Engineering Management
Field Project

Project Schedule Optimization in a Telecommunications Environment

By

Chris Taliaferro

Fall Semester, 2013

Tom H. Bowlin
Tom Bowlin
Committee Chairperson

[illegible]
Mike Kelly
Committee Member

[illegible]
Kevin Buie
Committee Member

Date accepted: 11/18/13

Acknowledgements

I would like to first thank the KU faculty for their guidance, wisdom, and support throughout my journey in the Master's of Engineering Management program. Specifically, I would like to recognize Tom Bowlin and Mike Kelly for the hours they spent as committee members on this Field Project.

I would be negligent to not also acknowledge my employer. They provided me the tuition assistance that made even the notion of pursuing graduate education possible. Also, they put in my path the quality of managers that I have found myself fortunate to learn and study from. Specifically, I would like to thank Kevin Buie who has been a mentor, friend, and committee member.

To my wife and daughter, I believe yours was the greater sacrifice while I pursued this degree. Thank you for the love and support through many busy nights and lost weekends over the last four years.

Finally, to my God who blesses beyond measure, thank you for the opportunity and the endurance to have that opportunity realized.

Executive Summary

Veritas Corporation is a thriving company in the fast-paced telecommunications industry. To keep up with the dynamic industry, a significant quantity of projects must be scheduled in a short time period without impacting each other. While Veritas succeeds in this, they do so at the cost of a significant number of management hours.

The objective of this research is to produce a quantitative method to quickly and accurately produce an optimized schedule for the vast number of projects that Veritas implements.

This was accomplished by making iterative binary integer linear programming models. The first model was to serve as a proof of concept and was conducted on a small market. After successful completion, the model was expanded to a larger market and verified.

Outstanding results were seen in both iterations. By using historical data, it is evident that the optimized model outperformed traditional manual scheduling. The larger and more complicated the situation became the greater the improvement provided by the optimized model.

Table of Contents

Chapter 1 – Introduction

Veritas Corporation is a wireless telecommunications company. The telecommunications industry has a dynamic environment that has evolved and grown at a staggering rate in the last several years. Because of its inherent nature of change, there is a rapid, constant flow of projects conducted at its switching centers throughout the United States.

Each switching center handles numerous services through its various network elements. These services range from the different technologies that customers are provided by a wireless carrier to Veritas' internal services that enable it to monitor the network and operate more efficiently. The distinct technologies that a wireless carrier typically provides are the different generations of wireless technology. 4G relates to internet via LTE or WiMax; 3G relates to internet via EVDO or UMTS; 2G relates to voice calls via CMDA or TDMA.

In Veritas, there are different levels of switching centers. Two of such centers are an Access Switch (AS) and a Core Switch (CS). Both centers provide the same type of service to its customers and for internal use. However, a CS is also an aggregation point for surrounding AS's. These AS's funnel their traffic to a CS. This CS is then connected to other CS's around the United States, which allows Veritas traffic to traverse the nation or beyond.

Because wireless customers expect services to be available 24x7, the network must always be functioning. To achieve this, two measures are taken. The first is that projects that could affect the customer's ability to use the network are only done from 12:00 AM to 5:00 AM. This is referred to as the maintenance window and is when any outage will have the least impact on its customers since this is when the fewest number of customers are on the network. Also, to achieve a high resiliency even when implementing these projects, telecommunication companies utilize a high level of equipment and service redundancy. By doing so, they increase the reliability of the network.

To illustrate, Figure 1 depicts a network of three switching centers. A CS in Kansas City is a hub for an AS in Wichita and an AS in Omaha. For redundancy, each office has two of the same type of equipment to provide service.

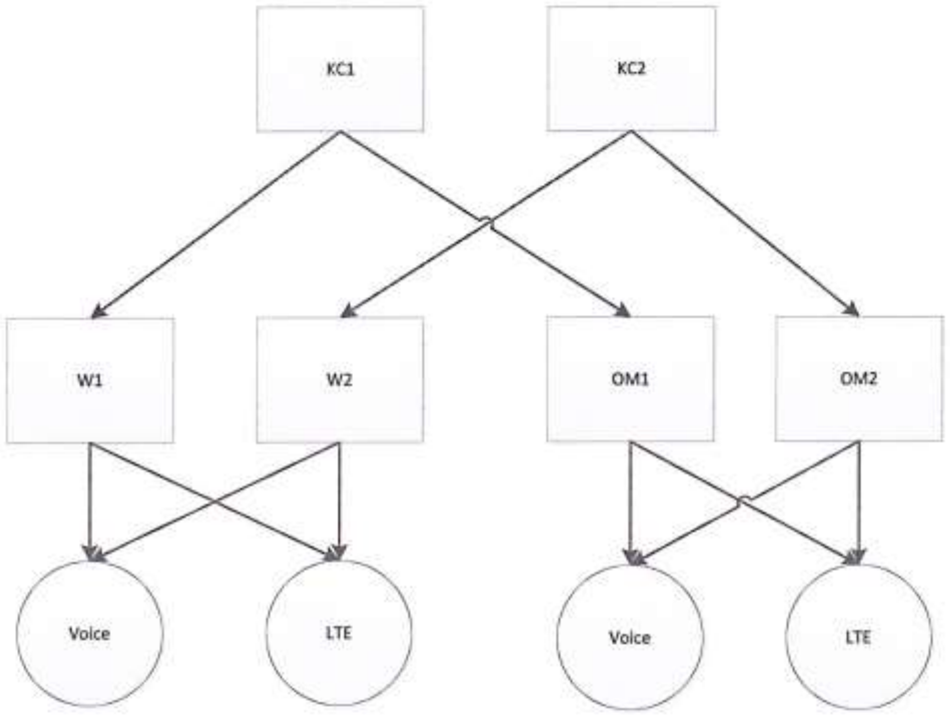

Figure 1 – Multiple Switch Redundancy Design Example

However, many projects cannot be implemented on a network element without temporarily disrupting a function of the wireless service that is running across it. When this device's functionality is disrupted, services are redirected to its redundant pair so that network functionality is still maintained. These projects, by themselves, are always designed to maintain service. The potential issue that one must continuously be wary of is how projects will impact each other if done simultaneously.

During a certain maintenance window, Kansas City is doing a project that impacts both Voice service and LTE service to its market and all of the AS's that it serves. During that same maintenance window, Wichita is implementing a project that impacts Voice service and Omaha is doing a project that impacts LTE service.

If these projects are not coordinated or spread out, Kansas City could first impair device KC1 while Wichita and Omaha could impair W2 and OM2. The result would be a loss of Voice service in Wichita and LTE service in Omaha.

The example described is shown in Figure 2. It portrays how the lack of coordination could lead to multiple service outages in the three switching center network design.

