RIASSUNTO

La presente Tesi di Dottorato affronta lo studio delle tematiche relative al Project Management e al Project Scheduling sotto incertezza.

Il problema affrontato, che riveste un ruolo di notevole importanza sia dal punto di vista scientifico che dal punto di vista pratico, consiste nella determinazione di un piano temporale delle attività cositituenti un progetto (schedula), che tenga nel contempo conto della disponibilità limitata di risorse e della minimizzazione del tempo totale di completamento del progetto stesso (makespan).

La schedula di progetto definita sulla base di dati deterministici può essere soggetta a numerosi cambiamenti, in ragione del fatto che molteplici fonti di incertezza investono l'intero ciclo di vita del progetto.

La presente tesi si propone di fornire una serie di strumenti quantitativi, basati essenzialmente su framework della programmzaione stocastica, in grado di supportare la fase di pianificazione temporale di progetti soggetti ad incertezza.

Nonostante il notevole interesse pratico ed applicativo della tematica affrontata, la letteratura scientifica riguardante il project scheduling sotto incertezza è ancora in fase embrionale. In particolare, risulta evidente la mancanza di una caratterizzazione probabilistica esplicita dell'incertezza e, come conseguenza, non vi è traccia di connesione di tale letteratura con l'ampio spettro di tecniche e metodi propri della branca della programmazione matematica sotto incertezza nota come programmazione stocastica.

La presente tesi offre un contributo innovativo in tale direzione, e si colloca pertanto nel quadro internazionale come il primo tentativo di affrontare la tematica del projet scheduling sotto incertezza con il framework della programmazione stocastica.

Nell'intraprendere lo studio di questa tematica nel Capitolo 3 si è ipotizzato che le risorse fossero comunque disponibili quando richieste, al fine di semplificare il problema e concentrare l'attenzione sulla gestione dell'incertezza.

In particolare il capitolo affronta il problema della caratterizzazione della funzione di distribuzione del makespan del progetto che, in condizioni di incertezza, può essere rappresentato da una variabile casuale con forma e caratteristiche non note.

Partendo dall'ipotesi che le durate delle attività costituenti il progetto siano variabili aleatorie la cui funzione di densità sia nota o possa essere stimata, sono stati proposti due metodi esatti per la soluzione del problema descritto.

L’efficienza dei due metodi propsti è stata poi testata su una batteria di problemi test tratti dalla letteratura e opportunamente modificati per tener conto dell’incertezza.
La ricerca svolta ha come caratteristiche distintive il superamento dell’approccio basato sul valore atteso su cui ad esempio il noto PERT si basa, contestualmente con l’introduzione di un’ottica avversa al rischio che ben si adatta alla gestione dei progetti in ambiti fortemente dinamici e competitivi. Inoltre, in modo originale rispetto alla letteratura, si è rilassata l’ipotesi di indipendenza delle durate delle attività, rappresentate come variabili aleatorie dipendenti con funzione di distribuzione discreta. L’originalità e la rilevanza dei risultati ottenuti è confermata dalla pubblicazione dei risultati della ricerca su rivista internazionale (Evaluating project completion time in project networks with discrete random activity durations- Computers & Operations Research 36(2009) 2716—2722).
Consolidati i concetti e le tecniche appresi in questa prima fase di studio, si è generalizzato il problema aggiungendo i vincoli sulle risorse.
Il problema in esame, noto in letteratura come Resource Constrained Project Scheduling problem –RCPSp- è stato oggetto di numerosi studi, finalizzati sia alla risoluzione del problema, nella sua versione deterministica.
Lo studio del problema in condizioni di incertezza (Robust Project scheduling) ha portato alla definizione di approcci per la determinazione di schedule robuste, capaci cioè di assorbire gli effetti di eventuali “disruptions”, ossia di eventi capaci di modificare la durata e/o l’assorbimento di risorse delle attività del progetto. Con il termine “robustezza della soluzione” o “stabilità della schedula” ci si riferisce alla differenza tra la schedula di base e la schedula che si è effettivamente realizzata. Tale differenza rappresenta una misura della performance dell’algoritmo usato per la definizione della schedula di base: l’obiettivo da perseguire è in questo caso non la minimizzazione del makespan bensì la generazione di una soluzione che non venga deteriorata dal verificarsi di eventuali “disruptions” o eventi imprevisti.
Diversamente dai contributi presenti in letteratura, nel Capitolo 4 la robustezza della schedula è introdotta nel modello sotto forma di vincolo probabilistico, piuttosto che come misura da effettuare a posteriori con l’ausilio di tecniche di simulazione.
Per la risoluzione di tale problema è stata proposta una metodologia euristica che ha quale punto di forza l’uso di un modello che integra le comuni euristiche per lo scheduling dei progetti con strumenti propri della programmazione

