Data Perturbation Analysis for IS Project Management Based on a Single Time Estimate

Hossein Arsham, Merrick School of Business, University of Baltimore, Baltimore, MD, USA

ABSTRACT

The Critical Path Method (CPM) is the most widely used tool for project management; however, it requires three estimates for the duration of each activity as its input. This is too much uncertain input requirement from the managers, making the critical path (CP) unstable causing major difficulties for the manager. A linear programming formulation of the project network is proposed for determining a CP, based on making one estimate for the duration of each activity. Upon finding the CP, data perturbation analysis (DPA) is performed using the constraints of the dual problem. This DPA set of uncertainties provides the manager with a tool to deal with the simultaneous, independent, or dependent changes of the input estimates that will preserve the current CP. The proposed procedure is easy to understand, easy to implement, and provides useful information for the manager. A numerical example illustrates the process.

Keywords: Critical Path Method (CPM), Data Perturbation Analysis, Enterprise Resource Planning, Largest Sensitivity Region, Linear Program, LP Degeneracy, Project Evaluation and Review Technique (PERT), Project Management

INTRODUCTION AND MOTIVATIONS

Project management is one of the fastest growing career fields in business today. Most of the growth in this field is in the information systems area, where there are widespread reports about most projects being late, many over budget, and all too often not satisfying design specifications. This paper is about information systems project management, although the principles apply to projects in any field. When proposing a new information system, the systems analyst will be confronted with many questions from top management, in particular “How much will it cost? And “When will it be done?” As many project managers know, these two questions are difficult to answer correctly.

A project involves getting a new, complex activity accomplished. Projects are purposeful, in that they are designed to accomplish something for the organization undertaking them. Because projects involve new activities, they typically involve high levels of uncertainty and risk. It is very difficult to predict what problems are going to occur in system development. There have been runaway information system projects. Barki, Rivard, and Talbot (1993) cited...
runaway information system project examples. In 1982, a major insurance company began development of an $8 million computer system from a major software provider. This system was intended to serve all of the computing needs of the insurance company and was due to be completed in 1987. However, a number of problems were encountered resulting in delays of completion until 1993, with a new estimated cost of $100 million. Pirdashti, Mohammadi, Rahimpour, and Kennedy (2008) used Delphi technique to elicit expert opinions about criteria for evaluating the network locations for a network design project.

Projects are systems consisting of interrelated parts working together to accomplish project objectives. There are a number of important roles within information systems projects. Project managers have to balance technical understanding with the ability to motivate diverse groups of people (the project team) brought together on a temporary basis. Projects are collections of activities. If one activity is late, other activities are delayed. If an activity is ahead of schedule, workers tend to slow down to meet the original completion date. Information systems projects have many similarities to generic projects. They consist of activities, each with durations, predecessor relationships, and resource requirements. They involve high levels of uncertainty and often suffer from time and cost overruns, while rarely experiencing time and cost under runs. However, information systems projects are different from generic projects in some aspects. While each project is unique, there are usually many, many replications of information systems project types. Most are served by a standard methodology, with the need to identify user requirements, followed by design of a system, production of the system, testing of the system, training and implementation, and, ultimately, maintenance of the system. These steps are not always serial; there are often many loops back to prior stages.

Defining project success is in itself difficult. There are many views of what makes a project successful. Successful implementation has been found to require mastery of the technical aspects of systems, along with understanding key organizational and behavioral dynamics. There has been a great deal of research into information systems project failure. Failure can occur when design objectives are not met. The difference between successful and failed information systems projects often lies in planning and implementation. A great deal of research has been performed to identify factors that lead to project success. These factors include planning, user involvement, and good communication. Additional factors that are reported as important in information systems project success repeatedly include top management support and clear statement of project objectives.

Information systems project management can involve a wide variety of tasks. Typical information systems project types include maintenance work, conversion projects, and new systems implementation. Maintenance projects are by far the most common type of information systems project. They can arise from need to fix errors or to add enhancements to some system, or they can involve major enhancements. Conversion projects involve changing an existing system. Most conversion projects consist of moving an application from one computer platform to another. New systems development involves different management characteristics by type of system. The types of systems include transaction processing, management control, decision support systems, data warehousing and data mining, group support system, executive information systems, Internet commerce, and enterprise resource planning (ERP).

Ahmad, Zakaria, and Sedera (2011) state that companies face the challenges of expanding their markets, improving products, services and processes, and exploiting intellectual capital in a dynamic network. Therefore, more companies are turning to an enterprise system for support across the entire lifecycle, from selection and implementation to use. Chen and Wang (2012) are aware of numerous difficulties associated with learning in a project. For facilitating knowledge learning, an integrated project management model is constructed. Hanafizadeh, Gholami,
Dadbin, and Standage (2010) also considered the implementation of enterprise resource planning systems, requiring huge investments, while ineffective implementations of such projects are commonly observed. A considerable number of these projects have been reported to fail or take longer than initially planned, while previous studies show that the aim of rapid implementation of such projects has not been successful; and the failure of the fundamental goals in these projects have imposed huge amounts of costs on investors. Sternad, Bobek, Dezelak, and Lampret (2009) are aware of the facts that ERP implementation is a complex process that requires substantial resources and efforts, and yet the results are very uncertain. Lai (2006) rightly emphasizes that it is very important to identify what are the key factors across different steps within the ERP implementation. Zarei and Naeli (2010) give warning that it may entail new hazardous challenges if enterprise resource planning cannot be well managed to achieve success. Critical success factors are project management, top management support, business process reengineering, change management, and training. Orlowski and Kowalczyk (2006) explicitly outline the difficulties involved in software project management, in particular regarding the planning and control of processes and the project-team manager. Agile methodology is an approach to project management software development for dealing with business realities, such as changing requirements during development (Kendall, Kong, & Kendall, 2010).

Information systems projects have high levels of uncertainty. The size of the project is usually not well understood until systems analysis has been completed. Most of the unexpected delay in these projects occurs during the latter stages of testing. Almost one third of the time used in typical projects was required for the planning phase, and coding typically consisted of one-sixth of the project. Coding is the most predictable portion of the project. The last half of the project was testing — one-quarter for component testing and one-quarter for system testing. The activity most difficult to predict was testing. Project managers currently use the critical path method (CPM) and/or project evaluation and review technique (PERT) in order to provide a planning and control structure for projects. CPM helps managers understand the relationships among project activities (see a sample of general project management textbooks and those with MIS applications included in the References section). Key personnel and other resources can be allocated to activities on the critical path. These activities can be closely monitored to avoid completion delays.

CPM identifies the sequence of activities that will have the longest completion time of the entire project. PERT extends CPM’s scope to deal with uncertainties inherent in any project. Project activity network models ranked above all other quantitative decision making tools in terms of the percentage of firms who used them. Although these techniques are widely used in practice and have provided economic benefit to their users, they are not problem-free. Problems derive from the nature of the underlying statistical assumptions, the potential for high computational expense, and from the assumption that factors determining activity duration are essentially probabilistic rather than determined by managerial action.

