Abstract

AGRAWAL, MANI KANT. Diffusion Activity Networks. (Under the direction of Professor Salah Eliddin Elmaghraby.)

An activity network (*AN*) is a directed acyclic graph with *n* nodes and *A* arcs. The nodes are numbered from *1* to *n* so that an arc always leads from a smaller numbered node to a higher numbered node. The graph has only one node with no incident arcs, which is called the starting node and numbered *1*. Node *n* is the only node with no emanating arcs and is named the terminal node. An arc represents an activity and a node the start or the culmination of that activity. The terminal node represents the end of the project. These kinds of graphs are also referred to as Activity on Arc (*AoA*) representation of *AN*.

In DiAN the process represented by the arcs is a diffusion process, the state of which is identified with the remaining work content (rwc). The process starts at time '0' at rwc = 1 with a negative drift coefficient. An absorbing barrier is placed at rwc = 0 to identify with the end of the process. The completion time of an activity is thus the first passage time of such a diffusion process.

The paradigm of DiAN, while offering an enhanced modeling concept, raises many questions regarding computational challenges, definition of project management metrics and applicability of such a tool in areas beyond project management. The thesis primarily focuses on understanding such foundational issues. These issues are as follows.

- Development of a conceptual framework to model the projects using DiAN techniques. A related issue was to develop a framework to translate the common managerial knowledge to the parameters of the DiAN model.

- Computational aspects of various performance metrics under the paradigm of DiAN. We focus mainly on calculation of completion time of a project to show the computational feasibility of this technique.

- Controlling an activity that has deviated substantially from its intended course of execution. We propose and evaluate multiple strategies from both planning and execution point of view.

- Generation of the test sets of a given complexity index to evaluate the effectiveness of this technique in an objective manner.

- Finally, some generalizations, extensions, and variants of DiAN are presented to stimulate future research on this topic.

The work presented here reflects the development of a new paradigm. In so, it offers more questions than answers. We have focused on the development of the framework along with enough analysis to validate its practical applicability.

DIFFUSION ACTIVITY NETWORKS

BY

MANI KANT AGRAWAL

A THESIS SUBMITTED TO THE GRADUATE FACULTY OF
NORTH CAROLINA STATE UNIVERSITY
IN PARTIAL FULFILLMENT OF THE
REQUIREMENTS FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

GRADUATE PROGRAM IN OPERATIONS RESEARCH

RALEIGH
SEPTEMBER 1999

APPROVED BY:

John Bishir B. B. Bhattacharyya

CHAIR OF ADVISORY COMMITTEE

Biography

Mani Kant Agrawal was born on june 5, 1966 in Ram Sanehi Ghat, India. He completed his undergraduate studies in civil engineering at Indian Institute of Technology at Kanpur, India. The studies culminated with a Bachelor of Technology degree in 1987.

The graduate education continued at Vanderbilt University in the area of transportation engineering with minor work in operations research. A Master of Science degree was conferred in 1989. The next several years were spend in consulting industry with increasing level of experience and responsibility.

In 1993, he joined North carolina State University to pursue his doctoral studies in Operations Research. During the studies he continued to work as a consultant.

In 1997 he left to join University of Illinois at Urbana Champaign as a visiting faculty member. This was a short stay soon after followed by working at Manugistics, Inc.

Acknowledgements

The process of obtaining a doctorate would not have been possible without the guidance, support, coaching and mentoring of my advisor Dr. Salah Elmaghraby. He not only guided me through this process but also taught many valuable lessons of the life. I would like to extend my appreciation to committee members, Drs. McEneaney, Bhattacharya, Reeves, Bishir and Damerdji. They were always there to offer helping hand through out the process.

Special thanks go to my parents, Dr. P.S. Agarwal and Mrs. Prabha Agarwal for a continuing moral and emotional support. Junior, my Bulldog was a constant companion through out the process. He shared many sleepless working nights with out complaning or destroying the work.

Lastly, thanks go to many of my friends whose support kept the spirit alive when it was needed the most.

Table of Contents

List of Tables vii

List of Figures viii

1 Introduction: Literature Review and Background Theory 1
- 1.1 PERT Model . . . 1
- 1.2 The Motivation . . . 3
- 1.3 Activity Networks Literature Review . . . 6
  - 1.3.1 The Completion Time of a Project . . . 6
  - 1.3.2 The Expected Value $E[T_n]$ . . . 14
- 1.4 Diffusion Processes . . . 16
  - 1.4.1 Simple Random Walk . . . 16
  - 1.4.2 The Standard Wiener Process (or Brownian Motion) . . . 17
  - 1.4.3 Differential Representation . . . 20
  - 1.4.4 Joint Distributions . . . 21
  - 1.4.5 A Differential Equation . . . 23
  - 1.4.6 Brownian Motion with Drift . . . 24
  - 1.4.7 Differential Equation Representation . . . 26
  - 1.4.8 Ito Processes and Ito's Lemma . . . 26
  - 1.4.9 Differential Equation Representation . . . 28
- 1.5 Thesis Outline . . . 29

2 Project Management and DiAN 31
- 2.1 Modeling an Activity as a Diffusion Process . . . 33
  - 2.1.1 Estimating Activity Parameters . . . 34
  - 2.1.2 Empirical Analysis . . . 36
  - 2.1.3 The Model . . . 42
- 2.2 Project Completion Time Problem . . . 45
  - 2.2.1 Bounding the Project Duration . . . 45

2.2.2 Project Completion Time Computations 46
2.3 Conclusion 54

3 Controlling the Activity **55**
3.1 Activity Control Strategies 56
3.1.1 Do Nothing 56
3.1.2 Apply Additional Resources 57
3.1.3 Split the Activity 59
3.1.4 Re-specify the Activity 60
3.2 Simulation of Strategies 60
3.3 Comparative Study of Strategies 62
3.3.1 Impact on Completion Time (T) 63
3.3.2 Cost Implications 64
3.4 Conclusion 67

4 Generation of Random Networks **69**
4.1 Comments on Network Generators 70
4.2 On Measures of Complexity 71
4.2.1 The KSD Measure of "Network Complexity" C 71
4.2.2 Thesen's NR and its Derivatives 74
4.3 The Directed Acyclic Graph Generator (DAGEN) 78
4.3.1 The Skeleton 79
4.3.2 The Generation of the Network 80
4.3.3 Description of DAGEN 82
4.3.4 DAGEN Operation 82

5 The Convolution of Random Variables **85**
5.1 Polynomials Approximation 86
5.2 The Convolution 87
5.2.1 $l_x \leq l_y;\ u_x \leq u_y;\ u_x + l_y \leq u_y + l_x$ 88
5.2.2 $l_x \leq l_y;\ l_x + u_y \leq u_x + l_y$ 88
5.2.3 The Calculation of Integrals 89
5.2.4 The Algorithm 91
5.2.5 Examples 93
5.3 Approximation Through Discretization 94
5.4 Example of Discrete Convolution 97

6 Summary, Conclusions and Extensions **99**

A Impact of Error in Simulating the Diffusion Processes **102**

B Impact of Strategies on Completion Time of an Activity 106

C Impact of Strategies on Cost of an Activity 110

D Data Requirements For SPD Network Generator 120

E GRAPH1.F: Source Code 123

F CINDEX.F: Source Code 140

G Instructions For Running The Program 151

H Sample Input and Output 154

List of References 156

List of Tables

2.1 Project # 1 36
2.2 Project # 2 36
2.3 Project # 3 37

3.1 Simulation Validation 62

A.1 Probability of reaching the control state 103
A.2 Probability of crossing the control state undetected 104

B.1 $\psi = 0.25$ case 107
B.2 $\psi = 0.5$ case 108
B.3 $\psi = 1.0$ case 109

C.1 $\psi = 0.25$ and $\alpha = 1.01$ case 111
C.2 $\psi = 0.25$ and $\alpha = 1.05$ case 112
C.3 $\psi = 0.25$ and $\alpha = 1.10$ case 113
C.4 $\psi = 0.50$ and $\alpha = 1.01$ case 114
C.5 $\psi = 0.50$ and $\alpha = 1.05$ case 115
C.6 $\psi = 0.50$ and $\alpha = 1.10$ case 116
C.7 $\psi = 1.0$ and $\alpha = 1.01$ case 117
C.8 $\psi = 1.0$ and $\alpha = 1.05$ case 118
C.9 $\psi = 1.0$ and $\alpha = 1.10$ case 119

List of Figures

1.1 AoA: Activities in Series. 8
1.2 AoA: Activities in Parallel. 9
1.3 AoA: The Interdictive Graph 9
1.4 Symmetric random walk over three time periods 19

2.1 Project # 1 39
2.2 Project # 2 40
2.3 Project # 3 41
2.4 DiAN, Project 1 47
2.5 Results of Monte Carlo Sampling, Project 1 49
2.6 DiAN, Project 2 50
2.7 DiAN, Project 3 50
2.8 Results of Monte Carlo Sampling, Project 2 52
2.9 Results of Monte Carlo Sampling, Project 3 53

3.1 Strategy: add more resources 58
3.2 Strategy: split the activity 59

4.1 Interdictive Graph (AoN) 73
4.2 Series 73
4.3 Series-Parallel 73
4.4 Example 1 of Anomaly: AoA (above) and AoN (below) 75
4.5 Example 2 of Anomaly: AoA (above) and AoN (below) 76
4.6 The structure of the skeleton graph 80
4.7 Illustration of Theorem 4.1 with $\eta = 2$ 81

Chapter 1

Introduction: Literature Review and Background Theory

The seminal paper by Malcolm, Roseboom, Clark, and Fazar [43] (1959) gave birth to the field of project management under stochastic duration of activities. The concepts proposed in that paper are widely known as Project Evaluation and Review Technique (*PERT*). *PERT* while capable of modeling a large variety of projects, is unable to cope with a number of issues as discussed in subsequent sections. The research undertaken for the thesis proposes a new project management paradigm, namely diffusion activity networks (*DiAN*). This proposed paradigm shows a promise towards providing answers to some of the still open questions under *PERT*. In this thesis we present the need for such a modeling paradigm, the results obtained, the difficulties encountered in achieving the desired results, and the directions for future research.

1.1 PERT Model

The *PERT* model was the first work to recognize the stochastic variability in the estimate of the activity durations. The model laid the groundwork for the analytical development of stochastic project planning. However, as true with any new paradigm, the analytical methods of Malcolm *et. al.* lacked requisite mathematical sophistication. MacCrimmon & Ryavec [41] provide a detailed account of such shortcomings.

The most notable of these are:

- The activity durations are modeled using the beta distribution with no previous empirical results to support this hypothesis. Moreover, the beta distribution was approximated using three parameters, namely pessimistic, optimistic, and a most likely value. The general beta distribution requires four parameters.
- The expected value of the project completion time as estimated by the *PERT* model is an optimistic estimate of the true expected value. In other words the *PERT* estimate is a lower bound on the true expected value. The magnitude of the error in the estimation is unknown.
- The *PERT* estimate of the standard deviation of the completion time could be higher or lower than the actual standard deviation. The magnitude of the error is also unknown.

In the following, the mathematical notation and the computational issues related to *PERT* or stochastic activity networks (AN) are introduced.

A stochastic AN is a two-terminal directed acyclic graph (dag), $G(N, A)$ composed of $|N| = n$ nodes (events) and $|A| = m$ arcs (activities). The nodes in the network are numbered so that an arrow points from a smaller numbered node to a larger one. The network has a starting node[1] 1 and a terminal node[2] n. Each node $i \neq 1, n$ is connected from above and below; hence there is at least one path from node 1 to each node and from each node to node n. Let Y_a denote the duration of an arc $a \in A$ which has a cumulative distribution function (*cdf*) $F_a(y)$. Also let P denote the set of all paths in the network and $Z(\pi_l)$ denote the duration of a path $\pi_l \in P$, then

$$Z(\pi_l) = \sum_{a \in \pi_l} Y_a. \tag{1.1}$$

As evident from equation (1.1), the *pdf* of $Z(\pi_l)$ is the sum of the r.v.'s along the path. The project completion time (time of realization of node n) T_n can now be

[1] Also known as the source node

[2] Also known as the sink node

written as

$$T_n = \max_{\pi_l \in P} [Z(\pi_l)] . \tag{1.2}$$

Therefore,

$$F_{T_n}(t) = P(T_n \leq t) = P(Z(\pi_l) \leq t \ \forall \pi_l \in P) . \tag{1.3}$$

which is extremely difficult to evaluate.

In fact, Hagstrom [28] proved this problem (1.3) to be NP-Hard by taking a discrete probability mass function (pmf) for Y_a that is defined over two distinct data points. He also proved that even finding the expected value of F_{T_n} is NP-Hard. The difficulty in computation of the completion time distribution F_{T_n} led researchers to focus on two main issues: bounding the cdf (F_{T_n}) or the expected value $E[T_n]$ from below and above, and approximating the cdf by various methods of known or unknown error bounds. The researchers also extended their methodology to pdf's other than the beta distribution.

1.2 The Motivation

An area of AN largely ignored by researchers is the estimation of the probability density function of the activity itself. This issue has generally been left to be decided by practitioners [59]. The most common technique used in practice is the estimation of statistical parameters from either the past data or from the opinion of the experts. The estimated parameters are typically the mean and variance or the three parameters asked by the $PERT$ model for the beta distribution. Needless to say, this issue of estimating the requisite parameters of an activity is of great importance to practitioners (project managers). Thus far, research in project management has not been able to address this issue in any significant manner.

One of the primary contributions to this is attributed to Lau and Somarajan [39]. They present a methodology to evaluate the parameters of a beta distribution using

the quantiles. This paper discusses at length the strategies in picking the quantiles and a numerical method to evaluate the parameters of the beta distribution. The second part of the paper discusses the use of Ramberg-Schmeiser distribution as a complement to the beta distribution. Use of Ramberg-Schmeiser distribution permits one to model the area, not covered by beta distribution, in "skewness-kurtosis" graph (see [30]).

While the work propsed by Lau and Somarajan proposes a systematic approach to computing the parameters of a distribution function, it comes short on two accounts. One being the number of parameters (quantiles in this case) that a project manager need to provide. The second shortcoming is the identification of the distribution function. The analyst needs to have a prior knowledge of the distribution function that would be appropriate for the current project.

In this work, we take a radical departure from the traditional method of modeling an activity using its probability distribution functions. We instead offer a paradigm that models the actual progression of an activity throughout its life time. Consider the following observations.

- Any activity is executed in a continuous manner.
- In a small interval of time the activity is expected to achieve a small change in its state. This change in state is stochastic.

These observations when combined with the assumptions that the change in the status of an activity over non-overlapping intervals is independent then we could model the activity as a diffusion process. The diffusion model permits us to draw upon the vast mathematical resources in modeling the actual behavior of an activity.

Modeling a project as a diffusion network is a novel approach with no precedent research.

Expanding on the diffusion model, we propose an approach that can be used in lieu of identifying the proper distribution function for an activity and its associated parameters. This approach could also be effectively used in modeling the activities where little or no accumulated knowledge exists or if such knowledge cannot be used

effectively. In the following we discuss the information available to a project manager and how to best utilize it in modeling an activity.

From a practical perspective, it is observed that a project manager can identify (or estimate, based on available resources) the rate of progress and the error of estimates over the planning horizon of an activity (project) as opposed to estimating the true *pdf* of its duration. This is in a sharp contrast to the required information sought by *PERT* and its offshoot from a project manager. The standard *pdf*'s (such as beta, triangular etc.) are incapable of utilizing this information. The assumption of an apriori fixed standard deviation leads to an erroneous *pdf* for the activity at hand: in reality, variability increases with time. Evidently, the representation of a time dependent uncertainty (as reflected in the variance) with a distribution of fixed uncertainty (variance) as in standard *PERT* analysis is unsatisfactory. These distributions (whether beta or otherwise), while capable of modeling a wide variety of situations, fail to capture the inherent variability (over time) in the variance of the completion time of an activity. In short, the traditional approach of modeling the uncertainties in the planning of a project using standard (fixed variance) distributions is not able to resolve this problem.

The software development projects are a particularly good example of this variability-over-time in the variance of an activity completion time. Here, the estimates of the remaining work content (rwc) change with time both in expectation and variance. Research and development, and large scale construction are also typical examples of such projects.

Concentrating on the information available to management, it is clear that the estimate of the duration of an activity has to be derived from the knowledge of the rate of progress and the error of estimating this rate. An obvious choice for modeling such a phenomenon is "diffusion process." That is what we focus on in this thesis. The completion time of an activity can be treated as analogous to the first passage time of a diffusion process (or its many variants) in the presence of an absorbing barrier. The diffusion activity network (*DiAN*) can now be defined as a two-terminal directed

acyclic graph with activities modeled as a diffusion process. *DiAN* is explored in detail in Chapter 2.

The following section 1.3 is devoted to the review of the estimation of the completion time of a project. Section 1.4 gives a brief introduction to diffusion theory. Finally, section 1.5 outlines the remaining chapters in this thesis.

1.3 Activity Networks Literature Review

We present a review of the state of the art in estimation of the completion time of a project when the activities durations are random variables.

1.3.1 The Completion Time of a Project

Estimation of the completion time distribution has been one of the most active areas of research in project management. Research in this area varies across approximations of the exact *pdf* to bounding the moments of the completion time distribution. The original *PERT* method [43] approximated the expected value of the project completion time by substituting the mean of activity durations for their distribution. The expected value of the completion time can thus be approximated by the elementary shortest path computations. This method is computationally efficient and can be easily applied to very large networks. On the negative side the results obtained provide only poor lower bounds to the actual expected value as shown by MacCrimmon and Ryavec [41].

Over time various shortcomings of the original *PERT* method [43] have been overcome. In the following we discuss the developments in this field that has taken place since the last comprehensive overview of the field by Elmaghraby[18].

The literature on completion time distribution estimates can be divided into three main categories to facilitate discussion. These are

- exact computation of the density/distribution function of T_n,
- analytical schemes to approximate/bound the *cdf* of T_n (i.e., F_{T_n}), and

- numerical techniques, such as Monte Carlo sampling, to estimate the *cdf* of T_n or any of its parameters, such as the expected value or the density function.

In the following we discuss these subcategories in grater detail.

Estimation of The Distribution Function F_{T_n}

Calculation of the exact *pdf* of the completion time is proven to be a computationally onerous proposition. Researchers over the years have proposed various methods for special cases or methods which can be applied for very small networks. Some of these methods were later proven to be incorrect or seriously handicapped in their computational feasibility for anything but trivial networks. As mentioned earlier, Hagstrom [28] in 1988 proved that computation of F_{T_n} is NP-Hard.

Charnes & Cooper [7] proposed a "chance constrained" model for estimating the exact *pdf* of T_n. Their approach, however, does not estimate exact *pdf* but an over-estimate; a fact presented in great detail in Elmaghraby [18].

A rather unknown paper (in the research community) is due to Hopfinger [33]. He proposed a recursive computational scheme for estimating F_{T_n}. The method is applicable to networks with multiple sinks and sources and for a general class of activity duration *pdf*. The algorithm can be applied in forward or backward recursion with equal computational requirement.

Hopfinger divides the original network (called V) into two subnetworks V_1 and V_2 such that $V = V_1 \cup V_2$. Initially $V_1 = \phi$ and $V_2 = V$. The recursion starts by adding the source node (or sink for backward recursion) to V_1 and setting the initial completion time as zero. Subsequently nodes are added to set V_1 recursively and the completion time distribution for the subnetwork V_1 is computed for all of its sinks. At all times $V_2 = V \setminus V_1$. In the constructed subnetwork (V_1) there is no arc with origin node in V_2 and terminating node in V_1. Clearly this subnetwork will have one source node and possibly many sink nodes.

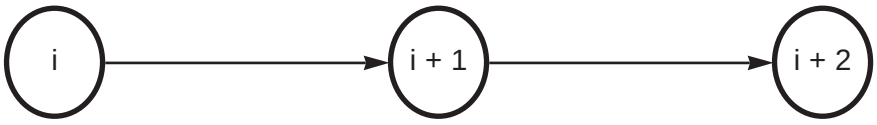

Figure 1.1: AoA: Activities in Series.

A closer examination of algorithm reveals that this method is nothing but a recursive scheme of computing the completion time distribution. The algorithm is not a dynamic programming formulation as claimed by the author. The method can be represented as taking the maximum of the distribution of incoming arcs at a node. Thus making it parallel to the original formulation of equation (1.3) in computational requirements. Research on this line of argument was not pursued further.

The calculation of the exact *pdf* of T_n becomes computationally easier if the network under consideration exhibits a series-parallel topology. A network is said to be series-parallel if it can be reduced to a single arc via repetitive application of series or parallel reductions [57].

As an example consider two arcs in series (see Fig. 1.1). The completion time of node $i+2$ can be obtained by convolving the *pdf* of the two arcs $(i, i+1)$ and $(i+1, i+2)$.

Similarly, consider two arcs in parallel (see Fig. 1.2). The completion time of node $i+1$ is the maximum of two *r.v.*'s corresponding to the arcs between nodes i and $i+1$.

If a network can be represented by arcs in series or parallel then it is referred to as a series-parallel network. In other words the network does not contain any interdictive graphs (see Fig. 1.3)[3]. The graphs which contain an interdictive graph are known as "irreducible" networks.

Most of the networks that are encountered in practice are not series-parallel.

[3]A similar discussion and construct applies to *AoN* networks

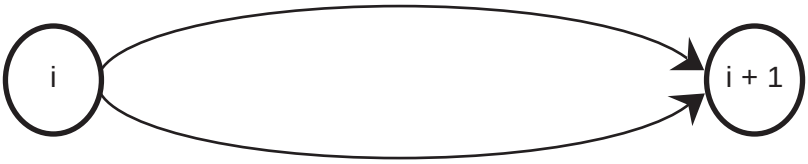

Figure 1.2: AoA: Activities in Parallel.

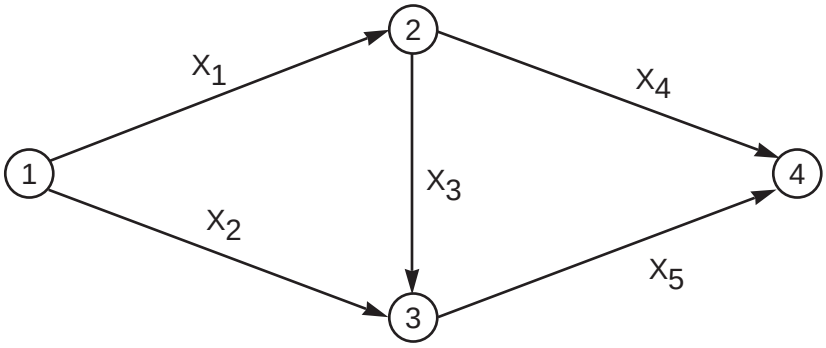

Figure 1.3: AoA: The Interdictive Graph

Researchers over the years proposed various schemes to convert (reduce) a network under consideration to a series-parallel one. Hartley & Wortham [32] proposed a method to identify "crossed network" which we call interdictive graph. Ringer [49] extended this methodology to include "double Wheatstone bridge" and "Criss-cross" networks[4]. At this point it became evident that the research in this area (of identifying all possible combinations of interdictive graphs) is not feasible. Thus we are left with the task of identifying the minimum number of such interdictive graphs in a given network[5].

Martin [44] exploits the irreducible property of AN's to translate the network into a series-parallel one. The method is applicable when the project is represented as an AoA network and the activity durations are either polynomial density functions or can be represented as such. The two main contributions of Martin's work are, (i) the reduction of a given network to a series-parallel network, and (ii) to compute the convolution of polynomial density functions.

The main advantage in Martin's method is the representation of convoluted polynomials. The method, however, does not alleviate the computational burden involved in evaluating convolutions. Some of these shortcomings are listed below.

- The convolution method involves evaluation of multiple nested integrals, making it computationally difficult for all but trivial cases.
- The convolution of two polynomial density functions of order α and β can result in a polynomial of order $\alpha+\beta$. Thus controlling the numerical error in large order polynomials becomes challenging.
- The number of subintervals over which the convolution is defined grows exponentially as $O(3^{n-1})$. Here n is the number of polynomials to be convoluted.

[4]Out of the four criss-cross networks presented in the paper, two are of complexity index one. Thus contradicting the authors earlier claim of multi-crossed networks. The author fixes three arcs for a double wheatstone bridge. The double wheatstone bridge is of complexity two therefore only two arcs need to be fixed.

[5]A question answered by Bein et. al. [4].

Martin's method, while proposes some promising directions for computing F_{T_n}, raises some serious questions on its applicability for any appreciable size network. An improved convolution scheme is proposed in Chapter 5; (also see Agrawal and Elmaghraby[1]). Moreover, Martin's method does not guarantee the minimum number of reductions to render the network series-parallel [57].

Dodin [16] expanded upon the idea of transforming the network into series-parallel. He secured a lower bound on F_{T_n}. Dodin, however, could not guarantee the minimum number of reductions required to make the network series-parallel. This was later achieved by Bein, Kamburowski & Stallmann [4].

A special case of *PERT* networks occurs when all activity durations are exponentially distributed. This network is aptly referred to as Markovian *PERT* networks by Kulkarni & Adlakha [38]. The F_{T_n} of a *PERT* network can be modeled as first passage time distribution in a continuous time Markov chain, constructed on uniformly directed cutsets of the *PERT* network. The approach, while aesthetically appealing, breaks down for large networks as the number of cutsets grow exponentially. Magott and Skudlarski [42] offer a detailed overview of this material as well as an extension to this method *via* hyperexponential distribution.

Bounds on the Distribution Function (F_{T_n})

Kleindorfer [36], Robillard and Trahan [51] and Shogan [54] proposed schemes for lower and the upper bounds on F_{T_n}. These methods rely on various stochastic inequalities to analytically derive the bounding distributions. Elmaghraby [21] provides a detailed discussion of these works.

After Shogan in 1977, who is the last of the three, there was relatively little work towards bounding the distribution function. The main reason is the realization of the computational complexity involved in such an endeavor. The next development in this area is attributed to the result of Bein, Kamburowski & Stallmann (BKS) [4]. They developed a polynomial time scheme to transform an irreducible AoA network into a series-parallel one. The algorithm also identifies the number of nodes to reduce

(alternately arcs to fix) to make a network series-parallel. The result led to a renewed interest in this field.

Consider two activities in series (Fig. 1.1), if we were to replace the *pdf* of arc $(i, i+1)$ by its expected value then the convolution of the two arcs is a lower bound on the true *cdf*. The upper bound is a natural outcome of the resulting series-parallel[6] network. Kamburowski[35] capitalized on this idea to bound F_{T_n} from both below and above. Elmaghraby [23] independently proposed a similar method. Elmaghraby's method is used to bound $E[T_n]$. Elmaghraby [23] shows that this method provides the tightest bounds to date. This method requires repeated convolution and maximum operations thus making it suitable for small networks if activities follow a continuous distribution. For large networks this method can be used with discrete distributions. In this thesis this method is employed extensively. Convolution and approximation schemes are developed to make this method applicable for large networks (see Chapter 5).

Hagstorm [29] transforms the network such that each realization of an arc is represented as an arc in itself. Naturally this method can be used only for discretely distributed arc durations. Moments of T_n can now be estimated on this transformed network by recursively computing moments at successively numbered nodes. In this algorithms as the number of realizations of an arc or the number of arcs grow, so does the computational requirement. The result is not surprising as the author also proved that finding the expected value of T_n is NP-Hard (see [28]).

Analytical approximation of the Distribution Function F_{T_n}

The earlier work in computing moments of F_{T_n} can be attributed to Clark [8]. The salient feature of his approach is an explicit recognition of the correlation between two activities, which is in sharp contrast to the dominant culture: it can safely be said that most research in this area assumes independence among activities (or independence among activities terminating or originating at a node). The approach has been mostly

[6] After reduction with BKS process.

ignored by researchers due to computational difficulties introduced by the correlation among activities.

The central theme of Clark's paper is that the maximum of two normal random variables can be approximated by a normal random variable with a fair degree of accuracy. Using this argument the first four moments of T_n can be estimated in a single scan of the network. For large scale networks the accuracy of such an approximation can not be guaranteed. In other words one does not know if the resulting moments are upper or the lower bounds on their respective 'exact' values. Also one does not know the magnitude of the error of such an approximation.

Sculli [53] extends the work of Clark [8] by transforming a given network such that no more than two arcs terminate at any node. This can be accomplished by inserting appropriate dummy activities and/or nodes in the network. The mean and variance of any node is then computed by taking the maximum of two normal random variables. Elmaghraby [21] provides a detailed review of these two works.

It has been successfully argued that the central limit theorem can not effectively be used in estimating the *pdf* of T_n since there need not be one dominant path in the network. This was the main argument against the original *PERT* model. Consider a large network. There should be several competing paths, referred to as 'dominant' paths. Each one of these paths should be approximately normal, by the central limit theorem. Dodin and Sirvanci [17] computed the completion time distribution of the project by taking the maximum over these dominant paths in the network. They show that this maximum can be represented by an extreme value distribution. Thus for a large network, the completion time distribution can be approximated by an extreme value distribution given that we know the number of dominant paths, a fact which the paper fails to address.

Monte Carlo Approximation of the Distribution Function F_{T_n}

Van Slyke [58] uses 'crude' Monte Carlo techniques to approximate the *pdf* of T_n. The method involves sampling all activities followed by a deterministic computation

of completion time. The network was sampled enough times to ensure statistical accuracy. Monte Carlo techniques can be successfully applied from medium to fairly large networks for computing the *cdf* along with other pertinent network parameters. It will be computationally inefficient for large networks due to the large sampling requirements. Note that the sampling requirements (to maintain the statistical accuracy) grow linearly with number of arcs in the network.

Burt & Garman [6] and Sigal *et al.* [55] improved upon the Monte Carlo technique of Val Slyke by incorporating improved (variance reduction) sampling techniques. In Burt & Garman's conditional Monte Carlo technique, only the activities that lie on two or more paths in the network are sampled. Sigal *et al.* sample only the activities which do not fall on a 'maximal uniformly directed cutset' (UDC)[7]. The net effect of these two sampling techniques is reduced number of arcs that need to be sampled. Elmaghraby [21] provides a critical review of these two sampling schemes.

1.3.2 The Expected Value $E[T_n]$

The paper of Malcolm *et al.* led to a flurry of research in this field, mainly in improving the estimate of the expected value of the completion time. Shogan [54] and Elmaghraby [21] provide an excellent review of the work in this area up to their respective times. We do not intend to duplicate the work of these two authors but to present the papers that have appeared since then. Papers reviewed in [54] and [21] are presented here for historical perspective only.

One of the first authors to offer an improvement over the *PERT* estimate was Fulkerson [26]. He computed a lower bound on $E[T_n]$ which is better than *PERT*'s estimate. Clingen [9] extended the Fulkerson's approach to include the case when the activity durations are continuous random variables. Elmaghraby [19] improved on Fulkerson's lower bound by realizing that the $E[T_n]$ is invariant under the reversal of

[7] A Uniformly Directed Cutset separates the start node from the terminal node and there is no arc which starts from the part containing the terminal node and terminates at the part containing the start node. Naturally such a partition is a set of arcs. The maximal UDC is the set of maximum cardinality.

arcs. The lower bound on the expected value of the terminal node, however, need not remain the same. Lindsey [40] proposed an improvement over Fulkerson's method but did not guarantee if it is a lower or an upper bound. Robillard and Trahan [50] presented a general method based on Jensen's inequality [48, pp. 46] for the lower bound estimates. Their method encompasses that of Fulkerson and Elmaghraby.

Kleindorfer [36] proposed a method to compute both the lower and the upper bounds on $E[T_n]$ in addition to the bounds on the *cdf* (see §1.3.1). Kleindorfer did not compare his bounds with those of earlier researchers. Shogan [54] later proved that the expected value bounds proposed by Kleindorfer are tighter than the original *PERT* estimate but not as tight as that of Fulkerson's, and *a fortiori* Elmaghraby's, bounds.

All of the aforementioned work explicitly recognizes the distribution function of activity durations for estimating $E[T_n]$. A second line of research focuses on utilizing just the moments of the activity durations. Devroye [14] computes an upper bound on the expected value and the variance of T_n by using only the mean and the variance of the activity durations.

Kamburowski [34] computed an upper bound on $E[T_n]$ by using a stochastically larger New Better than Used in Expectation (NBUE)[8] distribution for each activity. The NBUE distribution used here is exponential with the same mean as the mean of the original activity distribution. A simple recursive relationship permits the computation of upper bound in a single pass for all the nodes in the network. This makes it a feasible approach for even large scale networks. Elmaghraby [21] provides a detailed overview of this work.

Magott & Skudlarski [42] compute the upper bound on $E[T_n]$ by using an Increasing Failure Rate Average (IFRA)[9] distribution for all activities. The hyperexponential distribution is IFRA. Computational efficiency comes from realizing that the convolution of two hyperexponential variables is hyperexponential and the maximum of

[8] The *cdf* F of non-negative random variable is NBUE if F has finite mean and $\int_t^\infty \bar{F}(x)dx \leq \mu\bar{F}(t)$ for $t \geq 0$

[9] The *cdf* F of non-negative random variable is IFRA if $-\frac{\bar{F}(t)}{t}$ is non-decreasing in $t \geq 0$

two hyperexponentials can be approximated by a hyperexponential. For the maximum case the first two moments and the coefficient of variation are matched. The algorithm can be employed in a single scan of the network, thus making this method polynomial in computational complexity. This represents a substantial improvement over [38] where the state space expands exponentially (see [21]).