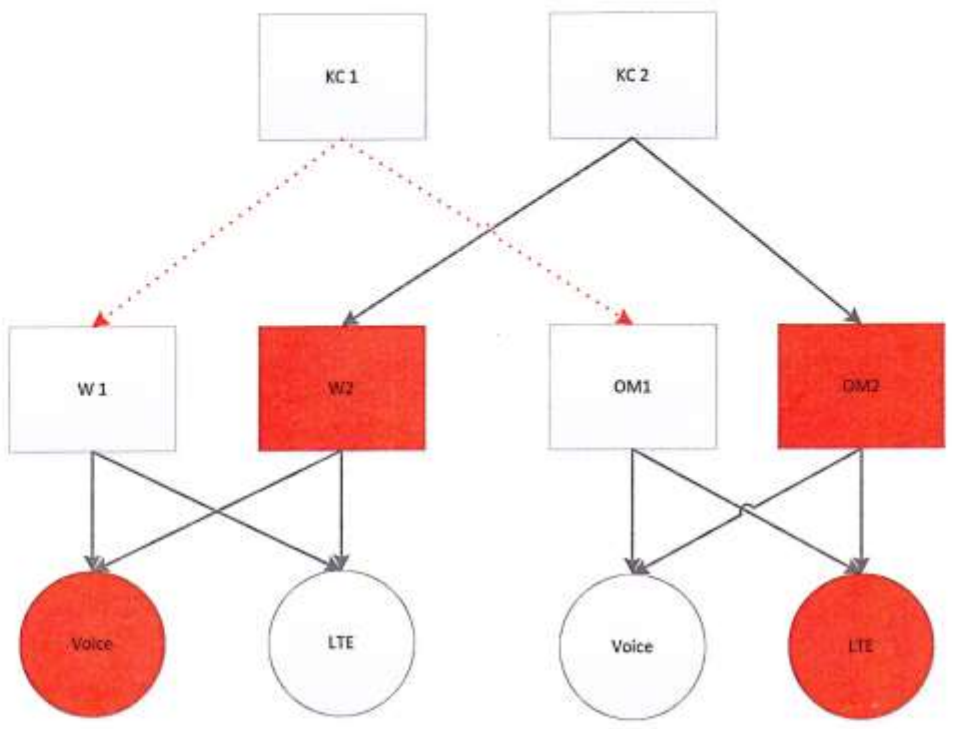

Figure 2 – Multiple Switch Project Impact Example

The fast-paced industry demands that these projects be completed on a rigorous timeline to keep up with customer needs. However, the requirement of reliability necessitates that customers never lose service. Therefore, a significant number of hours are put into planning the implementation of these projects so that no two projects that could affect each other are done simultaneously.

Research Purpose

The purpose of this research is to investigate modeling as a potential means of simplifying the task of scheduling the numerous and complex projects that are needed in a switching center or set of switching centers. While it's an essential business task, scheduling is also a daily routine that could be automated and should be optimized. By decreasing the amount of management hours spent on this process, those hours could be reallocated to focus on other business critical tasks or planning.

To meet the purpose of this research, an attempt will be made to develop a linear programming model that will take all the projects occurring within a switching center or set of switching centers and provide an optimized schedule for project implementation. For the model to be successful, it must be able to be executed in under five minutes, produce the same or better results than a manually-produced schedule, schedule three weeks out, and handle at least 40 projects.

Research Challenges

One of the main challenges to this research will be obtaining accurate, quantified project data to develop a model. Veritas does well in their scheduling, but they do a poor job in documenting what is involved in a project, what its impact will be, how long the duration of the project lasts, and how many resources will be required. Therefore, for this research to be successful, a significant amount of effort will be required to transform the existing project information into a useable form.

After the model is developed and proven, there could be resistance to its acceptance. Scheduling is a key component to the job function of many employees, including managers. Diminishing that role could be seen as a threat to their purpose. To avoid this, buy-in must be obtained from upper management, and a plan to reallocate their energies must be ready before merging it into day-to-day activities.

Chapter 2 – Literature Review

Schedule optimization has taken on many forms in various industries. A literature review was conducted to gain a better understanding of this problem, its solutions, and how these solutions can be applied to Veritas. This research focused on:

- current scheduling methods
- linear programming techniques
- tools used in scheduling

Current Scheduling Methods

In-depth research was conducted on how and why linear programming should be pursued in scheduling. Traditional methods of manual scheduling are effective in many organizations for most purposes. However, when fast-paced organizations grow and their dependencies multiply, the scheduling process becomes more complex.

Accounting for this complexity through manual methods has resulted in the frequent rescheduling of activities. As a consequence, more employee hours, overtime, and company resources are consumed by these activities. Therefore, in these scenarios, traditional methods prove to be time intensive, expensive, and inefficient (Taylor, 1996, 218).

Quantitative techniques are the solution to handling an ever shifting schedule that requires numerous variables to be considered. Amongst the quantitative techniques, linear programming is one of the more effective tools that can be used to solve this problem.

In a study to determine the most efficient method to schedule computer processing cycles, Nahir and Ziv considered linear programming as opposed to CSP (Constraint Satisfaction Problems). In their research, both techniques were used over various computer processing scenarios. They found that using the linear programming model was significantly larger than its CSP counterpart with more variables and constraints. However, it outperformed CSP in both accuracy and efficiency. In all tests, the linear programming model produced the optimal schedule successfully while CSP was only able to do so in 25% of the tests. Also, the linear programming model ran an average of six times faster than the CSP model (Djamel, et al. 2006, 70).

Similar accuracies were also found in the research that Fuller performed to develop methods to assist in production scheduling for building medium-sized trucks. Using a linear programming model, the production schedule was always improved over the traditional, manual scheduling methods that had been used. Also, the plans produced by the model provided the company an average cost savings of 9.6% (Fuller, 1975, 135).

Linear Programming Techniques

Linear programming is a quantitative problem-solving technique that was developed to help make decisions. It uses a mathematical model of linear constraint limitations to find a minimized or maximized solution of a linear function. (Anderson, et al. 2008, 16) The components of this model are the objective function, constraints, and variables.

The objective function is the linear equation that is to be minimized or maximized by the mathematical model to produce the optimal result. In a linear programming model, the constraints are mathematical functions that enforce boundaries for how large or small the solution to the objective function can be.

Variables consist of the values that are used in both the constraints and objective functions. They are broken down into two categories: uncontrollable inputs and decision variables. Uncontrollable inputs (also referred to as constants) are factors that cannot be changed by the decision maker. Decision variables, though, are the factors that the decision maker can influence to obtain an optimal solution. (Anderson, et al. 2008, 8)

Decision variables can be further broken down into non-integer, integer, and binary variables. Non-integer variables can be standard decimal numbers and will provide the most optimal solution. In some scenarios, though, decision variables cannot be broken into decimal units. In these cases, integer variables should be specified in the model. In situations where decisions need to be made, binary variables can provide an option to enable or disable a factor. This is done by constraining the variable to be either a 1 or a 0.

The common linear programming technique applied to scheduling problems is Mixed Integer Linear Programming (MILP). This adaptation of linear programming has both integer and non-integer variables. This form creates soft constraints by relaxing the model while still improving the quality of the solution. (Djamel, et al. 2006, 65)

However, when dealing with problems of significant magnitude, binary variables should be incorporated into the model. They can be used as decision points to account for the numerous potential outcomes of large systems. This provides enhanced flexibility and applicability to linear programming models. It should be noted, though, that while this technique allows for every variable to be accounted for, it will also be taxing on the linear programming software and system running that software (Matsuoka, et al., 2007, 239).