**Critique of CPM/PERT**

Here “critique” means the scope and limitations. The most widely used CPM/PERT algorithm is based on the following assumptions and procedure:

- Activity duration is rarely (if ever) known with certainty; the CPM asks the manager for three time estimates;
- Three time estimates are determined (by guesses, etc.) for each activity:
  - a = an optimistic completion time;
  - m = a most likely completion time (this is what we will use in this paper);
  - b = a pessimistic completion time;
- Activity approximations;
An approximation for the distribution of an activity’s completion time is a BETA distribution;

An approximation for the mean completion time for an activity is a weighted average \((1/6, 4/6, 1/6)\) of the three completion times; so it is \((a + 4m + b)/6\);

An approximation for the standard deviation for the completion time for an activity is its Range/6 or \((b – a)/6\);

The variance of an activity’s completion time is the square of the standard deviation = \([(b – a)/6]\)^2;

Project assumptions;

1. The distribution of the project completion time is determined by the critical path using the mean activity completion times;

2. The activity completion times are independent;

3. There are enough activities (at least 30, to make sure) on the critical path so that the central limit can be used to determine the distribution, mean, variance and standard deviation of the project;

Project distribution. Given the above assumptions, this means;

1. The project completion time distribution is normal;

2. The mean (expected) completion time, \(\mu\), of the project is the sum of the expected completion times along the critical path;

3. The variance of the completion time, \(\sigma^2\), of the project is the sum of the variance in completion times along the critical path;

4. The standard deviation of the completion time, \(\sigma\), of the project is the square root of the variance of the completion time of the project.

The critical path method requires three time estimates from the manager for each activity in the project. In a dialog with this author, a manager said that it is difficult enough to offer one estimate for the duration of one activity. By asking for three estimates for each activity in the project, more uncertainty is introduced. It makes sense to the manager to ask for one good estimate.

An alternative to the CPM/PERT is a stochastic activity network, Arsham (1992). This approach assumes the same project is to be performed repeatedly, and there is full knowledge of the time distribution function to be used in Monte-Carlo simulation experiments to approximate the expected CPM/PERT duration. This approach suffers from even more severe practical pitfalls, in addition to being computationally expensive.

The validity of the above assumptions has been a long-standing question, Kallo (1996), Kamburowski, (1997), Keefer and Verdini (1993), and Kuklan (1993). The solution based on inaccurate inputs result in the incorrect designation of the CPM/PERT, and the accuracy of the results decreases in relation to increased project complexity (Nakashima, 1999).

While Hasan and Gould (2001) support the sense-making activity of managers, more than half a century after the debut of CPM and PERT, they still are requiring complex user input. While, modern decision support systems for project management are more sophisticated and comprehensive than CPM/PERT, show insufficient progress in dealing with uncertainties (Trietsch & Baker, 2012). Czuchra (1999) also has some recommendations on optimizing budget spending for software implementation and testing. Yaghoubi, Noori, Azaron, and Tavakkoli-Moghaddam (2011) consider dynamic PERT networks where activity durations are unrealistically independent random variables with exponential distributions. Mouhoub, Benhocine, and Belouadah (2011), proposed to reduce the number of dummy activities as much as possible, but the result is even more complex. Herrerias -Velasco, Herrerias -Pleguezuelo, and René van Dorp (2011), being aware of the dif-
Difficulties with the interpretation of the parameters of the beta distribution, suggest an alternative to the PERT variance expression regarding the constant PERT variance assumption.

Martin et al. (2012), proposed a two-sided power and the generalized biparabolic distributions as an alternative to the mixture of the uniform and the beta distributions to the beta distribution in PERT methodology.

Yakhchali (2012) expands the work of Nasution (1994), while addressing the problem of determining the degree of possible and necessary criticality of activities, as well as determining paths in networks that have fuzzy activity durations.

Oke et al. (2009) applied neurofuzzy (i.e., uncertainty representation and fuzzy inferences) to a maintenance scheduling network that involves selection of alternate preventive maintenance schedules and operations.

D’Aquila (1993) recommends simultaneous use of CPM/PERT. The two assumptions that (a) activity times are mutually independent random variables and (b) path completion times are mutually independent. Mehrotra, Chai, and Pillutla (1996) suggest approximating the moments of the job completion. Banerjee and Paul (2008), use multivariate statistical tools analysis to measure the error in the classical PERT method of estimating mean project completion time when correlation is ignored. Premachandra (2001) yet provides approximations for the mean and the variance of activity based on “pessimistic,” “optimistic,” and “most likely” time estimates to get away from the beta distribution assumption.

Even ordinary sensitivity analysis is rarely performed in activity network projects, because the existing CPM/PERT algorithms do not contain enough information to perform the calculations necessary for this simplest form of Data Perturbation Analysis (DPA). To the best of our knowledge NETSOLVE is the only software available capable of dealing with ordinary sensitivity analysis for CPM (Jarvis & Shier, 1990; Phillips & Garcia-Diaz, 1990). The existing stochastic PERT software packages require a separate run for each scenario if any parameters are changed (Higgs, 1995; Lewis, 2007).

The probabilistic nature in both PERT models assumes random factors (e.g., bad luck, good luck) are present and does not allow for any managerially planned point of view to deal with uncertainty. In other words, much of the variability for activity duration may result from management decisions, rather than from random acts of nature.

**An Alternative Computational Approach**

This paper discusses a non-statistical approach, referred to as Data Perturbation Analysis (DPA), to calculate a variety of activity duration uncertainties. DPA deals with a collection of managerial questions related to uncontrollable environmental factors in project estimation tasks. The underlying mathematics of the DPA is a linear program based formulation. This approach provides:

1. an assessment and analysis of the stability of the critical path(s) under uncertainty;
2. monitoring of the admissible range of activity durations that preserve the current CP;
3. disclosure of the useful limits on simultaneous, dependent departures from the activity duration estimates to determine maximum “crashing” of critical activity durations, the maximum “slippage” for non-critical activity durations; and the impacts of such departures on the entire project completion time.

These benefits allow the manager more leverage in allocating project resources. Knowing the stability of the critical path(s) under uncertainty allows the manager to perceive a range of timing changes before a new path of activities
is critical. Monitoring the admissible range of activity durations that preserve the current CP aids in determining how resources should be applied among the activities to expedite the completion of the project and to smooth out workloads. Disclosure of information about departure times for critical and non-critical activities allows the manager to anticipate the consequences of slippage wherever it might occur in the project and to pre-determine back-up practices.

This paper develops a simplex-type solution algorithm to find the CP. The algorithm is easy to use and does not introduce any slack or surplus variables (as in the dual simplex method), or any artificial variables (Arsham, 1997a, 1997b); therefore it is computationally practical and stable to implement. A unified approach is presented to cover all aspects of the DPA.

The remainder of this article is divided into five parts. First, based on one estimate for the duration of each activity, the CP is obtained by a linear program formulation and an efficient solution algorithm. This follows by an illustrated numerical example. The data manipulation leading to the CP provides the necessary information for the Data Perturbation Analysis (DPA) that provides the largest sensitivity region, in the next section. This is followed by the parametric analysis, which includes the ordinary sensitivity analysis and the so-called 100% rule. Then the tolerance analysis is developed and, as its by-products, we obtain the individual and symmetric tolerance analysis together with a discussion of some potential applications. The last section is devoted to some concluding remarks. Throughout the paper the emphasis is on constructive procedures, proofs and a small numerical example since these lead directly to an efficient computer program for implementation. Furthermore, this algorithm facilitates PDA, including structural changes in the nominal project network.