1.4 Diffusion Processes

We motivate the development of a diffusion model *via* discrete (in both time and state) random walk from which the desired model is derived through a limiting process. We start by modeling the Brownian motion process, then proceed to model a *proportionality* relationship between the parameters, mean and variance, and time, that is, we seek a functional relationship of the form $E(X(t)) = at$ and $V(X(t)) = bt$. Finally we point out some alternative models, which are more elaborate and, naturally, more demanding analytically.

1.4.1 Simple Random Walk

One of the simplest examples of a stochastic process is the *discrete-time discrete-state simple random walk*. Here, a particle, say, starts at 0 and at time $\tau = 1, 2, 3, \cdots$ takes a step of size 1 either to the right or to the left, each with probability $\frac{1}{2}$. The movements are independent. Let ξ_τ denote the position of the particle at time τ. Then the dynamics of ξ_τ can be described by the following equation

$$\xi_\tau = \xi_{\tau-1} + X_\tau$$

Where X_τ is a random variable with the probability mass function (pmf)

$$P(X_\tau = 1) = P(X_\tau = -1) = \frac{1}{2};\ \tau = 1, 2, 3, \ldots.$$

ξ_τ is discrete-state because it can take on only the discrete values: for odd values of τ, the possible values of ξ_τ are $-\tau, \cdots, -3, -1, 0, 1, 3, \cdots, \tau$, and for even values of τ,

the possible values of ξ_τ are $-\tau, \cdots, -2, 0, 2, 4, \cdots, \tau$. The probability distribution of ξ_τ is found from the binomial distribution. In τ steps, the probability that there are n steps to the right and $\tau - n$ steps to the left is $\begin{pmatrix} \tau \\ n \end{pmatrix} 2^{-\tau}$, which is the probability that ξ_τ will take on the value $\tau - 2n$,

$$P\left(X_\tau = \tau - 2n\right) = \begin{pmatrix} \tau \\ n \end{pmatrix} 2^{-\tau}.$$

We shall use these concepts of random walk in the derivations that follow. At this time, note that the range of the possible values of ξ_τ (which is equal to $2\tau + 1$) increases with τ, so does the variance of ξ_τ (which is equal to $2 - 2^{-\tau}$). Hence ξ_τ is a nonstationary process.

1.4.2 The Standard Wiener Process (or Brownian Motion)

The Wiener process is a stochastic process with three important properties. First, it possesses the *Markov property*, namely, its future development depends only on its current state and does not depend on previous history. Second, it has (time) independent increments; *i.e.*, the probability distribution for its evolution over any (finite) time interval is independent of any other nonoverlapping time interval. Third, changes in the process over any finite interval of time are normally distributed, with a variance that increases linearly with the time interval.

Consider a particle that starts in position 0 and follows a symmetric random walk in one dimension. It takes (in time τ) one step h to the right with probability $\frac{1}{2}$ or one step $-h$ to the left with probability $\frac{1}{2}$. Let $X(t)$ denote the position of the particle at time t, measured only at discrete time epochs $1, 2, \cdots, \left\lfloor \frac{t}{\tau} \right\rfloor$, then (recall that $X_0 = 0$),

$$X(t) = h\left(X_1 + X_2 + \cdots + X_{\lfloor t/\tau \rfloor}\right), \tag{1.4}$$

where

$$X_i = \begin{cases} +1 & \text{if the } i\text{th step is to the right} \\ -1 & \text{if it is to the left} \end{cases}$$

Assume the X_i are independent. Since $E(X_i) = 0$ and $V(X_i) = 1$, we easily see from (1.4) that

$$\begin{aligned} E\left(X(t)\right) &= 0 \\ V\left(X(t)\right) &= h^2 n, \text{ where } n = \left\lfloor \frac{t}{\tau} \right\rfloor \end{aligned} \tag{1.5}$$

We wish to let h, the size of the step, and τ, the time interval between steps, go to 0 (whence n goes to ∞) in such a way that the result is meaningful[10]. To this end, let

$$h = \sigma\sqrt{\tau} \text{ for some } \sigma > 0$$

which leads to

$$E\left(X(t)\right) = 0 \tag{1.6}$$

as before, and[11]

$$V\left(X(t)\right) = \left(\sigma^2\tau\right)\left\lfloor \frac{t}{\tau} \right\rfloor \to \sigma^2 t \tag{1.7}$$

This process has the three desired characteristics listed above; refer to Fig. 1.4 which is plotted for three time periods:

1. It is certainly "memoryless." The position of the particle at time $t = 3$ in Fig. 1.4 depends only on its position at time 2 and not on how it arrived there.

2. $\{X(t),\ t \geq 0\}$ has stationary independent increments. This follows from the fact that for any i,j, $1 \leq i \leq j \leq n-1$, $\sum_{r=i}^{j} X_r$ is independent of $\sum_{r=0}^{i-1} X_r$ and $\sum_{r=j+1}^{n} X_r$.

[10] For instance, if we let $\Delta h = \tau$ and then let $\tau \to 0$, then both $E\left(X(t)\right)$ and $V\left(X(t)\right)$ shall converge to 0, and therefore $X(t) = 1$ with probability 1, a not very interesting process to study!

[11] Note that as $\tau \to 0$ the value of $\left\lfloor \frac{t}{\tau} \right\rfloor \to \frac{t}{\tau}$.

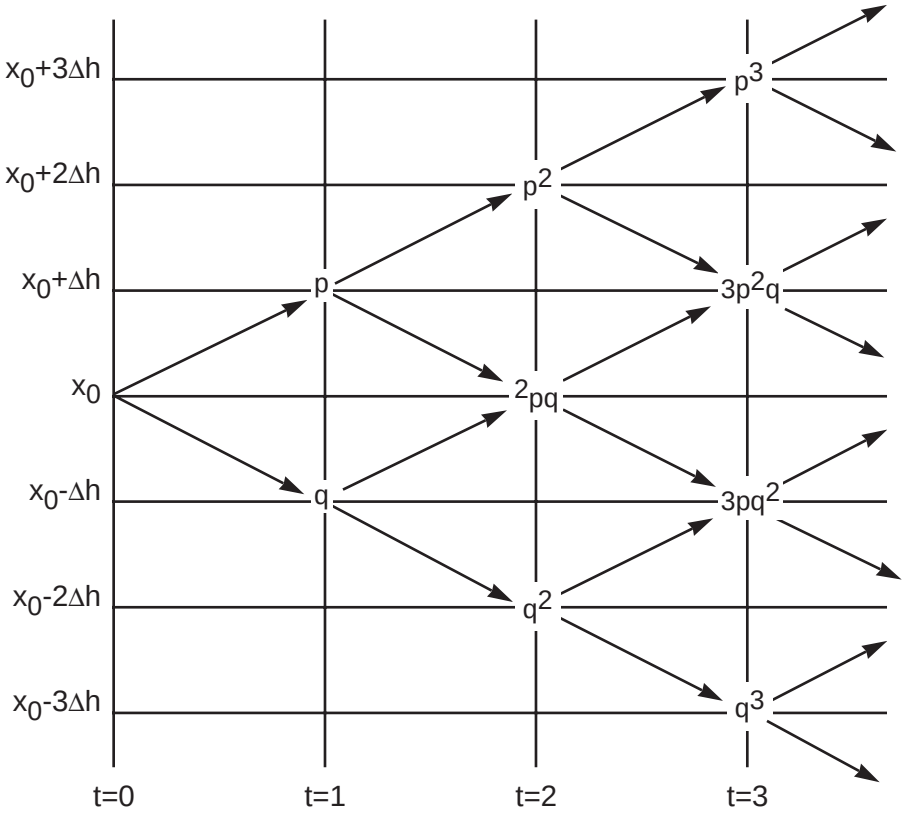

Figure 1.4: Symmetric random walk over three time periods

3. Invoking the central limit theorem, $X(t)$ is normal with mean given by (1.6) and variance given by (1.7). The same is true for any sum $\sum_{r=i}^{j} X_r$ when τ is small enough.

When $\sigma = 1$ the Brownian motion is called the *standard* Brownian motion. Since conversion to the standard Brownian motion is always possible through considering the process $\frac{X(t)}{\sigma}$ instead of $X(t)$, we may limit our discussion to the standard Brownian motion without loss of generality.

The above intuitive discussion glosses over two points of great theoretical importance. The first concerns the fact that the interpretation of the Brownian motion as the limit of the random walk (1.4) suggests that $X(t)$ should be a continuous function

of t. This, in fact, is the case.[12] *Stochastic processes that are almost everywhere continuous are called diffusion processes.* Interestingly enough, though the sample path $X(t)$ is always continuous, it is nowhere differentiable. While continuity is physically necessary and intuitively acceptable, the latter property (non-differentiability) cannot be shared by the physical phenomenon which we are modeling unless the particle has zero mass. So, the suggested model of the Wiener process is a poor model for the movement of particles over a short time interval; fortunately, over longer periods of time it reflects experimental results well and thus is an excellent model. The second is the issue of *existence* of such processes. In other words, the following question is of significant theoretical weight: does there exist a probability space[13] and a Gaussian process W thereon[14], satisfying the three properties listed above? This is a non-trivial question; the answer, however is affirmative.[15]

The Markov property may be translated into the following sequence of equalities,

$$P\left\{X(t+s) \leq a \,|\, X(s) = x_s,\ X(u),\ 0 \leq u < s\right\};\ t > 0 = P\left\{X(t+s) \leq a \,|\, X(s) = x_s\right\}$$

Which states that the conditional distribution of a future state $X(t+s)$ given its state at time s and its past history for all time u prior to time s depends only on the current state at time s.

1.4.3 Differential Representation

We introduce the differential notation here, as eventually we shall be using differential equations to represent the process parameters of interest to us. By letting τ become infinitesimally small we can represent the increment of a Wiener process, dw, in continuous time as

[12] To be more precise, every path is continuous "almost surely," in the sense that all paths, and there are uncountable number of them, are continuous except perhaps finitely many.

[13] A probability space is defined by the triplet $(\Omega, \mathcal{F}, P)$ where Ω is the sample space, $\mathcal{F}$ is the $\sigma-$algebra defined on Ω, and P is the probability measure defined on the $\sigma-$algebra.

[14] A stochastic process $\{X(t), t \geq 0\}$ is said to be Gaussian if the variables $X(t_1), \cdots, X(t_n)$ possess a multivariate normal distribution for all $n \geq 1$ and all $t_1, \cdots, t_n$.

[15] For a constructive proof of the existence of the Wiener process, see Grimmett & Strizaker [27], pp. 497-499.

$$dw = \epsilon_t \sqrt{dt} \tag{1.8}$$

in which ϵ_t is a random variable which follows a standard normal distribution. Therefore $E(dw) = 0$, and $V(dw) = dt$. This process has no derivative in the conventional sense; $\frac{\Delta w}{\Delta t} = \frac{\epsilon_t}{\sqrt{\Delta t}}$ which becomes infinite as Δt approaches 0.

1.4.4 Joint Distributions

Since $X(t)$ of equation 1.4 is normal with mean 0 and variance t (recall that we assume $\sigma = 1$), its density function is given by

$$f_t(x) = \frac{1}{\sqrt{2\pi t}} \exp\left(\frac{-x^2}{2t}\right)$$

We have assumed that the process starts at position 0. The *increments* of the process are independent and stationary, each possessing the above density function. Consequently the joint density of $X(t_1), X(t_2), \cdots, X(t_n)$ is given by

$$f(x_1, x_2, \cdots, x_n) = f_{t_1}(x_1) f_{t_2 - t_1}(x_2 - x_1) \cdots f_{t_n - t_{n-1}}(x_n - x_{n-1}) \tag{1.9}$$

In principle, one can use (1.9) to compute any desired probabilities. To illustrate, consider the (backward) determination of the conditional distribution of $X(s)$ given that $X(t) = x_t$, where $s < t$. [16] The desired conditional density is given by

$$f_{s|t}(x_s|x_t) = \frac{f_s(x_s) f_{t-s}(x_t - x_s)}{f_t(x_t)} \tag{1.10}$$

Now, $f_t(x)$ is given above,

$$f_s(x_s) = \frac{1}{\sqrt{2\pi s}} \exp\left(\frac{-x_s^2}{2s}\right)$$

and

[16] This is characterized as "backward" determination because it is assumed that we know the position at time $t > s$ and we wish to deduce the previous position at time s. Naturally, that position can be determined only in a probabilistic sense.

$$f_{t-s}(x_t - x_s) = \frac{1}{\sqrt{2\pi(t-s)}} \exp\left(\frac{-(x_t - x_s)^2}{2(t-s)}\right) \tag{1.11}$$

Substituting in (1.10) and simplifying, we secure

$$f_{s|t}(x_s|x_t) = \left[\frac{\sqrt{t}e^{-\frac{x_t^2}{2t}}}{\sqrt{2\pi s(t-s)}}\right] \exp\left\{\frac{x_s^2}{2s} - \frac{(x_t - x_s)^2}{2(t-s)}\right\}$$

which may be further simplified to

$$f_{s|t}(x_s|x_t) = \left[\sqrt{t}\,\exp\left(-\frac{x_t^2}{2t}\right)\exp\left(\frac{s(t-s)}{t^2}x_t^2\right)\right]\frac{1}{\sqrt{2\pi s(t-s)}}\exp\left(-\frac{t\left(x_s - \frac{sx_t}{t}\right)^2}{2s(t-s)}\right)$$

Hence, except for the constant term between square brackets, the (backward) conditional distribution of $X(s)$ given that $X(t) = x_t$ for $s < t$ is normal with mean and variance given by

$$\begin{aligned} E\left(X(s)|X(t) = x_t\right) &= \frac{sx_t}{t} \\ V\left(X(s)|X(t) = x_t\right) &= \frac{s(t-s)}{t} \end{aligned}$$

Note that the conditional variance of $X(s)$, given that $X(t) = x_t$, $s < t$, is *independent* of x_t.

To simplify notation, let $s/t = \alpha$, then the conditional distribution of $X(s)$ given $X(t)$ is normal with mean αx_t and variance $\alpha(1-\alpha)t$. Since $\alpha \in (0,1)$, it may be interpreted as a probability of "success" of a toss of a coin over the distance x_t in time t; its mean is α times the distance x_t, and its variance is the variance of a Bernoulli trial $\alpha(1-\alpha)$ times the time t.

We continue with one more property of this simple and fundamental process. Consider the covariance of the two random variables $X(s)$ and $X(t)$, $s \leq t$,

$$\text{Cov}\left(X(s), X(t)\right) = \text{Cov}\left(X(s), X(s) + X(t) - X(s)\right)$$

$$\begin{aligned} &= \text{Cov}\,(X(s), X(s)) + \text{Cov}\,(X(s), X(t) - X(s)) \\ &= s \end{aligned}$$

since $\text{Cov}\,(X(s), X(s)) = V\,(X(s)) = s$ and $X(s)$ are independent of the increment $X(t) - X(s)$. For instance, the correlation coefficient $\rho\,(X(t), X(s))$ is given by

$$\begin{aligned} \rho\,(X(s), X(t)) &= \frac{Cov\,(X(s), X(t))}{\sigma\,(X(s))\,\sigma\,(X(t))} \\ &= \sqrt{\frac{s}{t}} \end{aligned}$$

Thus, at "half time" to t the correlation coefficient is 0.707.

A property of Brownian motion is that as $\tau \to 0$ the expected total distance traveled by the particle over any finite interval of time becomes infinite! To see this, let Δx denote the infinitesimal distance covered in time τ. Clearly, $|\Delta x| = h$, whence $E\,(|\Delta x|) = h$. Then the total expected length of the path over a time interval of duration $t > 0$ is given by (recall that we put $h = \sigma\sqrt{\tau}$)

$$nh = \frac{h}{\tau}t = \frac{\sigma}{\sqrt{\tau}}t$$

which goes to infinity as τ goes to zero.

1.4.5 A Differential Equation

Consider for a moment the density function (1.11), which we re-write, for ease of notation, as

$$f(y, t|x, s) = \frac{1}{\sqrt{2\pi(t-s)}} \exp\left(\frac{-(y-x)^2}{2(t-s)}\right) \tag{1.12}$$

in which the conditioning in the expression of f means that the particle was in position x (instead of x_s) at time s. The density function of the position of the particle at time $t > s$ is denoted by f. This density function is the derivative of the probability distribution function $F(y, t\,|\,x, s)$; *i.e.,*

$$f(y,t|x,s) = \frac{\partial}{\partial y}\left[F(y,t|x,s)\right]$$

This is a function of four variables, *viz.*, x, y, s, and t. It turns out that f is the solution of the following differential equations:

$$\begin{aligned} \frac{\partial f}{\partial t} &= \frac{1}{2}\frac{\partial^2 f}{\partial y^2} : \text{ forward diffusion equation} \\ \frac{\partial f}{\partial s} &= \frac{1}{2}\frac{\partial^2 f}{\partial x^2} : \text{ backward diffusion equation} \end{aligned} \tag{1.13}$$

The names are indicative of the approach to derive the equations. Recall that the Wiener process is a Markov process. The forward equation assumes we know $w(t)$ and we secure $w(t+\tau)$ conditioned upon our knowledge of $w(t)$ for an infinitesimal time $\tau > 0$, then let $\tau \downarrow 0$. The backward equation assumes we know $w(s+\tau)$ and we secure $w(s)$ conditioned upon our knowledge of $w(s+\tau)$, then let $\tau \downarrow 0$. The coefficients in (1.13) are constants independent of the four parameters that define the density function, x, y, s, and t; this reflects the fact that the Wiener process is homogeneous in state and time, see properties (1) and (2) in section (1.4.2) above. This characteristic is not shared by other diffusion processes that are *not* homogeneous in either state or time, as we shall see below.

It can be shown that (1.12) is the unique density function that solves the forward or backward diffusion equations of (1.13) when either the initial or the terminal states are specified (see [27]).

1.4.6 Brownian Motion with Drift

In this section we wish to model a process that has mean and variance that are functions of time, in particular, *proportional* to time. The following development follows the steps of the previous one and motivates the final result *via* a random walk process.

Consider a particle that starts in position 0 and follows an asymmetric random walk in one dimension. In an infinitesimal time τ the particle takes one step of infinitesimal size h to the right with probability p, $0 \leq p \leq 1$, or one step $-h$ to the left with probability $q = 1 - p$. The value of p determines the propensity of the particle to "drift" to the right or to the left, hence the name. Let $X(t)$ denote the position of the particle at time t, measured only at discrete time epochs $1, 2, \cdots, \left\lfloor \frac{t}{\tau} \right\rfloor$, then (recall that $X_0 = 0$),

$$X(t) = h\left(X_1 + X_2 + \cdots + X_{\left\lfloor \frac{t}{\tau} \right\rfloor}\right) \tag{1.14}$$

where

$$X_i = \begin{cases} +1 & \text{if the } i\text{th step is to the right} \\ -1 & \text{if it is to the left} \end{cases}$$

Assume the X_i are independent. Since $E(X_i) = p - q$, and $V(X_i) = 4pq$, we easily see from (1.14) that

$$\begin{aligned} E\left(X(t)\right) &= hn(p-q), \text{ where } n = \left\lfloor \frac{t}{\tau} \right\rfloor \\ V\left(X(t)\right) &= 4npqh^2 \end{aligned}$$

We wish to let h, the size of the step, and τ, the time interval between steps, go to 0 (whence n goes to ∞) in such a way that the result is meaningful and independent of the (arbitrary) values of h, τ, p. To this end, let

$$h = \sigma\sqrt{\tau}$$

and

$$p = \frac{1}{2}\left[1 + \frac{\alpha}{\sigma}\sqrt{\tau}\right], \qquad \text{and} \qquad q = \frac{1}{2}\left[1 - \frac{\alpha}{\sigma}\sqrt{\tau}\right]$$

This seemingly artificial definition of the parameters h and p leads to

$$p - q = \frac{\alpha}{\sigma}\sqrt{\tau} = \frac{\alpha}{\sigma^2}h$$

so that

$$E\left(X(t)\right) = \frac{t}{\tau}\left(\frac{\alpha}{\sigma}\sqrt{\tau}\right)\left(\sigma\sqrt{\tau}\right) = \alpha t$$

and

$$V\left(X(t)\right) = \left[1 - \left(\frac{\alpha}{\sigma}\right)^2 \tau\right]\frac{t}{\tau}\left(\sigma^2\tau\right) \to \sigma^2 t$$

which indicate a process with mean at time t equal to αt and variance $\sigma^2 t$, both proportional to time, as desired. Let $\sigma = 1$ to secure the *standard Brownian motion with drift*. This latter could also be defined as

$$X(t) = \alpha t + B(t) \tag{1.15}$$

where $\{B(t)\}$ is the standard Brownian motion developed in section (1.4.2).

1.4.7 Differential Equation Representation

It is easy to see that, by letting the time increment τ be infinitesimally small, the Brownian motion with drift can have the following differential equation:

$$dx = \alpha dt + \sigma dw \tag{1.16}$$

where dw is the increment of a Wiener process as defined in equation 1.8.

1.4.8 Ito Processes and Ito's Lemma

The Wiener process with drift discussed in the previous section can be generalized in several ways, all of which are special cases of the following differential equation

$$dx = a(x,t)dt + b(x,t)dw \tag{1.17}$$

where dw is the increment of a (standard) Wiener process, and $a(x,t)$ and $b(x,t)$ are deterministic functions of state and time. It is evident that the "Wiener process with drift" of section (1.4.6) is a special case when $a(x,t) = \alpha$ and $b(x,t) = \sigma$, as easily seen in (1.16).

The mean and variance of increments of this process are easily evaluated as follows. Since $E(dw) = 0$, $E(dx) = a(x,t)dt$. $V(dx) = E[dx]^2 - (E[dx])^2$, which contains terms in dt, in $(dt)^2$, and in $(dt)(dw)$ which is of order $(dt)^{3/2}$. Ignoring higher powers of dt beyond the first, the variance is given by $V(dx) = (b(x,t))^2$. We refer to $a(x,t)$ as the instantaneous *drift rate* and to $b(x,t)$ as the instantaneous *variance rate*.

The Ito process of (1.17) is continuous in time but *not differentiable*. However, we often need to work with functions of the Ito process in which the differentials of such functions is necessary. To do this, we need to make use of Ito's Lemma, which is discussed next.

Suppose that $x(t)$ follows the process of equation (1.17), and consider a function $F(x,t)$ that is at least twice differentiable in the state x and once in t. The total differential of F is given by

$$dF = \frac{\partial F}{\partial x}dx + \frac{\partial F}{\partial t}dt + \frac{1}{2}\frac{\partial^2 F}{\partial x^2}(dx)^2 + \cdots \tag{1.18}$$

We secure $(dx)^2$ from (1.17),

$$(dx)^2 = a^2(x,t)(dt)^2 + 2a(x,t)b(x,t)(dt)^{3/2} + b^2(x,t)dt + \cdots$$

where the "$\cdots$" represent terms of higher order in dt. Since terms in dt of higher order than 1 go to zero faster than dt as it becomes infinitesimally small, they can be ignored and we may write

$$(dx)^2 = b^2(x,t)dt$$

Terms in (1.18) of higher order than $(dx)^2$ (assuming that F is more than twice differentiable) will also include terms of higher order than linear in dt, which can be ignored. Substituting for dx and $(dx)^2$ in (1.18) and collecting terms, we get

$$dF = \left[a(x,t)\frac{\partial F}{\partial x} + \frac{\partial F}{\partial t} + \frac{1}{2}b^2(x,t)\frac{\partial^2 F}{\partial x^2}\right]dt + b(x,t)\frac{\partial F}{\partial x}dw \tag{1.19}$$

This is Ito's Lemma. Note that (1.18) differs from the common chain rule for differentiation in ordinary calculus in that it contains the term $\frac{1}{2}\frac{\partial^2 F}{\partial x^2}(dx)^2$. This is to account for the convexity, or concavity, of the function F in the determination of its expectation. Indeed, suppose for the moment that $a(x,t) = 0$; then $E(dx) = 0$, so that the expectation of the first term in (1.18) is zero. Suppose, additionally, that $\partial F/\partial t = 0$; then the expectation of the second term in (1.18) is also zero. However, $E(dF) \neq 0$ but equal to the expectation of $\frac{1}{2}\frac{\partial^2 F}{\partial x^2}b^2(x,t)$. If F is convex (concave), $\frac{\partial^2 F}{\partial x^2} > 0$ (< 0) and dF will also be > 0 (< 0). This is an implication of Jensen's inequality: the expected value of a convex (concave) function of a random variable is larger (smaller) than the function of the expected value of the variable itself. For Ito processes, $(dx)^2$ behaves like dt, so the effect of convexity or concavity cannot be ignored in the evaluation of the expected value of the differential of F. The third term in (1.18) $\frac{1}{2}\frac{\partial^2 F}{\partial x^2}b^2(x,t)$ captures this effect.

1.4.9 Differential Equation Representation

Let $D \equiv \{D(t) : t \geq 0\}$ denote a diffusion process that progresses following (1.17). It can be given the following interpretation: over an infinitesimal interval of time $\tau > 0$, we specify the mean and variance of increments $D(t+\tau) - D(t)$ as follows:

$$\begin{aligned} P\left(|D(t+\tau) - D(t)| > \varepsilon \,|\, D(t) = x\right) &= o(\tau) \\ E\left(D(t+\tau) - D(t) | D(t) = x\right) &= a(x,t)\,\tau + o(\tau) \\ E\left(\left[D(t+\tau) - D(t)\right]^2 | D(t) = x\right) &= b^2(x,t)\,\tau + o(\tau) \end{aligned}$$

In a manner similar to the derivation of equation (1.12), the conditional density function of $D(t)$ given that $D(s) = x$ is denoted by $f(y,t\,|\,x,s)$, and defined as

$$f(y,t\,|\,x,s) = \frac{\partial}{\partial y}P\left(D(t) \leq y \,|\, D(s) = x\right)$$

It satisfies the following differential equations ([27])

$$\begin{array}{l} \frac{\partial f}{\partial t} = -\frac{\partial}{\partial y}\left[a(y,t)\, f\right] + \frac{1}{2}\frac{\partial^2}{\partial y^2}\left[b^2(y,t)\, f\right] : \text{ forward diffusion equation} \\ \frac{\partial f}{\partial s} = -a(x,s)\,\frac{\partial f}{\partial x} - \frac{1}{2}\, b(x,s)\,\frac{\partial^2 f}{\partial x^2} : \text{ backward diffusion equation} \end{array} \tag{1.20}$$

As noted by [27], pp. 494, "It is an extraordinary fact that the density function f is specified as soon as the instantaneous mean a and variance b^2 are known; we need no further information about the distribution of a typical increment." Typically, these two parameters are often specified by the nature of the process itself in a natural manner.

The Wiener process is secured as special case if $a(y,t) = 0$ and $b^2(y,t) = \sigma^2$, for some $\sigma^2 > 0$. On the other hand, the Brownian motion with drift is another special case when $a(y,t) = \alpha$ and $b^2(y,t) = \sigma^2$. The forward diffusion equation becomes

$$\frac{\partial f}{\partial t} = -\alpha\,\frac{\partial f}{\partial y} + \frac{1}{2}\sigma^2\frac{\partial^2 f}{\partial y^2}$$

from which we deduce that the corresponding diffusion process is such that

$$D(t) = \alpha t + w(t)$$

where w is a Wiener process. (Compare with (1.16).)

1.5 Thesis Outline

The emphasis of this research is on developing a framework for analyzing projects under the paradigm of diffusion processes. A second goal is to demonstrate the computational feasibility of such a paradigm. Thus the thesis is divided into two main parts: the first, Chapters 2 and 3, treat two issues in activity networks from the point of view of diffusion theory, and the second, which comprises Chapters 4 and 5, resolves two 'side issues' that were needed to perform the first objective.

Completion Time Distribution: Chapter 2 focuses on developing a computationally feasible model for estimating the completion time estimates of *DiAN*. This model is then compared with those obtained from classical analysis.

Controlling the Activity: Chapter 3 deals with the issue of controlling an activity that has deviated substantially from its intended course. Various strategies were developed and evaluated for controlling the activities.

The Generation of Random Networks: Chapter 4 treats the problem of the generation of random networks of given complexity index, DAGEN. Relevant computer codes were developed to implement the proposed model. These codes are available to the scientific community at large.

The Convolution of Random Variables: Chapter 5 develops a computationally better algorithm for the determination of the *cdf* of the sum of two *r.v.*'s, which is achieved through convolution. The procedure has been programmed and the code is also available to the scientific community.

Summary and Conclusions: Chapter 6 terminates the thesis with a summary of the results achieved, a discussion of their significance, and suggestions for future research.

Chapter 2

Project Management and DiAN

Chapter 1 introduced the concepts of the use of directed acyclic graphs (*dag*) to represent the topological characteristics of a project. For the successful modeling of a project the following are required.

1. A finite set of activities, which, taken together, constitute the project,
2. the precedence relationship among activities, and
3. the relevant characteristics of each activity.

The process of breaking a project into a finite set of tasks is an art. It is based on the prior experience of the management with the similar projects, technological requirements, safety issues, availability of resources, among others. In this work we assume that for a given project one can create the set of activities and the precedence relations among them.

The relevant characteristic of an activity that interests us is its execution time (duration) estimate. A deterministic estimate of the duration, while common, is applicable only to a limited set of projects. If we accept that for most of the projects the duration of an activity would be a random variable then an obvious question arises: What is the nature of this process and how could it be quantified? A task that we wish to explore in this chapter.

Past literature in the field of project management has focused on topological issues and on analysis of these networks. It has largely ignored the issues related to the estimation of probability density function (*pdf*) of an activity. The common activity duration estimation techniques can be categorized in two broad sets. One being the statistical estimate of distribution from data. The second is an estimation based on expert opinion.

Statistical techniques inherently presume that the project on hand is similar to the past projects. In many industries, such as small scale construction, this technique would be adequate in establishing reasonable estimates of the activity durations. However, in many projects, such as software development, the data from the past projects is largely inapplicable. The variability between any two software development projects is large enough to render past information to a limited use, if not downright useless.

The second popular method is to ask domain experts for an estimate of a pessimistic, an optimistic and a most likely duration for an activity. These estimates are then used to derive the beta or a triangular distribution for the activity duration. In the event that multiple users offer varying estimates for various quantities an average (possibly weighted) number is obtained. The expert opinion of the duration of an activity is prevalent in the service industries. It is evident that such an estimate is highly subjective and unreliable.

Needless to say, the issue of estimating the requisite parameters and the distribution function of an activity is of great importance with practitioners (project managers). An estimate that a project manager could identify with relative ease is the rate of progress and the error of estimates over the planning horizon of an activity (project) as opposed to its "true" *pdf*. This information, when translated into a triangular and/or beta distribution loses much of its characteristic. A better approach would be to derive a distribution function based on the raw information and not on an a priori assumption of triangular/beta distribution.

A closer look at the quantities estimated by the practitioners reveals that the error estimates in the duration of an activity increase with the duration of an activity. In other words the estimate of the variance (or the error) in the duration of an activity changes with time. The traditional approaches of planning for an activity using standard fixed variances may not be able to deal with this issue effectively.

In short, the existing paradigm is incapable of modeling a phenomenon where only the rate of progress and the error of the estimate are available. These distributions, while capable of modeling a wide variety of situations, fail to capture the variability in the variance of the completion time of an activity. Thus the assumptions of a "standard distribution" leads to erroneous subsequent analyses.

The software development project is a particularly good example of this variability in the variance of an activity completion time. Here, the estimate of the remaining work content (*rwc*) changes with time both in expectation and variance. Research and development, and large scale construction, are also typical examples of such projects.

2.1 Modeling an Activity as a Diffusion Process

The development of this paradigm was based on realizing this inherent variability in estimating the error of activity duration estimates. A better measure of the activity status would be the level of work that remains to be done (or equivalently has been done) and how this estimate is changing with time. With this measure the completion time of an activity can be identified with the time when the amount of work that remains to be done is zero.

Estimating the overall progress of an activity comprises two elements. One being the deterministic aspect and the second being the stochastic variability. The deterministic part estimates the rate of progress. It is the rate at which the remaining work content (*rwc*) decreases with time. This rate of progress ignores any randomness in the estimate. The second part of the estimate, *estimation error*, provides the required randomness in the estimate of *rwc* over time. It should be noted that the estimation

error changes with time.

These parameters can be identified with the rate of completion or instantaneous mean, and a changing variance (the error of estimates). Thus modeling the progress of an activity with its *rwc* and the changes in *rwc* with time provides the basic building block for our model.

A fundamental difference between our approach and the traditional models, based on the *PERT* and its offshoot, is that the *rwc* need not be monotonically decreasing. Here an activity's *rwc* may increase or decrease as the time progresses. Thus *rwc* as the *state* of an activity allows one to model the previously unknown factors that may impact its duration.