stocastic. La metodologia proposta si pone come un’efficiente strumento di pianificazione da affiancare alle tradizionali procedure di pianificazione temporale dei progetti.
La validazione del modello proposto è stata operata attraverso un’ampia fase di sperimentazione, considerando quale misura di robustezza il “livello di confidenza” della schedula reale, cioè la probabilità che il makespan programmato rimanga tale almeno in misura pari a tale livello.
I risultati della ricerca hanno portato alla stesura di un articolo sottoposto a revisione internazionale ed attualmente in fase di terza revisione (A heuristic approach for resource constrained project scheduling with uncertain activity durations-sottomesso a Computers and Operations Research).

Infine, il Capitolo 5 presenta un caso reale di applicazione delle metodologie definite per la risoluzione dei problemi prima descritti, un progetto riguardante la costruzione di residenze per studenti universitari. L’obiettivo perseguito è consistito nella validazione di tale metodologia quale strumento efficace di pianificazione e gestione dell’incertezza nei progetti di costruzioni civili ed edili. La metodologia quantitativa applicata, è stata definita con l’obiettivo di ottenere una schedula quanto meno sensibile alle inevitabili perturbazioni nello svolgimento delle attività del progetto. L’applicazione di tale metodologia si è avvalsa di un DSS che ha permesso di identificare, analizzare e quantificare l'affidabilità della schedula definita e l'impatto su di essa di possibili eventi inattesi.
Siffatta metodologia si è quindi dimostrata più efficiente di quelle deterministiche comunemente usate, nello sfruttare le informazioni disponibili per fornire una pianificazione delle attività in condizioni di incertezza. Il valore aggiunto apportato da tale metodologia consiste nel rappresentare uno strumento in grado di supportare i manager nello sviluppo di una pianificazione delle attività progettuali efficace e realistica oltre a poter essere utilizzato come linea guida per il controllo e il monitoraggio dell’andamento del progetto. La ricerca condotta in questo Capitolo, ha portato alla stesura di un articolo accettato con revisioni minori in una rivista internazionale (A methodology for dealing with uncertainty in constructions projects-Engineering Computations).

Index

List Of Figures and Tables

Chapter 1

Introduction

The growing interest in the field of project management is confirmed by many new theories, techniques and computer applications designed to support project managers in achieving their objectives. Within project management, project scheduling aims to generate a feasible baseline schedule specifying, for each activity, the precedence and resource feasible start times used as a baseline for project execution. Baseline schedule helps manager to visualize project evolution, giving a starting point for both internal and external planning and communication.

Careful project scheduling has been shown to be a key factor to improve the success rate of the project. A recent study by Maes et al. (2000) has found that inferior planning is the third reason of company failure in the Belgian construction industry. This struggles researchers to further develop new project scheduling methods.

After the concepts of project management and project scheduling have been introduced, in the next section the standard problems in project scheduling will be shortly presented followed by a brief introduction of concepts of stable project scheduling.

1.1. The Resource Constrained Project Scheduling Problem –RCPSP-

The Resource Constrained Project Scheduling problem aims at minimizing the duration of a project subject to precedence and resource constrains in a deterministic environment. Precedence constrains are assumed belonging to the best-known type of precedence relationships, the finish-start zero-lag relationship. Subject to such type of constrains, each activity is forced to start when all its predecessors have been completed. As far as resources are concerned, we limit our dissertation on renewable resource constraints, assuming resources available on a period-by-period basis and for which only the total resource use in each time period is constrained for each resource type.
Many exact and heuristic algorithms have been described in the literature to construct workable baseline schedules that solve the deterministic RCPSp, that has been shown to be NP-Hard in a strong sense. For extensive overviews we refer to Herroelen et al. (1998), Kolisch & Padman (1999), Kolisch & Hartmann (1999), Brucker et al. (1999) and Demeulemeester & Herroelen (2002).