Tools for Scheduling

In order to validate the effectiveness of any model developed, linear programming software will need to be utilized to run the mathematical model created to solve the scheduling problem. For this research, two software packages were chosen.

Microsoft Excel Solver

Microsoft Excel Solver tool is a very effective tool for developing small models. It does have size limitations, though, that can be easily surpassed. In fact, it can only handle a maximum of 200 decision variables (Frontline Systems, Inc., 2013). For the purposes of scheduling, this can significantly hinder the ability for a model to be effective. For a model scheduling projects for a day in a five-day work week, Solver could only handle a maximum of 40 projects during that week. Therefore, for problems requiring a substantially sized model, Excel Solver can be best utilized as a proof of concept tool for early, smaller versions of the final product.

To solve the scheduling problem for a Core Switch in Veritas, a model of approximately 2,000 variables will be required.

LINDO Systems

LINDO Systems has a variety of linear programming suites that can best fit most programming purposes. In Taylor's research, he found LINDO to be the best software for his model of truck production scheduling (Taylor, 1996, 225).

LINDO also has licenses up to an unlimited number of variables and constraints. Therefore, this software tool has the ability to solve a large optimal scheduling model using binary variables. The capabilities of LINDO, then, remove the potential software limitation that might be encountered when solving the problem that Veritas is facing (LINDO SYSTEMS., 2013).

LINDO also supports software trial periods and educational licenses that can provide short term functionality and proof of concept opportunities.

Chapter 3 – Research Procedure

The process used to develop the optimal project scheduling model for Veritas Corporation followed Bowlin's 7-Staged Modeling Process that was presented in an Engineering Management course at the University of Kansas (Bowlin, 2011):

1. Identify Purpose/Functional Need
2. Collect Necessary Information
3. Formulate Model
4. Validate Model
5. Exercise Model
6. Report Results
7. Implement Model

For the purposes of this research, stages 5 and 7 will not be executed. All results will be discussed in Chapter 4.

Identify Purpose/Functional Need

The purpose of this research is to develop a method to optimally schedule projects for Veritas Corporation. Many hours are currently spent to manually schedule impending projects. The result of this research will be a method that can determine the optimal schedule while still meeting all of the restrictions for each project.

Collect Necessary Information

A literature review was first conducted to determine the methods that have been used to implement optimal scheduling. Best practices from the various industries were then compiled and utilized to solve the scheduling problems at Veritas Corporation.

For the development of the optimal scheduling model, Veritas data was collected from a variety of sources. Company-wide project information was gathered from a headquarters online directive tracker. Contracted projects were obtained from separate vendor reports. Local project information was collected from the work-order queues and shift reports of each local team.

For each of these types of information, the datum had to be molded into a quantified form that could be used by a linear programming model.

Formulate Model

After reviewing the material in the literature review, it was determined that the best approach for meeting the needs of Veritas Corporation was by creating an algebraic formulation of a binary integer model.

The objective of this model would be to minimize the number of maintenance windows needed to complete all of the projects assigned to the given market(s). This minimization would be done while maintaining the need to have no project impact overlap and respecting the market resources available in the given maintenance window.

Validate Model

The binary integer model was validated using Microsoft Excel Solver. To stay within the quantity of variables capabilities of Solver, the model was implemented on the smallest market, which had the least number of projects. This validation was first done by obtaining all of the assigned dates, due dates, completion dates, and technology/device impacts of each project that occurred during a three week span of the second quarter of the 2013. That data was then applied to the created algebraic formulation.

The model was then executed with the gathered data. All formulas were manually verified to be accurate. The produced schedule was also checked to confirm it stayed within the constraints necessary for Veritas Corporation. During each execution, the model run duration was observed and recorded.

After successfully verifying the model was running according to design, a comparison was done to determine if the model produced a schedule that was an improvement over the historical schedule.

Report Results

All results and conclusions are presented in Chapter 4.

Chapter 4 – Results

The model developed as a result of this research was successfully applied to the separate scenarios of an Access Switch and a Core Switch. The results of these two "proof of concept" demonstrations are provided in three sections. The first lays the foundation by providing the details of the formula development for the schedule optimization model. The second section differentiates the optimization model from the manually-produced model by displaying the results derived from the successful application of each scenario of the model. The final section provides a summary of the results of this research and illustrates the benefits this research can provide to Veritas.

Model Development

This section specifies the assumptions, variables, constants, objective function, and constraints of the model developed for an Access Switch and Core Switch. While the model is the same for each scenario, the magnitude and complexity of the Core Switch is far greater.

Assumptions

Before delving into the model, it is important to first list model assumptions, as follows:

Teams outside of the influence of a given market may be required for some projects. It is assumed that these teams will be available to implement the given project on the date that the model suggests.

On certain key projects, the implementation schedule is governed by headquarters. In such cases, the given switch has limited control on what date the project will be completed. This model assumes that there are no such projects being scheduled.

In most offices, project implementation only occurs during the maintenance windows of weekday mornings. Therefore Saturday morning and Sunday morning are only utilized in case of emergency. This model assumes that no emergency scheduling will be needed for the provided projects.

Variable Definitions

$M_{i,j}$ ≡ whether project *i* will be performed on maintenance window *j* <binary>
Where $i = 1, 2, \ldots, DL$
$j = 1, 2, \ldots, PL$

Constant Definitions

Let Tn_i ≡ whether or not project *i* will impact Technology *n* <binary>
Let Dn_i ≡ whether or not project *i* will impact Device *n* <binary>
Let H_i ≡ the number of hours needed to complete project *i*
Let R_i ≡ the number of market resources needed to complete project *i* <integer>
Let A_j ≡ the number of market resources scheduled maintenance window *j* <integer>
Let W_k ≡ the number of hours in week *k* needed for small projects and maintenance <integer>
Let C_i ≡ the number of maintenance windows before project *i* is due <integer>

Objective Function

The purpose of the objective function is to dictate the model to schedule the projects in the earliest maintenance window possible. The first step in this optimization is to minimize the sum of all the binary variables that indicate if a project will be completed on a given night. To encourage the scheduling of the projects in the earliest possible maintenance window, though, weights had to be added to the calculations. This weighting was applied by multiplying the summation of the number of projects being conducted in a given maintenance window by the index of that maintenance window. Therefore, the objective function is a minimization of the summation of the weighted products across all maintenance windows.

$$MINIMIZE \qquad \sum_{j=1}^{DL} \sum_{i=1}^{PL} M(i,j) * j$$

Constraint #1

One of the leading factors to the number of management hours needed in current manual scheduling is the determination of whether projects will impact the same technology as another project. Therefore, this constraint provides the rules to prevent the scheduling of two similar impacting projects. Quantified binary data is provided from Veritas for each project as constants. These inputs specify whether a project impacts a given technology.

$$\sum_{i=1}^{PL} M(i,j) * Tn(i) \leq 1 \qquad for\ all\ Maintenance\ Windows, j$$

Constraint #2

Similar to Constraint #1, this constraint prevents the simultaneous scheduling of two projects that impact the same device. Quantified binary data is provided from Veritas for each project as constants. These inputs specify which devices are impacted by the project. The number of devices in this constraint is dependent on which switch the model will be implemented on.

$$\sum_{i=1}^{PL} M(i,j) * Dn(i) \leq 1 \qquad for\ all\ Maintenance\ Windows, j$$

Constraint #3

To prevent overloading a given night with too many projects, this constraint limits the number of projects that can be performed with the number of scheduled resources for a given night. Quantified data is provided by Veritas for each project as constants. These inputs are the hours needed to complete a project, the integer number of market resources that are required to be actively working the project, and the integer quantity of market resources scheduled for a given night.

$$\sum_{i=1}^{PL} M(i,j) * H(i) * R(i) \leq Aj * 5 \qquad for\ all\ Maintenance\ Windows, j$$

Constraint #4

This constraint is a model relaxation constraint. When the model was first developed, the smallest market produced over 1,000 variables. A large portion of these variables were the result of small projects that had minimal technology impact and required no more than two hours of a maintenance window.

