scheduled to work during periods when they are off. An implicit assumption for this project is

that the schedule is cut off at the beginning and end of the workday and continues on to the next day. The schedule that is produced in this way may not be as good as a schedule that explicitly assumes that there are pre-emptive tasks. For larger tasks, it would be reasonable to assume that assignment of the task may transfer between administrators so that the first portion of the task is handled by the first and the latter portion of the task is handled by the second.

One of the assumptions of this manner of scheduling is that response times, travel times, and setup times are included in the RTO. This assumption could lead to a schedule being less than optimal if employee travel times become a significant factor in the duration of tasks. In the case of recovery in an IT department, one administrator may handle multiple groups of users that are separated into clusters within work areas. It may be more efficient for the administrator to handle several tasks in one area at a time, rather than traveling between many tasks in different areas.

Without some form of constraint on the number of hours that a worker may be assigned to tasks, and without taking their personal schedules into account, the schedules that are produced may not be feasible. This means that one administrator may be eligible to perform a task at one time and not another. Constraints may need to be used to make sure that the amount of work that is assigned to different employees is more evenly distributed.

6.5 Conclusion

In this project, a disaster recovery problem was treated as a Multi-Mode Resource Constrained Project Scheduling Problem. A method for forming MMPSPs from FISMA compliance documents was proposed. Test problems taken from PSPLib were extended to include employee assignment information relevant to this disaster recovery scheduling problem. The resulting problems were then used to demonstrate the feasibility of solving this problem for small cases. A Constraint Programming solver (Google OR-Tools) was used to search for solutions to these problems. The author's implementation of the solver was able to find feasible solutions, but was not able to obtain good lower bounds on the makespan of the project due to difficulties in implementing the search procedure.

This implementation struggles to find solutions to problems with 120 tasks and 10 employees, when IT disaster recovery problems involving upwards of 10,000 tasks have been encountered by real world organizations. FISMA compliance does not make strong enough requirements for this method to work well with the data that is expected to exist in the corporate world. Additional organizational policies to enforce the recording of precedence constraints will be needed. The organizations will also need to digitize paper records used to track employee permissions and make them accessible. The problem formulation requires additional work in order for this method of disaster recovery scheduling to be feasible. A rescheduling method is needed to respond to unexpected events such as missing or incorrect precedence requirements. Better handling

of employee schedules is needed, to prevent employees from being scheduled to work when they should not be. Heuristic methods will be needed to find solutions to this problem quickly.

BIBLIOGRAPHY

[1] N. Altay and W. G. Green III. "OR/MS research in disaster operations management". In: *European Journal of Operational Research* 175.1 (Nov. 2006), pp. 475–493.

[2] C. Artigues, S. Demassey, and E. Néron. *Resource-constrained project scheduling : models, algorithms, extensions and applications*. 1st. Control Systems, Robotics, and Manufacturing Series. Wiley, 2008.

[3] P. Brucker et al. "Resource-constrained project scheduling: Notation, classification, models, and methods". In: *European Journal of Operational Research* 112.1 (Jan. 1999), pp. 3–41.

[4] P. Cichonski et al. *Computer Security Incident Handling Guide*. Special Publication 800-61 Revision 2. National Institute of Standards and Technology, 2012.

[5] E. L. Demeulemeester and W. Herroelen. *Project Scheduling: A Research Handbook*. Boston: Kluwer Academic Publishers, May 2002.

[6] E. Demeulemeester, M. Vanhoucke, and W. Herroelen. "RanGen: A random network generator for activity-on-the-node networks". In: *Journal of Scheduling* 6.1 (2003), pp. 17–38.

[7] L. E. Drezet and J. C. Billaut. "A project scheduling problem with labour constraints and time-dependent activities requirements". In: *International Journal of Production Economics*. Special Section on Recent Developments in the Design, Control, Planning and Scheduling of Productive Systems 112.1 (Mar. 2008), pp. 217–225.

[8] S. E. Elmaghraby. "Activity nets: A guided tour through some recent developments". In: *European Journal of Operational Research* 82.3 (May 1995), pp. 383–408.

[9] *Federal Information Security Management Act (FISMA) Implementation Project*. http://www.nist.gov/itl/csd/soi/fisma.cfm.

[10] *Google OR-Tools, GitHub*. https://github.com/google/or-tools.

[11] S. Hartmann and D. Briskorn. "A survey of variants and extensions of the resource-constrained project scheduling problem". In: *European Journal of Operational Research* 207.1 (Nov. 2010), pp. 1–14.

[12] P. V. Hentenryck. *Constraint Satisfaction in Logic Programming*. Cambridge Massachusetts: MIT Press, 1989.

[13] P. V. Hentenryck, R. Bent, and C. Coffrin. "Strategic Planning for Disaster Recovery with Stochastic Last Mile Distribution". en. In: *Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems*. Ed. by A. Lodi, M. Milano, and P. Toth. Springer Berlin Heidelberg, Jan. 2010, pp. 318–333.