The strength of this modeling paradigm resides in its ability to construct the *pdf* of activity completion time from the minimal information received from the practitioners and to adopt dynamically to actual progress. In the following section we present a systematic approach for obtaining these parameters for an activity duration.

2.1.1 Estimating Activity Parameters

In the previous section we presented an overview of the parameters that are required to estimate the activity completion time under this model. In this section these issues are explored in greater detail.

The main parameters for a diffusion model are the drift rate (also known as instantaneous mean) and the variance parameter (also known as instantaneous variance). These two parameters are referred to as $a(x,t)$ and $b^2(x,t)$ (or μ and σ^2 if constant) respectively.

The instantaneous mean in terms of an activity duration is the deterministic estimate of the rate of completion of an activity. The time unit of measurement is dependent on the project at hand. In general, a daily or weekly estimate is typically adequate for project planning and control. If a manager estimates that it would take 100 days to complete an activity in absence of any stochastic variations, then the instantaneous mean $\mu = -0.01$. A statistical estimate of the instantaneous mean can

be made if it can be assured that the performance of the past activities is applicable to the work at hand.

The instantaneous variance coefficient can be obtained by knowing the maximum allowed deviation from the deterministic rate of progress. For this work we have assumed $b^2(x,t)$ to be constant, which is denoted by σ^2. We require that the manager provides an allowable spread (ψ) at some time (from that beginning of the activity, t_0) $t_0 < t < T$. Here T denotes the expected completion time of the activity. Given that an activity starts at $rwc = 1$ at time $t = 0$, $T = -\frac{1}{\mu}$. Given that rwc follows a diffusion process, it immediately follows that $\sigma \equiv \frac{\psi}{6\sqrt{t}}$. A closer look at this estimate of instantaneous variance reveals that the error of estimate is zero at the beginning of an activity and linearly increases until the end of the activity.

The above methods provide a straightforward approach to estimating of instantaneous mean and variance. Both of these two parameters essentially provide a constant value. In many situations this constant value may not be suitable. As an example, if the duration of an activity is linked to the stock price movement then the instantaneous mean can be better represented as a function of both current state and time. The model presented here is indifferent to the type of function selected for either instantaneous mean or variance.

This estimate of rate of completion and of variance can be updated during the life of an activity to compute the updated distribution function of completion time. This presents a powerful tool to management for a what-if analysis. In Chapter 3 we used these techniques to analyze activities that have significantly deviated from their intended course.

This work focuses on diffusion processes with constant drift and variance parameters. The methodology developed here, however, is not restricted to constant value of these parameters. The work of securing the results for other type of diffusion process is left for the future researchers.

Table 2.1: Project # 1

Month	Labor Cost	Est. of Remaining Cost	Exp. remaining cost	Deviation in cost
1	920	32080	28875	3205
2	6158	25922	24750	1172
3	6689	16734	20625	-3891
4	9889	9345	16500	-7155
5	7192	2000	12375	-10375
6	1394	2000	8250	-6250
7	156	200	4125	-3925
8	0	200	0	200

Table 2.2: Project # 2

Month	Labor Cost	Est. of Remaining Cost	Exp. remaining cost	Deviation in cost
1	3414	72080	68631.8	3448.2
2	5128	66954	61768.6	5185.4
3	16026	50930	54905.5	-3975.5
4	11887	39040	48042.3	-9002.3
5	15690	23350	41179.1	-17829.1
6	13066	20285	34315.9	-14030.9
7	13267	12000	27452.7	-15452.7
8	14181	12000	20589.6	-8589.6
9	14148	2000	13726.4	-11726.4
10	8056	500	6863.2	-6363.2
11	151	0	0	0

2.1.2 Empirical Analysis

We received the data on actual performance of three projects from a local construction company. These projects consists of only one activity. The goal is to study this data in light of *DiAN*. In particular the question that we were trying to answer was if the progress of these projects could be explained using the principles of *DiAN*. Unfortunately, the data we received is incomplete to perform the analysis to its required rigor. We nonetheless decided to analyze the data to to see if any meaningful conclusions can be drawn. The data along with the calculations for these three activities is presented in Tables 2.1, 2.2 and 2.3 respectively.

Table 2.3: Project # 3

Month	Labor Cost	Est. of Remaining Cost	Exp. remaining cost	Deviation in cost
1	4990	130000	126562.5	3437.5
2	6663	123000	118125	4875
3	7798	115000	109687.5	5312.5
4	6921	131000	101250	29750
5	8518	122000	92812.5	29187.5
6	15783	106000	84375	21625
7	9379	97000	75937.5	21062.5
8	14506	84000	67500	16500
9	21144	63000	59062.5	3937.5
10	21108	48000	50625	-2625
11	15394	30000	42187.5	-12187.5
12	19011	20000	33750	-13750
13	9456	10000	25312.5	-15312.5
14	6744	3000	16875	-13875
15	2012	1000	8437.5	-7437.5
16	65	1000	0	1000

The original estimates for the three activities were \$33,000, \$75,495 and \$135,000 respectively. The actual final cost of the three activities is \$32,398, \$113,012 and \$169,492 respectively. In all three tables, the entries represent the actual expenses for the month occurred, the estimated remaining cost, the expected remaining cost in absence of any uncertainty and the deviation observed from the expected remaining cost. The last two columns of the tables were computed as a part of the analysis detailed below.

The important piece of information that is missing from the data is an initial estimate of the instantaneous mean and variance. The absence of an estimate of instantaneous mean and variance requires us to use the given data to derive a meaningful, albeit of questionable validity, information on these two parameters.

The instantaneous mean for the three projects was computed by dividing the initial estimate by the observed duration of the project. These are $-4,125$, $-6,863.2$ and $-8,437.5$ respectively for projects $1, 2$ and 3. The expected remaining cost in Tables

2.1, 2.3 and 2.2 was calculated using the instantaneous mean values thus derived. For instance, for project # 1, the expected remaining cost would be $\$33,000 - \$4,125 = \$28,875$ for first month and $\$28,875 - \$4,125 = \$24,750$ for second month.

An estimate of instantaneous variance for three activities was estimated by taking the information from first half life of the activities. We computed the difference between the expected remaining cost and actual remaining cost for the activities. These values are shown in the last columns of the Tables 2.1, 2.2 and 2.3. The instantaneous variance is thus the variance calculated for the first half of the entries in the deviation column. These standard deviations are 4720.90, 9422.96 and 10787.16 respectively for projects $1, 2$ and 3. For instance, for project # 1, the standard deviation was calculated for the numbers $3205, 1172, -3981$ and -7155.

In the analysis we would like to examine if the behavior of the activities could be explained using the concepts of *DiAN*. Consider Figures 2.1, 2.2 and 2.3. These figures show the progression of the activities as observed in the field. As evident from the figures that the progression of the projects are within one-sigma deviation from the mean except in few cases. The deviation never crosses the two-sigma deviation lines. The figure also shows three sets of extra lines. The two extreme set of dotted lines show the envelope of σ and 2σ deviation on either side of the line representing the mean. The standard deviation at a given time was calculated as follows.

$$\sigma_t = \sigma\sqrt{t}$$

The comparison of the actual progression of the activities with *DiAN* clearly indicates that the behavior of three activities can be explained using the principals of *DiAN*, and there is no reason to reject the *DiAN* hypothesis.

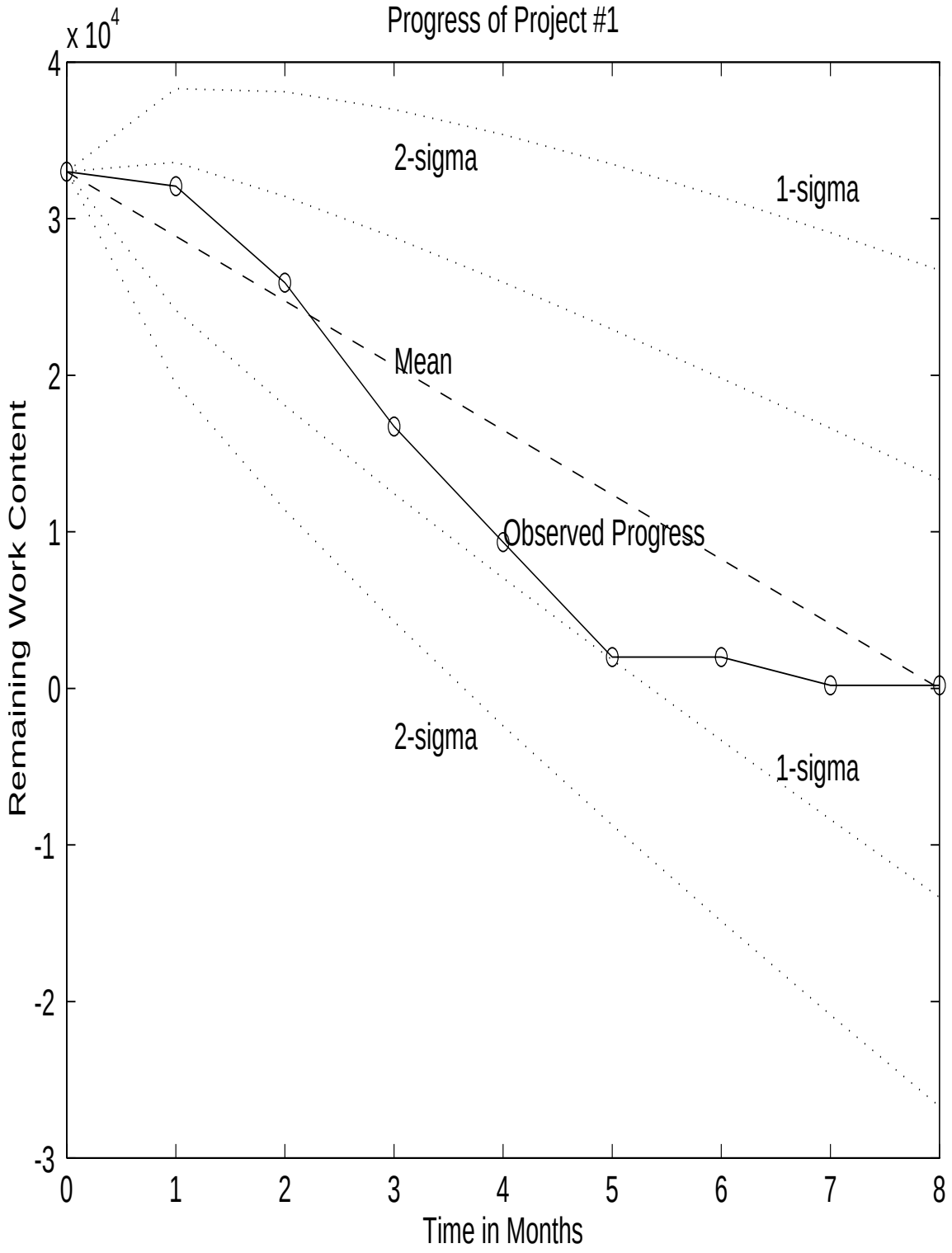

Figure 2.1: Project # 1

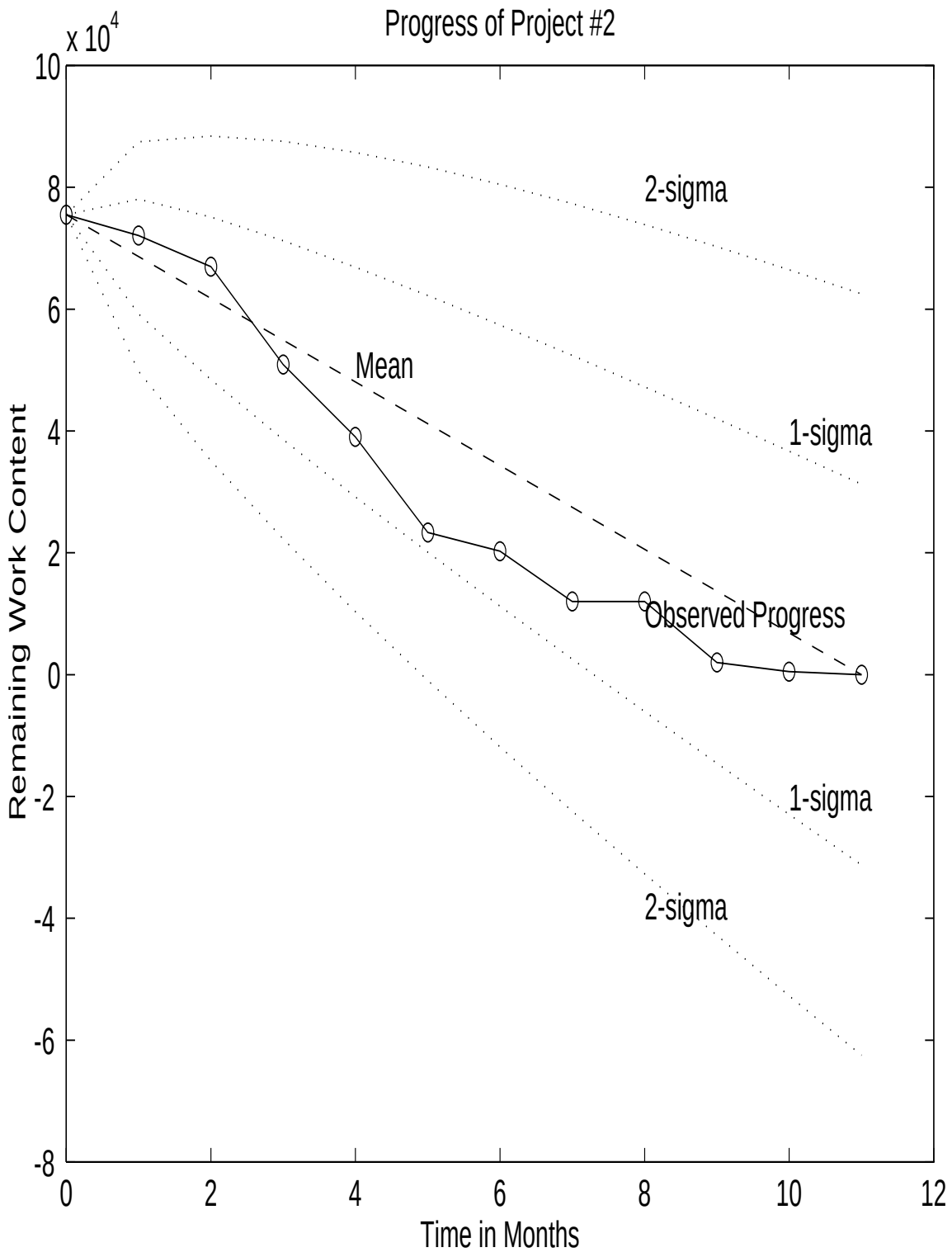

Figure 2.2: Project # 2

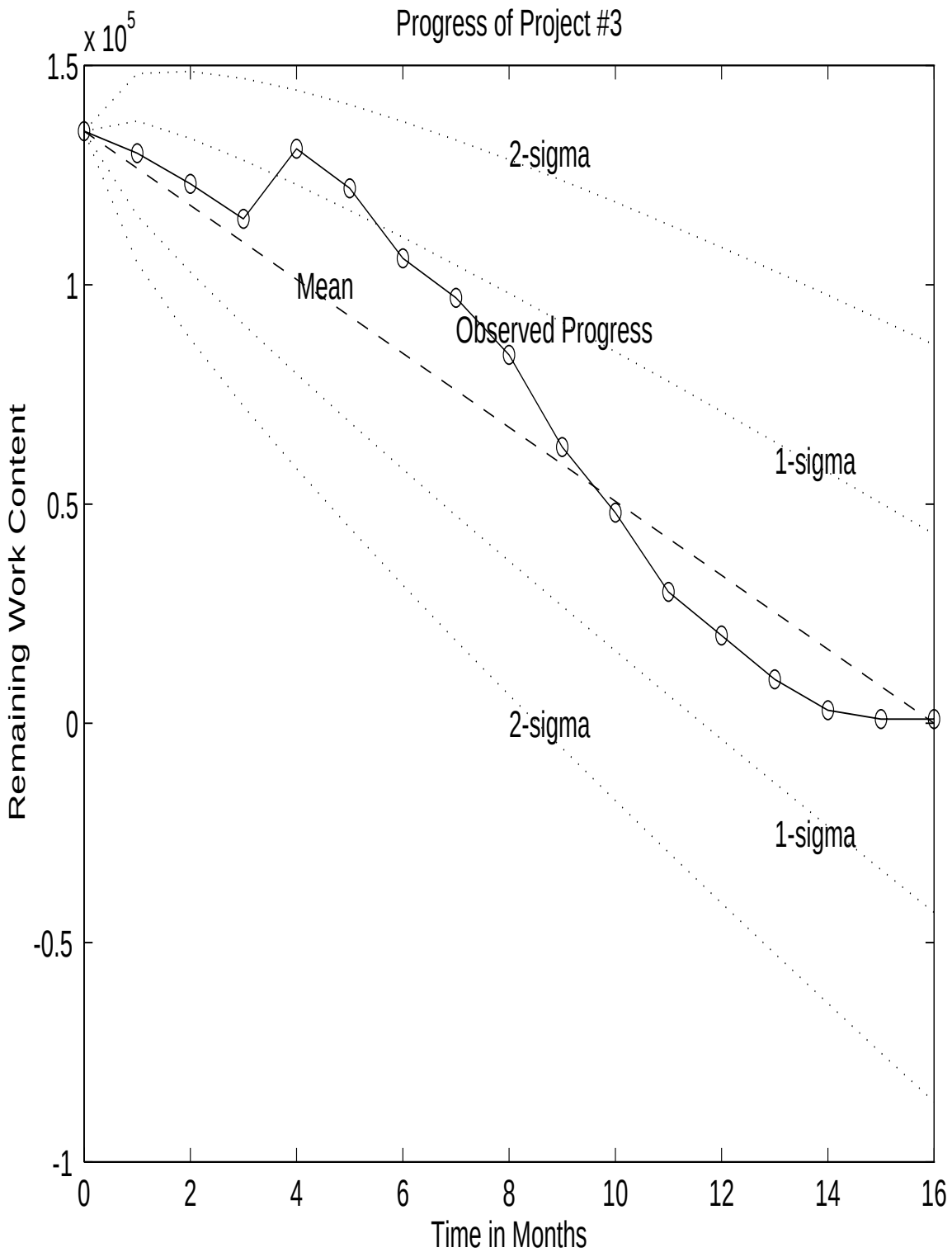

Figure 2.3: Project # 3

The question that still remains to be answered is that if we would have failed to reject other probability distributions. Unfortunately, we could not test this part of the hypothesis due to the lack of the required information. We did not know the probability distributions and the associated parameters for the three projects. Moreover, we do not know if the projects were designed with stochastic behavior in mind. The only data available is the final cost of completing the project. The single data point for a project precludes us from performing the requisite analysis.

2.1.3 The Model

The state of the system is modeled as *rwc*, the starting time $rwc = 1$, and the completion as $rwc = 0$. To represent the completion of an activity, an absorbing barrier is placed at $rwc = 0$.

The following discussion assumes an *AoA* representation of projects [20]. Some, or all of the activities in the project are assumed to follow a diffusion process. The *state* of system, *rwc*, is represented by x or y.

The state of an activity as a diffusion process can now be written as

$$\begin{aligned} dx &= a(x,t)\,dt + b(x,t)\,dz \\ dz &= \epsilon_t\sqrt{dt} \\ \epsilon_t &= N(0,1) \end{aligned} \tag{2.1}$$

Here, t is time; dz is the increment of the Wiener process; $a(x,t)$ is the instantaneous mean of the process; $b^2(x,t)$ is its instantaneous variance; and $N(0,1)$ is standard normal variate. The activity is complete (*i.e.*, its *rwc* is zero) when the process first reaches the absorbing barrier.

Let $D(t,x)$, $t \geq 0$, represent the duration of the activity with *rwc* of x at time t. The following (standard) assumptions are made on D.

Assumption 1. D is "regular;" *i.e.,*

$$P[D(t,x) > 0 \,|\, x > 0] = 1$$

In words: it is almost sure that with *rwc* $x > 0$ the time it takes to complete the activity is strictly > 0.

Assumption 2. $x(0) = 1$. That is, the initial work content (at time 0) is 1.

Assumption 3. An absorbing barrier is placed at $x = 0$. This denotes the completion of the project.

Assumption 4. $D(t,0) = 0 \Rightarrow D(t',0) = 0 \;\forall\, t' > t$. In words, if the duration of the activity is zero because it has been completed, then its "duration" stays at zero thereafter.

The last assumption permits us to define T, a *rv*, as the completion time of the activity, given by

$$T = \inf\{t \,|\, D(t,0) = 0\}$$

Let $f(x,t)$ denote the probability density function of D. It can be obtained by solving the Chapman-Kolmogorov forward equation with appropriate boundary conditions [27]; which may be written as

$$\frac{\partial f}{\partial t} = -\frac{\partial[a(x,t)\,f]}{\partial t} + \frac{1}{2}\frac{\partial^2[b(x,t)\,f]}{\partial x^2} \tag{2.2}$$

The boundary conditions are:

$$f(0,t) = 0,\ \forall\, t > 0 \tag{2.3}$$

$$f(x,0) = \delta_d(x),\ x \geq 0 \tag{2.4}$$

where δ_d is the Dirac delta function, $\delta_d = 1$ at $x = 1$ and $= 0$ elsewhere[1]. The *pdf* of completion time of an activity can be determined using equation (2.2). It may be difficult to solve (2.2) analytically for general functions $a(x,t)$ and $b(x,t)$. However, numerical solutions are attainable.

Consider the case when $a(x,t) = \mu$, a constant and $b(x,t) = 1$. The diffusion equation now simplifies to

$$dx = \mu dt + dz.$$

The Chapman-Kolmogorov equation (2.2) is now given by

$$\frac{\partial f}{\partial t} = -\mu \frac{\partial f}{\partial x} + \frac{1}{2}\frac{\partial^2 f}{\partial^2 x}$$

Let $f(y,t \mid x,0)$ denote the density *pdf* at time t when the process is initiated in state x (*i.e.,* at time 0). The above differential equation is now solved [27] to yield

$$f(y,t \mid x,0) = \frac{1}{\sqrt{2\pi t}} \exp\left[\frac{-1}{2t}(y - x + \mu t)^2\right];\ t > 0 \tag{2.5}$$

The $f(y,t)$ is a linear combination of $(y,t \mid +x,0)$ and $f(y,t \mid -x,0)$, and can be written as

$$f(y,t) = f(y,t \mid +\ x,0) - f(y,t \mid -x,0)\ \exp(-2\mu x) \tag{2.6}$$

The activity completion time density *pdf* for a given initial *rwc* x is given by

$$f(t) = \frac{x}{\sqrt{2\pi t^3}} \exp\left[\frac{-1}{2t}(x + \mu t)^2\right];\ t > 0 \tag{2.7}$$

Recall that the initial *rwc* is always $x = 1$, therefore (2.7) shall be written as

$$f(t) = \frac{1}{\sqrt{2\pi t^3}} \exp\left[\frac{-1}{2t}(1 + \mu t)^2\right];\ t > 0 \tag{2.8}$$

The *cdf* $F(t)$ may be written as

[1]The more precise definition of the Dirac delta function is as follows. Let $x \in \Re$; then $\delta_x(y) = 0$ if $y \neq x$, and $\int_{-\infty}^{+\infty} g(y)\,\delta_x(y)\,dy = g(x)$ for all integrable $g : \Re \to \Re$.

$$F(t) = P(T \leq t) = 1 - \Phi\left(\frac{\mu t + x}{\sqrt{t}}\right) + e^{-2\mu x}\,\Phi\left(\frac{\mu t - x}{\sqrt{t}}\right);\ t > 0 \qquad (2.9)$$

where $\Phi \sim N(0,1)$. For $a(x,t) = \mu$, a constant and $b(x,t) = \sigma$, a constant, we can write the activity completion time density function *pdf* as

$$f(t) = \frac{1}{\sigma\sqrt{2\pi t^3}} \exp\left[\frac{-1}{2\sigma^2 t}(1+\mu t)^2\right];\ t > 0 \qquad (2.10)$$

2.2 Project Completion Time Problem

2.2.1 Bounding the Project Duration

Given a project $G(N, A)$ one can compute the *pdf* of all activities under DiAN by using the concepts of section (2.1.3). The next step is to obtain the expected completion time of the last node $|N| = n$. The bounding scheme used here is due to Elmaghraby [23]. The details of this algorithm are presented below.

The bounding scheme proposed by Elmaghraby [23] is based on the concept of the "control network" of the given project. It provides tighter upper and lower bounds on the expected value of completion time than other approaches proposed to date in the literature. The lower and the upper bounds, are denoted by $\underline{T_n}$ and $\overline{T_n}$, respectively.

The algorithm also provides the convexly lower and the upper bounds on the *pdf* of completion time. Hence, in turn, the bounds on the expected value of completion time can be computed. The algorithm can be used recursively to bound the completion of all nodes in an activity network.

The complete algorithm (for upper bound) can be summarized in the following steps.

1. Find the complexity index of the network using the BKS [4] approach. Also determine the sequence of the nodes to reduce,

2. Replace, in a sequential manner, the node to be fixed (all the incoming and outgoing arcs) with new arcs which distribution is the convolution of the incoming and outgoing arcs[2],

3. Compute the upper bound on the completion time of each node by successive application of convolution and maximum operations.

The lower bound can be computed by replacing the distribution of the arc to be reduced in step 2 above with its expected value. This algorithm is programmed in MATLAB for discrete *df's*. Chapter 5 presents a detailed discussion of the issues related to discretization of continuous density functions. Issues related to convolution of discrete functions are also discussed in that chapter

2.2.2 Project Completion Time Computations

In this section we present several examples of such an analysis. The sample networks in this analysis were generated using the method outlined in Chapter 4.

Example 1

Consider the "interdictive graph" in AoA representation, Fig.2.4, which represents the following precedence relations among the activities (where "$\prec$" denotes "precede"): $1 \prec 3, 4$; $2 \prec 5$; $3 \prec 5$.

Here, the set of nodes is $N = \{1, 2, 3, 4\}$; whence $n = |N| = 4$, and the set of arcs (activities) is $A = \{1, \ldots, 5\}$, whence $m = |A| = 5$. Assume that the instantaneous mean of each activity $a_k(x, t)$ is constant over time, denoted by μ_k, for $= 1, \ldots, 5$[3]; and that the instantaneous variance $b_k^2(x, t)$ is also constant over time and $= 1$[4]. $\mu_k = [-\frac{1}{5}, -\frac{1}{10}, -\frac{1}{10}, -\frac{1}{8}, -\frac{1}{3}]$ for $k = 1, \ldots, 5$.

[2]Note that reducing a node may involve 'fixing' a "composite" arc: i.e., an arc that is the result of prior node reductions.

[3]This implies that the expected value of the duration of the activity increases, or decreases, linearly over time depending on the sign of μ_k.

[4]This implies that the variance of the duration of the activity increases linearly over time at rate $= 1$.

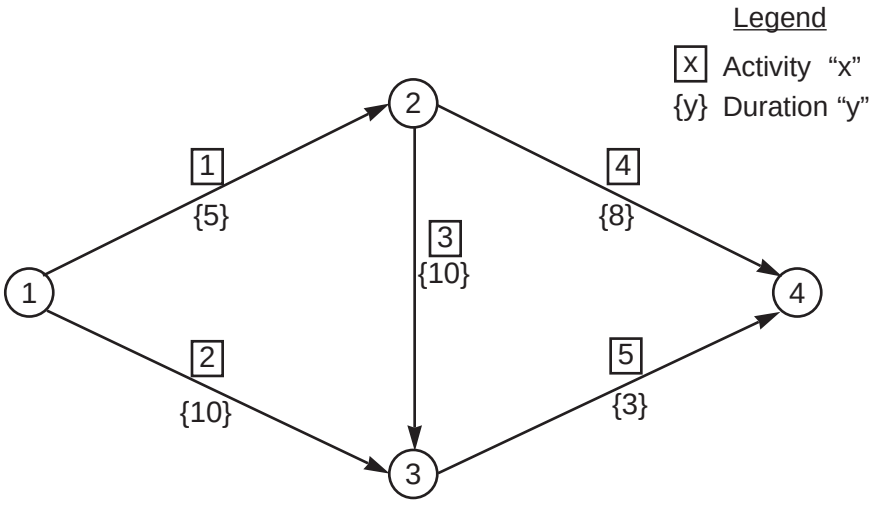

Figure 2.4: DiAN, Project 1

Using equation (2.8) the *pdf* of each activity is calculated. It is computationally difficult to compute the maximum or convolutions of two such density functions. This problem was dealt with by approximating each activity by a discrete probability mass function as described in Chapter 5.

The lower and the upper bounds on the expected value of the completion time of the project were estimated using the procedure outlined in section (2.2.1). The result is as follows:

$$l.b. = 14.2915 < E(T_4) = 15.3888 < u.b. = 15.5141$$

A Monte Carlo sampling of various activities of the project was performed to determine the 'exact' expected value $E[T_4] = 15.3888$.[5] The estimate of the expected value based on the lower and the upper bounds is their mean, 14.9028. The complete *pdf* of the completion time is shown in Fig. 2.5. The Fig. 2.5 also contains two other *pdf's* which would be obtained if the activities were assumed to have the poisson or

[5] A "crude" Monte Carlo sampling was used with no attempt at *its* variance optimization. The experiment was based on 1600 runs to guarantee that the resulting estimate does not vary from its "true" value by more than 5% at least 95% of the time.

the uniform distributions, following the classical approach of specifying the activity duration. More about these curves is presented in Sec. 2.2.2 below.

example 2

The second project network consists of 9 activities and 6 nodes and has complexity index 2, shown in Fig. 2.6. This network was generated using the procedure outlined in Chapter 4. In this project we have taken $b(x,t) = 1$ for all activities.

The bounds on the completion time of Project 2 following the *DiAN* model are as follows,

$$l.b. = 14.7733 < E[T_6] = 17.7080 < u.b. = 19.2423$$

As before, the "exact" value of the expected duration was secured via Monte Carlo sampling. The estimate of the expected duration based on the lower and upper bound is 17.7080.

Example 3

The third project shows an example of $b(x,t) \neq 1$ generated using the techniques of Chapter 4. The project consists of 7 nodes, 11 activities and has a complexity index of 3. The nodes to be reduced are $2, 3$ and 5. The project is shown in Figure 2.7.

The bounds on the completion time of Project 3 following the *DiAN* model are as follows,

$$l.b. = 30.5468 < E[T_7] = 30.5764 < u.b. = 31.8963$$

As before, the "exact" value of the expected duration was secured via Monte Carlo sampling.