LINEAR PROGRAM FORMULATION WITH A NEW SOLUTION TECHNIQUE

Suppose that in a given project activity network, there are m nodes, n arcs (i.e., activities) and an estimated duration, tij, associated with each arc (ij) in the network. Without loss of generality, it is assumed that the activities durations are continuous. The beginning node of an arc corresponds to the start of the associated activity and the end node to its completion. To find the CP, define the binary variables Xij, where Xij = 1, if the activity ij is on the CP, and Xij = 0 otherwise. The length of the path is the sum of the durations of the activities on the path. Formally, the CP problem is to find the longest path from node 1 to node m, i.e.:

Maximize \[ \sum_{i=1}^{m} \sum_{j=1}^{m} t_{ij} X_{ij} \]
subject to:
- \[ X_{ij} = 1 \]
- \[ X_{ij} = 0 \] for \( i \neq 1 \) or \( m \)
- \[ \sum_{i=1}^{m} X_{ij} = 1 \]
- \[ \sum_{k=1}^{m} X_{ij} = 0 \] for \( j \neq 1 \) or \( m \)

where the sums are taken over existing arcs in the network. The first and the last constraints are imposed to start (node 1) and complete the project (node m) by critical activities, respectively; while the other constraints provide that if any node is arrived at by a critical activity, then it must be left by a critical activity. Note that the integrality conditions (i.e., \( X_{ij} = 0 \) or 1) are changed to
X_{ij} \geq 0 since it is known that the optimal solution to these types of LP problems satisfy these conditions. Note also that one of these m constraints is redundant; e.g., the first constraint is the sum of all other constraints.

The Critical Path Finder

The following notation is used in the new algorithm:

- BVS - Basic Variable Set
- GJP - Gauss-Jordan Pivoting
- PR - Pivot Row (Row to be assigned to the variable to come in BVS)
- PC - Pivot Column (Column associated with variable to come in BVS)
- PE - Pivot Element
- OR - Open Row. A row not yet assigned to a variable. Labeled (?).
- (?) - Label for a row that is not yet assigned a variable (Open Row)
- RHS - Right Hand Side
- C/R - Column Ratio, RHS/PE
- R/R - Row Ratio, Last row/PR

The algorithm consists of preliminaries for setting up the initialization followed by three main phases: Basic Variable Set Augmentation, Optimality, and Feasibility. The Basic Variable Set Augmentation Phase develops a basic variable set (BVS) which may or may not be feasible. Unlike simplex and dual simplex, this approach starts with an incomplete BVS initially, and then variables are brought into the basis one by one. This strategy pushes towards an optimal solution. Since some solutions generated may be infeasible, the next step, if needed, pulls the solution back to feasibility. The Optimality Phase satisfies the optimality condition, and the Feasibility Phase obtains a feasible and optimal basis. All phases use the Gauss-Jordan pivoting (GJP) transformation used in the standard simplex and dual simplex algorithms (Arsham, 2005). The proposed scheme is as follows:

**Step 1. SET UP:** The initial tableau may be empty, partially empty, or contain a full basic variable set (BVS).

**Step 2. PUSH:** Fill-up the BVS completely by pushing it toward the optimal vertex.

**Step 3. PUSH FURTHER:** If the BVS is complete, but the optimality condition is not satisfied, then push further until this condition is satisfied; i.e., a primal simplex approach.

**Step 4. PULL:** If the BVS is complete, and the optimality condition is satisfied but infeasible, then pull back to the optimal vertex; i.e., a dual simplex approach.

Not all project networks must go through the Push Further and Pull sequence of steps, as shown in the numerical example. In essence, this approach generates a tableau containing all the information we need to perform all parts of DPA.

The large number of equality constraints, with zero value as their right-hand-side, raises concern for primal (pivotal) degeneracy that may cause pivotal cycling. In the proposed solution algorithm, the BVS Augmentation Phase does not replace any BVs, and the Feasibility Phase uses the dual simplex rule, therefore, there is no pivotal degeneracy in these two phases. However, degeneracy (cycling) may occur in the Optimality Phase after the BVS Augmentation Phase and the Feasibility Phase are completed. This strategy reduces the possibility of any cycling. In the case of cycling in the Optimality Phase, this could be treated using the simple and effective anti-cycling rule for simplex method. Out of the many problems by this algorithm, no cycling was encountered.

It is common in applications of CPM/PERT to have side-constraints in the nominal model. Network-based approaches to problems with side constraints require an extensive revision of the original solution algorithms to handle even one side-constraint. Additional difficulties arise from multiple side-constraints. In the proposed algorithm, when the optimal solution without the side constraints does not satisfy
some or all of the side constraints, the “most” violated constraint can be brought into the final tableau by performing the “catch-up” operation using the dual simplex rule. After doing so, the Feasibility Phase can be used to generate the updated final tableau. To ensure the integrality of the solution appropriate cutting planes can be introduced, if needed.

As part of PA, there is also interest in any structural changes to the nominal project network. There appear to be only a few references that deal with adding an arc to the network, and furthermore, updating the optimal solution requires solving a large number of sub-problems. The dual problem is used to determine whether the new arc changes the CP and if so, the new network is re-optimized. A distinction between basic and non-basic deleted arcs is made. If the deleted arc is a non-basic variable, then the solution remains unchanged. However, if the deleted arc is a basic variable, then the proposed algorithm replaces the deleted variable with a currently non-basic variable by using the optimality phase.

Set Up: Identify the largest $t_{ij}$ for the starting and finishing arcs. Subtract these largest values from the starting and finishing arcs activities respectively. Eliminate the first or the last constraint, whichever has the largest number of activities. Break any ties arbitrarily. Set up the initial simplex tableau without adding any artificial variables, and then start Phase I.

Phase I: Push Phase

1.0. Push Phase Termination Test

IF (?) Label exists, there are Open Rows. THEN continue the BV Iteration. OTHERWISE BVS is complete, start Push Further Phase (Step 2.0).

1.1. PE Selection

PC: Select the Largest $t_{ij}$ and any ties as candidate column(s).

Phase II: Push Further Phase

2.0. Push Further Termination Test

IF any $t_{ij}$ is positive, THEN continue the OP Iteration, OTHERWISE OP is complete, start Pull Phase Phase (Step 3.0).

2.1. PE Selection

PC: Select the Largest $t_{ij}$ and any ties as candidate column(s).

2.2. Push Further Iteration

a. Save PC outside the tableau.
b. Perform GJP.
c. Exchange PC and PR labels.
d. Replace the new PC with old PC with all elements multiplied by -1 except the PE. Continue Push Further Iteration Loop back to 2.0.
Phase III: Pull Phase

3.0. Pull Phase Iteration Termination Test

IF RHS is non-negative, THEN Tableau is Optimal. Interpret the results. OTHERWISE continue Pull Phase Iteration (Step 3.1)

3.1. PE Selection

PR: row with the most negative RHS. Tie Breaker arbitrary
PC: column with a negative element in the PR. Tie Breaker: column with the smallest t_{ij}. Further Tie Breaker arbitrary.

3.2. Pull Phase Transformation

a. Save PC outside the tableau.
b. Perform usual GJP.
c. Exchange PC and PR Labels.
d. Replace the new PC with old PC saved in (a).
   Continue Pull Phase Iteration (Loop back to 3.0)

The final tableau generated by this algorithm contains all of the information needed to perform the DPA. There is also additional useful information in the final tableau; e.g., the absolute value of the last row in the final tableau provides the slack times for the non-critical activities. Such information is useful to project managers because it indicates how much flexibility exists in scheduling various activities without affecting succeeding activities. Clearly, the critical activities have slack time equal to zero. However, the reverse statement is not necessarily true; an activity can have slack-time equal to zero while being non-critical. This can happen whenever there are multiple critical paths.