[14] Joint Task Force Transformation Initiative. *Security and Privacy Controls for Federal Information Systems and Organizations*. Special Publication 800-53 Revision 4. National Institute of Standards and Technology, 2013.

[15] Y. Kadrou and N. Najid. "A new heuristic to solve RCPSP with multiple execution modes and Multi-Skilled Labor". In: *IMACS Multiconference on Computational Engineering in Systems Applications*. Oct. 2006, pp. 1302–1309.

[16] R. Kolisch and A. Sprecher. "PSPLIB - A project scheduling problem library: OR Software - ORSEP Operations Research Software Exchange Program". In: *European Journal of Operational Research* 96.1 (Jan. 1997), pp. 205–216.

[17] H. Li and K. Womer. "Scheduling projects with multi-skilled personnel by a hybrid MILP/CP benders decomposition algorithm". en. In: *Journal of Scheduling* 12.3 (Sept. 2008), pp. 281–298.

[18] K. Marriott and P. J. Stuckey. *Programming with Constraints: An Introduction*. Cambridge Massachusetts: MIT Press, 1998.

[19] L. Mercier and P. Van Hentenryck. "Edge Finding for Cumulative Scheduling". In: *INFORMS Journal on Computing* 20.1 (Feb. 2008), pp. 143–153.

[20] *Minimum Security Requirements for Federal Information and Information Systems*. Federal Information Processing Standard 200. National Institute of Standards and Technology, 2006.

[21] T. Nayak et al. "End-to-end disaster recovery planning: From art to science". In: Apr. 2010, pp. 357–364.

[22] *NetworkX, GitHub*. https://networkx.github.io/.

[23] K. Neumann, C. Schwindt, and J. Zimmermann. *Project Scheduling with Time Windows and Scarce Resources: Temporal and Resource-Constrained Project Scheduling with Regular and Nonregular Objective Functions*. Berlin: Springer, 2003.

[24] K.-M. (Noel) Bryson et al. "Using formal MS/OR modeling to support disaster recovery planning". In: *European Journal of Operational Research* 141.3 (Sept. 2002), pp. 679–688.

[25] *Oil giant Saudi Aramco back online after 30,000 workstations hit by malware | Naked Security*. https://nakedsecurity.sophos.com/2012/08/27/saudi-aramco-malware/.

[26] *Project Scheduling Problem Library - PSPLib*. http://www.om-db.wi.tum.de/psplib/main.html.

[27] J.-C. Régin. "Global Constraints: A Survey". en. In: *Hybrid Optimization*. Ed. by P. v. Hentenryck and M. Milano. Springer Optimization and Its Applications 45. Springer New York, 2011, pp. 63–134.

[28] E. Rolland et al. "Decision support for disaster management". en. In: *Operations Management Research* 3.1-2 (Mar. 2010), pp. 68–79.

[29] F. Rossi, P. v. Beek, and T. Walsh. *Handbook of Constraint Programming*. Elsevier, 2006.

[30] J. Scott. "Filtering algorithms for discrete cumulative resources". PhD thesis. Uppsala: Uppsala University, 2010.

[31] *Standards for security categorization of federal information and information systems*. Federal Information Processing Standard 199. National Institute of Standards and Technology, 2004.

[32] M. Swanson, J. Hash, and P. Bowen. *Guide for Developing Security Plans for Federal Information Systems*. Special Publication 800-18 Revision 1. National Institute of Standards and Technology, 2006.

[33] M. Swanson et al. *Contingency Planning Guide for Federal Information Systems*. Special Publication 800-34 Revision 1. National Institute of Standards and Technology, 2010.

[34] L. P. Taylor. *FISMA Compliance Handbook*. Second Edition. Waltham MA: Syngress, 2013.

[35] P. Vilím. "Edge Finding Filtering Algorithm for Discrete Cumulative Resources in O(kn log n)". 2009.

[36] P. Vilím. "Global Constraints in Scheduling". PhD thesis. Prague: Charles University, 2012.

[37] P. Vilím. *Max Energy Filtering Algorithm for Discrete Cumulative Resources*. 2009.

[38] K. Wang et al. "A Mathematical Approach to Disaster Recovery Planning". In: Nov. 2005, pp. 46–46.

[39] E. Zambon et al. "A Model Supporting Business Continuity Auditing and Planning in Information Systems". In: *Second International Conference on Internet Monitoring and Protection, 2007. ICIMP 2007*. July 2007, pp. 33–33.

[40] W. Zeng et al. “Research on Formal Representation and Decision Theoretic Planning of Emergency Plan”. In: *GCIS ’10 Proceedings of the 2010 Second WRI Global Congress on Intelligent Systems*. Vol. 1. Dec. 2010, pp. 161–164.

SCHEDULING DISASTER RECOVERY OPERATIONS IN INFORMATION TECHNOLOGY UNDER FISMA

by

Joshua Brashear

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the last page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and may require a fee.