Validity of DiAN Model

In proposing a new paradigm for the study of of the complex projects we are keenly aware of the need to demonstrate two propositions. The first is the difference between

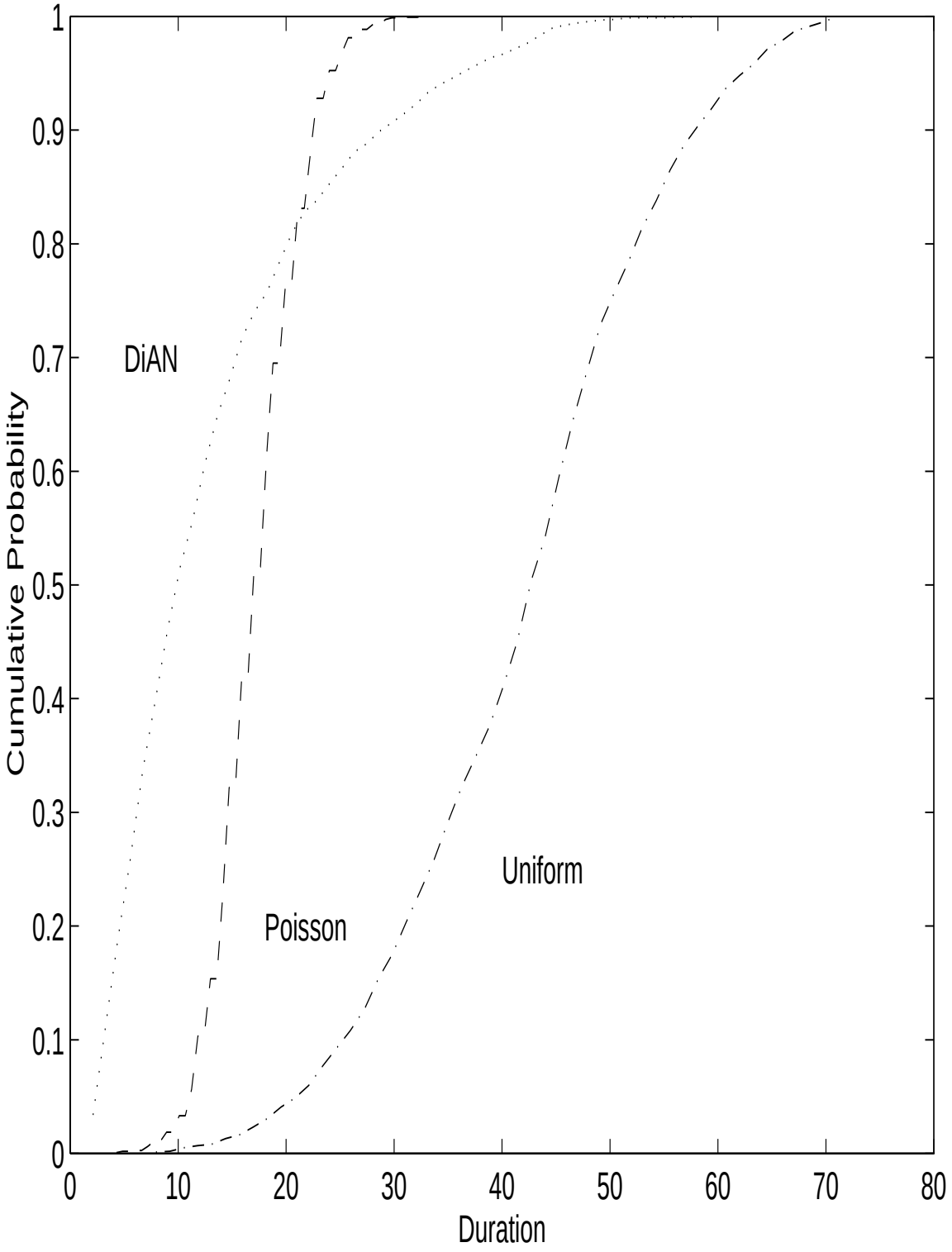

Figure 2.5: Results of Monte Carlo Sampling, Project 1

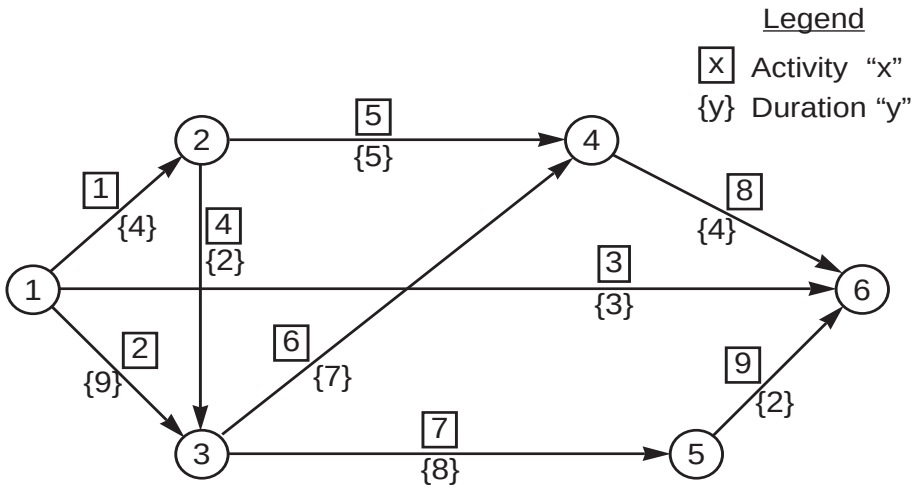

Figure 2.6: DiAN, Project 2

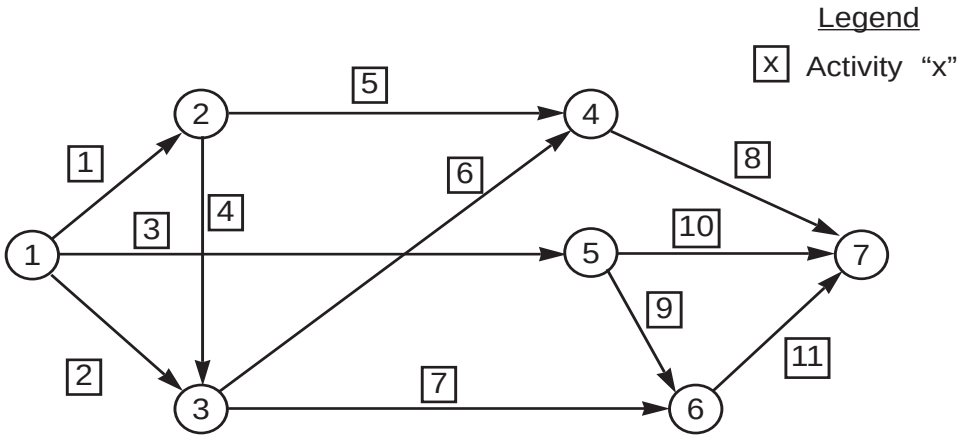

Figure 2.7: DiAN, Project 3

the results obtained via the *DiAN* model and the standard *PERT* approaches or its variants. Indeed, if there is no significant difference in the results then there is no need for the new paradigm, or at least there is diminishing need for it; the old would serve as well. And the second is the validity of the new paradigm as a "better model" of real life projects. Again, if it is of no greater help to the practitioner it may as well be deleted from the roster of tools available for project planning and control.

To address the first concern and to demonstrate the differences between the results obtained under the *DiAN* are substantially different from those obtained under the classical *PERT* paradigm, two radically different distributions for the activities were assumed. The first is Poisson with parameter λ equal to the mean of the presumed activity duration. This immediately resulted in a variance of the activity duration equal to the maximum variance under the *DiAN* model (recall that we have assumed in first two examples that $\sigma = 1$ in the diffusion equation, which results in the variance increasing equally with time). The second is Uniform over the (truncated) duration of the activity under the *DiAN* model.

The results of this analysis are shown in Figs. 2.5, 2.8 and 2.9. The plot of the *DiAN* cumulative distribution function (*cdf*) for all three examples lies significantly to the left of the *cdf* plots of other two distributions. Thus the expected value of the completion time of the project under *DiAN* is approximately half to one third of that for uniform distribution. As a specific example, the probabilities that project 2 shall complete in 30 days are approximately 0.05 for *Uniform*, 0.90 for *DiAN* and 1.0 for *Poisson*.

It is evident that the *DiAN* model gives *pdf*'s (and consequently, all the statistics that are based on them) that are quite different from those secured by standard *PERT* analysis.

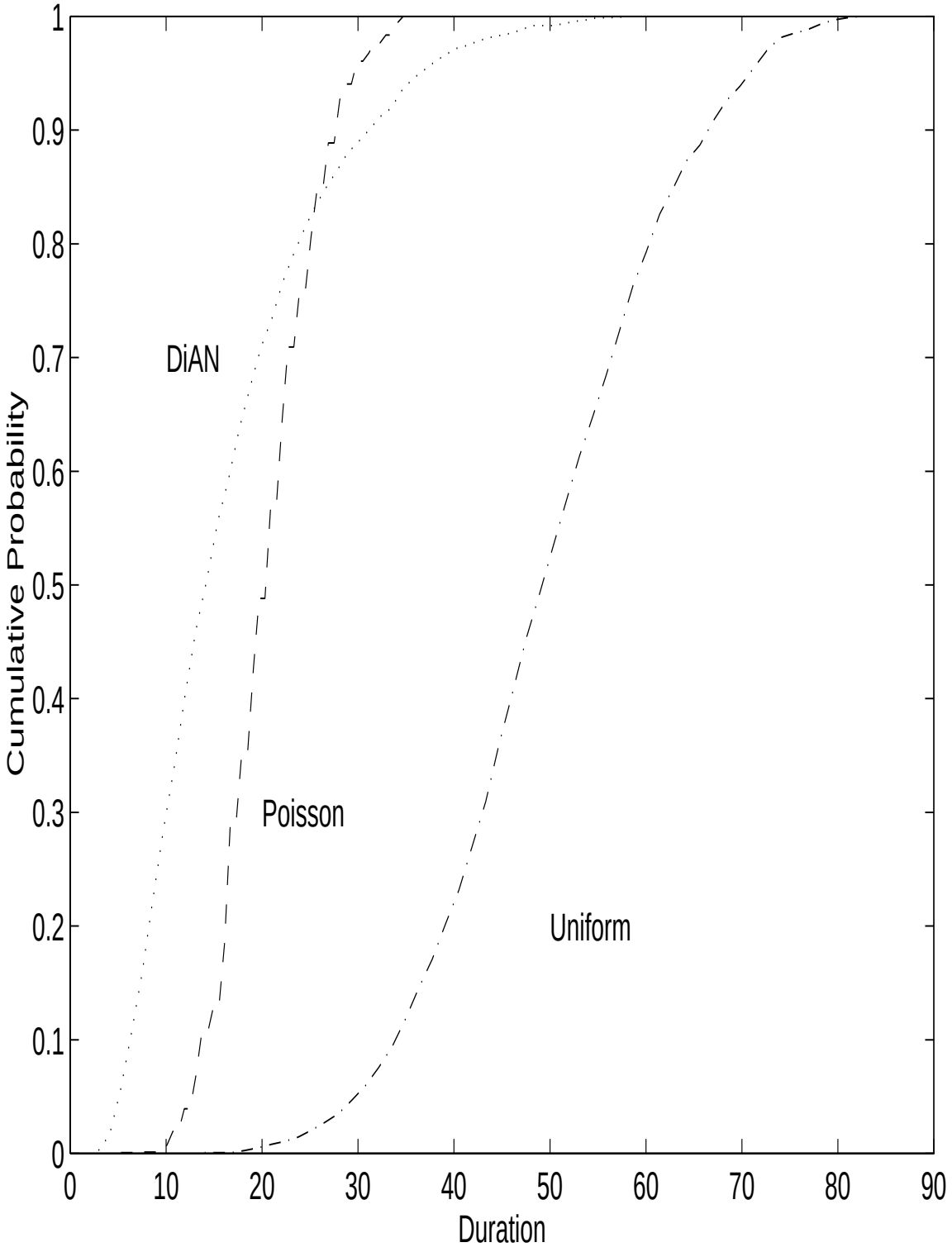

Figure 2.8: Results of Monte Carlo Sampling, Project 2

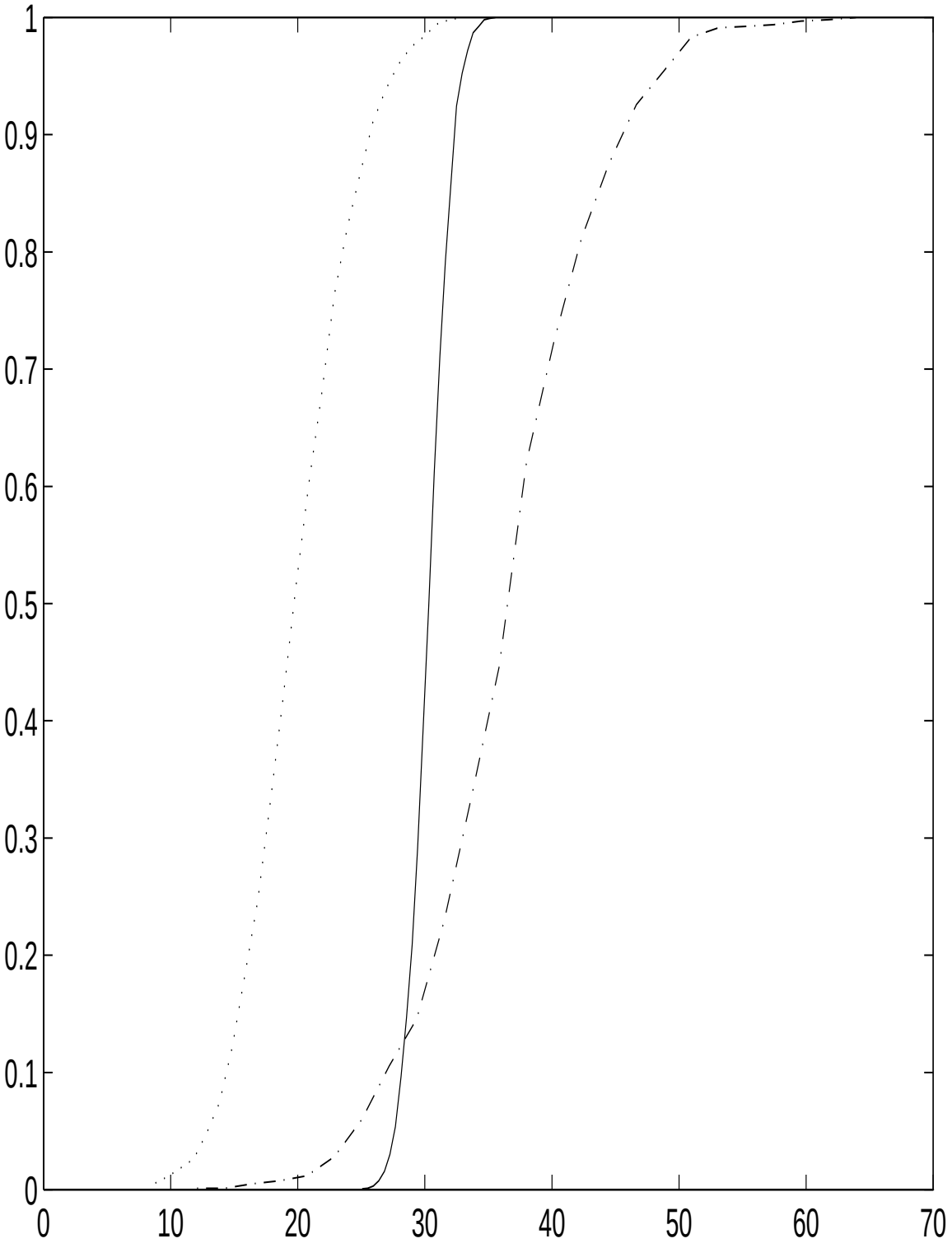

Figure 2.9: Results of Monte Carlo Sampling, Project 3

2.3 Conclusion

The *DiAN* paradigm we are proposing demands different parameters in its specification than classical *PERT* model or its variants. Here, the analyst must specify the best estimate for mean of the activity durations as well as changes in variance over time. Once these are specified, the *pdf* of the activity duration is derived. On the basis of such *pdf*'s one can then proceed to estimate the various parameters of interest.

We did not have sufficient field data available to study the validity of the *DiAN* model at this time. Hopefully, future studies may shed more light on applicability of this proposed paradigm under real time data. However, based on *a priori* reasoning we are convinced that the classical paradigm of fixed-variance distributions is unable to reflect the inherent uncertainty in the activity duration. Also the *DiAN* model, as it may progressively refined, satisfy this gap in the modeling ability.

Chapter 3

Controlling the Activity

The literature in the field of project management has largely ignored the issues related to the dynamic control of a project. A study of the practice of project management would reveal that it is rare that a project proceeds as planned from the outset. In other words, it is rare that an activity executes in absence of any randomness. Thus it can be expected that one would experience deviation from the intended course. In this chapter we would like to address this issue of identification of an activity that has deviated substantially from its intended course, and present ways to cope with such a scenario within the context of *DiAN*.

In the framework of *DiAN* the progress of an activity is measured in terms of its remaining work content (*rwc*) as it changes with time. To recall, the completion of an activity is associated with *rwc=0*. In the absence of any random variation, the progression of the activity (the *rwc*) should proceed according to the instantaneous mean.

The deviation[1] that is positive in that it accelerates the completion of an activity does not concern us. It is the negative deviation that increases the *rwc* of an activity that could be detrimental to the timely completion of an activity, and hence the project. A natural measurement for excessive negative deviation from the intended path would be if *rwc* reaches a predefined state at any time. We identify such a state

[1] the difference between the planned state and the actual state of the activity

as a control state or a flag. For example, a flag is placed at $rwc = 1.2$ if the state of the activity under consideration reaches this upper flag, at which time the activity is considered to be "out-of-control."

A simulation approach is undertaken to further study this behavior in the planning and execution of an activity. Various strategies were investigated for their effectiveness in controlling an out-of-control activity. Finally the cost implication of each strategy is investigated again using simulation.

The research conducted during the course of this investigation concerns the control of an activity, not the whole project. We recognize that at any given instance of the project execution there could be more than one activity that is active. Our research is focused on only one activity and leaves the investigation of the total project control for future research (see Chapter 6).

3.1 Activity Control Strategies

This investigation focuses on the selection of various activity control strategies at either the planning level (before the start of the activity) or once the activity has been detected as out-of-control. Various strategies for controlling an activity were investigated and their relative pros and cons are discussed in the subsequent sections. The selection of these strategies was based on the prevalent industry practices in this area.

3.1.1 Do Nothing

A do nothing strategy ignores any warning signs of the activity under consideration being out-of-control. In essence we do not place a control flag. The activity starts at $rwc = 1.0$ and is absorbed at $rwc = 0.0$ without any interference from management.

It is obvious that during the execution of the activity it may reach or cross the control state. This strategy is none-the-less one of the more prevalent strategies in practice due to many reasons. First, due to lack of appropriate monitoring systems.

And second, because of the inherent inertia of the management to change the resources allocated to an activity while it is being executed. From our point of view, it serves as a baseline strategy to compare the effectiveness of other 'proactive' strategies.

3.1.2 Apply Additional Resources

Under this strategy, the manager places additional resources to bring the activity to its intended termination time. The manager specifies the control state at which an activity is deemed to have deviated significantly from its intended course of execution[2]. We assume that the instantaneous variance of an activity remains constant by adding more resources to an activity. The only change due to the addition of resources is in the instantaneous mean of the activity (see figure 3.1).

The amount of resources added depends upon the original instantaneous mean ($\mu < 0$), the "crash rate" ($r < 0$), and the time (t) at which the activity reaches the control state (α). The *crash rate*, refers to the maximum allowable change in *rwc* per unit time. Let T represent the target completion time of the activity in absence of any randomness, then the new instantaneous mean (ν) is given as follows:

$$\nu = \left\{ \begin{array}{ll} \max\left[r, -\frac{\alpha}{T-t}\right] & \text{if } t < T \\ r & \text{otherwise} \end{array} \right\}$$

The starting state of the activity is the control state α.

The goal of this strategy is to add enough resources such that the target completion time of the activity does not change going forward. This goal however, may not be realizable if there is a certain maximum rate (due to other technical or policy reasons) that the activity can not exceed. In other words, the introduction of crash time places an upper bound on the additional resources that could be added.

[2] We have assumed a control state at $rwc \geq 1.0$ for illustrative purposes

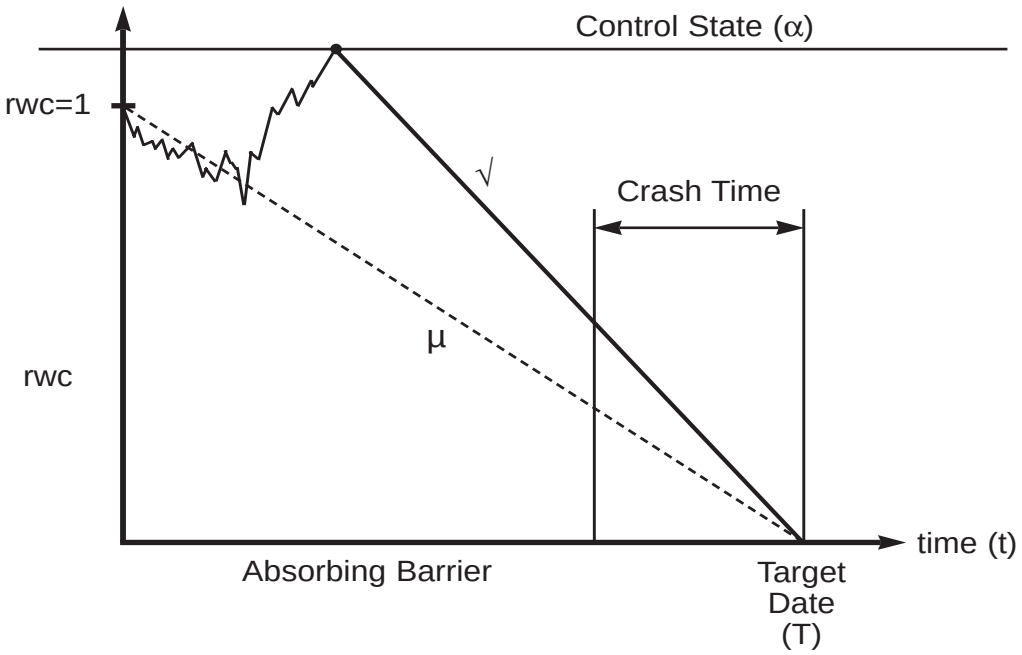

Figure 3.1: Strategy: add more resources

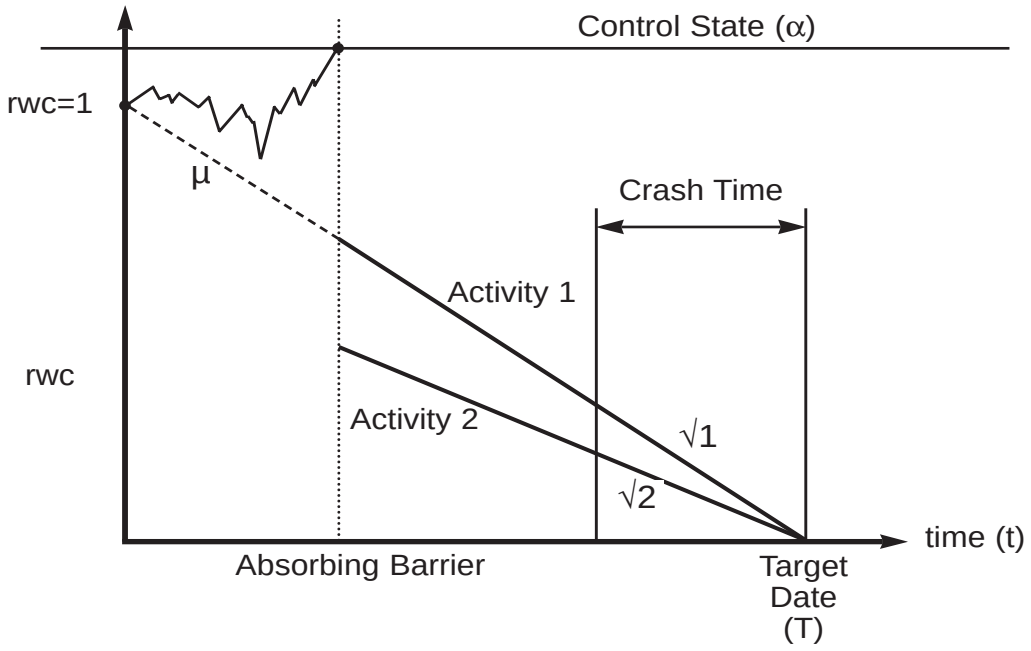

Figure 3.2: Strategy: split the activity

3.1.3 Split the Activity

A popular strategy in service industry is to split the out-of-control activity into two separate distinct activities (see Figure 3.2). The original activity is considered to be complete when the two new activities (after split) are complete. We assume that the two split activities are independent of each other in all executional aspects. We further assume that the instantaneous variances of the split activities are the same as that of the original activity. Let the instantaneous means of the two activities be ν_1 and ν_2. Then following the notation of previous section,

$$\nu_1 = \mu,$$

$$\nu_2 = \left\{ \begin{array}{ll} \max\left[r, -\frac{\alpha+\mu t}{T-t}\right] & \text{if } t < T \\ r & \text{otherwise.} \end{array} \right\}$$

The starting states (s_1 and s_2) of these two split activities is

$$s_1 = \left\{ \begin{array}{ll} -\mu t & \text{if } t < T \\ 0 & \text{otherwise.} \end{array} \right\}$$

$$s_2 = \left\{ \begin{array}{ll} \alpha + \mu t & \text{if } t < T \\ \alpha & \text{otherwise} \end{array} \right\}$$

As evident from the above discussion, this strategy shifts enough work from the original activity to new activity in order to get back on course. At the same time enough resources are allocated to the new activity so that it is completed on the original target time.

The strategy is not limited to the selected splitting scheme mentioned here. Alternate splitting schemes can be used to either suit the problem at hand or potentially optimal scheme can also be developed. However, the methodology developed here would be equally applicable to the new splitting scheme.

3.1.4 Re-specify the Activity

Under the re-specification strategy the scope of the activity is reduced to meet the original expected completion time. In essence the activity does not perform the total task as originally specified. This strategy is prevalent in software industry where the scope of a project is often reduced to meet the original completion date.

The instantaneous mean and variance of the activity remain the same. The state (s), however, is changed to $\alpha + \mu t$.

3.2 Simulation of Strategies

The strategies mentioned in the previous section do not permit a close form solution for completion time estimates. We opt to simulate various strategies to further investigate their impact on the completion time and cost estimates of an activity.

Following the notation of Chapter 2, let x represent the rwc of an activity. Then the progress of an activity can be represented as

$$\begin{aligned} dx &= a(x,t)\,dt + b(x,t)\,dz \\ dz &= \epsilon_t\sqrt{dt} \\ \epsilon_t &= N(0,1) \end{aligned}$$

In our case we have taken $a(x,t)$ and $b(x,t)$ to be constant. The instantaneous mean $a(x,t) < 0$ is represented as μ. The typical value for this simulation is $\mu = -0.1$ resulting in expected completion time ($E[T]$) of an activity to be 10. The instantaneous variance is calculated using the procedure outlined in section 2.1.1.

Knowing the instantaneous mean and the variance the simulation equation can now be written as

$$x_{t+\Delta t} = x_t + \mu\,\Delta t + \sigma\sqrt{\Delta t}\,N(0,1)$$

A crucial parameter in determining the accuracy in the simulation is the time increment Δt. The smaller the Δt the longer a simulation takes. Appendix A discusses this issue in more detail.

The accuracy of the simulations is verified by comparing the mean and the variance of the completion time of the "Do-Nothing" strategy with that of analytical expression. The mean $E[T]$ and the variance $var[T]$ of the completion time of an activity under this strategy are given as

$$E[T] = -\frac{\alpha}{\mu}$$

$$var[T] = -\frac{\alpha\sigma^2}{\mu^3}$$

We conducted 100 simulation runs with $\mu = -0.1, \alpha = 1.1$, and the allowable spread $\psi = 0.5, 1.0$ at $t = 5$ (i.e., at 50% of expected duration). The results of the

Table 3.1: Simulation Validation

	Analytical $E[T]$	Analytical $var[T]$	Simulated $E[T]$	Simulated $var[T]$
$\psi = 0.5$	10	1.388	10.2030	1.3727
$\psi = 1.0$	10	5.8978	10.0021	5.9424

simulation are given in table 3.1. It is evident from the results in the table that the assumption of $\Delta t = 0.1$ gives results within reasonable accuracy.

The simulation was also used to determine other metrics of interest such as the average number of times an activity reaches the control state.

3.3 Comparative Study of Strategies

The four strategies were simulated for different scenarios to study their impact on various measures of interest, mainly, the cost and activity completion time estimates. The scenarios simulated are as follows:

- Instantaneous mean $\mu = -0.1$
- Incremental simulation time $\Delta t = 0.1$
- Allowable spread $\psi = 0.25, 0.5$ and 1.0 at $t = 5$ (i.e., 50% of expected duration)
- Control state $\alpha = 1.01, 1.05, 1.1, 1.25$ and 1.5
- Starting state $s = 1.0$
- Crash time $t_c = 2, 5$
- Crash rate $r = -0.5, -0.2$
- Operating cost $C_o = 1$
- Setup cost $C_s = 5$ and 10
- Reset cost $C_r = 2$ and 5
- Penalty cost of delay $C_d = 2, 5$ and 10
- Number of runs $N = 100$

3.3.1 Impact on Completion Time (T)

One of the most important measures of a strategy is its impact on the completion time of an activity. Knowing how a strategy impacts the completion time of a project the management could take proactive measures by preparing for the selected strategy. We simulated these strategies with starting state $s = 1.0$ and starting time zero. The results of the analysis are presented in Appendix B.

For a large number of cases the completion time was similar to that of strategy # 1 (do nothing). The $\psi = 0.25$ case, in particular, gives the same results for all strategies. This is not surprising as with small values of allowable spread ψ, hence instantaneous variance, the probability of reaching the control state is near zero. As a result none of the activities were ever activated.

The cases where the control state was reached and various activities were activated present the other aspect of this analysis. This scenario occurred when the control state was close to 1.0 and the allowable spread was high (such as $\psi = 0.5$ or 1.0). As expected the higher the allowable spread and lower the control state the higher is the probability of reaching the control state hence the activation of strategies.

The splitting strategy was the worst performer in almost all cases. The mean completion time was roughly 20% higher for $\psi = 0.5$ case and 20% to 60% higher for $\psi = 1.0$ case when compared to the do nothing strategy. The impact on the standard deviation was even higher, roughly 6 to 11 times for $\psi = 0.5$ case and 11 to 16 times for $\psi = 1.0$ case. In other words the strategy of splitting the activity would not be a recommended alternative as a proactive measure as not only it has a negative impact on the completion time but the uncertainty of completion time also increases significantly.

In comparison, the other three strategies (do nothing, add resources and re-specify the activity) performed essentially in similar manner for both mean and standard deviation of the completion time. The re-specification strategy performed slightly better when the probability of reaching the control state is high. This is to be expected as under the re-specification strategy, if activated, only part of the activity is completed.

The crash time has a significant impact on the variability and mean of the completion time for the two strategies of adding more resources and splitting. A small value of crash time is equivalent of a high crash rate. Which in turn implies a high instantaneous mean. A high value of the mean could in turn significantly reduce the uncertainty. Thus higher crash rates induce less variability in the completion time estimates if these two strategies are activated.

Based on the aforementioned observation, we can draw the following conclusions for designing a proactive strategy if the completion time was the main concern.

1. use the do-nothing strategy whenever the allowable spread keeps *rwc* well below the control state.

2. For cases when the allowable spread brings the *rwc* closer to the control state, use re-specification strategy if it is a feasible alternative, else use the "add resources" strategy.

3. The splitting strategy should not be used as it performs worst among all strategies considered here.

3.3.2 Cost Implications

Another criterion that is frequently employed in evaluating the strategies is the cost of using such actions. This criterion is typically used during the execution phase of the activity as a reactive measure to an out-of-control measure.

Among the strategies considered here, one should note that under the re-specification strategy, we do not consider the indirect cost of not meeting the original work content of the activity. This indirect cost may come from loss of goodwill and market share, among others. This could also have an impact on activities succeeding the current activity, which, in some instances, may paralyze the project. The very nature of these indirect costs makes it difficult to analyze them formally within the context of this investigation. We would recommend that management consider these costs on a project by project basis along with the guidelines provided here.

The cost of completing the strategies after they have reached the control state is calculated as the sum of costs at any Δt until $rwc = 0$. A penalty cost is applied to the original cost if the activity completion time exceeds the expected completion time.

Strategy 1

$$\Delta C_1 = \left\{ \begin{array}{ll} C_o\,\Delta t & \text{if } t < T \\ (C_o + C_d)\,\Delta t & \text{otherwise} \end{array} \right\}$$

Under this strategy the incremental cost is the same as the operating cost as no changes in the working conditions of this activity have been made.

Strategy 2

$$\Delta C_2 = \left\{ \begin{array}{ll} \frac{\nu}{\mu} C_o\,\Delta t + C_r & \text{if } t < T \\ \left(\frac{\nu}{\mu} C_o + C_d\right)\,\Delta t + C_r & \text{otherwise} \end{array} \right\}$$

The incremental cost, in the first equation, represents the resetting cost of the activity (the second term) and the operating cost (the first term) which is proportional to the change in the instantaneous mean.

Strategy 3

$$\Delta C_3 = \left\{ \begin{array}{ll} \left(\frac{\nu_2}{\mu} + 1\right) C_o\,\Delta t + C_r + C_s & \text{if } t < T \\ \left(\left(\frac{\nu_2}{\mu} + 1\right) C_o + C_d\right)\,\Delta t + C_r + C_s & \text{otherwise} \end{array} \right\}$$

The first term in the incremental cost, in the first equation, reflects the cost of running two activities. The split activity's operating cost is changed in ratio of means. The second and the third terms represent the resetting cost of original activity and the setup cost of the new activity.

Strategy 4

$$\Delta C_4 = \left\{ \begin{array}{ll} C_o\,\Delta t + C_r & \text{if } t < T \\ (C_o + C_d)\,\Delta t + C_r & \text{otherwise} \end{array} \right\}$$

The incremental cost, in the first equation, includes a term for resetting the cost along with its original operating cost.

It should be noted that for strategy 3, the cost is accumulated until both activities are completed. Another point should be noted that for this simulation we assumed that the activities do not reach the control state again.

The simulation results for evaluating the cost of using various strategies is given in Appendix C.

The following can be deduced from the simulation results.

- The strategy 4, re-specification, is always the lowest cost strategy. This, however, can be misleading. By reducing the original scope of the activity there could be tangible but not easily measurable costs that are not accounted for here. Care must be taken in using this strategy.
- An interesting observation to note is that the expected cost of completion was higher even if the process does not reach the control state. This was in comparison to what the cost would be if the calculations were made in absence of any randomness. This has important managerial implications as the typical cost calculations are done based on average values. A 20% to 30% increase in cost could be substantial even if none of the control measures were activated.
- As expected, strategy 3 (splitting) was the worst performer across all cases. Moreover, the standard deviation of the cost was an order of magnitude higher than other strategies. This strategy is, therefore, not recommended as a reactive measure.
- Strategy 2 (add resources) performed as good as or better than the do-nothing strategy in all cases. Taking a deeper look into various cost elements that impact the behavior of this strategy one can draw the following conclusions.