1.2. Uncertainty in project scheduling

In a real life context, project execution is subject to a considerable uncertainty. Uncertainty can originate from multiple source: resources can become temporarily unavailable (Lambrechts et al. 2007,2008) and Drezet 2005), activities may have to be inserted or dropped (Artigues & Roubellat 2000), due dates may change, activities may take longer or less long than original expected, etc. As a consequence, although usefulness of baseline schedule is unquestionable, project will never execute exactly as it was planned due to uncertainty. The common practice of dealing with these uncertainties by taking deterministic averages of the estimated parameters might lead to serious fallacies (Elmaghraby 2005).
When duration of activities is assumed stochastic, we move to the field of stochastic RCPSp. A solution for the stochastic RCPSP needs to define the

appropriate reactive action for every possible disruptive event during project execution, given the current state of the project and the a priori knowledge of future activity distributions. Möhring et al. (1984, 1985) call such a reactive way of generating a solution a *scheduling policy* or *scheduling strategy*: a policy makes dynamic scheduling decisions during project execution at stochastic decision points, usually corresponding to the completion times of activities. In pure *dynamic scheduling* (Stork 2001), the use of schedules is even eliminated altogether.

The absence of a schedule has some consequences from an economic point of view. The baseline schedule (pre-schedule, predictive schedule) namely serves a number of important functions (Aytug et al. 2005), Mehta & Uzsoy 1998), such as facilitating resource allocation, providing a basis for planning external activities (i.e. contracts with subcontractors) and visualizing future work for employees. The baseline schedule needs to be sought before the beginning of the project as a prediction of how the project is expected to unfold. It has been observed, (Yang 1996), that using the baseline schedule together with a dispatching rule, i.e. *proactive-reactive scheduling* detailed in the following paragraph, often leads to a lower expected makespan than pure dynamic scheduling procedures. The following table distinguishes between three basic approaches for the development of a baseline schedule (Herroelen & Leus 2005).

Baseline schedule	During project execution
(i) No baseline schedule	(i) Dynamic scheduling (scheduling policies)
(ii) Baseline scheduling with no anticipation of variability	(ii) Reactive scheduling
(iii) Baseline scheduling with anticipation of variability	(iii) Proactive (robust) scheduling

Table 1-1: Different methods for schedule generation under uncertainty

1.3. Proactive VS Reactive project scheduling

In general, there are two approaches to deal with uncertainty in a scheduling environment (Davenport and Beck 2002; Herroelen & Leus 2005): proactive and reactive scheduling.

Proactive scheduling constructs a predictive schedule that accounts for statistical knowledge of uncertainty. The consideration of uncertainty information is used to make the predictive schedule more robust, i.e., insensitive to disruptions. *Reactive scheduling* involves revising or reoptimizing a schedule when an unexpected event occurs. At one extreme, reactive scheduling may not be based on a predictive schedule at all: allocation and scheduling decisions take place dynamically in order to account for disruptions as they occur. A less extreme approach is to reschedule when schedule breakage occurs, either by completely regenerating a new schedule or by repairing an existing predictive schedule to take into account the current state of the system.

It should be observed that a proactive technique will always require a reactive component to deal with schedule disruptions that cannot be absorbed by the baseline schedule. The number of interventions of the reactive component is inversely proportional to the *robustness* of the predictive baseline schedule. Many different types of robustness have been identified in the literature.

In the next chapter, notations and definitions will be provided to formally describe the problem that will be tackled in the remaining chapters.

Chapter 2

Definitions and Problem Formulation

In this chapter, a definition of the proactive project scheduling problem is given. Section 2.1 introduces project representations that help us in illustrating the procedures that will be proposed in later chapters. Afterwards, we propose a rigorous definition of the concepts of quality and solution robustness, which both will be main issues throughout this thesis (Section 2.2).

2.1. Project representations

A project consists of a number of events and activities or tasks that have to be performed in accordance with a set of precedence and resource constraints. The deterministic expected duration of activity j will be expressed as d_j, while in an uncertain scheduling environment, the stochastic activity durations will be denoted by $\mathbf{d}_j$. The activity-dependent weights w_j used in this dissertation represent the marginal cost of starting the activity j later or earlier than planned in the baseline schedule. Once a project schedule has been negotiated, constructed and announced to all stakeholders, modifying this schedule comes at a certain penalty cost. This cost corresponds to the importance of on-time performance of a task to avoid internal and external stability costs. Internal stability costs for the organization may include unforeseen storage costs, extra organizational costs or just a cost that expresses the dissatisfaction of employees with schedule nervousness. Costs related to (renegotiating) agreements with subcontractors, penalty clauses, goodwill damages, etc. are examples of stability costs that are external to the organization.