Theorem 1: The critical path is invariant under time reduction operations introduced in set-up phase.

Proof: Proof follows from the fact that multiplying the first and the last constraints by their maximum duration of starting and ending activities, respectively, and then subtracting from the objective function is equivalent to subtracting a constant from the objective function.

In real-life situations, it is common for a few side-constraints or some structural changes, such as deletion or addition of an arc. The final tableau may be updated after incorporating the side-constraint or the structural changes and then apply the Pull Phase if needed. The violation of the total unimodularity does not affect the solution procedure. If the solution is not integral, then cutting planes may be introduced to ensure an integral optimal solution.

The proposed algorithm operates in the space of the original variables and has a geometric interpretation of its strategic process. The simplex method is a vertex-searching method. It starts at the origin that is far away from the optimal solution. It then moves along the intersection of the boundary hyper-planes of the constraints, hopping from one vertex to the neighboring vertex, until an optimal vertex is reached in two phases. It requires adding artificial variables since it lacks feasibility at the origin. In the first phase, starting at the origin, the simplex hops from one vertex to the next vertex to reach a feasible one. Upon reaching a feasible vertex; i.e., upon removal of all artificial variables from the basis, the simplex moves along the edge of the feasible region to reach an optimal vertex, improving the objective value in the process. Hence, the first phase of simplex method tries to reach feasibility, and the second phase of simplex method strives for optimality. The simplex works in the space of \( n+(m-1) \) dimensions, \( n, t_{ij} \) and \( m-1 \) artificial variables, where \( m \) is the number of nodes and \( n \) is the number of arcs.

In contrast, the proposed algorithm strives to create a full basic variable set (BVS); i.e., the intersection of \( m-1 \) constraint hyper-planes that provides a vertex. The initialization phase provides the starting segment of a few
intersecting hyper-planes and yields an initial BVS with some open rows. The algorithmic strategic process is to arrive at the feasible part of the boundary of the feasible region. In the Push Phase, the algorithm pushes towards an optimal vertex, unlike the simplex, which only strives for a feasible vertex. Occupying an open row means arriving on the face of the hyper-plane of that constraint. Any successive iteration in the Push Phase augments the BVS by including another hyper-plane in the current intersection. By restricting incoming variables to open rows only, this phase ensures movement in the space of intersection of hyper-planes selected in the initialization phase only until another hyper-plane is hit. Recall that no replacement of variables is done in this phase. By every algorithm’s iteration the dimensionality of the working table is reduced until the BVS is filled, indicating a vertex. This phase is free from pivotal degeneracy. The selection of an incoming variable is as the one having the largest tij helps push toward an optimal vertex. As a result, the next phase starts with a vertex.

At the end of the Push-Further phase the BVS is complete, indicating a vertex which is in the neighborhood of an optimal vertex. If feasible, this is an optimal solution. If this basic solution is not feasible, it indicates that the push has been excessive. Note that, in contrast to the first phase of the simplex, this infeasible vertex is on the other side of the optimal vertex. Like the dual simplex, now the Pull Phase moves from vertex to vertex to retrieve feasibility while maintaining optimality; it is free from pivotal degeneracy since it removes any negative; non- zero, RHS elements. The space of the algorithm is m-1 dimensions in the Push Phase and n dimensions in the Push Further and Pull Phases. Note that m-1 is the number of constraints and n is the number of arcs.

**Theorem 2:** The Push and Pull Phases are free from pivotal degeneracy that may cause cycling.

**Proof:** As it is known, whenever a RHS element is zero in any simplex tableau (except the final tableau), the subsequent iteration may be pivotal degenerate when applying the ordinary simplex method, which may cause cycling. In the Push phase, we do not replace any variables. Rather, we expand the basic variable set (BVS) by bringing in new variables to the open rows marked with “?” . The Pull Phase uses the customary dual simplex rule to determine what variable goes out. This phase is also free from pivotal degeneracy since its aim is to replace any negative, non-zero RHS entries.

**Theorem 3:** The solution algorithm terminates successfully in a finite number of iterations.

**Proof:** The algorithm consists of the Set-up Phase to generate an initial tableau that contains some basic variables, followed by three phases. The Push Phase is a BVS augmentation process that develops a basic solution, which may or may not be feasible. The Push Further Phase aims to satisfy the optimality condition. If the BVS is not feasible, the Pull Phase is activated to obtain a feasible optimal solution. All phases use the usual GJP, but differ in the method used to select the pivot element. The Push Phase uses modified simplex column selection criteria to enter one variable at a time into an open row, rather than replacing a variable, while moving towards a vertex that is “close” to the optimal vertex. This strategy pushes toward an optimal solution, which may result in pushing too far into non-feasibility. The Pull Phase, if needed, pulls back to a feasible solution that is optimal.

The theoretical basis for the proposed algorithm rests largely upon the total unimodularity of the constraints coefficient matrix and it remains unimodal under the GJP operations. By the LP formulation, optimality is attained when all tij in the last row in a tableau are non-positive and the algorithm terminates successfully. The current algorithm starts with some non-positive tij. Removing the redundant constraint turns a column into a unit vector identifying a basic variable for the set-up phase.
Since we are adding (not replacing) variables to the BVS in the Initialization and Push Phases, deletion of basic columns is permissible. This reduces the complexity significantly and results in a smaller tableau. In the Pull Phase, if an RHS is negative, there exists at least one element of -1 in that row. If this were not the case, an inconsistent constraint would exist, which is impossible. In this phase, the reduced pivoting rule produces the same results as the usual pivoting with a smaller tableau. The proof follows from the well-known reduced pivoting rule in GJP.

The proposed algorithm converges successfully since the path through the Push, Push-Further and Pull Phases does not contain any loops. Therefore, it suffices to show that each phase of the algorithm terminates successfully. The Set-up Phase uses the structure of the problem to fill-up the BVS as much as possible without requiring GJP iterations. The Push Phase constructs a complete BVS. The number of iterations is finite since the size of the BVS is finite. Push-Further Phase uses the usual simplex rule. At the end of this phase, a basic solution exists that may not be feasible. The Pull Phase terminates successfully by the well-known theory of dual simplex.

AN ILLUSTRATIVE NUMERICAL EXAMPLE

This section illustrates this new algorithm by walking through the project as shown in Figure 1. The LP formulation of this project network is:

Max $9X_{12} + 6X_{13} + 0X_{23} + 7X_{34} + 8X_{35} + 10X_{45} + 12X_{56}$
subject to:

- $X_{12} + X_{13} = 1$,
- $X_{12} - X_{23} = 0$,
- $X_{13} + X_{23} - X_{34} - X_{35} = 0$,
- $X_{34} + X_{45} - X_{56} = 0$,
- $X_{56} = 1$, and all $X_{ij} \geq 0$.

Phase I: Set Up Phase

Subtract the largest duration of starting and ending activities (which are 9 and 12) from the starting and ending activity durations. Eliminate the first constraint. The reduced problem is:

Max $-3X_{13} + 7X_{34} + 8X_{35} + 10X_{45}$
subject to:

- $X_{12} - X_{23} = 0$,
- $X_{13} + X_{23} - X_{34} - X_{35} = 0$,
- $X_{34} + X_{45} - X_{56} = 0$,
- $X_{56} = 1$, and all $X_{ij} \geq 0$.