    1. As the control state is lowered, representing stricter control, the probability of reaching the control state goes up. This implies that the resetting cost would have a higher impact on the overall cost of the activity.

2. Cost of delay and the operating cost would counter-balance each other. In general the lower the crash rate (higher crash time) the lower the cost would be, all else being equal.

3. The location of the control state plays an important role. For lower control state the cost of resetting has more impact compared to the cost of delay. Similarly, if the control state is higher than the cost of delay plays a more predominant role.

4. The cost increases in relation to the allowable spread (ψ).

In general we recommend the use of the do-nothing strategy if the resetting cost is high compared to the delaying cost of an activity. The strategy of adding more resources is an overall favorite strategy. The care must be taken as how much resources to add which is determined by the location of control state and crash time. Addition of resources must be balanced against the cost of delaying the activity.

In practical terms if the ramp-up cost is high - as in case of software development - and if a small delay (10 to 20%) in completion would not significantly hurt the project, then use the do-nothing strategy otherwise addition of more resources would be a better choice of strategy.

3.4 Conclusion

In this chapter we present an analysis of a rather neglected issue in the area of activity networks. Controlling an activity, hence a project, is an important executional aspect. It is not uncommon to see budget and time overruns due to poor emphasis on control aspects of a project.

The simulation of these strategies does not incorporate some of the subtle issues as they would appear in the practical application of these strategies. The most notable of these are the time lag between implementation of a strategy and its full effectiveness. We have taken this time lag to be zero. Another factor would be the use of multiple

strategies at various point in time. In our case, the practitioner chooses one strategy for the entire discourse of the activity. The lack of these consideration, however, does not limit the usefullness of the work in practice. These issues are mentioned here to further stimulate the research in this area.

The research in this area is rather sparse. There are no systematic ways in which various control strategies are designed or implemented. The strategies that are investigated here are representative of real world scenario. They are, however, not exhaustive in any sense of the world.

Chapter 4

Generation of Random Networks

As discussed in Chapter 1, a need to generate testsets in various fields has been recognized by both researchers and practitioners to evaluate algorithms. The field of activity networks is not immune to such needs. The past work on this topic is rather limited and can be attributed to Patterson [46], Demeulemeester et. al. [13], Kolisch, Sprecher & Drexl (KSD) [37] and the latest offering by Ferreira et. al. [25]. The details of these are discussed in subsequent sections of this chapter.

Our offering of a methodology and the associated software to generate activity networks is motivated by the fact that none of these testsets took cognizance of the more recent result of Bein, Kamburowski & Stallmann (BKS) [4]. Briefly, BKS establishes the "optimal node reduction," which we refer to as "node reduction index" (*n.r.i.*), as a measure of non-conformity of a network to the series/parallel [57] topology. We assert that the *n.r.i.* is an important parameter that adds one more dimension to the measure of a network's "difficulty." This has been confirmed by the work of Elmaghraby [22] and De Reyck & Herroelen [12]. We expect it to receive additional confirmation as researchers achieve more familiarity with the result of BKS.

The software is intended to serve as a research tool to investigate the performance of the proposed approaches or to test new software. We stop short of labeling the resultant network as "randomly generated" because it is not guaranteed that the topology of the network is a random selection from among the population of all

possible networks with the specified number of arcs, nodes, and *n.r.i.*, despite the fact that during the construction of the sample network several decisions are made based on random selection from among the possible alternatives. The same comment holds true for the other network generators mentioned above.

Apart from the independent value of the procedure developed in this chapter, this methodology was used to generate the example networks in Chapter 2. Using *DAGEN* we were able to generate the networks of varying number of nodes, arcs and node reduction index.

Discussion in this chapter can aptly be divided into two main sections. In the first section a comprehensive analysis of the existing measures of network complexity is presented. This section also covers an in depth review of the issues related to the generation of networks by the KSD procedure. The second section details the *DAGEN* network generation procedure.

4.1 Comments on Network Generators

KSD's network generator produces AoN network, denoted by *(V,A)*, where V is the set of activities (nodes), $|V| = n$, and A is the set of arcs representing the precedence relations, which number lies between lower and upper bounds, as described below. It is well known that the translation of this mode of representation into the more familiar (to managers) AoA mode is rather tricky, requiring, in most cases, the addition of "dummy arcs" to maintain the integrity of the precedence relations specified by the AoN. As a consequence, the resulting AoA is *not* unique, which may add to the confusion of the network generation exercise and the interpretation of subsequent results.

The KSD generator scrupulously avoids the creation of *redundant* arcs since they contribute nothing to the logic of the network. The definition of redundancy is the usual one: an arc (i, j) is *redundant* if it is deducible, by transitivity, from existing precedence relations; *i.e.,* if it is an element of the transitive closure of

$((V, A)\backslash\{(i, j)\})$. Denote the minimal and maximal number of non-redundant arcs in the network by A^{min} and A^{max}; respectively (see section 4.2.1).

The KSD network generator requires the specification of a large number of parameter values. The complete roster of parameters to be specified by the analyst is given in Appendix D, and we estimate that there are $17 + 9\ (|R| + |N| + |D|)$ parameters, where R is the set of renewable resources, N is the set of non-renewable resources, and D is the set of doubly-constrained resources (*i.e.*, resources that are of limited availability in each period as well as in total value over the life of the project). This is rather excessive. For instance, if the resources are given as follows: $|R| = 2$, $|N| = 1$, and $|D| = 1$, the analyst must specify 53 parameters! One can hardly conceive of an experimental design that would estimate even the main effects of all these parameters. If each is evaluated at only two levels, a full factorial experimental design would require 2^{53} cells! The details of the network generator are presented in Appendix D.

4.2 On Measures of Complexity

4.2.1 The KSD Measure of "Network Complexity" C

KSD define the *network complexity*[1] C as "the average number of non-redundant arcs per node (including the super-source and super-sink[2])." Presumably, C represents a measure of the "intensity" of the precedence relations among the activities. We have,

$$A^{min} = n - 1$$

and for $n \geq 6$[3]

[1] We interpret "network complexity" to be synonymous with "computing difficulty of analysis."

[2] The "super-source" is a dummy activity that immediately precedes all start activities, and the "super-sink" is a dummy activity that immediately succeeds all finish activities.

[3] In our independent calculations, we derived $A^{max} = 2(n-2)$ for $n \geq 3$ which produces smaller values for A^{max} than the formula of KSD.

$$A^{max} = \begin{cases} n-2+(\frac{n-2}{2})^2 & \text{if n is even} \\ \\ n-2+(\frac{n-1}{2})(\frac{n-3}{2}) & \text{if n is odd} \end{cases}$$

Consequently, one can easily deduce that their network complexity C is bounded by,

$$1-\frac{1}{n} \leq C \leq \begin{cases} 1-\frac{2}{n}+\frac{1}{4n}(n-2)^2 & \text{if n is even} \\ \\ 1-\frac{2}{n}+\frac{1}{4n}(n-1)(n-3) & \text{if n is odd} \end{cases} \tag{4.1}$$

To gain more insight into these expressions, consider a project with $n = 500$ activities (a modest size project); then C is approximately bounded by,

$$1 \leq C \leq 125$$

It is not immediately obvious what a number C in this range represents, as far as the "difficulty in analysis" is concerned. On the other hand, suppose one normalizes this parameter by dividing throughout by A^{max}; the lower bound then becomes (asymptotically) 0 and the upper bound $=1$ always. Call the new parameter C' to distinguish it from C. A choice of $C' \in (0,1]$ would then be a normalized representation of the choice of the "intensity" of the precedence relations among the activities. One would imagine that complexity would increase as $C' \to 1$. Unfortunately, this is not necessarily true, which casts doubt on the validity of C as a measure of network "complexity" in the first place. To illustrate, consider the well known "interdictive graph" of 5 activities shown in Fig. 4.1, augmented by the super-source and the super-sink, so that we have, in all, 7 activities.

Here, $A^{min} = 6$[4] and $A^{max} = 11$[5]. Choosing $C' = \frac{6}{11} \approx 0.545$ results in the chain of Fig. 4.2; and choosing $C' = 1$ results in the network shown in Fig. 4.3.

[4] when the activities are in series, see Fig. 4.2.

[5] when the activities are in parallel, see Fig. 4.3

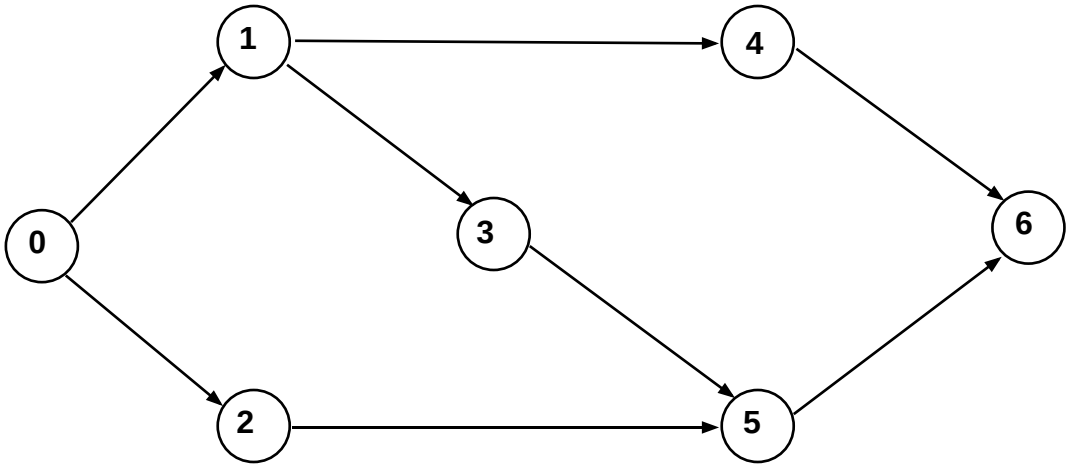

Figure 4.1: Interdictive Graph (AoN)

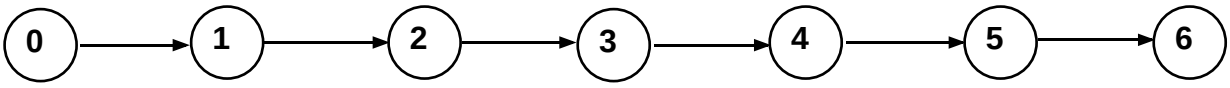

Figure 4.2: Series

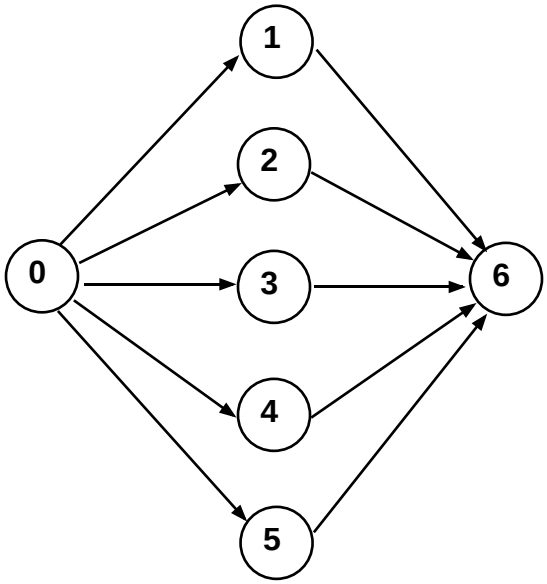

Figure 4.3: Series-Parallel

Interestingly enough, the network in Fig. 4.3 is *series-parallel* which, for many applications[6], is *easier* to solve than the *IG* of Fig.4.1 which is of complexity $\frac{8}{11} \approx 0.727$!! It is relatively easy to construct examples of such anomalous behavior; Figs. 4.4 and 4.5 exhibit two networks of identical C' $(= \frac{8}{11} = 0.727)$ but with radically different degrees of difficulty! Almost all the criticisms levied by KSD against the RS measure of Cooper [10] (section 4.2 of the KSD paper) apply to their measure of complexity C. The RS measure is ratio of available capacity of a resource to the sum of resource requirements by all activities. Naturally, the low values of RS are associated with the infeasibility.

We concede that it is difficult to measure network complexity by one parameter, which explains why the issue has occupied researchers for such long time (see Elmaghraby & Herroelen[24] for a more detailed discussion of this issue). We submit that it takes *several* parameters to capture the full import of "network complexity;" one of them may be the parameter C' of KSD. We discuss the others next.

4.2.2 Thesen's NR and its Derivatives

In 1977 Thesen[56] tried to capture the essence of the "difficulty" in analyzing ANs, especially when sequencing activities under limited resources, the so-called "resource constrained project scheduling problem (RCPSP)," by proposing a *measure of network restrictiveness* (*NR*). The idea stems from the observation that if the activities are unrelated by any precedence relations there shall be $n!$ possible sequences, while a chain of activities permits only one sequence. Let π denote the number of possible sequences; then $1 \leq \pi \leq n!$. Thesen defined the restrictiveness of an activity network (V,A), still within the AoN mode of representation, as

$$NR = 1 - \frac{\log \pi}{\log n!}$$

It is easy to see that $NR = 1$ when $\pi = 1$, implying maximal restrictiveness, and that $NR = 0$ when $\pi = n!$, implying minimal restrictiveness. It is argued that the

[6] For instance, in the optimal allocation of resources; see Elmaghraby [22]

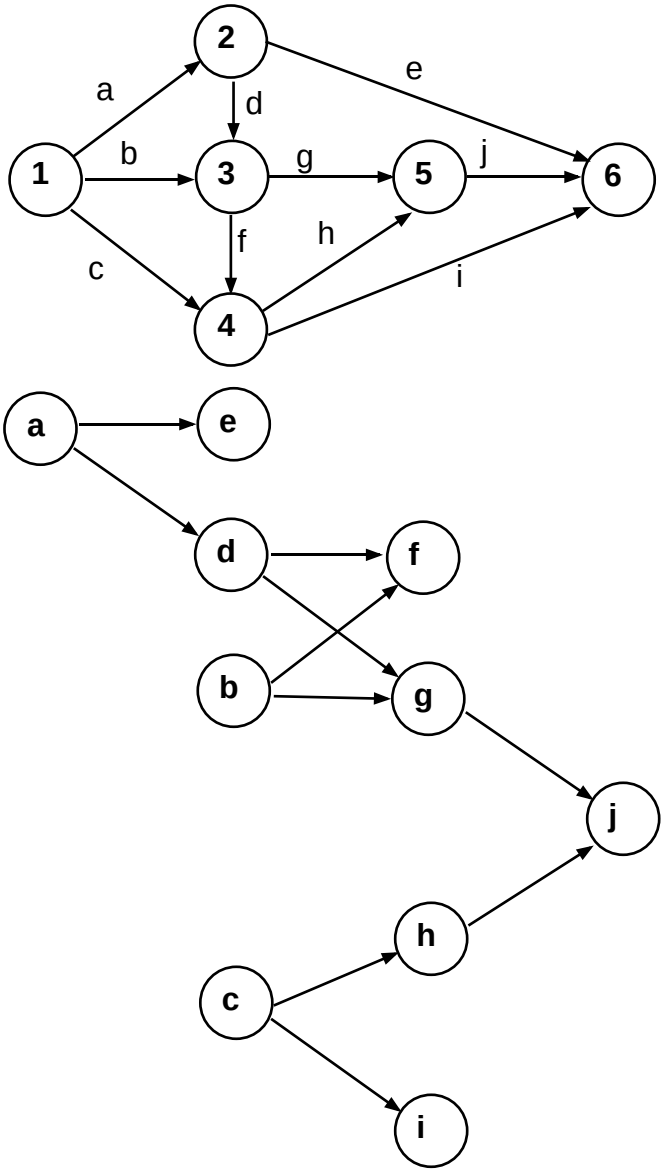

Figure 4.4: Example 1 of Anomaly: AoA (above) and AoN (below)

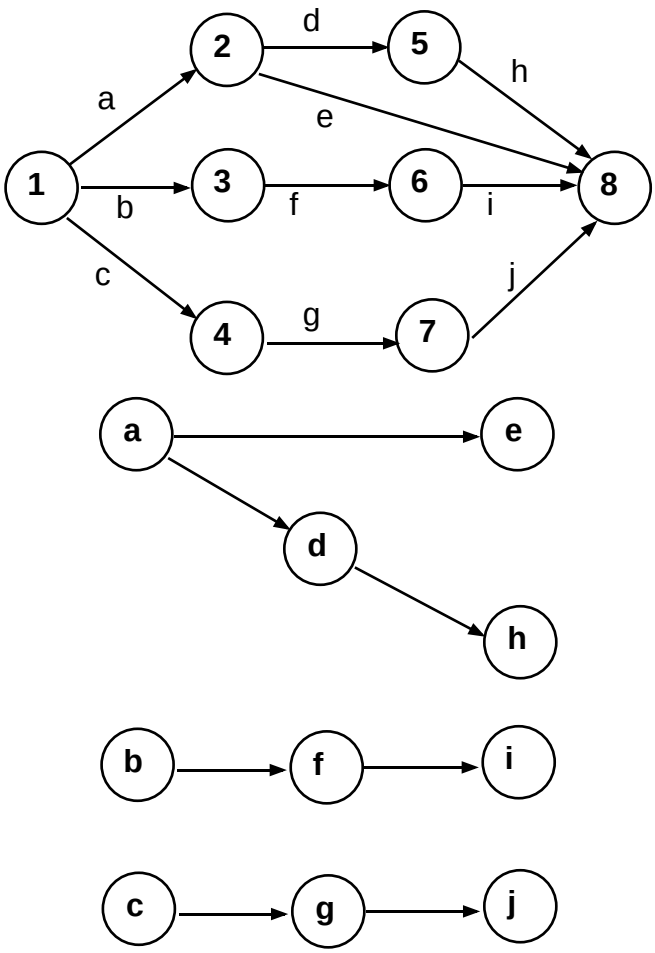

Figure 4.5: Example 2 of Anomaly: AoA (above) and AoN (below)

difficulty in analysis is inversely proportional to NR. The only remaining issue then is to *measure* the functional relation between the effort it takes to perform the required analysis (*e.g.*, schedule the activities to minimize the completion time of the project) and the measure NR.

This measure suffers from at least two major drawbacks. The first is the extreme difficulty in determining π except in the two extreme cases cited above. This difficulty was recognized by Thesen and prompted him to test several approximations to π. One of such measures is RT

$$RT = 1 - \frac{2\sum_{i,j \in V} r_{ij} - 6(n+1)}{n(n-1)};\ r_{ij} = \begin{cases} 1 & \text{if there is a path from } i \text{ to } j \\ 0 & \text{otherwise} \end{cases}$$

The second is its lack of correspondence to intuition. For instance, halving the number of sequences (*i.e.*, $\pi = n!/2$) would result in $NR = 1 - \frac{\log(n!/2)}{\log n!} = \frac{\log 2}{\log n!}$, which is meaningless! (If $n = 500$, and the precedence relations are such that there are $500!/2$ sequences, the expression would result in $NR \approx 0$.) For these, and other, reasons Elmaghraby & Herroelen [24] discounted this parameter as an appropriate measure of network complexity.

Interestingly enough, an "approximate" version of this measure was resurrected in the recent work of Schwindt[52], who asserted its appropriateness as a measure of network complexity in RCPSP. He also provided empirical evidence to substantiate his assertion. Additional evidence of the relevance of this measure is also reported by De Reyck[7].

We submit that the desire to take the "density of the precedence relations" into account, which is a natural and legitimate concern, can be implemented by much simpler expressions such as,

$$\rho = \frac{2|\bar{A}|}{n(n-1)} \tag{4.2}$$

where $\bar{A}$ is the transitive closure of the precedence relations A, and $|A|$ is the cardinality of A. The logic behind this expression is as follows: if A were a chain, then

[7] *op. cit.*

$|\bar{A}| = n(n-1)/2$ since every activity i would have an arc to each succeeding activity $j > i$, $\Rightarrow \rho = 1$, implying maximal restrictiveness[8]. At the other extreme, if $A = \emptyset$, the empty set, then $\bar{A} = \emptyset$ also; whence $|\bar{A}| = 0 \Rightarrow \rho = 0$, implying minimal restrictiveness. It is easy to verify that our measure ρ is identical to the *order strength* (OS) measure given by

$$OS = \frac{r}{\frac{n(n-1)}{2}} = \frac{2r}{n(n-1)};\ r_{ij} = \begin{cases} 1 & \text{if there is a path from } i \text{ to } j \\ 0 & \text{otherwise} \end{cases}$$

used by Mastor [45] for generating and evaluating assembly line balancing problems since the early seventies. Dar-El [11] proposed a measure

$$FR = \frac{\text{Number of zero entries in precedence matrix}}{\text{Total number of matrix entries}}$$

which is essentially the same as *OS*.

De Reyck has already verified the equivalence between *OS* and *RT*. And, as mentioned above, he has verified experimentally the relevance of this factor to the measurement of network complexity as reflected in the time to complete the analysis for the RCPSP.

4.3 The Directed Acyclic Graph Generator (DAGEN)

DAGEN is a procedure to generate activity on arc (AoA) networks of given number of nodes (n), arcs (m) and the *n.r.i.* (η). Optionally, the resource requirements for each activity can be generated within a given range of values. The networks generated can be described as near random as the underlying structure (skeleton) is deterministic upon which the rest of network is built. DAGEN is thus a two step procedure: (i) construction of the skeleton; (ii) completion of the network.

[8] It is assumed, as it is commonly the case, that the activities are numbered topologically so that an arc always leads from a small numbered node to a larger numbered one.

The underlying philosophy is to start with a smallest network of given complexity and add nodes and arcs while maintaining the complexity index of the resulting network the same. At first we conjectured that there would be a library of skeletons from which one can randomly choose a network [2, 3]. Later we demonstrate that a smallest network of given *n.r.i.* can be constructed to replace the library of the networks.

In the following we describe the generation of skeleton and the rest of the network. A brief user's manual for the software follows the generation procedure.

4.3.1 The Skeleton

The skeleton is a network of given complexity η that employs a minimal number of nodes and arcs. It represents the backbone for the rest of the network. The generation of a skeleton proceeds as follows.

Construction of the Skeleton

- For a given node reduction index $\eta \geq 1$, let $n = \eta + 3$. Number the nodes 1 to n.
- The set of arcs are constructed as follows: from node i, $i = 1, 2, \cdots n-2$ there is an arc to nodes $i+1$, $i+2$, and from node $n-1$ there is an arc to node n; see Fig. 4.6.

As can be easily verified, this construction leads to every successive four nodes constituting an Interdictive Graph (IG). Note that if η is the *n.r.i.*, then the number of arcs $m = 2\eta + 3 = 2n - 3$.

Properties of the Skeleton

The following statement asserts that the skeleton *dag* is a minimal graph of the specified complexity.

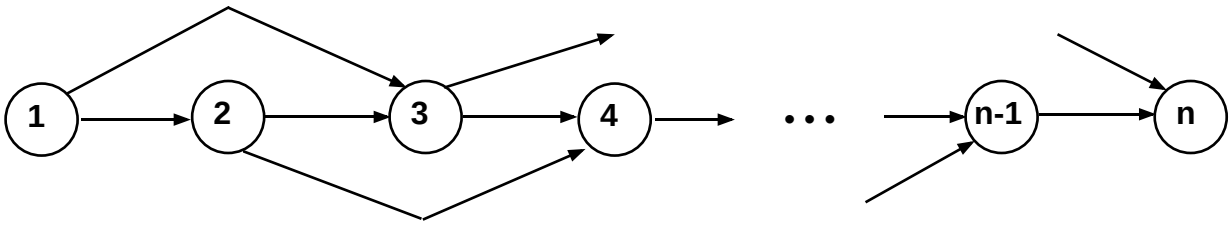

Figure 4.6: The structure of the skeleton graph

Theorem 4.1 *For a given* n.r.i. η *the skeleton, constructed according to the procedure specified above, is the smallest* dag *in both the number of nodes and number of arcs of that* n.r.i..

proof By contradiction. (The proof is illustrated in Fig. 4.7, which is constructed for $\eta = 2$.) It is easy to see the validity of the proposition for $\eta = 1$, since the skeleton is the *IG* which we know is the smallest irreducible *dag* of complexity 1; no economy in either nodes or arcs is possible. Now consider a skeleton *dag* constructed for $\eta > 1$. Any economy in the size of the *dag* must be the consequence of removing some node i or some arc (i, j). But removal of any node i would result in the skeleton graph of complexity $\eta - 1$. The same is true for the removal of any arc (i, j), since then node i shall have only one arc out of it, which would immediately destroy the *IG* structure of nodes i, $i + 1$, $i + 2$, $i + 3$. □

4.3.2 The Generation of the Network

In the second step of the procedure, modules consisting of at most one node and at most three arcs are inserted into the network at hand such that the *n.r.i.* of the resulting network remains unaltered. The modules represent the basic building blocks of the network. A total of five modules are considered for this purpose, as described next:

Series Expansion: If an arc (i, j) already exists between two randomly selected nodes i and j, $i \prec j$, then a node k can be inserted between nodes i and j such

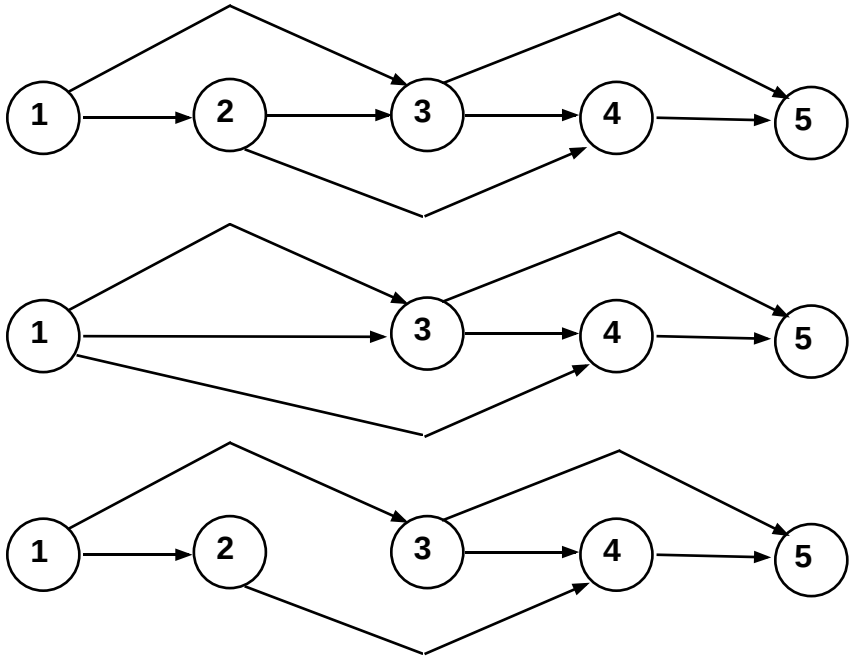

Figure 4.7: Illustration of Theorem 4.1 with $\eta = 2$

that the original arc (i, j) is *replaced* by two arcs (i, k) and (k, j). This results in a net addition of one node and one arc.

Parallel Expansion: If an arc (i, j) exists between two randomly selected nodes i and j, $i \prec j$, then a node k can be inserted by adding two arcs (i, k) and (k, j) respectively. This results in a net addition of one node and two arc (note that arc (i, j) is not deleted here).

Transitive Direct Expansion: If the randomly selected nodes i and j are not connected by a direct arc then an arc may be inserted between nodes i and j. The insertion of such an arc is permissible if $i \prec j$ and the complexity of the resulting network (*i.e.*, after inserting the arc) remains unaltered. This results in a net addition of one arc and zero nodes.

Transitive Series Expansion: If the randomly selected nodes i and j are not connected by a direct arc then a node k and two arcs (i, k) and (k, j) may be

inserted between nodes i and j. The insertion of this module is permissible if $i \prec j$ and the complexity of the resulting network (*i.e.*, after inserting the arc) remains unaltered. This results in a net addition of one node and two arcs.

Transitive Parallel Expansion: If the randomly selected nodes i and j are not connected by a direct arc then a node k and three arcs (i, j), (i, k) and (k, j) may be inserted between nodes i and j. The insertion of this module is permissible if $i \prec j$ and the complexity of the resulting network (i.e., after inserting the arc) remains unaltered. This results in a net addition of one node and three arcs.

4.3.3 Description of DAGEN

The computer program "graph1.f" to automate the procedure of constructing the DAGEN with the specified parameters was developed in FORTRAN using gnu g77 compiler. A program (CINDEX) to compute the *n.r.i.* of a network was also developed to verify the accuracy of the generated network. (Procedure CINDEX may be used independently to determine the *n.r.i.* of any given network.) The source code for the two programs are provided in Appendices E and F.

4.3.4 DAGEN Operation

Running the program can be described in five distinct steps.

- In the first step, user inputs are received in an interactive mode. The user is prompted for the desired *n.r.i.* of the network (range 1 to 97). A range of the possible number of nodes is then provided to the user. The minimum number of nodes in the network is controlled by the *n.r.i.* while the maximum number of nodes and the maximum *n.r.i.* are limited only by practical programming considerations. Next, the user is supplied with an option to select either the *number of arcs* or the *restrictiveness* ρ (see equation 4.2) as the method of generating the network. In either event, the user is provided with a range to choose from. The number of arcs (hence

restrictiveness) in the network are limited by the number of nodes selected earlier in the specifications. User supplied parameters of the network are checked for input correctness. If the user has supplied correct information then the program continues to the next step; otherwise the execution is halted with an error message. Finally, the user is prompted for the costs and resource(s) specifications (see below) from which values shall be randomly drawn to complete the specification of the project. The program is limited to one non-renewable resource and five renewable resources.

- The second step of the program deals with the generation of the skeleton network, as described in section 4.3.1. If the user supplied more nodes and/or arcs than required to construct just the skeleton then the program proceeds to the third step. If the given number of nodes and arcs are exactly sufficient to construct the skeleton network, the program prints the result in the user supplied file and stops.
- The third step of the program deals with the construction of the network proper. If there are nodes to be inserted (at least one) then the program continues with this step; otherwise it goes to the fourth step.

  In this step two nodes are selected randomly. A module (see section 4.3.2) is selected randomly if the selection criteria are satisfied. This process continues until there are no more nodes to be added. The selection of the modules is also restricted such that at the end of this step the "number of remaining nodes" to be inserted is zero while the "number of remaining arcs" are zero or positive. The insertion of transitive modules is restricted to the skeleton.
- The fourth step inserts transitive direct arcs in the network skeleton. This step selects a node i and then randomly checks among all nodes in the skeleton to select a node k such that arc *(i,k)* does not exist. If there are no more arcs which need to be inserted then the program goes to the next step.

- The fifth and final step of the program renumbers all the nodes to be topologically ordered. It randomly assigns values to the cost and resources parameters for all the activities. Since the program is limited to one non-renewable resource and five renewable resources, the quantity required of each, and the fixed as well as the variable costs for its utilization, are drawn from uniform distributions over the specified ranges. (The ranges are requested as input parameters; see Appendix G.) The execution time of the program (exclusive of input and output of data) is determined in this step.

The results are reported in three files: (i) *Dagen.Net*, which contains the detailed output file; (ii) *Dagen.Mat*, which contains the adjacency matrix of the network generated; and (iii) *Dagen.CI*, which contains the results of network reduction index calculations. A facsimile of these outputs is shown in Appendix H for a net with $n.r.i. = 3,\ N = 10,\ A = 20$.

The programs GRAPH1 and CINDEX have been extensively tested for various ranges of inputs. A "formal and complete" testing, however, was not undertaken. Typical execution time for GRAPH1 ranges from a fraction of second for small nets (about $n.r.i. = 3,\ |N| = 10$ nodes, and $|A| = 25$ arcs) to nearly 100 seconds for large nets (about $n.r.i. = 20,\ |N| = 100$ node, and $|A| = 150$ arcs).

Chapter 5

The Convolution of Random Variables

In the mathematical analysis, such as in Chapter 2 of activity networks (*AN's*) one is usually faced with the problem of evaluating the sums and the maxima of independent random variables (*r.v.*'s). The maximum operation is easily performed by multiplying the probability cumulative distribution functions (*cdf)* of the *r.v.*'s involved. The sum operation, however, leads in a natural way to the evaluation of the convolution of the *r.v.*'s, which is a rather demanding operation, and constitutes the main subject matter of this chapter.

The convolution operation on continuous probability density functions (*pdf*'s) is difficult except in very few cases (such as the normal distribution). Therefore, we propose to use two approximation techniques for evaluating the convolution of such density functions. The first is *polynomial approximation* and the second is based on the *discretization* of the continuous density function $f(t)$.

In sections 5.1 we discuss various means of approximating *pdf*'s with polynomials. Section 5.2 focuses on developing an efficient way to compute the convolution operation when the *pdf*'s are represented by polynomials, or piecewise polynomials. The problem of 'discretization' is discussed in the section 5.3 of this chapter. An example of discretization process is presented in section 5.4.