2.1.1. Project network

A project network is a graphical representation of the events, activities and precedence relations of the project. A network is a directed graph $G = (N,A)$, consisting of a set of nodes $N=\{0,..,N\}$ and a set of arcs A. The transitive closure of a graph $G = (N,A)$ is a graph $TG = (N, TA)$ which contains an edge from i to j whenever there is a directed path from i to j in the original graph. The main focus of the project network is the representation of the precedence relationships between the activities of the project.

There are two network notation schemes commonly used in project scheduling. The activity-on-arc (AoA) representation uses the set of nodes N to represent events and the set of arcs A to represent the activities, while in the activity-on-node (AoN) notation scheme, the set N denotes the activities and the set A represents the precedence relationships. The arcs TA of the transitive closure $TG = (N, TA)$ represent in this case all direct and transitive precedence relationships in the original network. Now we introduce a project that will be used as our vehicle for definition and problem formulation. Let consider a project consisting of 10 activities (activity 0 and activity 9 are dummy activities representing the project start and finish) subject to finish-start, zero-lag precedence constraints and a single renewable resource constraint. The single renewable resource is assumed to have a constant per period availability a of 10 units. Expected activity durations d_j, activity weights w_j and resource requirements are also given. Figure 2.1 denotes the AoN project network for the project described in Table 2.1.

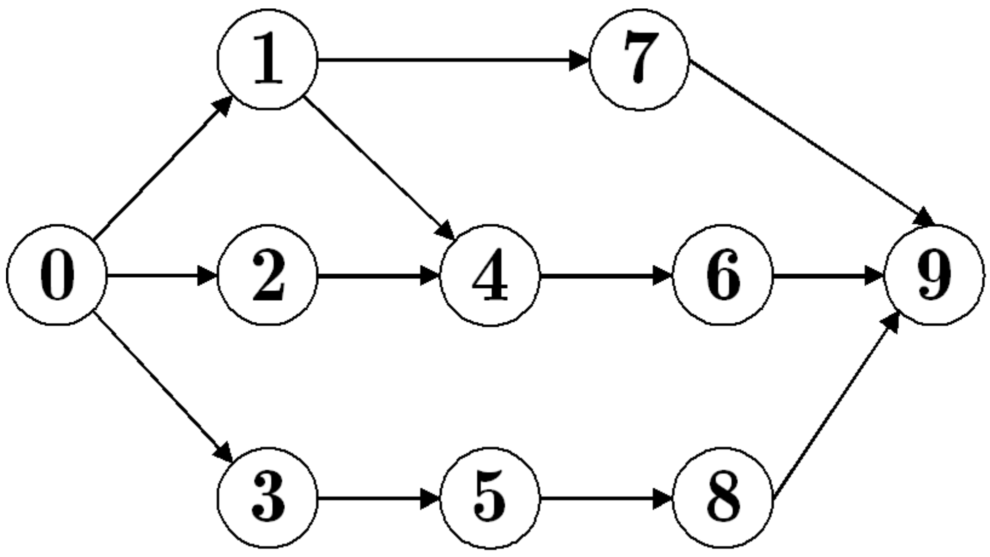


Figure 2-1: Example Project Network

2.1.2. Project Schedule

A schedule S is defined in project scheduling as a list $S = (s_0, s_1, \ldots, s_n)$ of intended start times $s_j \geq 0$ for all activities $j \in N$. A Gantt chart (introduced by H. Gantt in 1910) provides a typical graphical schedule representation by drawing the activities on a time axis.

A schedule is called *feasible* if the assigned activity start times respect the constraints imposed on the problem. In deterministic project scheduling, a feasible schedule is a sufficient representation of a solution.