The initial tableau and the consequence tableaux are as follows:

<table>
<thead>
<tr>
<th>Node</th>
<th>Var</th>
<th>$X_{12}$</th>
<th>$X_{13}$</th>
<th>$X_{34}$</th>
<th>$X_{35}$</th>
<th>$X_{45}$</th>
<th>$X_{56}$</th>
<th>RHS</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td>?</td>
<td>1</td>
<td>-1</td>
<td>-1</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td>?</td>
<td>1</td>
<td>1</td>
<td>-1</td>
<td>-1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>?</td>
<td>1</td>
<td>1</td>
<td>-1</td>
<td>-1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>?</td>
<td>1</td>
<td>1</td>
<td>-1</td>
<td>-1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td>?</td>
<td>1</td>
<td>1</td>
<td>-1</td>
<td>-1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td>?</td>
<td>1</td>
<td>1</td>
<td>-1</td>
<td>-1</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

Tentative Initial Simplex Tableau

<table>
<thead>
<tr>
<th>Node</th>
<th>Var</th>
<th>$X_{12}$</th>
<th>$X_{13}$</th>
<th>$X_{34}$</th>
<th>$X_{35}$</th>
<th>$X_{45}$</th>
<th>$X_{56}$</th>
<th>RHS</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td></td>
<td>?</td>
<td>1</td>
<td>1</td>
<td>-1</td>
<td>-1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>?</td>
<td>1</td>
<td>1</td>
<td>-1</td>
<td>-1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>?</td>
<td>1</td>
<td>1</td>
<td>-1</td>
<td>-1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td>?</td>
<td>1</td>
<td>1</td>
<td>-1</td>
<td>-1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td>?</td>
<td>1</td>
<td>1</td>
<td>-1</td>
<td>-1</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

First Tableau

The initial tableau and the consequence tableaux are as follows:

<table>
<thead>
<tr>
<th>Node</th>
<th>Var</th>
<th>$X_{12}$</th>
<th>$X_{13}$</th>
<th>$X_{34}$</th>
<th>$X_{35}$</th>
<th>$X_{45}$</th>
<th>$X_{56}$</th>
<th>RHS</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td></td>
<td>X_{12}</td>
<td>-1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>?</td>
<td>1</td>
<td>1</td>
<td>-1</td>
<td>-1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>?</td>
<td>1</td>
<td>1</td>
<td>-1</td>
<td>-1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td>?</td>
<td>1</td>
<td>1</td>
<td>-1</td>
<td>-1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td>?</td>
<td>1</td>
<td>1</td>
<td>-1</td>
<td>-1</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

Second Tableau

<table>
<thead>
<tr>
<th>Node</th>
<th>Var</th>
<th>$X_{12}$</th>
<th>$X_{13}$</th>
<th>$X_{34}$</th>
<th>$X_{35}$</th>
<th>$X_{45}$</th>
<th>$X_{56}$</th>
<th>RHS</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td></td>
<td>X_{12}</td>
<td>-1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>?</td>
<td>1</td>
<td>1</td>
<td>-1</td>
<td>-1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>?</td>
<td>1</td>
<td>1</td>
<td>-1</td>
<td>-1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td>?</td>
<td>1</td>
<td>1</td>
<td>-1</td>
<td>-1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td>?</td>
<td>1</td>
<td>1</td>
<td>-1</td>
<td>-1</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

Third Tableau

<table>
<thead>
<tr>
<th>Node</th>
<th>Var</th>
<th>$X_{12}$</th>
<th>$X_{13}$</th>
<th>$X_{34}$</th>
<th>$X_{35}$</th>
<th>$X_{45}$</th>
<th>$X_{56}$</th>
<th>RHS</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td></td>
<td>X_{12}</td>
<td>-1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>?</td>
<td>1</td>
<td>1</td>
<td>-1</td>
<td>-1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>?</td>
<td>1</td>
<td>1</td>
<td>-1</td>
<td>-1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td>?</td>
<td>1</td>
<td>1</td>
<td>-1</td>
<td>-1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td>?</td>
<td>1</td>
<td>1</td>
<td>-1</td>
<td>-1</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

Fourth Tableau

Copyright © 2012, IGI Global. Copying or distributing in print or electronic forms without written permission of IGI Global is prohibited.
Figure 1. An illustrative numerical example

Node | Var | X_{12} X_{23} RHS |
-----|-----|-------------------|
2    | X_{12} | 1 | 1 |
3    | X_{23} | 1 | 1 |
4    | X_{34} | 1 | 1 |
5    | X_{45} | 1 | 1 |
6    | X_{56} | 1 | 1 |
    | \( t_{ij} \) | -3 | -9 | -38 |

End of Push Iteration

**Phase II: Push Further**

2.0. Push Further Iteration
Termination Test

All \( t_{ij} \) are non-positive therefore end of OP Iteration.

**Phase III: Pull Phase**

3.0. FE Iteration Termination Test

All RHS are non-negative. Therefore terminate FE Iteration. Tableau is optimal.

Results are: \( X_{12} = 1, X_{23} = 1, X_{34} = 1, X_{45} = 1, X_{56} = 1 \) and all other \( X_{ij} = 0 \). That is, the CP path is \( 1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5 \rightarrow 6 \) with a project completion time of 38 time units. The slack times for the two non-critical activities namely \( (1 \rightarrow 3) \) and \( (3 \rightarrow 5) \) are given as the absolute value of the last row in the final tableau that are 3 and 9, respectively. It is noteworthy that the proposed solution algorithm is more general than it appears. Specifically, having obtained the final tableau, it can handle revisions of the nominal project. Clearly, this capability would be useful in coping with the situation where some activities are deleted or some new activities are added to the nominal project. Rather than starting the algorithm again to find the CP, the current final tableau can be updated. Moreover, if interest is in finding all critical paths, or counting (which is combinatorial) the number of CPs, the last row in the final tableau provides the necessary information. If any \( t_{ij} = 0 \), then there might be alternative critical paths. By bringing any \( X_{ij} \) with \( t_{ij} = 0 \) into the BVS, a new CP may be generated. Clearly, in such a case, the DPA results are valid for the current CP and may not be correct for the others.

**DATA PERTURBATION ANALYSIS**

Given the outcome of a linear program formulation and calculation for the set of project activities, a series of analyses can provide valuable management information. These uncertainty ranges can be obtained by per-
forming the following different types of Data Perturbation Analysis (DPA) depending on the nature of the uncertainty: perturbation analysis; tolerance analysis; individual symmetric tolerance analysis; symmetric tolerance analysis; parametric sensitivity analysis; and ordinary sensitivity analysis.

**Construction of Perturbation Analysis Set**

Simultaneous and independent changes in the estimated activity durations in either direction (over or under estimation) for each activity that will maintain the current CP. This provides the largest set of perturbations. Inclusion of all actual activity durations in this set preserves the current CP.