In computing the completion time distribution of a project, a topic of Chapter 2, the maximum and convolution operations are used repeatedly. To elaborate, two

activities in series can be replaced by an activity with a distribution that is sum of two original activities. Similarily, two activities in parallel can be replaced by an activity with distribution as the maximum of two original activities. The number of required convolution and maximum operations increases geometrically with the increase in the size of a project. It thus becomes paramount to develop an efficient scheme for the two operations. The techniques for computing the convolution, developed in this chapter, are used in all the examples of Chapter 2 which was necessary on two grounds. The first was to manage the growth in the data storage requirements for discrete distributions and the second to be able to convolve the density function in equations 2.8 and 2.10 using polynomial and/or discrete approximations.

To the best of our knowledge, the only reference to the computing of the convolution under the assumption of polynomial *pdf*'s is due to Martin [44]. A detailed overview of Martin's method is presented in Chapter 1. In this chapter we present an alternative formulation that generates the cumulative density function (*cdf*), which we believe to be simpler and more intuitive.

5.1 Polynomials Approximation

The polynomial approximation we propose is for *finite* and *bounded* domains of the independent variable t. We recognize that the *pdf* $f(t)$ is often defined for $t \in [0, \infty]$. Typically, $f(t)$ approached zero asymptotically. For modeling purposes of almost all real life activitis we take the range to be $[0, \bar{T}]$, where $\bar{T}$ is the upper bound on the duration of the activity. The value of $\bar{T}$ can be determined to any desired accuracy p by solving the equation: $F(\bar{T}) = p$. Typically, $p = 0.90$ to 0.99. Normalization is achieved by dividing $f(t)$ in the range $[0, \bar{T}]$, by p. Clearly, by truncating the range of t at $\bar{T}$ we are somewhat distorting the "true" density function $f(t)$.[1]

The literature on numerical analysis is replete with polynomial approximation

[1] A problem may arise in such approximation when the polynomial aproximation results in negative values. Care must be taken to eliminate such occurrences and re-normalize the polynomial approximation to yield a *bona fide* density function.

schemes; see Dierckx [15] and Hamming [31] for details. The choice of method depends on the situation under study and the analyst's preference; there is no "one best way." Here, we propose a rather simple approach based on the Vendermonde's determinant [31].

Given N+1 points $(t_i,\ f(t_i));\ i = 0, 1, 2, \ldots, N$ and $0 \leq t_0 < t_1 < t_2 < \ldots < t_N = \bar{T}$, we can fit a polynomial of degree N to the data, referred to as $P(t) = \sum_{k=0}^{N} a_k t^k$. The condition that the polynomial $P(t)$ passes through the given data points dictates that $f(t_i) = P(t_i) \forall i$. It can be demonstrated [31] that the Vandermonde determinant V_{N+1} is well defined and is non-zero; resulting in a unique solution $P(t);\ t_0 \leq t \leq \bar{T}$.

The selection of the N+1 sample points is left to the discretion of the analyst. One popular method is to select the points at uniform intervals. Another equally popular method is to select them as Chebychev points [5], which is what we have adopted. The details of computing the Chebychev points is given on page 96.

5.2 The Convolution

The proposed algorithm can be used successfully for small size AN's. For medium to large size networks, the computational burden places a heavy penalty on its use. This increase in the computational complexity is due to two reasons. The first is the growth of the order of the polynomial with each convolution operation. And the second is the exponential increase, $O(3^{n-1})$, in the number of subintervals n over which the convolution is defined.

We ealborate on the convolution of two *r.v.*'s. Extension to several *r.v.*'s is straightforward.

Let the two random variables be X and Y, with $f(x) = \sum_{i=0}^{n} c_i x^i;\ 0 \leq l_x \leq x \leq u_x$, and $g(y) = \sum_{i=0}^{m} d_i y^i;\ 0 \leq l_y \leq y \leq u_y;\ u_x$ and u_y are finite. Let $G(y) = \int_{l_y}^{y} g(y)\, dy$; the *cdf* of Y. Also define $Z = X + Y$ and the *cdf* of Z as $F(z)$. Without loss of generality, assume that $l_x \leq l_y$. Then there could be two possible convolution cases.

5.2.1 $l_x \leq l_y;\ u_x \leq u_y;\ u_x + l_y \leq u_y + l_x$

The breakpoints are $l_x + l_y, u_x + l_y, u_y + l_x$ and $u_x + u_y$ with $0 \leq l_x + l_y \leq u_x + l_y \leq u_y + l_x \leq u_x + u_y$. We can now write $F(z)$ as

$$\begin{aligned} F(z) &= F_1(z);\ l_x + l_y \leq z \leq u_x + l_y \\ &= F_2(z);\ u_x + l_y \leq z \leq u_y + l_x \\ &= F_3(z);\ u_y + l_x \leq z \leq u_x + u_y \end{aligned} \tag{5.1}$$

where,

$$F_1(z) = \int_{x=l_x}^{z-l_x} f(x)\, G(z-x)\, dx;\ l_x + l_y \leq z \leq u_x + l_y \tag{5.2}$$

$$F_2(z) = \int_{x=l_x}^{u_x} f(x)\, G(z-x)\, dx;\ u_x + l_y \leq z \leq u_y + l_x \tag{5.3}$$

$$F_3(z) = \int_{x=l_x}^{z-(u_x+l_y)} f(x)\, dx + \int_{x=z-(u_x+l_y)}^{u_x} f(x)\, G(z-x)\, dx;\ u_y + l_x \leq z \leq u_x + u_y \tag{5.4}$$

5.2.2 $l_x \leq l_y;\ l_x + u_y \leq u_x + l_y$

The breakpoints are $l_x + l_y, u_y + l_x, u_x + l_y$ and $u_x + u_y$ with $0 \leq l_x + l_y \leq u_y + l_x \leq u_x + l_y \leq u_x + u_y$. We can now write $F(z)$ as

$$\begin{aligned} F(z) &= F_1(z);\ l_x + l_y \leq z \leq u_y + l_x \\ &= F_2(z);\ u_y + l_x \leq z \leq u_x + l_y \\ &= F_3(z);\ u_x + l_y \leq z \leq u_x + u_y \end{aligned} \tag{5.5}$$

where,

$$F_1(z) = \int_{x=l_x}^{z-l_x} f(x)\, G(z-x)\, dx;\ l_x + l_y \leq z \leq u_y + l_x \tag{5.6}$$

$$F_2(z) = \int_{x=l_x}^{u_x} f(x)\, G(z-x)\, dx;\ u_y + l_x \leq z \leq u_x + l_y \tag{5.7}$$

$$F_3(z) = \int_{x=l_x}^{z-(u_y+l_x)} f(x)\, dx + \int_{x=z-(u_y+l_x)}^{u_x} f(x)\, G(z-x)\, dx;\ u_x+l_y \leq z \leq u_x+u_y \tag{5.8}$$

5.2.3 The Calculation of Integrals

We now focus on the calculation of the integrals $F_1(z)$, $F_2(z)$ and $F_3(z)$ in equations (5.1)-(5.8). Since the integrands in these integrals are polynomials, it is clear that the resulting integrations would be polynomials too. The calculations shown here are for case 5.2.1, *i.e.*, $l_x \leq l_y;\ u_x \leq u_y;\ u_x + l_y \leq u_y + l_x$. Calculations for case 5.2.2 are essentially similar.

The *cdf* $G(y)$ can be computed by integrating $g(y)$.

$$G(y) = \int_{l_y}^{y} g(z)dz = \delta_0 + \delta_1 y + \cdots + \delta_{m+1} y^{m+1} \quad l_y \leq y \leq u_y$$

where,

$$\delta_0 = -\sum_{i=1}^{m+1} \frac{d_{i-1}}{i} (l_y)^i \tag{5.9}$$

$$\delta_i = \frac{d_{i-1}}{i};\ for\ i \geq 1 \tag{5.10}$$

Knowledge of $G(x)$ leads to writing $G(z-x)$ as follows:

$$G(z-x) = \sum_{j=0}^{m+1} \left[h_j(z) x^j \right] \tag{5.11}$$

where,

$$h_k(z) = (-1)^k \sum_{j=k}^{m+1} \binom{j}{k} \delta_j z^{j-k} \quad for\ k = 0, 1, 2, \ldots, m+1 \tag{5.12}$$

We write the integrand $f(x)G(z-x)$ as a polynomial

$$\begin{aligned} P(z,x) &= f(x)G(z-x) = \left[c_0 + c_1 x + \cdots + c_n x^n\right]\left[h_0(z) + h_1(z)x + \cdots + h_{m+1}(z)x^{m+1}\right] \\ &= \alpha_0(z) + \alpha_1(z)\,x + \cdots + \alpha_{m+n+1}(z)\,x^{m+n+1} \end{aligned}$$

where $\alpha_q(z)$ are given by

$$\alpha_q(z) = \sum_{k=r(q)}^{k=s(q)} c_{q-r} h_k(z) \tag{5.13}$$

with

$$\begin{aligned} s(q) &= \min[q, m+1], \text{ and} \\ r(q) &= \max[0, q-n] \end{aligned}$$

The convolution integral can now be written as:

$$\begin{aligned} \int_L^U f(x)G(z-x)dx &= \int_L^U P(z,x)dx \\ &= \sum_{i=0}^{m+n+1}\left[\alpha_i(z)\int_L^U x^i dx\right] \end{aligned} \tag{5.14}$$

where $L = l_x + l_y$ and $U = u_x + u_y$.

Recalling the definition of $h_k(z)$ in (5.12), we obtain

$$\alpha_q(z) = \sum_{k=r(q)}^{k=s(q)} c_{q-r} h_k(z) = \sum_{i=0}^{m+1-r(q)} \beta_i^q z^i;\ q = 0, \ldots, m+n+1$$

where,

$$\beta_i^q = \left\{ \begin{array}{c} (-1)^i \sum_{j=r(q)}^{j=s(q)} \left(\begin{array}{c} i+j \\ j \end{array} \right) c_{q-j}\delta_{i+j} \quad i = 0, 1, 2, \ldots, m+1-s(q) \\ (-1)^i \sum_{j=r(q)}^{j=i} \left(\begin{array}{c} i+j \\ j \end{array} \right) c_{q-j}\delta_{i+j} \quad i = m+1-\{s(q)-1\}, \ldots, m+1-r(q) \end{array} \right\}$$

5.2.4 The Algorithm

The algorithm to compute the convolution can now be summarized in terms of additions and multiplication of coefficients of $f(x)$ and $g(y)$.

1. calculate δ_i for $i = 0, 1, 2, \ldots, m+1$.
2. calculate $h_k(z)$ for $k = 0, \cdots, m+1$.
3. calculate $\alpha_q(z)$ for $q = 0, 1, 2, \ldots, m+n+1$.
4. calculate $\int_L^U x^i dx$ for each i.
5. sum the coefficients of z^i to get the convolution.

Calculation of $F_1(z)$

$$\begin{aligned} F_1(z) &= \int_{x=l_x}^{z-l_x} f(x)G(z-x)dx \qquad l_x + l_y \le z \le u_x + l_y \\ &= \sum_{i=0}^{m+n+1} \left[\alpha_i(z)\gamma_i(z)\right] \\ &= \sum_{i=0}^{m+n+1} \sum_{k=0}^{m+1-r(i)+(i+1)} \left[b_k^i z^k\right] \end{aligned}$$

where,

$$\alpha_i(z) = \beta_0^i + \beta_1^i z + \cdots + \beta_{m+1-r(i)}^i z^{m+1-r(i)}$$

$$\gamma_i(z) = a_0^i + a_1^i z + \cdots + a_i^i z^{i+1}$$

a_j^i can be computed as follows:

$$a_0^i = (l_x)^{i+1}\left[(-1)^{i+1} - 1\right]\frac{1}{i+1}$$

$$a_j^i = \frac{i}{i+1}\begin{pmatrix} i+1 \\ j \end{pmatrix}(-1)^{i+1-j}(l_x)^{i+1-j}, \quad j = 1, 2, \ldots, i+1$$

To calculate b_k^i, let

$$\begin{aligned} q(k) &= \min\{k, i+1\}, \text{ and} \\ p(k) &= \max\{0, k-(m+1-r(i)\} \end{aligned}$$

then

$$b_k^i = \sum_{l=p(k)}^{l=q(k)} \beta_{k-l}^i a_k^i$$

$F_1(z)$ can now be computed as the sum of polynomials.

Calculation of $F_2(z)$

$$\begin{aligned} F_2(z) &= \int_{x=l_x}^{u_x} f(x)G(z-x)dx, \quad u_x + l_y \le z \le u_y + l_x \\ &= \sum_{i=0}^{m+n+1} \left[\alpha_i(z) \int_{l_x}^{u_x} x^i dx\right] \\ &= \sum_{i=0}^{m+n+1} \left(\frac{u_x^i - l_x^i}{i+1}\right) \alpha_i(z) \end{aligned}$$

Thus the calculation of $F_2(z)$ reduces to that of sum of the coefficients of the polynomials.

Calculation of $F_3(z)$

$$F_3(z) = \int_{x=l_x}^{z-(u_x+l_y)} f(x)dx + \int_{x=z-(u_x+l_y)}^{u_x} f(x)G(z-x)dx, \quad u_y + l_x \le z \le u_x + u_y$$

The second integration is very similar to the calculation of $F_1(z)$. Consider

$$\int_{x=z-(u_x+l_y)}^{u_x} f(x)G(z-x)dx = -\int_{x=u_x}^{z-(u_x+l_y)} f(x)G(z-x)dx$$

The coefficients a_j^i would however change as follows:

$$a_0^i = \frac{-1}{i+1}\left[(-1)^{i+1}(u_x + l_y)^{i+1} - (u_x)^{i+1}\right]$$

$$a_j^i = \frac{-1}{i+1}\left[\begin{pmatrix} i+1 \\ j \end{pmatrix}(-1)^{i+1-j}(u_x + l_y)^{i+1-j}\right], \quad j = 1, 2, 3, \ldots, i+1$$

The first integration can be computed as follows:

$$\begin{aligned}\int_{x=l_x}^{z-(u_x+l_y)} f(x)dx &= \sum_{i=0}^{n} c_i \int_{l_x}^{z-(u_x+l_y)} x^i dx \\ &= \sum_{i=0}^{n}\sum_{j=0}^{i+1} e_j^i z^j\end{aligned}$$

where,

$$e_0^i = \frac{c_i}{i+1}\left[(-1)^{i+1}(u_x + l_y)^{i+1} - (l_x)^{i+1}\right]$$

$$e_j^i = \frac{c_i}{i+1}\begin{pmatrix} i+1 \\ j \end{pmatrix}(-1)^{i+1-j}(u_x + l_y)^{i+1-j}$$

Thus the calculation of $F_3(z)$ reduces to that of sum of the coefficients of the polynomials.

5.2.5 Examples

Example 1.

This is the same example given in Martin [44]. Given $f(x) = 1;\ 0 \le x \le 1$ and $g(y) = 1;\ 0 \le y \le 1$, then the breakpoints are $0, 1$ and 2. We have,

$$F_1(z) = \int_{x=0}^{z} (z - x)\, dx = z^2/2;\ 0 \le z \le 1$$

and,

$$F_2(z) = \int_{x=0}^{z-1} dx + \int_{x=z-1}^{1} (z - x)\, dx = -\frac{3}{2} + z(3 - z) + \frac{(z-1)^2}{2}\ 1 \le z \le 2$$

Note that Martin obtains the density functions: $f_1(z) = z$ and $f_2(z) = 2 - z$.

Example 2.

This is the same example given in Ramat, Lenté & Tacquard [47]. Given $f(x) = 1/2;\ 0 \leq x \leq 2$ and $g(y) = 1/3;\ 0 \leq y \leq 8$, then the breakpoints are $5, 7, 8, 10$. We have,

$$F_1(z) = \int_{x=0}^{z-5} \frac{1}{2}\,\frac{z-5-x}{3}\,dx = \frac{(z-5)^2}{12};\ 5 \leq z \leq 7$$

$$F_2(z) = \int_{x=0}^{2} \frac{1}{2}\,\frac{z-5-x}{3}\,dx = \frac{z-6}{3};\ 7 \leq z \leq 8$$

and,

$$\begin{aligned} F_3(z) &= \int_{x=0}^{2} \frac{1}{2}\,\min\{1,\,\frac{z-5-x}{3}\} \\ &= \int_{x=0}^{z-x} \frac{1}{2}\,dx + \int_{x=z-x}^{2} \frac{1}{2}\,\frac{z-5-x}{3}\,dx \\ &= \frac{z-8}{2} + \frac{1}{6}\left[2(z-5) - 2 - (z-5)(z-8) + \frac{(z-8)^2}{2}\right];\ 8 \leq z \leq 10 \end{aligned}$$

This is the result secured by Ramat, Lenté & Tacquard by a different (and computationally more demanding) approach.

5.3 Approximation Through Discretization

Discretization of a continuous *pdf* is a popular method for implementig project planning algorithms. The popularity is attributed to the ease of convolution (and maximum) operations albeit at the cost of reduced accuracy. Nonetheless, the computational ease enables one to model large scale systems.

The popular methods to discretize a continuous density function, along with their relative pros and cons, are the following:

Moment Matching: The moments of the function to be discretized are matched with the moments of the discrete points to give the location of discrete points

and their probabilities. This leads to a system of non-linear equations which quickly becomes computationally impossible to solve. Also, this method can not be used successfully where the requisite moments do not exist. The positive aspect of this method is the reduced error of subsequent computations that are based on the discrete approximation.

Equal Probability: The distance between the discrete points is selected so that each one of them has equal probability. The resulting points have equal probabilities and (usually) variable distance between them. This method requires inversion of continuous *pdf*'s; hence it is feasible only for functions which are easily invertible, thus limiting its use in genaral. Numerical inversion techniques can be applied to broaden the use of this procedure.

Equal Distance: The discrete points are chosen such that the distances between them are equal. The probability mass assigned to each point varies with the point. This method is computationally the easiest of all methods. On the negative side, it is the least accurate of the three mentioned here.

The discretization of density function has its own shortcomings. The obvious one is the loss of information in translation from the continuous version of *pdf*. This is the shortcoming of discretization that is most often acknowledged in the literature and practice. There are, however, two more sources of error and/or computational bottlenecks associated with discretization. They are:

1. **Support Shrinkage** The shrinking of the support occurs because, normally, the discrete *pmf* is defined over a shorter range of the continuous *pdf* to ensure that the mean of the discrete probability mass function (*pmf*) is equal to the mean of the continuous function. Further shrinking of the support occurs with each convolution (and maximum) operation. This shrinking of the support can be minimized by increasing the number of points used for discretization. However, the increased number of discrete points creates storage problems as described in the next reason below.

2. **Storage Issues** With each convolution (and maximum) operation the number of discrete points grows often multiplicatively, creating a storage problem during implementation. There is, therefore, need to recreate a set of points which represent the discrete *pmf* and its properties, and maintain the representation to no more than n points, say. Needless to say, this action further reduces the accuracy of the approximations, and is a potential for additional support shrinkage.

It is evident from the above discussion that all three types of errors are related to each other. Thus a scheme is developed where a given number of discrete points can be collapsed into a smaller number of points with the same expected value, while maintaining the orginal support.

We propose to use the Chebychev points for a given interval $[a, b]$ to discretize the continuous *pdf*, or to collapse a discrete *pmf* in N discrete points to another discrete *pmf* in n points, with $n \ll N$. The discretization scheme is as follows.

Recall that the objective here is to compute the location $x_i,\ i = 1, \ldots, n$, and the corresponding probabilities $p_i,\ i = 1, \ldots, n$ that best approximate the continuous *pmf*. The x_i's can be computed by dividing the semi-circle over the range $[a, b]$ into n equal segments. The mid points of these n segments are then projected onto the support to determine the location of the x_i's. Thus,

$$x_i = a + \left(\frac{b-a}{2}\right)\left[1 - \cos\frac{\pi(2i-1)}{2n}\right],\ i = 1, \ldots, n \tag{5.15}$$

where $[a, b]$ is the original support of the *pdf* (or the *pmf*).

The probability mass at each of these x_i's may be computed, for example, by satisfying the first $n-1$ moments of the discrete *pmf* [2]. The last equation comes from realizing that the sum of all probability masses should be one. Let m_k denote the k^{th} moment of the original *pdf*. The p_i's are now determined from the solution of the following set of linear equations:

[2] For a continuous *pdf* we do not need to calculate the moments. The probabilities can be calculated by integration.

$$\begin{aligned} \sum_{i=1}^{n} p_i x_i^k &= m_k,\ k = 1, \ldots, n-1 \\ \sum_{i=1}^{n} p_i &= 1. \end{aligned} \tag{5.16}$$

At any step, the shrinking of the support can be avoided by expanding the range $[a, b]$ into a new range $\left[a - sin(\frac{\pi}{2n}), b + sin(\frac{\pi}{2n})\right]$ and redistributing the points. The collapsing of the points to a smaller number follows a similar logic.

This method of discretization provides an improved scheme where not only the shrinking of the support is eliminated but also the first $n-1$ moments are matched throughout the process.

5.4 Example of Discrete Convolution

Consider two *pmf*'s Z_1 and Z_2. Where Z_1 takes on values $[1, 3, 5, 7, 9]$ and Z_2 takes values $[6, 7, 8, 9, 10]$. Both with probabilities $[1/5, 1/5, 1/5, 1/5, 1/5]$. Let $Z = Z_1 + Z_2$. Then Z takes on values

$$[7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19]$$

with probabilities

$$[0.04, 0.04, 0.08, 0.08, 0.12, 0.08, 0.12, 0.08, 0.12, 0.08, 0.08, 0.04, 0.04].$$

It is evident from the above calculations that the storage requirements for Z is significantly higher than that for Z_1 and/or Z_2. In situations where large number of *r.v.*'s are to be convolved, the storage requirement would grow geometrically with number of *r.v.*'s. The preferred course of action would be to reduce the number of points on which Z is defined from 13 to a smaller number, say 5. Call this new (after reduction) *r.v.* as ζ.

A straight forward application of equation 5.16 with $a = 7, b = 19$ and $n = 5$ results in

$$\zeta = [7.2937, 9.4733, 13.0000, 16.5267, 18.7063]$$

with probabilities

$$[0.0666, 0.2278, 0.4114, 0.2278, 0.0666].$$

Consider the support of $\zeta, [7.2937, 18.7063]$. It has shrunk from the support of Z which is $[7, 19]$. To avoid such shrinking of the support we need to redistribute the values of ζ and its associated probabilities. The redistributed values are

$$\zeta = [6.9998, 9.2917, 13.0000, 16.7083, 19.0002]$$

with probabilities

$$[0.0462, 0.2425, 0.4225, 0.2425, 0.0462].$$

The shrinking of the support could also have been avoided in the first place by taking $a = 7 - sin\left(\frac{\pi}{2n}\right)$ and $b = 19 + sin\left(\frac{\pi}{2n}\right)$ in the original calculations.

Chapter 6

Summary, Conclusions and Extensions

This thesis presents a new paradigm for modeling projects whose activities exhibit stochastic variability in their duration estimates. We focus on the modeling and the computational aspects of this methodology for successful implementation in practice. The following presents a brief overview of the work accomplished, conclusions reached and a guide to the extensions of this work.

The core of this work relates to the development of the *DiAN* model to better explain the stochastic variability of an activity, hence a project. In that we model the remaining work content (*rwc*) of an activity over time as a diffusion process in the presence of an absorbing barrier. The absorption time of which corresponds to the completion time of an activity. The bounds on the project completion time were estimated using the bounding scheme developed by Elmaghraby [23].

During the course of this work two auxiliary issues were also explored. One is the generation of test networks of given node reduction index (*n.r.i.*), the subject matter of Chapter 4. The other, in Chapter 5, is the convolution of two random variables via polynomial or discrete approximations. This work was necessary to ensure the computational effectiveness of the *DiAN* model in Chapter 2.

In Chapter 3 we present a methodology for controlling an activity that has substantially deviated from its intended course. Three control strategies were developed and compared against the do-nothing strategy for both completion time and cost.

During the course of this research we made several assumptions and gained an understanding of how this work can be further extended. In particular, we would like to outline several areas of research that we hope would guide the future researchers on this topic.

1. The current model, in Chapter 2, focuses on a specific topological structure of the network with strict precedence constraints which results in a directed acyclic graph (*dag*). This representation, while being the most popular, is not the sole representative of project management networks. We believe that this work could be extended to networks with generalized precedence relationship and to stochastic activity networks [20]. On a related note, this work focuses on diffusion processes with constant instantaneous mean and variance. We believe that, for more realistic representation, the *DiAN* model can be extended to the more general diffusion processes [27] with instantaneous mean and variances that are functions of time and/or state.

2. In Chapter 3, the control state remains fixed at a location (e.g., at *rwc=1*) for controlling an activity. In practice the location of the control state can vary with time. The shape of the control state can be used to represent various policy considerations. As an example a control state at 1.0 in the beginning of the activity and 0.5 towards the end of the activity can be used to represent the varying nature of risk taking capacity of an organization. This raises the issue of computing the distribution of the completion time of an activity in presence of time varying control states.

3. The "corrective measures" discussed in Chapter 3 relate to a single activity and not the whole project. These control strategies are myopic as they do not take into account the other activities that are being executed simultaneously or the ones that succeed the activity under consideration. We believe that this study can be extended in at least two directions. The first being the control

of the whole project as opposed to a single activity. The second extension is controlling the project in the presence of limited resources.

4. In Chapter 4 we presented a methodology for generating the Activity-on-Arc (AoA) networks of given node reduction index ($n.r.i.$). There is no known methodology of generating Activity-on-Node (AoN) networks of given $n.r.i.$ As a matter of fact finding the $n.r.i.$ of an AoN is itself an open issue. Such development would greatly assist in ameliorating the computing burden of network optimization under limited availability of resources.

In this research we have proposed a methodology for systematically modeling the activities, hence projects, with stochastic durations that exhibit increased variability with increased duration of an activity. The method developed here is computationally efficient to be used in the practice. However, as true with any new paradigm we have only scratched the surface of this topic. We hope that the $DiAN$ model would inspire and guide the future researchers in furthering this work.

Appendix A

Impact of Error in Simulating the Diffusion Processes

Diffusion processes are continuous and non-differentiable almost everywhere (see Chapter 1). The simulation of such a process, using discrete steps, samples it only at the finite number of points. This leaves the process unobservable in between two sample points. The following discussion focuses on the error of observing such a process only at finite intervals (Δt) and the resulting impacts on the control strategies presented in Chapter 3.

To document, the magnitude of such an error due to sampling at the discrete times, we used two techniques. The first numerically computes the probability of reaching the control state in the desired time interval (Δt). The other computes the upper bound on the probability of crossing the control state undetected at finite number of points in Δt.

For the first method, the probability density function of the time (t) of reaching the control state can be written as

$$f(t) = \frac{d}{\sigma\sqrt{2\pi t^3}} \exp\left[\frac{-1}{2\sigma^2 t}(d + \mu t)^2\right];\ t > 0.$$

Here, d is the distance from the control state, μ the instantaneous mean and σ^2 the instantaneous variance. The cumulative density function was obtained by numerically integrating the above equation. The results of the analysis are presented in Table A.1.

Table A.1: Probability of reaching the control state

d	Δt	μ			
		-0.01	-0.025	-0.05	-0.1
0.001	0.01	5.138×10^{-02}	2.974×10^{-02}	1.067×10^{-02}	9.246×10^{-04}
	0.05	2.933×10^{-01}	1.497×10^{-01}	3.713×10^{-02}	1.533×10^{-03}
	0.1	3.890×10^{-01}	1.821×10^{-01}	3.903×10^{-02}	1.533×10^{-03}
0.005	0.01	9.243×10^{-20}	7.475×10^{-21}	9.620×10^{-23}	8.694×10^{-27}
	0.05	1.050×10^{-05}	6.301×10^{-07}	2.821×10^{-09}	4.658×10^{-15}
	0.1	7.650×10^{-04}	3.329×10^{-05}	5.234×10^{-08}	8.477×10^{-15}
0.01	0.01	6.099×10^{-73}	4.341×10^{-75}	9.727×10^{-79}	2.659×10^{-86}
	0.05	3.101×10^{-17}	1.589×10^{-19}	1.101×10^{-23}	2.824×10^{-33}
	0.1	4.224×10^{-10}	1.465×10^{-12}	2.563×10^{-17}	3.853×10^{-29}

We have taken the $\sigma = 0.0745$ which corresponds to the spread $\psi = 1.0$ (see Chapter 3). The probability of reaching the control state is essentially negligible for $\psi = 0.25$ and 0.5, the other two values of ψ used in Chapter 3.

It can be observed from Table A.1 that except for $d = 0.001$, all other probability values are essentially negligible. The probability values are also negligible for $\mu < -0.1$. This stands to reason since for large values of instantaneous mean the deterministic part of the diffusion process takes over the stochastic part.

It can be argued that the probabilities calculated in Table A.1 do not represent the probability of crossing the control state undetected. These probabilities, however, do provide a significant insight into the problem of crossing the control state undetected. The control state can be crossed undetected (even detected) only if the diffusion process can reach the control state in the first place. This is what this analysis intends to show.

The second analysis aims to calculate an upper bound on the probability of crossing the state undetected. We calculate the probability that a process starting from time 0 crosses the control state at a given time $0 < t_i < \Delta t$ and comes below the control state in time $\Delta t - t_i$ assuming it was at the control state at time t_i. These probabilities were calculated at finite (2 and 4) number of points. The results of the analysis are presented in Table A.2.

Table A.2: Probability of crossing the control state undetected

# points	Δt	d		
		0.001	0.005	0.01
2	0.01	0.2086	0.0797	0.0124
	0.05	0.2238	0.1531	0.0843
	0.1	0.2240	0.1694	0.1137
4	0.01	0.6171	0.2213	0.0392
	0.05	0.6685	0.4418	0.2328
	0.1	0.6693	0.4958	0.3207

In the analysis presented in Table A.2 we have taken the instantaneous mean $(\mu) = -0.1$ and the spread $(\psi) = 1.0$. It is evident from the table that as the number of points increase so does the probability of crossing the state undetected. This is to be expected as we have summed the individual probabilities computed at the individual intermediate points. A better estimate would be to sum the conditional probabilities that the process has not reached the control state at previous points.

The analyses presented here reveal that the control state could be reached, with a significant probability, when the process is in the immediate viscinity of the control state, the instantaneous mean is small and the instantaneous variance is high. The examples for control strategies studied in Chapter 3 use $\mu = 0.1$ and $d \geq 0.01$ which results in the probability of reaching the control state negligibly. The upper bounds on the probability of crossing the the control state, presented in the second analysis, show a much higher value. This, however, is only an upper bound and a poor one at best.

This issue can also be addressed from the managerial perspective. The process can only be controlled if it is measurable. For most of the projects in construction, software developments and similar industries it is difficult, if not impossible, to measure the project at a granularity finer than 0.1 day. A second related issue would be the impacts on the project of knowing that it has indeed reached the control state undetected. We submit that this information would have little if any impact on the desired managerial objectives. It is of questionable value that the management needs

to react to the continuous changes in the status of a project.

In practical situations, we would recommend that the observation (measurement) interval of an activity Δt be separated from the decision making interval, which may be a multiple of Δt, if computational cost is not an issue. As an example, in this case one could take the decision making interval to be 0.1 or 1 day, while observing the process at $\Delta t = 0.01$ or even shorter time intervals. The decison making events could include the information as to how many times an activity has reached the control state since the last decision making event, though we suspected that such an information would be of little managerial value except, perhaps, in providing management with a base for determining the capacity requirements in future activities of a similar nature.

Appendix B

Impact of Strategies on Completion Time of an Activity

The simulation runs for the completion time comparison of various strategies are presented in this appendix. We did not present the simulation output of control state located at 1.5. In this scenario ($\alpha = 1.5$) the control state was never reached thus the results of the simulation do not significantly add to the information presented here.