Therefore, instead of using the model to schedule even the smallest of projects, this relaxation technique was introduced. This technique implements a constraint that provides a set number of hours in a week to small projects or maintenance instead of scheduling each of these separately. Then the small projects can be removed from the list of projects to be scheduled.

After applying this technique to the model, the number of projects decreased drastically, resulting in a 75% reduction in variables.

$$\sum_{j=1}^{5} Aj * 5 - \left(\sum_{i=1}^{PL} M(i,j) * H(i) \right) \geq Wk \qquad for\ all\ Weeks, k$$

Constraint #5

This constraint was developed to ensure that each project is completed by its set due date. Veritas provided constant integer inputs that specify the number of maintenance windows that can transpire before a project will hit its due date.

$$\sum_{j=1}^{Ci} M(i,j) \geq 1 \qquad for\ all\ Projects, i$$

Constraint #6

In instances where one project is dependent on another project's completion, this constraint limits the former projects *i* potential completion dates to after the latter project *h* is finished.

$$\left(\sum_{j=1}^{DL} M(i,j-1) \right) - M(h,j) \geq 1 \qquad for\ all\ Maintenance\ Windows, j$$

Constraint #7

In instances where one project must be done immediately after another project, this constraint forces the projects to be done back-to-back.

$$M(i,j) - M(h,j+1) = 0 \qquad for\ all\ Maintenance\ Windows, j$$

Model Results

The constraints, variables, and objective function were combined in linear models for both an Access Switch and Core Switch. The Access Switch model was first developed using Excel Solver. Due to Excel's functional limitations, the scope was limited. Once proven, the same model was developed using LINDO's LINGO 14.0 software package. The models in both software packages produced the same minimized objective function results.

The model for the Access Switch used attributes of eight projects that were implemented at the Access Switch. These projects occurred during three weeks, or 15 maintenance windows, of the second quarter of 2013.

After the model was run, the results were compared to the actual schedule that was manually produced over those three weeks. This schedule is shown below in Table 1.

	Maintenance Window														
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Prj 1	-	-	-	-	-	O	-	-	-	-	-	-	-	-	A-D
Prj 2	-	-	-	-	A	-	-	O	-	D	-	-	-	-	-
Prj 3	O	-	D	-	-	-	-	A	-	-	-	-	-	-	-
Prj 4	-	-	A	-	-	-	O	-	-	-	-	D	-	-	-
Prj 5	-	A	-	O	-	-	-	-	-	-	-	-	-	D	-
Prj 6	-	O	-	-	D	-	-	-	-	A	-	-	-	-	-
Prj 7	O	-	-	-	A-D	-	-	-	-	-	-	-	-	-	-
Prj 8	A	-	O	D	-	-	-	-	-	-	-	-	-	-	-

Table 1 – Actual, Optimized, and Due Dates for Access Switch

The key for Table 1is as follows:

- "A" – Actual date that was manually scheduled
- "O" – Optimized date produced by model

- "D" – Due date
- "A-D" – Actual date occurred on Due date
- "A-O" – Actual and Optimized date are the same
- "A-O-D" – Actual, Optimized, and Due date are the same
- "O-D" – Optimized and Optimized date are the same

While the Access Switch scenario was small in scope, it produced an improved schedule. The weighted cost objective function for the manually-produced schedule was 59. The optimized model produced a weighted cost of 32 – a 40.68% reduction from the manually-produced schedule. This reduction translates into two less maintenance windows being used and all projects being completed 7 maintenance windows earlier.

While this validated the model, it didn't adequately reflect the substantial value the model has for Veritas. The reason is attributed to the limited size and relative simplicity of the projects. However, by transitioning the Access Switch model to LINGO, the research was able to expand to a Core Switch model. LINGO's advanced modeling capabilities enabled the model to account for all 42 projects that occurred in 65 maintenance windows during the second quarter of 2013, which can be seen in Table 2. While the same constraint categories were used in both models, the Core Switch added nearly 20 times the number of variables and 10 times the number of constraints.

	Maintenance Window																																																																
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64	65
Prj 1	[illegible]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	[illegible]	-	-	-	-	-	[illegible]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Prj 2	[illegible]	[illegible]	-	-	-	-	[illegible]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Prj 3	-	-	[illegible]	-	-	-	-	[illegible]	-	[illegible]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Prj 4	-	-	-	-	[illegible]	-	-	-	-	-	[illegible]	-	-	-	-	-	-	-	-	[illegible]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Prj 5	-	-	-	-	-	-	-	-	-	-	-	-	-	-	[illegible]	[illegible]	-	-	-	-	-	[illegible]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Prj 6	-	[illegible]	-	-	-	-	-	-	-	-	-	-	[illegible]	-	-	[illegible]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Prj 7	-	[illegible]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	[illegible]	-	-	-	-	-	-	-	-	[illegible]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Prj 8	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	[illegible]	[illegible]	-	-	-	-	-	-	-	[illegible]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Prj 9	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	[illegible]	-	-	[illegible]	-	-	-	-	-	-	-	-	-	-	-	[illegible]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Prj 10	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	[illegible]	[illegible]	-	-	-	-	-	-	-	-	[illegible]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Prj 11	-	-	-	-	-	-	-	-	-	-	-	[illegible]	-	-	-	-	-	-	-	-	[illegible]	-	-	-	-	-	-	-	-	-	-	-	-	-	[illegible]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Prj 12	-	-	-	-	-	-	-	-	-	-	[illegible]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	[illegible]	-	[illegible]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Prj 13	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	[illegible]	-	-	-	-	-	[illegible]	-	-	-	-	-	-	-	-	-	-	[illegible]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Prj 14	-	-	-	-	-	-	-	-	-	-	-	-	-	[illegible]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	[illegible]	-	-	-	-	-	[illegible]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Prj 15	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	[illegible]	-	-	-	-	-	-	-	-	-	-	-	-	-	[illegible]	-	-	-	-	-	-	[illegible]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Prj 16	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	[illegible]	-	-	-	-	-	-	-	-	-	-	-	-	-	[illegible]	-	-	-	-	-	-	-	-	[illegible]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Prj 17	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	[illegible]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	[illegible]	-	-	-	-	-	-	-	[illegible]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Prj 18	-	-	-	[illegible]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	[illegible]	-	-	-	-	-	-	[illegible]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Prj 19	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	[illegible]	-	-	-	-	-	-	-	-	-	-	[illegible]	-	-	-	-	-	-	-	-	-	[illegible]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Prj 20	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	[illegible]	-	-	-	-	[illegible]	-	-	-	-	-	-	-	-	-	-	-	-	-	[illegible]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Prj 21	-	-	[illegible]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	[illegible]	-	-	-	-	-	-	-	-	[illegible]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Prj 22	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	[illegible]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	[illegible]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	[illegible]	-	-	-	-	-	-	-	-	-	-
Prj 23	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	[illegible]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	[illegible]	-	[illegible]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Prj 24	-	[illegible]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	[illegible]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	[illegible]	-	-	-	-	-	-
Prj 25	-	-	-	-	-	-	-	-	-	-	[illegible]	-	-	-	-	-	-	-	-	[illegible]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	[illegible]
Prj 26	-	-	-	-	-	[illegible]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Prj 27	-	-	-	-	-	[illegible]	[illegible]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Prj 28	-	-	-	-	-	-	-	[illegible]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Prj 29	-	-	-	-	-	-	-	-	[illegible]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Prj 30	-	-	-	-	-	-	-	-	-	[illegible]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Prj 31	-	-	-	-	-	-	-	-	-	-	-	-	-	[illegible]	-	[illegible]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Prj 32	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	[illegible]	[illegible]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Prj 33	-	-	-	[illegible]	-	-	-	-	-	-	-	-	-	-	-	-	-	[illegible]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Prj 34	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	[illegible]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	[illegible]	[illegible]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Prj 35	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	[illegible]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	[illegible]	[illegible]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Prj 36	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	[illegible]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	[illegible]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Prj 37	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	[illegible]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	[illegible]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Prj 38	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	[illegible]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	[illegible]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Prj 39	-	[illegible]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	[illegible]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Prj 40	-	-	-	-	-	-	-	-	-	-	[illegible]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	[illegible]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Prj 41	-	-	-	-	-	-	-	-	-	-	-	[illegible]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	[illegible]	-	-	-	-	-	-	-	-	-
Prj 42	-	-	-	-	-	-	-	-	-	-	-	-	[illegible]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	[illegible]	-	-	-	-	-	-	-	-