The LP formulation of this project network can be written as an LP, called the primal problem:

\[
\begin{align*}
\text{Max} & \quad 9X_{12} + 6X_{13} + 0X_{23} + 7X_{34} + 8X_{35} + 10X_{45} + 12X_{56} \\
\text{subject to:} & \quad X_{12} + X_{13} = 1, -X_{12} + X_{23} = 0, -X_{13} - X_{23} + X_{34} + X_{35} = 0, -X_{34} + X_{45} = 0, \\
& \quad -X_{35} - X_{45} + X_{56} = 0, -X_{56} = -1, \text{and all } X_{ij} \geq 0.
\end{align*}
\]

The Dual Problem is:

Minimize \(U_1 - U_6\) subject to:

\[
\begin{align*}
U_1 - U_2 & \geq 9 - \text{The Dual Constraint Related to Critical Activity } X_{12} = 1 \\
U_1 - U_3 & \geq 6 - \text{The Dual Constraint Related to Non-critical Activity } X_{13} = 1 \\
U_2 - U_3 & \geq 0 - \text{The Dual Constraint Related to Critical Activity, } X_{33} = 1 \\
U_3 - U_4 & \geq 7 - \text{The Dual Constraint Related to Critical Activity, } X_{34} = 1 \\
U_3 - U_5 & \geq 8 - \text{The Dual Constraint Related to Non-critical Activity } X_{45} = 1 \\
U_4 - U_5 & \geq 10 - \text{The Dual Constraint Related to Critical Activity } X_{45} = 1 \\
U_5 - U_6 & \geq 12 - \text{The Dual Constraint Related to Critical Activity } X_{56} = 1 \\
U_j's & \text{are unrestricted}
\end{align*}
\]

The constraints of the dual formulation suggest that for any activity, the difference between finish and starting times exceeds the duration of the activity. The above constraints are related to seven \(X_j\), as they appear in the objective function of the primal problem, respectively. Knowing the critical activities have zero slack time, the following constraints are binding:

\[
\begin{align*}
U_1 - U_2 & = 9 \\
U_2 - U_3 & = 0 \\
U_3 - U_4 & = 7 \\
U_4 - U_5 & = 10 \\
U_5 - U_6 & = 12
\end{align*}
\]

The parametric (i.e., perturbed) RHS of these constraints gives:

\[
\begin{align*}
U_1 - U_2 & = 9 + T_1 \\
U_2 - U_3 & = 0 + T_3 \\
U_3 - U_4 & = 7 + T_4 \\
U_4 - U_5 & = 10 + T_6 \\
U_5 - U_6 & = 12 + T_7
\end{align*}
\]

Solving these parametric system of equations we obtain:

\[
\begin{align*}
U_1 & = -U_6 + T_1 + T_3 + T_4 + T_6 + T_7 + 38 \\
U_2 & = -U_6 + T_3 + T_4 + T_6 + T_7 + 29 \\
U_3 & = -U_6 + T_4 + T_6 + T_7 + 22 \\
U_4 & = -U_6 + T_6 + T_7 + 12 \\
U_5 & = -U_6 + T_7 + 12
\end{align*}
\]

For the larger projects this parametric solution can be obtained by using the JavaScript:

- Solving Linear Parametric RHS: http://www.mirrorservice.org/sites/home.ubalt.edu/ntsbarsh/Business-stat/otherapplets/PaRHSSyEqu.htm

The following proposition formalizes the shifting of parametric bon-binding constraint.

**Proposition 1:** For any given point \(X^{o} = (X_1^o, X_2^o, ......., X_n^o)\) the parameter \(T\) value for any resource/production constraint is pro-
portional to the (usual) distance between the point $X^o$ and the hyper-plane of the constraint.

**Proof:** Proof follows from the fact that the distance from point $X^o = (X_1^o, X_2^o, \ldots, X_n^o)$ to any nonbinding hyper-plane, i.e.:

$$a_1 X_1^o + a_2 X_2^o + \ldots + a_n X_n^o = b + T$$

is

$$\text{Absolute } [a_1 X_1^o + a_2 X_2^o + \ldots + a_n X_n^o - b - T] / \sqrt{(a_1^2 + a_2^2 + \ldots + a_n^2)}$$

That reduces to:

$$\text{Absolute } T / \sqrt{(a_1^2 + a_2^2 + \ldots + a_n^2)}$$

Therefore the parameter $T$ value is proportional to the distance with the constant proportionality, that is $1 / \sqrt{(a_1^2 + a_2^2 + \ldots + a_n^2)}$. This is independent of point $X^o$. In the above example $X^o$ is the dual optimal vertex. This completes the proof.

Therefore, the sensitivity region for the two non-binding constraints are found by plugging in the shadow prices:

$$U_1 - U_3 \geq 6 + T, \quad 38 - 29 \geq 6 + T, \quad T_2 \leq 3$$
$$U_3 - U_5 \geq 8 + T, \quad 29 - 12 \geq 8 + T, \quad T_5 \leq 9$$

Putting all together, i.e., the union of all sensitivity regions, we obtain the largest sensitivity region for the duration of all activities:

$$S = \{ T_j, j=1, 2, 3, 4, 5, 6, 7 | T_1 \geq -9, T_2 \geq -6, T_3 = 0, T_4 \geq -7, T_5 \geq -8, T_6 \geq -10, T_7 \geq -12, T_1 + T_3 \geq -3, T_2 \leq 3, T_4 + T_6 \geq -9, T_5 \leq 9 \}$$

Notice that **Data Perturbation set** $S$ in convex and non-empty since it contains the origin, i.e., when all $T_j = 0$. Perturbation is said to be CP preserving, if the perturbed model has the same CP as the nominal project. Clearly, the DPA are concerned with CP preserving as outlined in the Introduction. This set could be used to check whether a given numerical perturbed activity durations has any impact on the current CP.

The following presents different types of popular sensitivity analysis for non-degenerate problems. For treatment of degeneracy, see Lin and Wen (2003), Arsham (2007), and Lin (2010, 2011).

**Parametric and Ordinary Sensitivity Analysis**

Parametric analysis is of particular interest whenever there are some kinds of dependency among the activity durations. This dependency is very common in project activity networks. This analysis can be considered as simultaneous changes in a given direction. Define a perturbation vector $P$ specifying a perturbed direction of the activity durations. Introducing a scalar parameter $a \geq 0$, it is required to find out how far the direction of $P$ can be moved while still maintaining the current CP.

In our numerical example let us assume the activity duration times $[9, 6, 0, 7, 8, 10, 12]$ are perturbed along the dependent vector $P = (0, 0, -1, 3, 1, -2, -1)$. The perturbed activity durations To find scalar parameter $a$, substitute for vector $T = [9a, 6a, 0, 7a, 8a, 10a, 12a]$ in the critical region

$$S = \{ T_j, j=1, 2, 3, 4, 5, 6, 7 | T_1 \geq -9, T_2 \geq -6, T_3 = 0, T_4 \geq -7, T_5 \geq -8, T_6 \geq -10, T_7 \geq -12, T_1 + T_3 \geq -3, T_2 \leq 3, T_4 + T_6 \geq -9, T_5 \leq 9 \}$$

Now all terms are converted in terms of parameter $a$. The smallest positive value for $a$ in 3.

Therefore the current CP remains critical for any perturbation $aP$, where $0 \leq a \leq 3$.