Table B.1: $\psi = 0.25$ case

Strategy	Control State	Crash Time	Completion Time	
			Mean	Std Dev
1	1.01	2	10.07	0.50
2	1.01	2	10.07	0.59
3	1.01	2	10.21	0.55
4	1.01	2	10.10	0.61
1	1.01	5	10.07	0.56
2	1.01	5	10.05	0.58
3	1.01	5	10.05	0.57
4	1.01	5	10.05	0.56
1	1.05	2	10.11	0.59
2	1.05	2	10.02	0.64
3	1.05	2	10.13	0.59
4	1.05	2	10.10	0.66
1	1.05	5	10.19	0.61
2	1.05	5	10.06	0.58
3	1.05	5	9.97	0.54
4	1.05	5	10.12	0.65
1	1.10	2	10.07	0.58
2	1.10	2	10.12	0.56
3	1.10	2	10.05	0.57
4	1.10	2	10.04	0.55
1	1.10	5	10.07	0.59
2	1.10	5	10.09	0.58
3	1.10	5	10.02	0.64
4	1.10	5	10.04	0.52
1	1.25	2	10.15	0.61
2	1.25	2	10.01	0.65
3	1.25	2	10.18	0.68
4	1.25	2	10.06	0.51
1	1.25	5	10.06	0.67
2	1.25	5	10.12	0.54
3	1.25	5	10.04	0.62
4	1.25	5	10.03	0.61

Table B.2: $\psi = 0.5$ case

Strategy	Control State	Crash Time	Completion Time	
			Mean	Std Dev
1	1.01	2	9.98	1.15
2	1.01	2	9.96	1.10
3	1.01	2	11.75	7.88
4	1.01	2	10.21	1.22
1	1.01	5	9.95	1.14
2	1.01	5	10.15	1.26
3	1.01	5	12.13	11.08
4	1.01	5	10.04	1.14
1	1.05	2	10.15	1.34
2	1.05	2	10.08	1.11
3	1.05	2	10.66	6.68
4	1.05	2	9.95	1.16
1	1.05	5	10.02	1.12
2	1.05	5	9.92	1.14
3	1.05	5	10.20	1.09
4	1.05	5	10.10	1.37
1	1.10	2	9.90	1.20
2	1.10	2	10.15	1.30
3	1.10	2	10.12	1.19
4	1.10	2	10.02	1.07
1	1.10	5	10.17	1.27
2	1.10	5	10.10	1.24
3	1.10	5	10.27	1.20
4	1.10	5	10.03	1.22
1	1.25	2	10.21	1.20
2	1.25	2	10.00	1.14
3	1.25	2	10.12	1.21
4	1.25	2	10.12	1.29
1	1.25	5	9.94	1.13
2	1.25	5	10.14	1.14
3	1.25	5	10.21	1.15
4	1.25	5	10.02	1.14

Table B.3: $\psi = 1.0$ case

Strategy	Control State	Crash Time	Completion Time	
			Mean	Std Dev
1	1.01	2	9.89	2.28
2	1.01	2	10.17	2.21
3	1.01	2	16.54	16.16
4	1.01	2	9.81	2.54
1	1.01	5	10.68	2.50
2	1.01	5	10.11	2.35
3	1.01	5	16.34	11.31
4	1.01	5	9.69	2.06
1	1.05	2	10.15	2.33
2	1.05	2	10.03	2.17
3	1.05	2	12.42	12.57
4	1.05	2	10.28	2.69
1	1.05	5	9.88	2.32
2	1.05	5	9.98	2.11
3	1.05	5	12.44	11.54
4	1.05	5	10.11	2.04
1	1.10	2	10.05	2.23
2	1.10	2	10.25	2.55
3	1.10	2	12.22	12.17
4	1.10	2	9.97	2.19
1	1.10	5	9.76	2.37
2	1.10	5	10.40	2.58
3	1.10	5	10.73	3.68
4	1.10	5	10.16	2.64
1	1.25	2	9.86	2.40
2	1.25	2	10.40	2.64
3	1.25	2	9.96	2.21
4	1.25	2	9.93	2.41
1	1.25	5	10.40	2.63
2	1.25	5	9.96	2.49
3	1.25	5	10.15	2.33
4	1.25	5	10.31	2.42

Appendix C

Impact of Strategies on Cost of an Activity

We conducted simulation runs under various scenarios to study the impact of the strategies on the cost of completing an activity. In the following we do not present the complete list of all the runs. The list is limited to the runs deemed significant in drawing the conclusions.

Table C.1: $\psi = 0.25$ and $\alpha = 1.01$ case

	Costs			Cost Under Strategies							
				Strategy 1		Strategy 2		Strategy 3		Strategy 4	
t_c	C_r	C_s	C_d	μ	σ	μ	σ	μ	σ	μ	σ
2	2	5	2	10.56	1.20	10.62	1.38	10.65	1.30	10.48	1.14
2	5	5	2	10.52	1.40	10.74	1.36	10.44	1.27	10.67	1.42
2	2	10	2	10.65	1.25	10.33	1.18	10.50	1.24	10.53	1.16
2	5	10	2	10.44	1.10	10.58	1.53	10.61	1.24	10.65	1.27
2	2	5	5	10.99	2.00	11.57	2.41	11.73	2.66	11.42	2.27
2	5	5	5	11.54	2.72	11.34	2.16	11.55	2.75	11.17	2.21
2	2	10	5	11.46	2.50	12.21	3.44	11.64	2.66	11.26	2.27
2	5	10	5	11.47	2.73	11.20	2.21	11.60	2.34	11.62	2.46
2	2	5	10	12.52	4.61	13.58	5.03	12.95	4.27	12.68	3.78
2	5	5	10	12.27	3.71	12.47	3.86	11.67	3.75	12.70	4.14
2	2	10	10	12.28	4.31	12.43	4.26	12.98	4.80	14.22	4.81
2	5	10	10	11.97	3.71	12.84	4.42	12.69	4.59	12.96	4.45
5	2	5	2	10.44	1.40	10.56	1.29	10.73	1.40	10.65	1.36
5	5	5	2	10.73	1.30	10.79	1.51	10.39	1.00	10.57	1.21
5	2	10	2	10.56	1.18	10.44	1.28	10.58	1.20	10.53	1.23
5	5	10	2	10.76	1.32	10.57	1.33	10.51	1.10	10.77	1.40
5	2	5	5	11.66	2.58	11.13	2.26	11.31	2.43	11.42	2.41
5	5	5	5	11.64	2.37	11.32	2.23	11.27	2.35	11.19	2.26
5	2	10	5	11.09	2.08	11.23	2.25	11.38	2.67	11.65	2.62
5	5	10	5	11.82	2.61	11.11	2.19	11.34	2.50	11.44	2.79
5	2	5	10	12.70	4.20	13.01	4.07	12.75	4.27	12.55	4.39
5	5	5	10	12.66	4.43	12.36	4.05	13.17	4.69	12.43	4.04
5	2	10	10	13.08	4.49	12.66	4.00	12.94	4.98	12.13	3.57
5	5	10	10	12.83	4.74	13.78	4.89	11.92	4.14	12.72	4.23

Table C.2: $\psi = 0.25$ and $\alpha = 1.05$ case

	Costs			Cost Under Strategies							
				Strategy 1		Strategy 2		Strategy 3		Strategy 4	
t_c	C_r	C_s	C_d	μ	σ	μ	σ	μ	σ	μ	σ
2	2	5	2	10.34	1.14	10.59	1.17	10.76	1.32	10.62	1.37
2	5	5	2	10.43	1.17	10.53	1.37	10.83	1.40	10.71	1.42
2	2	10	2	10.53	1.15	10.70	1.31	10.74	1.25	10.72	1.32
2	5	10	2	10.43	1.43	10.48	1.21	10.57	1.29	10.73	1.38
2	2	5	5	11.88	2.91	11.55	2.64	11.53	2.45	11.22	2.57
2	5	5	5	11.62	2.43	11.54	2.30	11.26	2.26	11.30	2.50
2	2	10	5	10.87	1.90	11.52	2.60	11.59	2.69	11.11	2.60
2	5	10	5	11.24	1.98	11.18	2.27	11.34	2.44	11.65	2.58
2	2	5	10	12.44	3.86	11.97	3.84	11.92	4.08	12.45	3.72
2	5	5	10	12.26	4.00	12.46	3.76	11.98	3.85	12.57	4.43
2	2	10	10	13.21	5.21	13.03	4.20	13.26	4.29	12.15	3.60
2	5	10	10	12.28	3.92	12.83	4.11	12.65	4.04	12.55	3.79
5	2	5	2	10.59	1.22	10.51	1.31	10.25	1.06	10.73	1.43
5	5	5	2	11.03	1.60	10.64	1.21	10.54	1.26	10.60	1.43
5	2	10	2	10.78	1.31	10.70	1.43	10.54	1.17	10.40	1.15
5	5	10	2	10.62	1.27	10.69	1.27	10.67	1.39	10.51	1.29
5	2	5	5	11.14	2.34	11.18	2.30	11.27	2.37	11.24	2.41
5	5	5	5	11.33	2.81	10.94	2.39	11.56	2.72	11.44	2.61
5	2	10	5	11.50	2.57	11.27	2.16	11.25	2.08	11.12	2.34
5	5	10	5	11.33	2.46	11.03	2.27	11.55	2.53	11.03	1.79
5	2	5	10	12.50	3.92	13.18	4.22	12.48	3.40	13.16	4.27
5	5	5	10	13.59	4.62	12.34	4.14	13.38	5.10	13.12	4.56
5	2	10	10	12.59	3.88	13.24	5.00	12.90	4.39	13.50	4.82
5	5	10	10	13.59	5.49	12.38	3.52	12.66	4.28	12.21	4.03

Table C.3: $\psi = 0.25$ and $\alpha = 1.10$ case

	Costs			Cost Under Strategies							
				Strategy 1		Strategy 2		Strategy 3		Strategy 4	
t_c	C_r	C_s	C_d	μ	σ	μ	σ	μ	σ	μ	σ
2	2	5	2	10.87	2.49	12.04	3.10	31.02	74.54	10.84	2.37
2	5	5	2	10.95	2.43	11.70	3.47	12.88	15.86	11.16	2.76
2	2	10	2	11.57	3.00	11.40	2.80	35.80	95.97	11.26	2.59
2	5	10	2	11.18	2.45	11.58	3.49	20.06	47.37	11.72	3.10
2	2	5	5	13.38	5.78	12.58	4.73	30.66	89.66	12.56	4.78
2	5	5	5	12.72	4.79	13.05	5.46	38.80	120.06	12.89	5.04
2	2	10	5	11.90	3.96	12.26	4.71	26.21	71.60	12.88	4.89
2	5	10	5	13.20	5.64	12.71	4.52	59.24	237.23	13.38	5.06
2	2	5	10	15.85	8.68	15.77	10.44	51.79	198.62	16.62	10.41
2	5	5	10	16.38	9.28	14.70	8.69	49.36	132.61	15.08	8.94
2	2	10	10	16.19	8.83	15.32	7.67	46.61	191.32	15.43	8.91
2	5	10	10	16.57	9.53	15.41	8.59	43.19	143.61	14.93	7.25
5	2	5	2	10.91	2.29	11.04	2.71	21.18	41.72	11.27	2.62
5	5	5	2	11.26	2.80	11.43	3.25	31.59	74.70	11.68	3.13
5	2	10	2	10.85	2.29	10.99	2.06	41.71	122.87	11.23	2.56
5	5	10	2	11.58	2.66	11.52	3.14	16.44	27.63	11.66	3.41
5	2	5	5	12.92	5.14	13.17	5.42	43.83	172.35	12.29	4.94
5	5	5	5	13.34	5.82	13.12	6.02	48.14	143.73	12.80	4.51
5	2	10	5	12.36	4.45	13.51	5.15	44.35	126.00	13.37	5.05
5	5	10	5	12.44	4.59	13.62	5.13	42.98	134.01	12.70	4.80
5	2	5	10	14.55	7.74	15.93	9.62	58.76	249.38	15.85	9.27
5	5	5	10	15.06	8.73	15.30	7.78	25.33	51.69	15.22	8.57
5	2	10	10	14.82	7.66	15.46	8.64	52.60	169.30	15.19	8.61
5	5	10	10	15.74	9.18	16.17	10.27	25.28	55.43	16.23	9.95

Table C.4: $\psi = 0.50$ and $\alpha = 1.01$ case

	Costs			Cost Under Strategies							
				Strategy 1		Strategy 2		Strategy 3		Strategy 4	
t_c	C_r	C_s	C_d	μ	σ	μ	σ	μ	σ	μ	σ
2	2	5	2	10.87	2.49	12.04	3.10	31.02	74.54	10.84	2.37
2	5	5	2	10.95	2.43	11.70	3.47	12.88	15.86	11.16	2.76
2	2	10	2	11.57	3.00	11.40	2.80	35.80	95.97	11.26	2.59
2	5	10	2	11.18	2.45	11.58	3.49	20.06	47.37	11.72	3.10
2	2	5	5	13.38	5.78	12.58	4.73	30.66	89.66	12.56	4.78
2	5	5	5	12.72	4.79	13.05	5.46	38.80	120.06	12.89	5.04
2	2	10	5	11.90	3.96	12.26	4.71	26.21	71.60	12.88	4.89
2	5	10	5	13.20	5.64	12.71	4.52	59.24	237.23	13.38	5.06
2	2	5	10	15.85	8.68	15.77	10.44	51.79	198.62	16.62	10.41
2	5	5	10	16.38	9.28	14.70	8.69	49.36	132.61	15.08	8.94
2	2	10	10	16.19	8.83	15.32	7.67	46.61	191.32	15.43	8.91
2	5	10	10	16.57	9.53	15.41	8.59	43.19	143.61	14.93	7.25
5	2	5	2	10.91	2.29	11.04	2.71	21.18	41.72	11.27	2.62
5	5	5	2	11.26	2.80	11.43	3.25	31.59	74.70	11.68	3.13
5	2	10	2	10.85	2.29	10.99	2.06	41.71	122.87	11.23	2.56
5	5	10	2	11.58	2.66	11.52	3.14	16.44	27.63	11.66	3.41
5	2	5	5	12.92	5.14	13.17	5.42	43.83	172.35	12.29	4.94
5	5	5	5	13.34	5.82	13.12	6.02	48.14	143.73	12.80	4.51
5	2	10	5	12.36	4.45	13.51	5.15	44.35	126.00	13.37	5.05
5	5	10	5	12.44	4.59	13.62	5.13	42.98	134.01	12.70	4.80
5	2	5	10	14.55	7.74	15.93	9.62	58.76	249.38	15.85	9.27
5	5	5	10	15.06	8.73	15.30	7.78	25.33	51.69	15.22	8.57
5	2	10	10	14.82	7.66	15.46	8.64	52.60	169.30	15.19	8.61
5	5	10	10	15.74	9.18	16.17	10.27	25.28	55.43	16.23	9.95

Table C.5: $\psi = 0.50$ and $\alpha = 1.05$ case

	Costs			Cost Under Strategies							
				Strategy 1		Strategy 2		Strategy 3		Strategy 4	
t_c	C_r	C_s	C_d	μ	σ	μ	σ	μ	σ	μ	σ
2	2	5	2	11.02	2.47	11.38	2.30	11.26	2.91	10.88	2.42
2	5	5	2	11.25	2.83	11.22	2.49	11.29	3.17	10.97	2.46
2	2	10	2	11.23	2.82	10.71	2.28	11.42	2.91	10.98	2.40
2	5	10	2	11.31	2.75	11.12	2.34	10.94	2.41	11.09	2.37
2	2	5	5	12.43	4.73	12.79	4.46	12.28	4.95	12.98	4.96
2	5	5	5	12.90	4.58	12.31	5.19	12.21	5.14	12.55	4.80
2	2	10	5	12.58	4.92	12.46	4.45	13.47	5.93	13.20	5.98
2	5	10	5	12.84	4.68	12.48	4.89	12.21	4.94	12.50	5.37
2	2	5	10	15.41	9.51	14.72	7.43	15.42	8.51	16.27	9.39
2	5	5	10	16.05	9.00	14.63	8.09	15.26	8.83	15.24	8.77
2	2	10	10	15.58	9.60	13.52	8.01	14.74	9.72	15.45	9.91
2	5	10	10	14.71	8.75	14.12	7.83	14.63	9.11	13.05	7.06
5	2	5	2	11.17	2.62	10.83	2.37	11.08	2.55	11.12	2.66
5	5	5	2	11.41	3.07	11.50	2.76	11.31	2.90	10.70	2.63
5	2	10	2	10.62	2.38	10.68	2.46	10.75	2.45	11.47	2.59
5	5	10	2	11.49	2.85	11.04	2.75	11.41	3.00	10.83	2.70
5	2	5	5	13.44	6.20	12.57	5.23	12.07	4.45	12.21	3.98
5	5	5	5	12.11	3.98	13.23	5.70	13.14	5.25	13.54	5.87
5	2	10	5	11.82	4.55	11.81	3.86	12.74	5.65	12.55	4.92
5	5	10	5	13.12	5.24	11.53	4.06	12.68	5.01	13.33	5.54
5	2	5	10	15.63	8.49	14.26	8.08	15.57	9.91	15.59	9.98
5	5	5	10	16.55	10.15	14.51	7.87	14.61	8.38	14.33	7.59
5	2	10	10	14.28	8.83	15.87	9.91	15.25	8.61	14.59	8.50
5	5	10	10	14.81	8.70	15.74	9.57	16.21	8.90	15.03	7.52

Table C.6: $\psi = 0.50$ and $\alpha = 1.10$ case

	Costs			Cost Under Strategies							
				Strategy 1		Strategy 2		Strategy 3		Strategy 4	
t_c	C_r	C_s	C_d	μ	σ	μ	σ	μ	σ	μ	σ
2	2	5	2	11.23	2.76	11.37	2.63	11.04	2.65	11.23	2.45
2	5	5	2	11.40	2.81	11.34	2.62	10.91	2.64	11.10	2.53
2	2	10	2	11.35	2.75	11.38	2.96	10.43	2.06	10.80	2.45
2	5	10	2	11.47	3.15	11.46	2.71	11.03	2.48	12.77	2.76
2	2	5	5	12.46	4.94	12.59	4.57	12.50	4.27	12.38	4.30
2	5	5	5	13.58	5.23	12.52	5.24	12.82	5.40	12.81	5.08
2	2	10	5	12.35	5.37	12.91	4.96	12.27	4.25	12.15	4.03
2	5	10	5	13.09	5.06	12.23	4.53	12.36	4.77	12.38	5.43
2	2	5	10	15.42	8.53	15.39	7.98	13.77	7.31	14.83	8.81
2	5	5	10	15.40	7.96	15.49	8.85	16.23	9.29	14.09	7.34
2	2	10	10	16.69	8.93	15.68	8.75	16.15	8.61	14.60	8.03
2	5	10	10	13.97	8.56	16.08	9.34	15.04	9.40	15.17	8.41
5	2	5	2	10.96	2.45	11.37	3.07	11.37	2.84	11.00	2.41
5	5	5	2	11.43	2.51	11.23	2.65	11.34	2.63	10.87	2.32
5	2	10	2	10.68	2.16	11.30	2.80	11.39	2.67	11.34	2.61
5	5	10	2	11.23	2.39	12.74	2.76	11.27	2.58	10.79	2.56
5	2	5	5	12.89	5.36	13.03	5.47	13.01	5.22	13.05	5.76
5	5	5	5	13.49	5.14	12.34	5.24	12.94	4.85	12.56	4.52
5	2	10	5	13.11	5.95	12.06	5.01	12.44	4.42	12.41	4.13
5	5	10	5	13.30	5.67	11.64	3.60	11.90	4.22	12.56	4.86
5	2	5	10	15.15	8.15	14.81	8.87	14.70	8.10	16.08	9.99
5	5	5	10	15.71	8.80	14.04	7.04	16.47	10.00	16.02	10.27
5	2	10	10	14.87	7.48	16.28	9.32	15.29	7.93	15.62	10.09
5	5	10	10	16.34	9.08	15.57	8.18	14.25	7.50	15.06	8.40

Table C.7: $\psi = 1.0$ and $\alpha = 1.01$ case

	Costs			Cost Under Strategies							
				Strategy 1		Strategy 2		Strategy 3		Strategy 4	
t_c	C_r	C_s	C_d	μ	σ	μ	σ	μ	σ	μ	σ
2	2	5	2	12.61	6.09	13.46	5.39	37.82	43.67	12.45	4.89
2	5	5	2	11.74	5.21	16.47	8.51	48.26	51.62	13.80	5.22
2	2	10	2	11.96	5.64	13.53	6.05	43.53	53.73	12.32	5.47
2	5	10	2	11.91	5.00	16.22	8.80	44.13	56.92	13.40	5.19
2	2	5	5	14.09	8.40	15.54	8.84	56.11	90.43	16.77	9.44
2	5	5	5	15.06	11.07	17.45	10.45	73.17	105.52	16.67	11.59
2	2	10	5	14.41	10.21	16.40	11.30	89.85	126.24	14.87	8.70
2	5	10	5	15.83	9.96	17.69	10.17	71.38	109.65	17.65	11.11
2	2	5	10	20.88	15.71	19.42	14.36	142.40	276.65	18.83	14.90
2	5	5	10	19.78	16.77	23.93	19.02	162.58	290.55	17.68	12.93
2	2	10	10	19.87	19.46	19.96	16.00	132.31	208.05	17.79	16.66
2	5	10	10	20.82	19.90	26.70	19.27	68.83	130.30	25.98	24.74
5	2	5	2	12.31	6.53	12.54	5.04	41.68	42.84	11.94	4.70
5	5	5	2	12.88	6.10	15.32	9.31	37.37	45.68	13.85	5.74
5	2	10	2	13.10	6.18	12.86	5.37	44.48	54.00	12.12	5.54
5	5	10	2	12.90	6.93	14.27	8.72	38.48	45.67	13.64	5.96
5	2	5	5	15.41	11.28	14.54	9.57	40.17	69.80	14.62	10.03
5	5	5	5	14.70	10.35	17.40	9.38	62.94	120.98	14.80	7.62
5	2	10	5	14.72	9.45	14.59	8.73	65.14	112.39	14.80	9.49
5	5	10	5	14.03	9.34	17.51	10.43	53.29	70.79	15.62	8.40
5	2	5	10	20.10	17.77	20.21	21.02	113.77	228.97	18.50	18.11
5	5	5	10	21.23	21.47	22.19	16.79	136.30	259.87	22.50	19.63
5	2	10	10	19.13	18.39	17.92	16.83	124.43	194.56	18.79	16.01
5	5	10	10	19.81	20.45	20.09	17.09	122.94	268.56	17.95	14.69

Table C.8: $\psi = 1.0$ and $\alpha = 1.05$ case

	Costs			Cost Under Strategies							
				Strategy 1		Strategy 2		Strategy 3		Strategy 4	
t_c	C_r	C_s	C_d	μ	σ	μ	σ	μ	σ	μ	σ
2	2	5	2	11.60	4.68	13.52	6.89	14.99	16.14	11.65	5.25
2	5	5	2	12.16	5.91	13.42	8.43	20.70	32.09	12.17	5.30
2	2	10	2	12.92	6.51	11.68	4.64	20.29	46.23	12.12	5.56
2	5	10	2	12.69	6.47	13.20	7.18	20.22	26.27	12.52	5.33
2	2	5	5	15.87	10.20	15.47	9.06	26.41	41.32	15.57	10.56
2	5	5	5	14.42	11.78	16.29	10.34	28.12	53.28	15.20	10.95
2	2	10	5	16.41	12.58	14.83	8.79	24.26	43.95	13.58	8.01
2	5	10	5	15.57	11.26	16.61	11.65	20.97	34.47	16.42	12.98
2	2	5	10	17.56	14.78	21.73	18.29	32.57	62.32	22.66	21.15
2	5	5	10	17.60	13.05	18.20	18.25	65.73	169.63	19.60	15.77
2	2	10	10	19.96	18.14	15.90	14.04	35.85	72.50	21.28	18.59
2	5	10	10	18.18	15.79	21.23	18.54	36.28	104.07	20.11	19.17
5	2	5	2	13.01	6.69	11.64	5.39	15.84	19.22	12.03	5.28
5	5	5	2	11.21	4.86	12.61	6.05	17.97	24.03	12.42	5.40
5	2	10	2	11.18	4.73	11.85	4.85	23.83	42.76	12.26	5.42
5	5	10	2	11.61	4.54	13.14	6.55	20.19	23.87	12.75	5.70
5	2	5	5	14.75	10.49	14.72	9.95	25.52	29.39	14.57	9.05
5	5	5	5	14.27	8.51	15.64	11.93	20.39	35.81	15.62	9.82
5	2	10	5	14.60	8.85	15.03	8.89	24.29	39.52	13.97	8.92
5	5	10	5	13.92	9.31	13.52	8.16	33.87	56.86	15.21	10.20
5	2	5	10	20.19	18.13	16.83	14.47	44.95	150.58	19.28	17.81
5	5	5	10	20.33	18.63	20.28	17.75	41.13	99.63	20.65	17.60
5	2	10	10	21.78	19.64	17.16	14.05	43.77	114.16	20.17	19.60
5	5	10	10	20.86	19.02	22.17	15.76	33.86	59.28	21.22	17.66

Table C.9: $\psi = 1.0$ and $\alpha = 1.10$ case

	Costs			Cost Under Strategies							
				Strategy 1		Strategy 2		Strategy 3		Strategy 4	
t_c	C_r	C_s	C_d	μ	σ	μ	σ	μ	σ	μ	σ
2	2	5	2	12.80	5.63	11.46	4.35	11.60	4.89	11.96	4.69
2	5	5	2	11.43	4.83	12.51	5.29	12.20	5.30	12.37	5.96
2	2	10	2	12.18	5.69	11.78	4.75	13.34	15.17	11.91	5.34
2	5	10	2	12.21	5.31	11.41	5.22	14.49	14.74	12.04	5.57
2	2	5	5	14.34	11.29	14.85	9.43	16.64	20.28	14.82	9.31
2	5	5	5	16.92	12.04	15.10	12.14	16.51	19.65	15.04	11.01
2	2	10	5	16.02	12.00	15.08	10.49	16.22	19.94	15.32	11.78
2	5	10	5	15.87	11.72	14.54	9.82	16.22	12.15	14.29	11.40
2	2	5	10	21.72	24.04	19.62	17.80	29.34	48.01	21.07	21.09
2	5	5	10	20.62	15.98	20.44	16.99	26.65	36.29	19.96	17.00
2	2	10	10	21.08	19.49	20.02	20.89	23.91	33.70	18.62	15.46
2	5	10	10	19.42	16.87	20.42	21.00	17.20	14.00	19.94	17.71
5	2	5	2	13.30	6.38	12.17	5.32	13.80	9.14	12.00	5.19
5	5	5	2	11.54	4.89	12.01	5.39	13.69	14.43	11.51	4.48
5	2	10	2	11.76	4.19	12.36	5.46	11.59	5.38	11.83	5.23
5	5	10	2	12.31	6.04	11.94	4.85	14.74	13.76	11.77	4.70
5	2	5	5	15.15	9.37	16.08	10.72	14.60	10.04	14.61	9.23
5	5	5	5	14.14	7.54	15.52	10.74	18.09	13.18	14.71	10.18
5	2	10	5	15.31	10.56	15.05	10.12	14.90	12.12	14.39	9.59
5	5	10	5	14.95	9.56	15.05	10.01	18.20	14.28	16.05	10.01
5	2	5	10	18.41	18.65	22.38	18.99	24.55	25.07	18.47	15.26
5	5	5	10	20.64	18.92	16.60	14.77	22.62	36.73	20.91	18.85
5	2	10	10	22.67	21.78	20.60	19.14	27.83	65.70	17.33	17.63
5	5	10	10	21.68	16.84	18.74	17.17	21.14	18.09	16.80	17.00

Appendix D

Data Requirements For SPD Network Generator

The following inputs are required by the SPD network generator (the number in square brackets at the end of each input is the number of input parameters required):

- minimal (maximal) number of activities; denoted by J^{min} (J^{max}), [2]
- minimal (maximal) number of modes to be used by the activities; denoted by M^{min} (M^{max})[1], [2]
- minimal (maximal) duration of the activities; denoted by d^{min} (d^{max}), [2]
- minimal (maximal) number of "start activities;" *i.e.*, activities with no predecessors; denoted by S_1^{min} (S_1^{max}), [2]
- minimal (maximal) number of "finish activities;" *i.e.,* activities with no successors; denoted by P_J^{min} (P_J^{max}), [2]
- maximal number of successor and predecessor activities to each activity j, denoted by S_j^{max} and P_j^{max} $j = 2, \cdots, n-1$; respectively, [2]
- *network complexity* C as specified in section 4.1, [1]
- tolerated *complexity deviation* ϵ_{net}; a measure of the permissible deviation from the stated complexity C expressed as a multiplier of C; typically $0 < \epsilon_{net} < 1$, [1]

[1] A "mode" is a euphemism for a (duration, resource utilization) pair. The implicit assumption is that the longer duration would require less resource.

- number of renewable resources $|R|$, and for each resource $r \in R$ the analyst must specify the number of modes available for its processing, which may be different from the number of modes accessible to each activity, $[|R|+1]$
- number of non-renewable resources $|N|$, and for each resource $r \in N$ the analyst must specify the number of modes available for its processing, which may be different from the number of modes accessible to each activity, $[|N|+1]$
- number of doubly-constrained resources$|D|$, and for each resource $r \in D$ the analyst must specify the number of modes available for its processing, which, again, may be different from the number of modes accessible to each activity, $[|D|+1]$
- minimal (maximal) number of resources of category $\tau \in \{R, N, D\}$, denoted by $|\tau|^{min}$ ($|\tau|^{max}$), $[2\,(|R|+|N|+|D|)]$
- minimal (maximal) number of resources of category $\tau \in \{R, N, D\}$ used by job-mode $[j, m]$; denoted by Q_τ^{min} (Q_r^{max}), $[2\,(|R|+|N|+|D|)]$
- minimal (maximal) number of resources of category $\tau \in \{R, N, D\}$; denoted by U_τ^{min} (U_τ^{max}), $[2\,(|R|+|N|+|D|)]$
- the *resource factor* of resources of category $\tau \in \{R, N, D\}$, denoted by RF_τ,[2]
  $[(|R|+|N|+|D|)]$

[2]The *resource factor* reflects the average portion of resources requested per activity. $RF = 0$ indicates that the activities require no resource; $RF = 1$ indicates that each job requires all the resources. For the single mode case, let k_{jr} denote the quantity required by activity j of resource r. Then RF is given by

$$RF = \frac{1}{|R|(J-2)} \sum_{j=2}^{J-1} \sum_{r \in R} \begin{cases} 1 & \text{if } k_{jr} > 0 \\ 0 & \text{otherwise} \end{cases}$$

For the multi-mode case the expression is generalized straightforwardly to

$$RF_\tau = \frac{1}{|\tau|(j-2)} \sum_{j=2}^{J-1} \frac{1}{M_j} \sum_{m=1}^{M_j} \sum_{r \in \tau} \begin{cases} 1 & \text{if } k_{jmr} > 0 \\ 0 & \text{otherwise} \end{cases}$$

where, as expected, k_{jmr} is the quantity required by activity j in mode m of resource r.