Table 2 – Actual, Optimized, and Due Dates for Core Switch

The results from the use of the Core Switch model showed significant improvements over the project implementation schedule that actually took place. That schedule produced a weighted cost objective function value of 1,189 while the optimized schedule generated a schedule with a value of 567 – a 52.31% reduction from the manually-produced schedule. The outcome was that the optimized schedule completed all projects 28 work days earlier and utilized 17 less maintenance windows. The actual schedule also missed three project deadlines and completed 45% of the projects within two days of the due date. The optimized schedule, though, did not miss any deadlines and was within two dates of being due only 21% of the time.

All LINGO model optimized results were carefully inspected to ensure that no constraints were violated. LINGO was also able to complete the linear programming calculations in an average elapsed time of 0.3 seconds.

Further result details can be found in the Appendix section. Data includes the project specifications provided by Veritas, the LINGO model developed in this research, and the LINGO results obtained from that model.

Summary

The task of scheduling and implementing projects in Veritas can be considerably improved by the integration of linear programming models into day-to-day activities. Through the model developed in this research, Veritas can decrease the time required to develop a model, reduce the risk of scheduling overlapping projects that could damage the network, and improve project implementation efficiency.

The task of scheduling projects at Veritas switching centers can become a drastically simpler task. Instead of significant employee hours being consumed by manually creating a project schedule that protects network integrity, this research developed a model that can produce a schedule in remarkably short time.

The significance of the time reduction improvement can be illustrated through an example. In this example, a manager spends an average of five hours a week on creating a manually-produced schedule and has a salary of $100,000.00 a year. By incorporating the model developed in this research into day-to-day activities, the time needed to invest into scheduling becomes negligible. The five hours a week saved through the transition from manual scheduling will alleviate 260 hours, or 32.5 days, and $12,500.00 in administrative energies a year per manager that could be repurposed.

Another critical benefit of this research is its inherent ability to reduce risk. The purpose behind the hours spent by managers manually scheduling project activities is to protect network reliability and ensure customer experience is never impaired. By utilizing each project's specific requirements and impacts, the model achieves the necessary protection every time.

Finally, the model developed through this research produces superior scheduling efficiencies than the traditional manual scheduling methods. In both an Access Switch and Core Switch situation, the model produced drastic reductions in days required to complete all projects. The greater improvement, though, was found in the larger, more complicated Core Switch. Therefore, the implementation of this model into day-to-day activities will provide benefits to all Switch types, but it will provide the greatest benefits as the project scheduling situation becomes larger and more complex.

Chapter 5 – Suggestions for Further Work

While this research produced measurable improvements in both Access Switch and Core Switch models, there are several aspects of model development that merit attention. This chapter focused on three of those areas: improved user interface, additional project options, and expanded scope for multiple markets.

User Interface

In its current format, the linear programming model developed can only be easily modified by someone who is familiar with both linear programming concepts and the LINGO programming language. This must change to become a tool used in day-to-day operations by multiple departments.

The problem can be remedied through one of the capabilities provided by LINGO. It offers the ability to pull specific data out of a database or Excel spreadsheet. This data can then be used in the model calculations and output the produced results to another program. Since Excel is used almost universally in businesses, it would be a natural transition to modify the LINGO model to use data from a specific Excel spreadsheet to build the model and then output the result to the same or another spreadsheet.

Project Options

As a scheduling tool, another beneficial course of improvement would be to add more options for each project instead of only looking for the most compressed timeline possible. This would allow Veritas to better customize their project schedule.

One avenue that could be pursued is the option of prioritizing projects. Certain milestones or projects may be more critical to the business than others. Such an option would allow accounting for prioritization of objectives.

Also, some projects may have dependencies beyond those incorporated in the model in this research. There can also be slack or lag time between those projects. Accounting for these relationships could provide a better fit for some schedules.

Finally, this model assumes that other teams required for the project will be available on the date the model produces. This will not always be the case. As such, a useful addition to the model would be the ability to account for the resource availabilities of other teams.

Multiple Markets

The model developed through this research successfully optimizes a project schedule for a market. As described in the introduction, multiple markets can have an impact on each other. Therefore, additional constraints could be added to the model to account for the inter-relationships between markets and their related projects.

References

Anderson, David R., Dennis J. Sweeney, Thomas A. Williams, and Kipp Martin. *An Introduction to Management Science: Quantitative Approaches to Decision Making.* Twelfth Edition. Mason, Ohio: Thomason South-Western, 2008.

Bowlin, Tom. "Applications of Quantitative Analysis in Decision Making." *Engineering Management.* University of Kansas, Fall 2011.

Bülbül, Kerem, and Philip Kaminsky. "A Linear Programming-Based Method for Job Shop Scheduling." Journal of Scheduling 16.2 (2013): 161-83. ProQuest. Web. 15 Aug. 2013.

Djamel, Nait Tahar, et al. "A Linear Programming Approach for Identical Parallel Machine Scheduling with Job Splitting and Sequence-Dependent Setup Times." International Journal of Production Economics 99.1 (2006): 63-73. ProQuest. Web. 15 Aug. 2013.