**Ordinary Sensitivity Analysis**

In this sub-section interest is in finding the range for any particular activity duration,
holding all other activity durations unchanged. Clearly, the ordinary sensitivity is a special case of parametric analysis where it is required to find the extent of the move in positive and negative directions of any one of the axes in the n-parametric space \( t_{ij} \). Here, \( \mathbf{P} \) is a unit row vector or its negative depending on whether an upper or a lower limit is required. The ordinary sensitivity analysis is summarized in Table 1. Alternatively, to find the allowable uncertainty \( T_i \) for any particular activity duration time \( i \), set all other \( T_j = 0 \) in perturbation set \( S \). The results are summarized in Table 1.

The above analysis deals with one duration-uncertainty at-a-time analysis. Suppose we want to find the simultaneous allowable overestimation error in all activity durations is needed.

**The 100% Rule**

The 100% rule states that simultaneous increase (decrease) changes is allowed as long as the sum of the percentages of the change divided by the corresponding maximum increase (decrease) allowable change in the range of ordinary sensitivity analysis for each coefficient does not exceed 100%.

Therefore, base on this rule, the CP will be preserved if

\[
\frac{T_2}{3} + \frac{T}{9} \leq 1,
\]

where the two denominators are the allowable increases from the sensitivity analysis for \( t_{13} \) and \( t_{35} \) respectively. That is, as long as this inequality holds, the current CP remains unchanged. Clearly, this condition is sufficient but not necessary. Similarly, the application of the 100% rule for underestimating all activity duration provides:

\[
\frac{T_1}{-3} + \frac{T_2}{-6} + \frac{T_4}{-7} + \frac{T_5}{-8} + \frac{T_6}{-9} + \frac{T_7}{-12} \leq 1,
\]

with a similar interpretation.

As mentioned earlier the Data Perturbation Set \( S \) is useful to check for any numerically known values of durations if current CP remains critical. However the algebra becomes messy if one tries to find, for example, what is the largest percentage change for all activities to maintain the current CP? Arsham (1990) give detail treatments of this and other useful sensitivity analyses with numerical examples including the followings.

**Tolerance Analysis**

Simultaneous and independent changes expressed as the maximum allowable percentage of the estimated activity duration in either direction (over or under estimation) for each activity that will maintain the current CP. Such an analysis is useful whenever the uncertainties in

<table>
<thead>
<tr>
<th></th>
<th>Lower Limit</th>
<th>Upper Limit</th>
</tr>
</thead>
<tbody>
<tr>
<td>( T_1 )</td>
<td>-3</td>
<td>M</td>
</tr>
<tr>
<td>( T_2 )</td>
<td>-6</td>
<td>3</td>
</tr>
<tr>
<td>( T_3 )</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>( T_4 )</td>
<td>-7</td>
<td>M</td>
</tr>
<tr>
<td>( T_5 )</td>
<td>-8</td>
<td>9</td>
</tr>
<tr>
<td>( T_6 )</td>
<td>-9</td>
<td>M</td>
</tr>
<tr>
<td>( T_7 )</td>
<td>-12</td>
<td>M</td>
</tr>
</tbody>
</table>

Table 1. One change at a time sensitivity limits
the estimated activity durations can be expressed as some percentages of their estimated values. Here, the lower and upper uncertainty limits, i.e., error in both directions: over and under estimation, on each \( t'_{ij} \) must be found. That is, for all \( ij \) activity, to maintain the current CP.

Let \( B \) denotes the matrix in the body of the final tableau, then denote matrix \( A \) as the row-wise augmented matrix \( B \), constructed by pending an identity matrix \( I \) of order \( k \) as its last \( k \) rows where \( k \) is the number of non-critical activities in the final tableau. That is, \( A = [B | I]^T \), where the superscript \( T \) means transpose. Denote \( \tau_j = C \cdot A_j \), where \( A_j \) is the absolute value of the \( j \)th column of \( A \), in the body of the final tableau, and \( T \) is the estimated activity duration time vector. From now on, the critical and non-critical activities must be differentiated.

**Tolerance Limits for Critical Activities**

For any critical activity duration, i.e., for any \( t_{sk} \), the upper tolerance limit is unbounded. However, the lower tolerance limit for \( t_{sk} \) is:

\[
t_{sk}^- = \max \{t_{sk} \cdot \left[ (\text{lower sensitivity limit})_j / \tau_j \right], \text{and the lower limit from the sensitivity analysis} \}
\]

where the max is over all \( j \) such that the \((t_{sk} \) row and the \( j \)th column) element of \( A \) is positive.

In the numerical example, the augmented matrix \( A \) is shown in Table 2 and \( C = (9, 0, 7, 10, 12, 6, 8) \). The element of matrix \( \tau = C \cdot A \), are \( \tau_1 = 15 \) and \( \tau_2 = 25 \).

For example, the lower tolerance limit for the activity duration \( t_{12} \) can be found as follows:

\[
t_{12}^- = \max \{9(-3)/15, 9(-9)/25, -3\} = -9/5
\]

**Tolerance Limits for Non-Critical Activities**

For any non-critical activity duration, i.e., for any \( t_{sk} \), the lower tolerance limit is the ordinary sensitivity limit, and its upper limit for \( t_{sk} \) is:

\[
t_{sk}^+ = \min \{t_{sk} \cdot \left[ (T_n*)_j / - \tau_j \right], \text{and the upper limit from the sensitivity analysis} \}
\]

where the min is over all \( j \) such that the \((t_{sk} \) row and the \( j \)th column) element of \( A \) is positive. For example, the upper tolerance limit for the activity duration \( t_{13} \) is:

\[
t_{13}^+ = \min \{6(-3)/-15, 6(-9)/-25, 3\} = 6/5.
\]

The tolerance limits of all activity duration for the numerical example are shown in Table 3. The symmetric tolerance limits reflect the maximum allowable equal percentage error in both directions (over and under estimations) that hold simultaneously over all activity durations. Clearly, the symmetric tolerance range is a subset of the tolerance range. From Table 3, it is clear that the symmetric tolerance limit for the numerical example is 20%.

Notice the symmetric tolerance provides no meaningful limits if any element in the last row of the final table is zero. In such a case, the symmetric tolerance range is zero for all activity durations. Clearly, this limits symmetric tolerance analysis application.

**A Discussion of the DPA Results for the Managers**

Table 4 summarizes some parts of our findings from analysis applied to the numerical example, using the sensitivity region:

\[
S = \{ T_j, j=1, 2, 3, 4, 5, 6, 7 | T1 \geq -9, T2 \geq -6, T3 = 0, T4 \geq -7, T5 \geq -8, T6 \geq -10, T7 \geq -12, T1 + T3 \geq -3, T2 \leq 3, T4 + T6 \geq -9, T5 \leq 9 \}
\]

As always, care must be taken when rounding the DPA limits. Clearly the upper limit and lower limit must be rounded down and up, respectively.