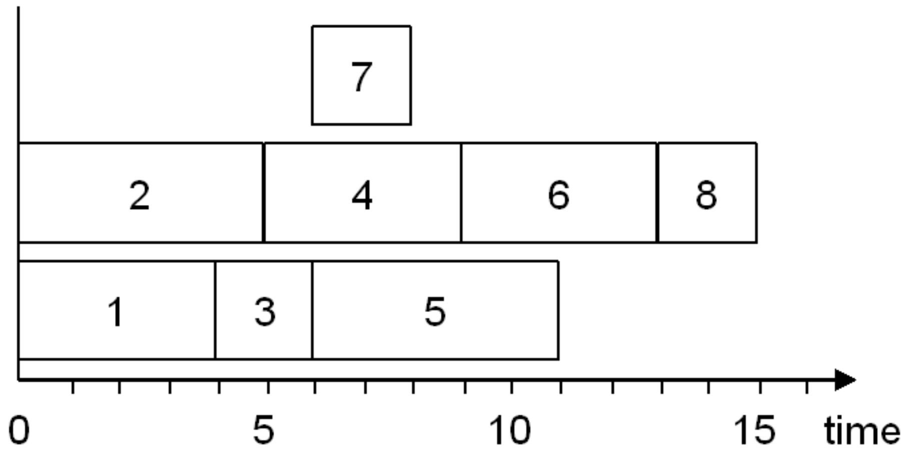


Figure 2-2: A minimum duration schedule

Figure 2.2 depicts a solution schedule for the example network presented.

2.1.2.1. Baseline schedule

A baseline schedule (pre-schedule or predictive schedule) is a list of activity start times generated under the assumption of a static and deterministic environment that is used as a baseline during actual project execution.
A baseline schedule is generated before the actual start of the project (time *0*) and will consequently be referred to as S_0.
It serves a number of important functions (Aytug et al. (2005), Mehta & Uzsoy (1998), Wu et al. (1993)). One of them is to provide visibility within the organization of the time windows that are reserved for executing activities in order to reflect the requirements for the key staff, equipment and other resources. The baseline schedule is also the starting point for communication and coordination with external entities in the company's inbound and outbound supply chain: it constitutes the basis for agreements with suppliers and subcontractors (e.g. for planning external activities such as material procurement and preventive maintenance), as well as for commitments to customers (delivery dates).

2.1.2.2. Realized schedule

A realized schedule S_T is a list of actually realized activity start times S_T that is generated once complete information of the project is gained.
The proactive-reactive scheduling decisions made during project execution influence the actually obtained realized schedule S_T. In a stochastic environment, the realized schedule will thus typically be unknown before the project completion time T. We will refer to this stochastic schedule by S_T.

2.1.3. Resource usage and representations

In resource-constrained project scheduling, project activities require resources to guarantee their execution. Multiple resource categories exist (Blazewicz et al. 1986) but in this thesis (as in the RCPSP), we will limit our scope to renewable resources that are available on a period-by-period basis and for which only the total resource use in each time period is constrained for each resource type.

Every activity j requires an integer per period amount r_{jk} of one or more renewable resource types k $(k = 1, 2, \ldots, K)$ during its execution. The renewable resources have a constant per period availability a_k. The resource constraints can thus be written as:

$$^{n}_{j=1} r_{jk} = a_k \quad \forall \; k = 1,2,..K$$

in which P_t denotes the set of activities that are active at time t.

The network (see Section 2.1.1) and schedule (see Section 2.1.2) representations of the project do not visualize the resource allocation. Hence, additional resource-based project representations are introduced in this section.

2.1.3.1. Resource profile

A resource profile is an extension of a Gantt chart that additionally indicates the variation in resource requirement of a single renewable resource type over time for each activity. Resource requirements and availability are denoted on the Y-axis. Each resource type requires its own resource profile. The resource profile for the single renewable resource type corresponding to the minimum duration schedule of Figure 2.2 is given in Figure 2.3.