Frontline Systems, Inc. "Standard Excel Solver - Dealing with Problem Size Limits." *Excel Solver, Optimization Software, Monte Carlo Simulation, Data Mining.* N.p., n.d. Web. 27 Sept. 2013. <http://www.solver.com/standard-excel-solver-dealing-problem-size-limits >

Fuller, Jack A. "Linear Programming Approach to Aggregate Scheduling." Academy of Management Journal (pre-1986) 18.1 (1975): 129. ProQuest. Web. 15 Aug. 2013.

LINDO SYSTEMS. "LINDO Systems - Optimization Software: Integer Programming, Linear Programming, Nonlinear Programming, Stochastic Programming, Global Optimization." *LINDO Systems - Optimization Software: Integer Programming, Linear Programming, Nonlinear Programming, Stochastic Programming, Global Optimization*. N.p., n.d. Web. 27 Sept. 2013. <http://www.lindo.com/>

Matsuoka, S., & Muraki, M. (2007). Short-term maintenance scheduling for utility systems. Journal of Quality in Maintenance Engineering, 13(3), 228-240. doi:http://dx.doi.org/10.1108/13552510710780267

Taylor, R. W. "A Linear Programming Model to Manage the Maintenance Backlog." Omega 24.2 (1996): 217. ProQuest. Web. 15 Aug. 2013.

Appendix

The purpose of the appendix is to provide more granular data to assist in understanding how the developed model was implemented. This will be done in three sections. The first will provide the Veritas data needed for each project. The next section will contain the LINGO model that was created to derive the optimized schedule. Finally, the last section will provide the results obtained by LINGO.

The data provided only pertains to the Access Switch "proof of concept" scenario. The addition of the Core Switch scenario would not have provided further value to the understanding of this research and would have increased the size of this report by 95 pages.

Veritas Project Data

The data provided in this section will serve as constants in the model developed through this research. This data includes specifications for each project and market resource availability for each maintenance window over the three week timespan.

Project	Due Date	Hrs in MW	Res Required
Project 1	15	5	1
Project 2	10	5	1
Project 3	3	2	1
Project 4	12	5	1
Project 5	14	5	1
Project 6	5	5	1
Project 7	5	2	1

Project 8	4	3	1

Table 3 – Due date, hours required, and resources required for each project

Project	Tech1	Tech2	Tech3	Tech4
Project 1	1	1	1	0
Project 2	1	1	1	1
Project 3	0	0	0	0
Project 4	0	1	0	0
Project 5	0	1	0	0
Project 6	0	0	0	0
Project 7	0	0	0	1
Project 8	1	1	1	1

Table 4 – Technologies impacted by each project (binary constants)

Project	Device1	Device2	Device3	Device4
Project 1	0	0	1	0
Project 2	0	1	0	0
Project 3	0	0	1	0
Project 4	0	0	1	0
Project 5	0	0	0	0
Project 6	0	0	0	0
Project 7	0	0	0	0
Project 8	1	0	0	0

Table 5 – Devices impacted by each project (binary constants)

Maintenance Window	Resources Available
1	1
2	1
3	1
4	1
5	1
6	2
7	2
8	2
9	2
10	2
11	1

12	1
13	1
14	1
15	1

Table 6 – Resources available for each maintenance window

LINGO Model

```
MODEL:

SETS:
Dates: MW;
projects: proj_num;
constants;
PxV(projects,constants): Veritas_Data;
MktRes: res_avail;
PxD(projects,Dates): V_schedule;
ENDSETS

DATA:
projects = 1..8;
proj_num = 1 2 3 4 5 6 7 8;
Dates = 1..15;
MW = 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15;
MktRes = 1..15;
res_avail = 1 1 1 1 1 2 2 2 2 2 1 1 1 1 1;

constants = DUE RES_REQ HOURS TECH_1 TECH_2 TECH_3 TECH_4 DEVICE_1
DEVICE_2 DEVICE_3 DEVICE_4;
DUE = 1;
RES_REQ = 2;
HOURS = 3;
TECH_1 = 4;
TECH_2 = 5;
TECH_3 = 6;
TECH_4 = 7;
DEVICE_1 = 8;
DEVICE_2 = 9;
DEVICE_3 = 10;
DEVICE_4 = 11;

Veritas_Data =
15 1 5 1 1 1 0 0 0 1 0
10 1 5 1 1 1 1 0 1 0 0
3  1 2 0 0 0 0 0 0 1 0
12 1 5 0 1 0 0 0 0 1 0
14 1 5 0 1 0 0 0 0 0 0
5  1 5 1 0 0 0 0 0 0 0
5  1 2 0 0 0 1 0 0 0 0
4  1 3 1 1 1 1 1 0 0 0;
ENDDATA
```

```
!Constraint#1;
 @FOR(Dates(date_index):
!There cannot be projects affecting TECH_1 occurring at the same time;
 @SUM(projects(proj_index):
V_schedule(proj_index,date_index)*Veritas_Data(proj_index,TECH_1))
 <= 1;
 !There cannot be projects affecting TECH_2 occurring at the same time;
 @SUM(projects(proj_index):
V_schedule(proj_index,date_index)*Veritas_Data(proj_index,TECH_2))
 <= 1;
 !There cannot be projects affecting TECH_3 occurring at the same time;
 @SUM(projects(proj_index):
V_schedule(proj_index,date_index)*Veritas_Data(proj_index,TECH_3))
 <= 1;
 !There cannot be projects affecting TECH_4 occurring at the same time;
 @SUM(projects(proj_index):
V_schedule(proj_index,date_index)*Veritas_Data(proj_index,TECH_4))
 <= 1);

!Constraint#2;
 @FOR(Dates(date_index):
 !There cannot be projects affecting DEVICE_1 occurring at the same
time;
 @SUM(projects(proj_index):
V_schedule(proj_index,date_index)*Veritas_Data(proj_index,DEVICE_1))
 <= 1;
 !There cannot be projects affecting DEVICE_2 occurring at the same
time;
 @SUM(projects(proj_index):
V_schedule(proj_index,date_index)*Veritas_Data(proj_index,DEVICE_2))
 <= 1;
 !There cannot be projects affecting DEVICE_3 occurring at the same
time;
 @SUM(projects(proj_index):
V_schedule(proj_index,date_index)*Veritas_Data(proj_index,DEVICE_3))
 <= 1;
 !There cannot be projects affecting DEVICE_4 occurring at the same
time;
 @SUM(projects(proj_index):
V_schedule(proj_index,date_index)*Veritas_Data(proj_index,DEVICE_4))
 <= 1);

!Constraint #3
  Prevent overloading of market resources by limiting the number of
projects
  based on hours required to complete;
 @FOR(Dates(date_index):
 @SUM(projects(proj_index): V_schedule(proj_index,date_index)*
      Veritas_Data(proj_index,RES_REQ)*Veritas_Data(proj_index,HOURS))
 <= 5*res_avail(date_index));

!Constraint #4
  Relaxation technique to reduce variables.  This allows for a set
amount of hours for
  maintenance or small projects in a given week;
 !Week 1;
 @SUM(Dates(date_index)|date_index #LT#6: 5*res_avail(date_index)) -
 @SUM(PxD(proj_index,date_index)|date_index #LT#6:
      V_schedule(proj_index,date_index)*Veritas_Data(proj_index,HOURS))
 >= 5;
!Week 2;
 @SUM(Dates(date_index)|date_index #GT# 5 #OR# date_index #LT# 11:
5*res_avail(date_index)) -
 @SUM(PxD(proj_index,date_index)|date_index #LT#6:
      V_schedule(proj_index,date_index)*Veritas_Data(proj_index,HOURS))
 >= 5;
!Week 3;
 @SUM(Dates(date_index)|date_index #GT# 10: 5*res_avail(date_index)) -
 @SUM(PxD(proj_index,date_index)|date_index #LT#6:
      V_schedule(proj_index,date_index)*Veritas_Data(proj_index,HOURS))
 >= 5;

!Constraint #5
 Projects must be completed- Still need to set the bounds;
 @FOR(projects(proj_index):
 @SUM(Dates(date_index)| date_index #LT# Veritas_Data(proj_index,DUE):
      V_schedule(proj_index,date_index))
 >= 1);

!Constraint#6;
 !V_schedule must be binary;
 @FOR(PxD(I,J): @BIN(V_schedule(I,J)));

!OBJECTIVE FUNCTION;
MIN = @SUM(Dates(date_index):
            @SUM(PxD(proj_index, date_index):
MW(date_index)*V_schedule(proj_index,date_index)));
END
```