Since project managers are concerned with the stability of the CP under uncertainty of the estimated durations, the output information from our various DPA should be of interest to
Table 2. Matrix A as a tool for tolerance analysis

\[ A = \begin{bmatrix}
    t'_{12} & 1 & 0 \\
    t'_{23} & 1 & 0 \\
    t'_{34} & 0 & 1 \\
    t'_{45} & 0 & 1 \\
    t'_{56} & 1 & 0 \\
    t'_{13} & 1 & 0 \\
    t'_{35} & 0 & 1 \\
\end{bmatrix} \]

Table 3. Tolerance limits for all durations

<table>
<thead>
<tr>
<th></th>
<th>Lower Limit</th>
<th>Upper Limit</th>
</tr>
</thead>
<tbody>
<tr>
<td>(t'_{12})</td>
<td>-9/5 (20%)</td>
<td>M</td>
</tr>
<tr>
<td>(t'_{13})</td>
<td>-6 (100%)</td>
<td>6/5 (20%)</td>
</tr>
<tr>
<td>(t'_{23})</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>(t'_{34})</td>
<td>-63/25 (36%)</td>
<td>M</td>
</tr>
<tr>
<td>(t'_{35})</td>
<td>-8 (100%)</td>
<td>72/25 (36%)</td>
</tr>
<tr>
<td>(t'_{45})</td>
<td>-18/5 (36%)</td>
<td>M</td>
</tr>
<tr>
<td>(t'_{56})</td>
<td>-12 (100%)</td>
<td>M</td>
</tr>
</tbody>
</table>

Table 4. Ranges for the Ordinary Sensitivity Analysis (OSA), Tolerance Analysis (TA), Individual Symmetric Tolerance Analysis (IST), and Symmetric Tolerance Analysis (STA) with critical activities in bold

<table>
<thead>
<tr>
<th></th>
<th>Crash Durations</th>
<th>Slippage Durations</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>SA</td>
<td>TA</td>
</tr>
<tr>
<td>(t_{12})</td>
<td>6</td>
<td>7.2</td>
</tr>
<tr>
<td>(t_{13})</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>(t_{23})</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>(t_{34})</td>
<td>0</td>
<td>4.48</td>
</tr>
<tr>
<td>(t_{35})</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>(t_{45})</td>
<td>1</td>
<td>6.4</td>
</tr>
<tr>
<td>(t_{56})</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
project managers. As long as the actual activity durations remain within these intervals, the current CP remains critical. Some specific implications from these results for the project manager are that they provide means for discussion and analysis by the project team members as to how to consider modifying the project prior to its implementation. The results given in Table 4 help the project manager to assess, analyze, monitor, and manage all phases of a project, i.e., planning, scheduling, and controlling. When the issues caused by the inherent uncertainty in any project are considered, there are other benefits to be gained by using the proposed DPA approach; e.g., it provides various ranges of uncertainty with the lower limits as the maximum “crashing” for critical activity durations and the upper limits as the maximum “slippage” for non-critical activity durations and their impacts on the entire project completion time. These results could be helpful in some large projects, e.g., if certain activities could be moved into certain intervals or be divisible into smaller sub-activities and “tucked in” at several locations.

The proposed approach can also provide a rich modelling environment for strategic management of complex projects using Monte-Carlo simulation experiments. In this treatment, an a priori duration distribution function may be selected for each activity with the support range produced by the tolerance analysis. Clearly, in the case of complete lack of knowledge uniform distribution function could be used. To use the three-point estimate PERT, the lower and upper limits of individual symmetric tolerance analysis found in this paper can serve as the largest lower bound for optimistic and the smallest upper bound for pessimistic estimates respectively.

Throughout the DPA it has been assumed that duration of all activities are continuous. Clearly, whenever some activity durations are required to be discrete (10), then this admissibility condition must be added to all parts of DPA. The discussion of DPA did not include other important managerial issues, such as time-cost-performance tradeoff, including penalty for delay, reward for timely completion of the entire project; resource allocation; leveling; and constrained resource scheduling; and the human side of project management. The proposed model should serve as an aid to management judgment; therefore the project manager’s experience must be incorporated in the proposed prescriptive model. Oke, Ayoola, Momodu, and Lawal (2009) considered the duration uncertainties as the most critical for their network travelling inspectors in different locations.

Winston (2003) stated correctly that: “The assumption that the critical path found by the critical path method will always be the critical path for the project network may not be justified. For example if activity A = 1 → 3 with duration 6 days was significantly delayed and activity B = 1 → 2 with duration 9 days was completed ahead of schedule, then the critical path might be different.” The main managerial question is: How much delay and how much ahead of schedule? Unfortunately, the answer is not generally given. However, this needed information is obtained using the DPA results given in Table 4 as follows: Using the TA upper limit = 7.2 and the lower limit = 7.2 for these two activities duration, respectively, the answer is a delay of at least 1.2 days (7.2 - 6 = 1.2) for A, and a crash of at least 1.8 days (9 - 7.2 = 1.8) for B.

This allows for other activities change too. However, since we are dealing with these two activities changes only, a better result can be obtained by using a parametric analysis along vector P = {-9a, 6b, 0, 0, 0, 0, 0}. Plugging into set S, we get a = 1/3, b = 1/2. Therefore a better answer is a delay of at least 3 days (i.e., t13 = 6 + 3 = 9) for A, and ahead of time of at most 3 days (i.e., t12 = 9 - 3 = 6) for B.

CONCLUSION

As long as the actual activity durations remain within intervals introduced in this paper the current critical path (CP) remains critical. By using the Data Perturbation Analysis (DPA)
approach, project managers are provided means for negotiation and analysis by the project team members to how to consider modifying the project prior to its implementation. Moreover, the project manager is able to assess, analyze, monitor, and manage various types of uncertainties in all phases of a project; i.e., planning, scheduling, and controlling. Oke, Ayoola, Momodu, and Lawal (2009) considered the duration uncertainties as the most critical and difficult part for their travelling inspectors at different locations.

When the issues caused by the inherent uncertainty in any project are considered, there are other benefits to be gained by using the proposed DPA approach. It provides various ranges of uncertainty with the lower limits as the maximum “crashing” for critical activity durations and the upper limits as the maximum “slippage” for non-critical activity durations, and their impacts on the entire project completion time. These results could be helpful in some large projects where certain activities could be moved into certain intervals or be divisible into smaller sub-activities and “tucked in” at several locations. Using CPM/PERT as a beginning, instead of a fully developed technique, Higgs (1995) enhances the effectiveness and sophistication of information systems planning.

The proposed approach can also provide a rich modeling environment for strategic management of complex projects using Monte-Carlo simulation experiments. In this treatment, a priori duration distribution function may be selected for each activity with the support domain produced by the tolerance analysis. In the case of complete lack of knowledge a uniform distribution function may be used. Therefore the DPA report to the manager can enhance any other method of project management.

When the three-point estimate CPM is used, then the lower and upper limits of individual symmetric tolerance analysis can serve as the largest lower bound for optimistic and the smallest upper bound for pessimistic estimates, respectively. This may release the project managers from guessing. Throughout the DPA, it has been assumed that the duration of all activities is continuous. As a result, whenever some activity durations are required to be discrete, then this admissibility condition to all parts of DPA must be added.

The DPA approach may be further extended by consideration of important managerial issues, such as time-cost-performance tradeoff, including penalty for delay, reward for timely completion of the entire project, resource allocation, leveling and constrained resource scheduling, and the human side of project management. Clearly, the proposed model in this paper should serve as an aid to, rather than substitute for, management judgment. Therefore, the project manager’s experience must be incorporated in the proposed prescriptive model. The proposed approach has the advantage of being computationally practical, easy for a project manager to understand, and provides useful practical information.

ACKNOWLEDGMENT

I am most appreciative for the reviewers’ comments and useful suggestions. The NSF Grant CCR-9505732 supports this work.

REFERENCES


Hossein Arsham is the Harry Wright Distinguished Research Professor at the University of Baltimore. He received his higher-education degrees, all concentrated on modeling, from three different continents. Dr. Arsham’s teaching, research, and consulting activities are multidisciplinary and interdisciplinary.