- the *resource scarceness* of category $\tau \in \{R, N, D\}$, denoted by RS_τ. It is a measure of the "tightness" of the resource availability *vis-a-vis* the demands for the resource. Its use requires the definition of two other parameters: K_r^{min} and K_r^{max} for *each* resource $r \in \tau$. Let the availability of resource r be denoted by K_r. Then K_r it is given by

$$K_r = K_r^{min} + RS_\tau \, (K_r^{max} - K_r^{min})$$

  $[(|R| + |N| + |D|)]$
- tolerated *resource factor deviation* ϵ_{RF}; a measure of the permissible deviation from the stated RF_τ; $\tau \in \{R, N, D\}$, expressed as a multiplier of RF_τ; typically $0 < \epsilon_{RF} < 1$. $[(|R| + |N| + |D|)]$

Appendix E

GRAPH1.F: Source Code

```
c***
c   Program to generate random graphs
c   of given complexity Index
c***
      integer ci,node,minnode,arc,minarc,maxarc
      integer nr,ar,nk,l1,l2,l3,counter,ni,nj,insnod,insarc
      integer tnode,tinsarc,tranval,upi,upj,links
      integer m(100,100),renumber(100),choice
      integer updatedm(100,100),testnj(100),testni(100)
      integer ihrs,imins,isecs,i100ths,ihrt,imint,isect,i100tht
      real elapstim,param(500,10),minres,maxres,restr
      open(unit=21,file='dagen.net')
      open(unit=22,file='dagen.mat')
      maxn=100
c***
c   Data Entry
c***
      write(*,1000)
1000  format(1x,'Please input the Complexity Index  <= 97')
```

```
      write(*,1001)
1001  format(1x,'Complexity Index = ')
      read(*,*) ci
      minnode=ci+3
      write(*,1005)minnode
      read(*,*) node
      minarc=2*node-3
      maxarc=2*(node-minnode)+(3*ci+2)+(ci*(ci-1)/2)+1
      minres=2*float(minarc)/float(node*(node-1))
      maxres=2*float(maxarc)/float(node*(node-1))
      write(*,1006)
      read(*,*)choice
      if(choice.eq.1) then
write(*,1010)minarc,maxarc
read(*,*)arc
      else
write(*,1007)minres,maxres
read(*,*)restr
arc=int(0.5*node*(node-1)*restr)
      endif
1010  format(1x,'Please input Number of arcs  >= ',i3,5x,' =< ',i3)
1005  format(1x,'Please input Number of Nodes  >=',i3,5x,' =< 100')
1006  format(1x,'Specify 1. Number of arcs   OR   2. Restrictiveness')
1007  format(1x,'Please input restrictiveness >=',f6.4,5x,'=<',f6.4)
c
c     error checks for input
c
      if((node.lt.minnode).or.(node.gt.maxn))then
      write(6,1106)
```

```
1106  format(1x,'Wrong Input of Number of Nodes')
      stop
      endif
      if((arc.lt.minarc).or.(arc.gt.maxarc))then
      write(6,1107)
1107  format(1x,'Wrong Input of Number of Arcs')
      stop
      endif
c
      call system_clock(isecs)
c***
c   Construct Linesr Skeleton of complexity Index ci
c***
      do 100 i=1,maxn
      do 105 j=1,maxn
      m(i,j)=0
      updatedm(i,j)=0
105   continue
      renumber(i)=0
100   continue
      nk=ci+3
      do 110 i=1,nk-2
      m(i,i+1)=1
      m(i,i+2)=1
110   continue
  m(nk-1,nk)=1
c***
c   nr=remaining nodes
c   ar=remaining arcs
```

```
c   nk=nodes in skeleton
c***
      nr=node-nk
      ar=arc-(2*ci+3)
      if(ar.le.1) then
      if(ar.eq.1)m(1,node)=1
      do 180 i=1,node
      do 190 j=1,node
      updatedm(i,j)=m(i,j)
190   continue
180   continue
      goto 500
      endif
c***
c   Select two nodes ni,nj for insertion
c   The seed for the random number is the 100th
c   of the second of the system clock
c***
200   tnode=node-nr
      call genrand(ni,tnode)
201   call genrand(nj,tnode)
c***
c       To check if an arc exists between ni and nj
c***
      if(ni.eq.nj) goto 201
c***
c     if only one arc need to be inserted
c***
      if((nr.eq.0).and.(ar.gt.0)) then
```

```
      do 610 j=1,node
        testni(j)=0
610   continue
      do 620 i=1,nk-1
630     call grandij(ni,1,nk-1)
        counter=0
        do 631 j=1,nk-1
          if(testni(j).ne.0)counter=counter+1
631     continue
        if(counter.ge.nk-1) then
        write(6,1105) ar
1105    format(1x,'Sorry Could not insert ',i3,' arcs')
        goto 401
        endif
        if((testni(ni).eq.1).or.(ni.ge.nk)) goto 630
        testni(ni)=1
        do 621 j=1,nk
          testnj(j)=0
621     continue
        counter=nk-ni
        do 650 while(counter.ne.0)
640       call grandij(nj,ni,nk)
          if((testnj(nj).eq.1).or.(nj.gt.nk)) goto 640
          testnj(nj)=1
          counter=counter-1
          if(m(ni,nj).eq.1) goto 650
          call transit(ni,nj,m,tnode,tranval)
          if(tranval.eq.0) goto 650
          ar=ar-1
```

```
          m(ni,nj)=1
          if(ar.le.0) goto 401
650     continue
620   continue
      endif
c***
c     if all solutions are possible
c***
      if(ar.le.0) goto 401
      if(m(ni,nj).eq.0) goto 300
c***
c       Inserting a unit if an arc bewtween ni and nj exists
c       insnod=node number to be inserted
c       insarc=number of arcs to be inserted
c***
      if((nr.le.0).or.(ar.lt.nr)) goto 200
      insnod=node-nr+1
      nr=nr-1
      m(ni,insnod)=1
      m(insnod,nj)=1
      m(ni,nj)=0
      if(ar.ge.(nr+2)) then
      call genrand(insarc,2)
         if(insarc.eq.2) then
         m(ni,nj)=1
         ar=ar-2
         else
         ar=ar-1
         endif
```

```
      else
      ar=ar-1
      endif
      goto 200
c***
c       inserting a unit if no arc between ni and nj
c***
300   continue
c
c       to check if ni and nj are inside the skeleton
c
      if((ni.ge.nk).or.(nj.gt.nk)) goto 200
c
c       check if transitive relation between ni and nj exists
c
      call transit(ni,nj,m,tnode,tranval)
      if(tranval.eq.0) goto 200
c
c       checking the three cases
c       1.  if all three units are possible
c       2.  if only series and one arc is possible
c       3.  if only one arc is possible
c
      if((ar.ge.3).and.(nr.le.(ar+2)).and.(nr.ge.1)) goto 310
      if((ar.eq.2).and.(nr.le.(ar+1)).and.(nr.ge.1)) goto 320
      if((ar.ge.1).and.(nr.le.ar)) goto 330
      goto 200
c
c       case 1
```

```
c
310   continue
      insnod=node-nr+1
      call genrand(tinsarc,3)
      if(tinsarc.eq.1) then
      nr=nr-1
      ar=ar-3
      call nulpar(ni,nj,insnod,m)
      else
         if(tinsarc.eq.2) then
         nr=nr-1
         ar=ar-2
         call nulser(ni,nj,insnod,m)
         else
         ar=ar-1
         call nulone(ni,nj,m)
         endif
      endif
      goto 200
c
c       case 2
c
320   continue
      insnod=node-nr+1
      call genrand(tinsarc,2)
      if(tinsarc.eq.1) then
      nr=nr-1
      ar=ar-2
      call nulser(ni,nj,insnod,m)
```

```
      else
      ar=ar-1
      call nulone(ni,nj,m)
      endif
      goto 200
c
c       case 3
c
330   continue
      ar=ar-1
      call nulone(ni,nj,m)
      goto 200
c***
c       renumber the nodes to follow the standard convention
c***
401   continue
      renumber(1)=1
      counter=1
      do 400 l1=1,10000
        do 410 l2=2,node
        if(renumber(l2).ne.0) goto 410
        do 420 l3=1,node
          if(m(l3,l2).eq.0) goto 420
          if(renumber(l3).eq.0) goto 410
420     continue
        counter=counter+1
        renumber(l2)=counter
410     continue
      if(renumber(nk).ne.0) goto 490
```

```
400   continue
c
c       update m
c
490   do 460 l1=1,node
        do 470 l2=1,node
        if(m(l1,l2).eq.0) goto 470
        upi=renumber(l1)
        upj=renumber(l2)
        updatedm(upi,upj)=m(l1,l2)
470     continue
460   continue
c***
500   call netdata(updatedm,node,links,param)
      call system_clock(isect)
      elapstim=(real(isect)-real(isecs))/100
      restr=2*arc/(node*(node-1))
      write(21,1100)ci,node,arc,restr,elapstim
      write(22,*)node
1100  format(1x,'ci=',i3,'; nodes=',i3,'; arcs=',i3,
     +      'Restrictiveness=  ',f6.4,'; ex time in sec=',f10.2)
       do 530 i=1,node
       write(22,*) (updatedm(i,j),j=1,node)
530    continue
      write(21,1115)
1115  format(1x,'Arc',5x,'From',5x,'To',7x,'Dur',8x,'Cost',5x,
     +'Res1',5x,'Res2',5x,'Res3',5x,'Res4',5x,'Res5',5x,
     +'Non-Ren Res')
      do 510 i=1,links
```

```
      write(21,1110)i,(param(i,j),j=1,10)
510   continue
1110  format(1x,i3,5x,f4.0,7x,f4.0,4x,f6.3,2x,f6.3,5(2x,f6.3))
      close(unit=22)
      close(unit=21)
      stop
      end
c****
c      subroutine to generate a random number
c      between 1 and n
c****
      subroutine genrand(intrand,n)
      integer n,intrand,isec
      real realrand
5000  call system_clock(isec)
      call random_seed(isec)
      call random_number(realrand)
      if(realrand.le.0.01)then
        intrand=nint(realrand*n*100)+1
      else
        intrand=nint(realrand*n)+1
      endif
      if((intrand.gt.n).or.(intrand.lt.1)) goto 5000
      return
      end
c****
c      subroutine to generate a random number between ni and nj
c****
      subroutine grandij(intrand,ni,nj)
```

```
      integer ni,nj,intrand,isec
      real realrand
5010  call system_clock(isec)
      call random_seed(isec)
      call random_number(realrand)
      if(realrand.le.0.01)then
        intrand=nint(realrand*(nj-ni+1)*100)+ni
      else
        intrand=nint(realrand*(nj-ni+1))+ni
      endif
      if((intrand.gt.nj).or.(intrand.lt.ni)) goto 5010
      return
      end
c****
c       subroutine to check the transitive relationship
c****
      subroutine transit(ni,nj,m,tnode,tranval)
      integer ni,nj,tnode,tranval,l1,l2,l3,l4,counter
      integer m(100,100),transi(100),transj(100)
      do 2000 l1=1,tnode
      transi(l1)=0
      transj(l1)=0
2000  continue
      transi(ni)=1
      tranval=0
      do 2010 l1=1,tnode
      counter=0
        do 2020 l2=1,tnode
          do 2030 l3=1,tnode
```

```
          if(transi(l3).eq.0) goto 2030
          if(m(l3,l2).eq.0) goto 2030
          if(l2.eq.nj) then
            tranval=1
            return
          else
            transj(l2)=1
            counter=1
          endif
2030      continue
2020    continue
      if(counter.eq.0) then
        tranval=0
        return
      else
        do 2040 l4=1,tnode
        transi(l4)=transj(l4)
        transj(l4)=0
2040    continue
      endif
2010  continue
      if(l1.eq.tnode) write(*,1060)
1060  format(1x,'error in transitive relationship')
      return
      end
c****
c       subroutine to insert parallel unit if no arc
c       between ni and nj exists.
c****
```

```
      subroutine nulpar(ni,nj,insnod,m)
      integer ni,nj,insnod,m(100,100)
      m(ni,nj)=1
      m(ni,insnod)=1
      m(insnod,nj)=1
      return
      end
c****
c       subroutine to insert series unit if no arc
c       between ni and nj exists.
c****
      subroutine nulser(ni,nj,insnod,m)
      integer ni,nj,insnod,m(100,100)
      m(ni,insnod)=1
      m(insnod,nj)=1
      return
      end
c****
c       subroutine to insert one arc if no arc
c       between ni and nj exists.
c****
      subroutine nulone(ni,nj,m)
      integer ni,nj,m(100,100)
      m(ni,nj)=1
      return
      end
c****
c Subroutine to generate the network parameters
c****
```

```
      subroutine netdata(updatedm,node,links,param)
      integer node,updatedm(100,100),i,j,k,links,maxlink
      integer nres
      real param(500,10),mindur,maxdur,costini,costprop,olddur
      real dur,res,minrenew,maxrenew,totnonrw,nonrw,oldres
c
c   get input from user
c
      write(*,6500)
6500  format(1x,'Please provide following information:'/
     +1x,'Minimum possible Duration'/
     +1x,'Maximum possible Duration'/
     +1x,'Initial setup cost'/
     +1x,'Proportional cost'/
     +1x,'Total Non-Renewable Resource'/
     +1x,'Minimum Renewable Resource'/
     +1x,'Maximum Renewable Resource')
c
      read(*,*) mindur,maxdur,costini,costprop,totnonrw,
     +minrenew,maxrenew
c
c   Assign arc numbers
c
      maxlink=500
      links=0
      call rrandxy(olddur,mindur,maxdur)
      do 6100 i=1,node
  do 6110 j=1,node
  if(updatedm(i,j).eq.0) goto 6110
```

```
  links=links+1
  do 6120 k=1,10
   param(links,k)=0.0
6120  continue
  param(links,1)=i
  param(links,2)=j
c
c duration and cost
c
6210  call rrandxy(dur,mindur,maxdur)
  if(abs(dur-olddur).le.0.01) goto 6210
  olddur=dur
  param(links,3)=dur
  param(links,4)=costini-dur*costprop
c
c renewable
c
  call genrand(nres,5)
  do 6130 k=5,4+nres
6220  call rrandxy(res,minrenew,maxrenew)
    if(abs(res-oldres).le.0.01) goto 6220
    oldres=res
    param(links,k)=res
6130   continue
c
c   non-renewable
c
  nonrw=totnonrw
  if(nonrw.gt.0) then
```

```
6230    call rrandxy(res,0.0,nonrw)
   if(abs(res-oldres).le.0.01) goto 6230
  else
   res=0.0
  endif
  oldres=res
  param(links,10)=res
  if(links.eq.maxlink) then
    write(*,6510)
6510     format(1x,'Too many arcs...Terminating')
    stop
  endif
6110 continue
6100  continue
      return
      end
c****
c      subroutine to generate a real random number between x and y
c****
      subroutine rrandxy(realrand,x,y)
      integer isec
      real realrand
5020  call system_clock(isec)
      call random_seed(isec)
      call random_number(realrand)
      if(realrand.le.0.01) realrand=realrand*(y-x)*100+x
      if((realrand.gt.y).or.(realrand.lt.x)) goto 5020
      return
      end
```

Appendix F

CINDEX.F: Source Code

```
c***
c       Program to calculate complexity index of a network
c***
      integer i,j,k,domnod,domprev,maxn,totnod,counter,tranval
      integer compind,rednode(100)
      integer m(100,100),dom(100),revdom(100),origm(100,100)
      integer row(100),col(100),admit(100,100)
c***
c       read data
c***
      open(unit=22,file='dagen.mat')
      open(unit=23,file='dagen.ci')
c      write(*,1000)
c1000  format(1x,'maximum number of nodes (=< 100) ???')
      read(22,*)maxn
      read(22,*)((origm(i,j),j=1,maxn),i=1,maxn)
      write(*,1008)
1008   format(1x,'Please wait while program executes')
```

```
c***
c       initialize matrices
c***
      do 100 i=1,maxn
      dom(i)=0
      revdom(i)=0
      row(i)=0
      col(i)=0
      do 110 j=1,maxn
      m(i,j)=origm(i,j)
      admit(i,j)=-1
110   continue
100   continue
c***
c       calculate dominator tree
c***
      dom(1)=1
      dom(maxn)=1
      do 200 k=1,maxn-2
      totnod=2
      do 210 j=2,maxn-1
      if(dom(j).gt.0) then
        totnod=totnod+1
        goto 210
      else if(m(1,j).eq.1) then
        dom(j)=1
        totnod=totnod+1
        goto 210
      endif
```

```
      counter=0
      do 220 i=1,j
      if(m(i,j).eq.1) then
        counter=counter+1
        domnod=i
      endif
220   continue
      if(counter.eq.1)then
        dom(j)=domnod
        totnod=totnod+1
        goto 210
      else
        domprev=dom(domnod)
        if(domprev.ne.0) then
          m(domnod,j)=0
          m(domprev,j)=1
        endif
      endif
210   continue
      if(totnod.eq.maxn) goto 230
200   continue
c***
c     calculate reverse dominator tree
c***
230   do 240 i=1,maxn
      do 250 j=1,maxn
      m(i,j)=origm(i,j)
250   continue
240   continue
```

```
      revdom(1)=maxn
      revdom(maxn)=maxn
      do 300 k=1,maxn-2
      totnod=2
      do 310 j=maxn-1,2,-1
      if(revdom(j).gt.0) then
        totnod=totnod+1
        goto 310
      else if(m(j,maxn).eq.1) then
        revdom(j)=maxn
        totnod=totnod+1
        goto 310
      endif
      counter=0
      do 320 i=maxn,j,-1
      if(m(j,i).eq.1) then
        counter=counter+1
        domnod=i
      endif
320   continue
      if(counter.eq.1)then
        revdom(j)=domnod
        totnod=totnod+1
        goto 310
      else
        domprev=revdom(domnod)
        if(domprev.ne.0) then
          m(j,domnod)=0
          m(j,domprev)=1
```

```
        endif
      endif
310   continue
      if(totnod.eq.maxn) goto 400
300   continue
c***
c       Generate complexity graph
c***
400   do 410 i=1,maxn
      do 420 j=1,maxn
      m(i,j)=origm(i,j)
420   continue
410   continue
      do 430 i=1,maxn
      do 440 j=1,maxn
      if(i.eq.1) then
        m(i,j)=0
        goto 440
      elseif(j.eq.maxn)then
        m(i,j)=0
        goto 440
        elseif(j.le.dom(i)) then
          m(j,i)=0
          goto 440
          elseif(j.ge.revdom(i))then
            m(i,j)=0
            goto 440
            elseif(m(i,j).eq.1)then
              goto 440
```

```
              else
              call transit(i,j,m,maxn,tranval)
              m(i,j)=tranval
              endif
440   continue
430   continue
c***
c       calculate max flow
c         labeling scheme for row and columns
c              -1    for dash ( - )
c               0    for unlabeled
c               #    for label number
c***
c     Initialize
c***
      do 450 i=1,maxn
      do 460 j=1,maxn
        if(m(i,j).eq.1) admit(i,j)=0
        if(row(i).eq.-1) goto 450
        if(col(j).eq.-1) goto 460
        if(m(i,j).eq.0) goto 460
        admit(i,j)=1
        row(i)=-1
        col(j)=-1
460   continue
450   continue
c***
c     Labeling rows with dashes (or -1 here)
c***
```

```
470   do 480 i=1,maxn
        row(i)=0
        col(i)=0
        do 490 j=1,maxn
          if(admit(i,j).eq.1) goto 480
490     continue
        row(i)=-1
480   continue
c***
c     labeling columns with numbers
c***
      do 500 i=1,maxn
        if(row(i).ne.-1) goto 500
        do 510 j=1,maxn
          if((admit(i,j).eq.0).and.(col(j).eq.0)) col(j)=i
510     continue
500   continue
c***
c     label row with numbers
c***
      do 520 j=1,maxn
        if(col(j).eq.0) goto 520
        do 530 i=1,maxn
          if((row(i).eq.-1).or.(row(i).gt.0)) goto 530
          if(admit(i,j).le.0) goto 530
          row(i)=j
530     continue
520   continue
c***
```

```
c     checking "breakthrough" condition
c***
      do 540 i=1,maxn
        if(row(i).lt.0) goto 540
        do 550 j=1,maxn
          if(admit(i,j).eq.1) goto 540
550     continue
        goto 560
540   continue
c***
c     Non-Breakthrough, maximal flow
c***
      compind=0
      do 600 i=1,maxn
        if(row(i).eq.0) then
          rednode(i)=1
          compind=compind+1
        endif
600   continue
      do 610 i=1,maxn
        if(col(i).eq.0) goto 610
        rednode(i)=1
        compind=compind+1
610   continue
      goto 900
c***
c     Breakthrough, reassign the flow and re-label
c***
560   admit(i,j)=1
```

```
      j=row(i)
      admit(i,j)=0
      i=col(j)
      admit(i,j)=1
      goto 470
c***
c     write results
c***
900   continue
      write(23,1010)compind
      do 910 i=1,maxn
        if(rednode(i).eq.0) goto 910
        write(23,1020) i
910   continue
1010  format(1x,'complexity Index = ',i3)
1020  format(1x,'Reduce Node   ',i3)
      stop
      end
c****
c       subroutine to check the transitive relationship
c****
      subroutine transit(ni,nj,m,tnode,tranval)
      integer ni,nj,tnode,tranval,l1,l2,l3,l4,counter
      integer m(100,100),transi(100),transj(100)
      do 2000 l1=1,tnode
      transi(l1)=0
      transj(l1)=0
2000  continue
      transi(ni)=1
```

```
      tranval=0
      do 2010 l1=1,tnode
      counter=0
        do 2020 l2=1,tnode
          do 2030 l3=1,tnode
          if(transi(l3).eq.0) goto 2030
          if(m(l3,l2).eq.0) goto 2030
          if(l2.eq.nj) then
            tranval=1
            return
          else
            transj(l2)=1
            counter=1
          endif
2030      continue
2020    continue
      if(counter.eq.0) then
        tranval=0
        return
      else
        do 2040 l4=1,tnode
        transi(l4)=transj(l4)
        transj(l4)=0
2040    continue
      endif
2010  continue
      if(l1.eq.tnode) write(23,2060)
2060  format(1x,'error in transitive relationship')
      return
```

```
      end
c****
```

Appendix G

Instructions For Running The Program

The FOTRAN source code for the two prorams is supplied with this report. An executable code for SUN operating system is also provided. To run the programs user can type the file name at the command prompt and follow the instructions. Users can recompile the programs to suit their specific needs.

The files include:

- "graph1.f" - source code for graph generation program; includes parameter generation module.
- "graph1" - executable file for "graph1.for" for unix systems.
- "cindex.f" - source code for *CI* analysis program.
- "cindex" - executable file for "cindex.for" for unix systems.

The files generated are:

- "dagen.net" - the detailed output file.
- "dagen.mat" - the adjacency matrix of the net.
- "dagen.ci" - the results of complexity index calculations.

Details of file "dagen.net"

The output includes the following:

- The number of nodes and arcs in the network
- The number of arcs which could not be inserted in a preset time.

- The restrictiveness.
- The time taken to generate network in 100th of seconds. This time does not include the time for input output or for the generation of other network parameters.
- Details of network parameters such as duration, cost, and resources.
- Network statistics

Details of file "dagen.mat"

Line 1: Number of nodes n

Lines 2 to "n+1": Adjacency matrix

The files generated are:

- "(filename).net" - the detailed output file.
- "(filename).mat" - the adjacency matrix of the net.
- "(filename).ci" - the results of complexity index calculations.
- "network.dat" - a temporary file that contains the values of the network parameter which can be used for different runs. The details of this file are given next.

Details of file "network.dat"

This file contains only one line with the following information separated with spaces:

Activity Min Possible Duration
Activity Max Possible Duration
Activity Min Setup Cost
Activity Max Setup Cost
Activity Min Proportional Cost
Activity Max Proportional Cost
Project Min Non-renewable Resource
Project Max Non-renewable Resource

Activity Min Renewable Resource

Activity Min Renewable Resource

Details of file "(filename).net"

The output includes the following:

The number of nodes and arcs in the network

The number of arcs which could not be inserted in a preset time.

The time taken to generate network in 100th of seconds. This time does not include the time for input output or for the generation of other network parameters.

Details of network parameters such as duration, cost, and resources.

Network statistics

Details of file "(filename).mat"

Line 1: Number of nodes n

Lines 2 to "n+1": Adjacency matrix

Appendix H

Sample Input and Output

DAGEN.MAT: The adjacency matrix of the network

```
10
0  1  1  1  0  0  1  0  1  1
0  0  1  0  0  0  0  0  0  0
0  0  0  1  1  0  1  0  0  0
0  0  0  0  0  1  0  0  1  1
0  0  0  0  0  0  1  0  0  0
0  0  0  0  0  0  1  0  0  0
0  0  0  0  0  0  0  1  1  1
0  0  0  0  0  0  0  0  1  0
0  0  0  0  0  0  0  0  0  1
0  0  0  0  0  0  0  0  0  0
```

DAGEN.NET: The network with resources

ci= 3; nodes= 10; arcs= 20; Restrictiveness= .4444; ex time in sec= 24.75

Arc	From	To	Dur	Cost	Res1	Res2	Res3	Res4	Res5	Non-Ren Res
1	1	2	7.678	96.161	48.285	.000	.000	.000	.000	.496
2	1	3	33.538	83.231	94.128	24.917	64.714	39.059	.000	.266
3	1	4	26.670	86.665	96.471	60.294	.000	.000	.000	.921
4	1	7	5.181	97.409	73.071	24.163	12.478	44.366	.000	.176
5	1	9	26.500	86.750	24.979	10.352	56.634	.000	.000	.087
6	1	10	23.910	88.045	38.972	69.007	72.642	21.709	.000	.866
7	2	3	48.041	75.980	31.032	28.313	10.056	27.746	81.153	.875
8	3	4	36.012	81.994	44.735	59.405	76.036	60.801	76.675	.512
9	3	5	11.095	94.453	69.496	83.162	27.886	90.644	24.204	.526
10	3	7	30.462	84.769	45.209	.000	.000	.000	.000	.750
11	4	6	6.098	96.951	74.635	10.830	.000	.000	.000	.550
12	4	9	36.312	81.844	46.855	34.663	51.955	24.036	.000	.212
13	4	10	16.131	91.934	65.568	55.262	75.257	49.584	.000	.921
14	5	7	9.598	95.201	79.666	78.894	.000	.000	.000	.655
15	6	7	48.714	75.643	98.274	11.912	.000	.000	.000	.571
16	7	8	44.343	77.828	17.655	.000	.000	.000	.000	.295
17	7	9	5.018	97.491	68.952	.000	.000	.000	.000	.089
18	7	10	49.780	75.110	61.576	87.159	.000	.000	.000	.090
19	8	9	6.758	96.621	42.345	55.383	49.557	59.466	.000	.374
20	9	10	18.909	90.545	45.000	13.632	35.579	10.833	87.441	.617

List of References

[1] M.K. Agrawal and S.E. Elmaghraby. On the convolution of piecewise polynomial distributions. Technical Report OR Report No. 313, North Carolina State University, 1996.

[2] M.K. Agrawal, S.E. Elmaghraby, and W.S. Herroelen. On the generation of testsets for project activity nets. In *Proceedings of 4th International Workshop on Project Management and Scheduling, Leuven, Belgium, July 12-15*, pages 202–220, 1994.

[3] M.K. Agrawal, S.E. Elmaghraby, and W.S. Herroelen. Dagen: A generator of testsets for project activity nets. *European Journal of Operational Research*, 90:376–382, 1996.

[4] W.W. Bein, J. Kamburowski, and M.F.M. Stallmann. Optimal reduction of two-terminal directed acyclic graphs. *Siam Journal of Computing*, 21:1112–1129, 1992.

[5] C.D. Boor. *A Practical Guide to Splines.* Springer-Verlag, New York, N.Y., 1978.

[6] J. M. Burt Jr. and M.B. Garman. Conditional monte carlo: A simulation technique for stochastic network analysis. *Management Science*, 18(3):207–217, 1971.

[7] A. Charnes, W. W. Cooper, and G. L. Thompson. Critical path analysis via

chance constrained and stochastic programming. *Operations Research*, 12:629–632, 1964.

[8] C. E. Clark. The greatest of a finite set of random variables. *Operations Research*, 9:145–162, 1961.

[9] C. T. Clingen. A modification of fulkersons PERT algorithm. *Operations Research*, 12(4):629–632, 1964.

[10] D.F. Cooper. Heuristics for scheduling resource-constrained projects: An experimental investigation. *Management Science*, 22:1186–1194, 1976.

[11] Dar-El. MALB-A heuristic technique for balancing large single-model assembly lines. *AIIE Transactions*, 5:343–356, 1973.

[12] B. De Reyck and W.S. Herroelen. On the use of the complexity index as a measure of complexity in activity networks. Technical Report Res. Report No. 9332, Department of Applied Economics, Katholieke Universiteit Leuven, Debériotstraat 36, B-3000 Leuven, Belgium, 1993.

[13] E. Demeulemeester, B. Dodin, and W. S. Herroelen. A random activity network generator. *Oparations Research*, 41:972–980, 1993.

[14] L. P. Devroye. Inequalities for the completion times of the stochastic PERT networks. *Mathematics of Operations Research*, 4:441–447, 1979.

[15] P. Dierckx. *Curve and Surface Fitting with Splines*. Oxford Science Publications, 1993.

[16] B. Dodin. Bounding the project completion time distribution in PERT networks. *Operations Research*, 33:862–881, 1985.

[17] B. Dodin and M Sirvanci. Stochastic networks and extreme value distribution. *Computers and Operations Research*, 17(4):397–409, 1990.

[18] S. E. Elmaghraby. Activty nets: A guided tour through some recent developments. *European journal of operational research*, 82:383–408, 1995.

[19] S.E. Elmaghraby. On the expected value of PERT type networks. *Management Science*, 13(5):299–306, 1967.

[20] S.E. Elmaghraby. *Activity Networks*. Wiley, 1977.

[21] S.E. Elmaghraby. The estimation of some network parameters in the PERT model of activity networks: review and critique. In R. Slowinski and J. Weglarz, editors, *Advances in Project Scheduling*, chapter 1, pages 371–432. Elsevier, New York, NY, 1989.

[22] S.E. Elmaghraby. Optimal resource allocation via dynamic programming in activity networks. *European Journal of Operational Research*, 64:1–17, 1993.

[23] S.E. Elmaghraby. Bounding the expected project duration: A control net approach. Technical Report OR Report No. 311, North Carolina State University, Raleigh, NC, 1995.

[24] S.E. Elmaghraby and W.S Herroelen. On the measurement of complexity in activity network. *European Journal of Operational Research*, 5:223–234, 1980.

[25] J. A. Ferreira, L. V. Tavares, and J. S. Coelho. A general generator of project networks in terms of their morphological features. In *Proceedings of 6th international workshop on project management and scheduling, Istanbul, Turkey, July 7-9*, 1998.

[26] D. R. Fulkerson. Expected critical path lengths in PERT networks. *Operations Research*, 10:808–817, 1962.

[27] G. R. Grimmett and D. R. Stirzaker. *Probability and Random Processes*. Oxford Science Publications, New York, N.Y., second edition, 1994.

[28] Jane N. Hagstorm. Computational complexity of PERT problems. *Networks*, 18:139–147, 1988.

[29] Jane N. Hagstorm. Computing the probability distribution of project duration in a PERT network. *Networks*, 20:231–244, 1990.

[30] G Hahn and S. Shapiro. *Statistical Models in Engineering.* Wiley, New York, 1967.

[31] R. W. Hamming. *Numerical Methods for Scientists and Engineers.* Dover Publications, New York, N.Y., second edition, 1986.

[32] H. O. Hartley and A.W. Wortham. S statistical theory for PERT critical path analysis. *Management Science*, 12(10):B469–B481, 1966.

[33] E. U. S. Hopfinger. *On the Exact Evaluation of Finite Activity Networks with Stochastic Durations of Activities.* Springer-Verlag, New York, N.Y., 1977.

[34] J. Kamburowski. An upper bound on the expected completion time of PERT networks. *European Journal of Operational Research*, 21:206–212, 1985.

[35] J. Kamburowski. Bounding the distribution of project duration in PERT network. *Operation Research Letters*, 12:17–22, 1992.

[36] G. B. Kleindorfer. Bounding distribution for a stochastic acyclic network. *Operations Research*, 7:1586–1601, 1971.

[37] R. Kolisch, A. Sprecher, and A. Drexel. Characterization and generation of a general class of resource-constrained project scheduling problems. *Management Science*, 41:1693–1703, 1995.

[38] V. G. Kulkarni and V. G. Adlakha. Markov and markov-regenerative PERT networks. *Operations Research*, 34(5):769–781, 1986.

[39] H. Lau and C. Somarajan. A proposal on improved procedures for estimating task-time distributions in PERT. *European Journal of Operational Research*, 85:39–52, 1995.

[40] J. H. Lindsey II. An estimate of expected critical-path length in PERT network. *Operations Research*, 20:800–812, 1972.

[41] Kenneth R. MacCrimmon and Charles A. Ryavec. An analytical study of the PERT assumptions. *Operations Research*, 12:16–37, 1964.

[42] J. Magott and K. Skudlarski. Estimating the mean completion time of PERT networks with exponentially distributed durations of activities. *European Journal of Operational Research*, 71:70–79, 1993.

[43] D.G. Malcom, J.H. Roseboom, Clark C.E., and W. Fazar. Application of a technique for research and development program evaluation. *Operations Research*, 7:646–669, 1959.

[44] J.J. Martin. Distribution of time through a directed acyscic network. *Operations Research*, 13:46–66, 1965.

[45] A.A. Mastor. An experimental and comparative evaluation of production line balancing techniques. *Management Science*, 16:728–746, 1970.

[46] J. Patterson. A comparison of exact approaches for solving the multiple constrained resource project scheduling problems. *Management Science*, 30:854–867, 1984.

[47] E. Ramat, C. Lenté, and C. Tacquard. Modélisation de l'incertidue dans les projets d'innovation: le modèle raih. Technical report, Laboratoire d'Informatique, Ecole d'Ingéniwuea en Informatique pour l'Industrie - Université de Tours-Technopôle, 37913 Tours Cedex 9, France, 1995.

[48] C.R. Rao. *Linear Statistical Inferences and its Applications.* Wiley, New York, N.Y., 1971.

[49] L.J. Ringer. Numeriacl operators for statistical PERT critical path analysis. *Management Science*, 16(2):B136–B143, 1969.

[50] P. (posthumous) Robillard and M. Trahan. Expected completion time in PERT networks. *Operations Research*, 24(1):177–182, 1976.

[51] P. (posthumous) Robillard and M. Trahan. The completion time of PERT networks. *Operations Research*, 25(1):15–29, 1977.

[52] C. Schwindt. Progen/max: A new problem generator for different resource-constrained project scheduling problems with minimal and maximal time lags. Technical Report Technical Report WIOR-449, Institut für Wirtshaftstheorie und Operations Research, Universität Karlsruhe, Germany, 1995.

[53] D. Sculli. The completion time of PERT networks. *Journal of Operations Research Society*, 34(2):155–158, 1983.

[54] A.W. Shogan. Bounding distributions for a stochastic PERT network. *Networks*, 7:359–381, 1977.

[55] C.E. Sigal, A.A.B. Pritsker, and J.J. Solberg. The use of cutsets in monte carlo analysis of stochatsic networks. *Mathematics and Computers in Simulation*, 21:376–384, 1979.

[56] A Thesen. Measures of the restrictiveness of project networks. *Networks*, 7:193–208, 1977.

[57] J. Valdes, R. Tarzan, and E. Lawler. The recognition of series parallel digraphs. *SIAM Jounal Computing*, 11:298–313, 1982.

[58] R. M. Van Slyke. Monte carlo methods and the PERT problem. *Operations Research*, 11(5):839–860, 1968.

[59] T. M. Williams. Practical use of distributions in network analysis. *journal of Operational Society*, 43(3):265–270, 1992.