LINGO Model Results

```
  Global optimal solution found.
  Objective value:                              32.00000
  Objective bound:                              32.00000
  Infeasibilities:                              0.000000
  Extended solver steps:                               0
  Total solver iterations:                            89
  Elapsed runtime seconds:                          0.06

  Model Class:                                      PILP

  Total variables:                    120
  Nonlinear variables:                  0
  Integer variables:                  120

  Total constraints:                  147
  Nonlinear constraints:                0

  Total nonzeros:                     720
  Nonlinear nonzeros:                   0
```

Variable	Value	Reduced_Cost
DUE	1	0
RES_REQ	2	0
HOURS	3	0
TECH_1	4	0
TECH_2	5	0
TECH_3	6	0
TECH_4	7	0
DEVICE_1	8	0
DEVICE_2	9	0
DEVICE_3	10	0
DEVICE_4	11	0
MW(1)	1	0
MW(2)	2	0
MW(3)	3	0
MW(4)	4	0
MW(5)	5	0
MW(6)	6	0
MW(7)	7	0
MW(8)	8	0
MW(9)	9	0
MW(10)	10	0
MW(11)	11	0
MW(12)	12	0
MW(13)	13	0
MW(14)	14	0
MW(15)	15	0
PROJ_NUM(1)	1	0
PROJ_NUM(2)	2	0
PROJ_NUM(3)	3	0
PROJ_NUM(4)	4	0
PROJ_NUM(5)	5	0
PROJ_NUM(6)	6	0
PROJ_NUM(7)	7	0
PROJ_NUM(8)	8	0
VERITAS_DATA(1,DUE)	15	0
VERITAS_DATA(1,RES_REQ)	1	0
VERITAS_DATA(1,HOURS)	5	0
VERITAS_DATA(1,TECH_1)	1	0
VERITAS_DATA(1,TECH_2)	1	0

VERITAS_DATA(1,TECH_3)	1	0
VERITAS_DATA(1,TECH_4)	0	0
VERITAS_DATA(1,DEVICE_1)	0	0
VERITAS_DATA(1,DEVICE_2)	0	0
VERITAS_DATA(1,DEVICE_3)	1	0
VERITAS_DATA(1,DEVICE_4)	0	0
VERITAS_DATA(2,DUE)	10	0
VERITAS_DATA(2,RES_REQ)	1	0
VERITAS_DATA(2,HOURS)	5	0
VERITAS_DATA(2,TECH_1)	1	0
VERITAS_DATA(2,TECH_2)	1	0
VERITAS_DATA(2,TECH_3)	1	0
VERITAS_DATA(2,TECH_4)	1	0
VERITAS_DATA(2,DEVICE_1)	0	0
VERITAS_DATA(2,DEVICE_2)	1	0
VERITAS_DATA(2,DEVICE_3)	0	0
VERITAS_DATA(2,DEVICE_4)	0	0
VERITAS_DATA(3,DUE)	3	0
VERITAS_DATA(3,RES_REQ)	1	0
VERITAS_DATA(3,HOURS)	2	0
VERITAS_DATA(3,TECH_1)	0	0
VERITAS_DATA(3,TECH_2)	0	0
VERITAS_DATA(3,TECH_3)	0	0
VERITAS_DATA(3,TECH_4)	0	0
VERITAS_DATA(3,DEVICE_1)	0	0
VERITAS_DATA(3,DEVICE_2)	0	0
VERITAS_DATA(3,DEVICE_3)	1	0
VERITAS_DATA(3,DEVICE_4)	0	0
VERITAS_DATA(4,DUE)	12	0
VERITAS_DATA(4,RES_REQ)	1	0
VERITAS_DATA(4,HOURS)	5	0
VERITAS_DATA(4,TECH_1)	0	0
VERITAS_DATA(4,TECH_2)	1	0
VERITAS_DATA(4,TECH_3)	0	0
VERITAS_DATA(4,TECH_4)	0	0
VERITAS_DATA(4,DEVICE_1)	0	0
VERITAS_DATA(4,DEVICE_2)	0	0
VERITAS_DATA(4,DEVICE_3)	1	0
VERITAS_DATA(4,DEVICE_4)	0	0
VERITAS_DATA(5,DUE)	14	0
VERITAS_DATA(5,RES_REQ)	1	0

VERITAS_DATA(5,HOURS)	5	0
VERITAS_DATA(5,TECH_1)	0	0
VERITAS_DATA(5,TECH_2)	1	0
VERITAS_DATA(5,TECH_3)	0	0
VERITAS_DATA(5,TECH_4)	0	0
VERITAS_DATA(5,DEVICE_1)	0	0
VERITAS_DATA(5,DEVICE_2)	0	0
VERITAS_DATA(5,DEVICE_3)	0	0
VERITAS_DATA(5,DEVICE_4)	0	0
VERITAS_DATA(6,DUE)	5	0
VERITAS_DATA(6,RES_REQ)	1	0
VERITAS_DATA(6,HOURS)	5	0
VERITAS_DATA(6,TECH_1)	1	0
VERITAS_DATA(6,TECH_2)	0	0
VERITAS_DATA(6,TECH_3)	0	0
VERITAS_DATA(6,TECH_4)	0	0
VERITAS_DATA(6,DEVICE_1)	0	0
VERITAS_DATA(6,DEVICE_2)	0	0
VERITAS_DATA(6,DEVICE_3)	0	0
VERITAS_DATA(6,DEVICE_4)	0	0
VERITAS_DATA(7,DUE)	5	0
VERITAS_DATA(7,RES_REQ)	1	0
VERITAS_DATA(7,HOURS)	2	0
VERITAS_DATA(7,TECH_1)	0	0
VERITAS_DATA(7,TECH_2)	0	0
VERITAS_DATA(7,TECH_3)	0	0
VERITAS_DATA(7,TECH_4)	1	0
VERITAS_DATA(7,DEVICE_1)	0	0
VERITAS_DATA(7,DEVICE_2)	0	0
VERITAS_DATA(7,DEVICE_3)	0	0
VERITAS_DATA(7,DEVICE_4)	0	0
VERITAS_DATA(8,DUE)	4	0
VERITAS_DATA(8,RES_REQ)	1	0
VERITAS_DATA(8,HOURS)	3	0
VERITAS_DATA(8,TECH_1)	1	0
VERITAS_DATA(8,TECH_2)	1	0
VERITAS_DATA(8,TECH_3)	1	0
VERITAS_DATA(8,TECH_4)	1	0
VERITAS_DATA(8,DEVICE_1)	1	0
VERITAS_DATA(8,DEVICE_2)	0	0
VERITAS_DATA(8,DEVICE_3)	0	0

VERITAS_DATA(8,DEVICE_4)	0	0
RES_AVAIL(1)	1	0
RES_AVAIL(2)	1	0
RES_AVAIL(3)	1	0
RES_AVAIL(4)	1	0
RES_AVAIL(5)	1	0
RES_AVAIL(6)	2	0
RES_AVAIL(7)	2	0
RES_AVAIL(8)	2	0
RES_AVAIL(9)	2	0
RES_AVAIL(10)	2	0
RES_AVAIL(11)	1	0
RES_AVAIL(12)	1	0
RES_AVAIL(13)	1	0
RES_AVAIL(14)	1	0
RES_AVAIL(15)	1	0
V_SCHEDULE(1,1)	0	1
V_SCHEDULE(1,2)	0	2
V_SCHEDULE(1,3)	0	3
V_SCHEDULE(1,4)	1	4
V_SCHEDULE(1,5)	0	5
V_SCHEDULE(1,6)	0	6
V_SCHEDULE(1,7)	0	7
V_SCHEDULE(1,8)	0	8
V_SCHEDULE(1,9)	0	9
V_SCHEDULE(1,10)	0	10
V_SCHEDULE(1,11)	0	11
V_SCHEDULE(1,12)	0	12
V_SCHEDULE(1,13)	0	13
V_SCHEDULE(1,14)	0	14
V_SCHEDULE(1,15)	0	15
V_SCHEDULE(2,1)	0	1
V_SCHEDULE(2,2)	0	2
V_SCHEDULE(2,3)	0	3
V_SCHEDULE(2,4)	0	4
V_SCHEDULE(2,5)	0	5
V_SCHEDULE(2,6)	0	6
V_SCHEDULE(2,7)	1	7
V_SCHEDULE(2,8)	0	8
V_SCHEDULE(2,9)	0	9
V_SCHEDULE(2,10)	0	10

V_SCHEDULE(2,11)	0	11
V_SCHEDULE(2,12)	0	12
V_SCHEDULE(2,13)	0	13
V_SCHEDULE(2,14)	0	14
V_SCHEDULE(2,15)	0	15
V_SCHEDULE(3,1)	1	1
V_SCHEDULE(3,2)	0	2
V_SCHEDULE(3,3)	0	3
V_SCHEDULE(3,4)	0	4
V_SCHEDULE(3,5)	0	5
V_SCHEDULE(3,6)	0	6
V_SCHEDULE(3,7)	0	7
V_SCHEDULE(3,8)	0	8
V_SCHEDULE(3,9)	0	9
V_SCHEDULE(3,10)	0	10
V_SCHEDULE(3,11)	0	11
V_SCHEDULE(3,12)	0	12
V_SCHEDULE(3,13)	0	13
V_SCHEDULE(3,14)	0	14
V_SCHEDULE(3,15)	0	15
V_SCHEDULE(4,1)	0	1
V_SCHEDULE(4,2)	0	2
V_SCHEDULE(4,3)	0	3
V_SCHEDULE(4,4)	0	4
V_SCHEDULE(4,5)	0	5
V_SCHEDULE(4,6)	0	6
V_SCHEDULE(4,7)	0	7
V_SCHEDULE(4,8)	1	8
V_SCHEDULE(4,9)	0	9
V_SCHEDULE(4,10)	0	10
V_SCHEDULE(4,11)	0	11
V_SCHEDULE(4,12)	0	12
V_SCHEDULE(4,13)	0	13
V_SCHEDULE(4,14)	0	14
V_SCHEDULE(4,15)	0	15
V_SCHEDULE(5,1)	0	1
V_SCHEDULE(5,2)	0	2
V_SCHEDULE(5,3)	0	3
V_SCHEDULE(5,4)	0	4
V_SCHEDULE(5,5)	0	5
V_SCHEDULE(5,6)	1	6

V_SCHEDULE(5,7)	0	7
V_SCHEDULE(5,8)	0	8
V_SCHEDULE(5,9)	0	9
V_SCHEDULE(5,10)	0	10
V_SCHEDULE(5,11)	0	11
V_SCHEDULE(5,12)	0	12
V_SCHEDULE(5,13)	0	13
V_SCHEDULE(5,14)	0	14
V_SCHEDULE(5,15)	0	15
V_SCHEDULE(6,1)	0	1
V_SCHEDULE(6,2)	1	2
V_SCHEDULE(6,3)	0	3
V_SCHEDULE(6,4)	0	4
V_SCHEDULE(6,5)	0	5
V_SCHEDULE(6,6)	0	6
V_SCHEDULE(6,7)	0	7
V_SCHEDULE(6,8)	0	8
V_SCHEDULE(6,9)	0	9
V_SCHEDULE(6,10)	0	10
V_SCHEDULE(6,11)	0	11
V_SCHEDULE(6,12)	0	12
V_SCHEDULE(6,13)	0	13
V_SCHEDULE(6,14)	0	14
V_SCHEDULE(6,15)	0	15
V_SCHEDULE(7,1)	0	1
V_SCHEDULE(7,2)	0	2
V_SCHEDULE(7,3)	1	3
V_SCHEDULE(7,4)	0	4
V_SCHEDULE(7,5)	0	5
V_SCHEDULE(7,6)	0	6
V_SCHEDULE(7,7)	0	7
V_SCHEDULE(7,8)	0	8
V_SCHEDULE(7,9)	0	9
V_SCHEDULE(7,10)	0	10
V_SCHEDULE(7,11)	0	11
V_SCHEDULE(7,12)	0	12
V_SCHEDULE(7,13)	0	13
V_SCHEDULE(7,14)	0	14
V_SCHEDULE(7,15)	0	15
V_SCHEDULE(8,1)	1	1
V_SCHEDULE(8,2)	0	2

V_SCHEDULE(8,3)	0	3
V_SCHEDULE(8,4)	0	4
V_SCHEDULE(8,5)	0	5
V_SCHEDULE(8,6)	0	6
V_SCHEDULE(8,7)	0	7
V_SCHEDULE(8,8)	0	8
V_SCHEDULE(8,9)	0	9
V_SCHEDULE(8,10)	0	10
V_SCHEDULE(8,11)	0	11
V_SCHEDULE(8,12)	0	12
V_SCHEDULE(8,13)	0	13
V_SCHEDULE(8,14)	0	14
V_SCHEDULE(8,15)	0	15