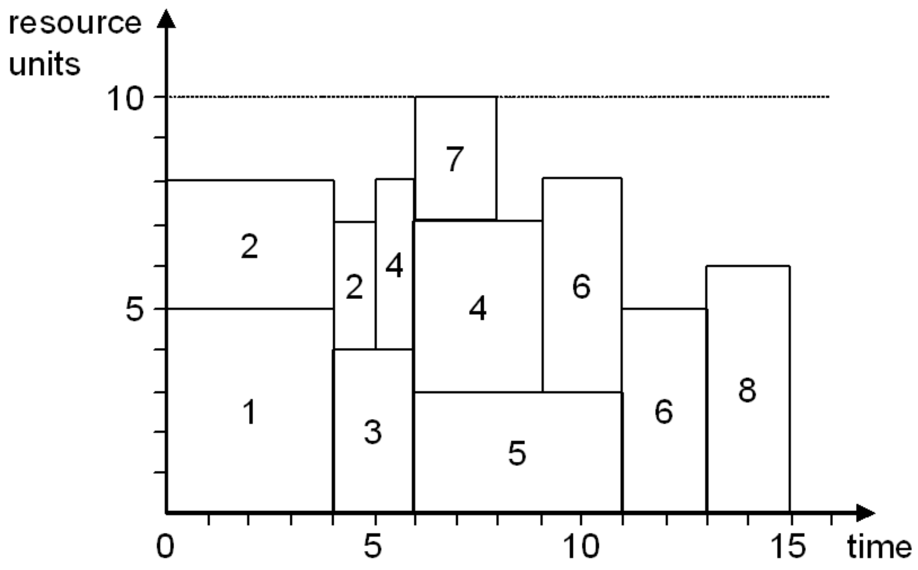


Figure 2-3: Resource profile for example project

2.1.3.2. Resource flow network

Artigues et al. (2003) define *resource flow networks (or transportation networks)* to identify the amount of resources transported from the end of one activity to the beginning of another activity after scheduling has taken place. f_{ijk} denotes the amount of resources of type k, flowing from activity i to activity j. The resource flow network is a network with the same nodes N as the original project network, but with arcs connecting two nodes if there is a resource flow between the corresponding activities, i.e.

$$\exists\, k: f_{ijk} > 0$$

We define R as the set of flow carrying arcs in the resource flow network. The resource arcs in R may induce extra temporal constraints to the project. We remark that a schedule may allow for different ways of allocating the resources so that the same schedule may give rise to different resource flow networks. Not every feasible resource allocation implies an equal amount of stability.

Relying on the one-pass algorithm of Artigues et al. (2003) to compose a resource flow network for the schedule of Figure 2.2, results in the resource flow network $G = (N,R)$ presented in Figure 2.4. Activity 8, for example, has a per period resource requirement of six units. It uses three resource units released by its predecessor activity 5, two units passed on by activity 7 and one unit released by activity 6. The arcs (1,3); (3,7); (6,8), (7,6) and (7,8) represent extra precedence relations that were not present in the original network. The arc (7,9) was present in A, but is not drawn in Figure 2.4 because there is no resource flow from 7 to 9.

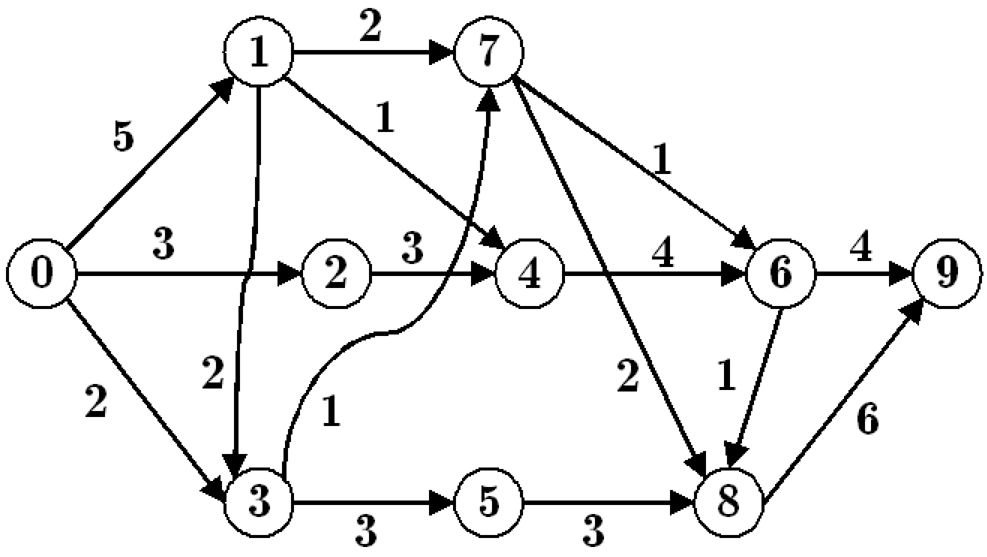


Figure 2-4: Resource flow network for the example project

In the following figure, -Figure 2.5-, the resource profile of Figure 2.3 is reported to illustrate the use of the individual resource units along the horizontal bands. This project representation includes both the resource profile and the resource flow network and will thus become our preferred representation in the remainder of this dissertation if there is only one resource type, as is the case in the example network.

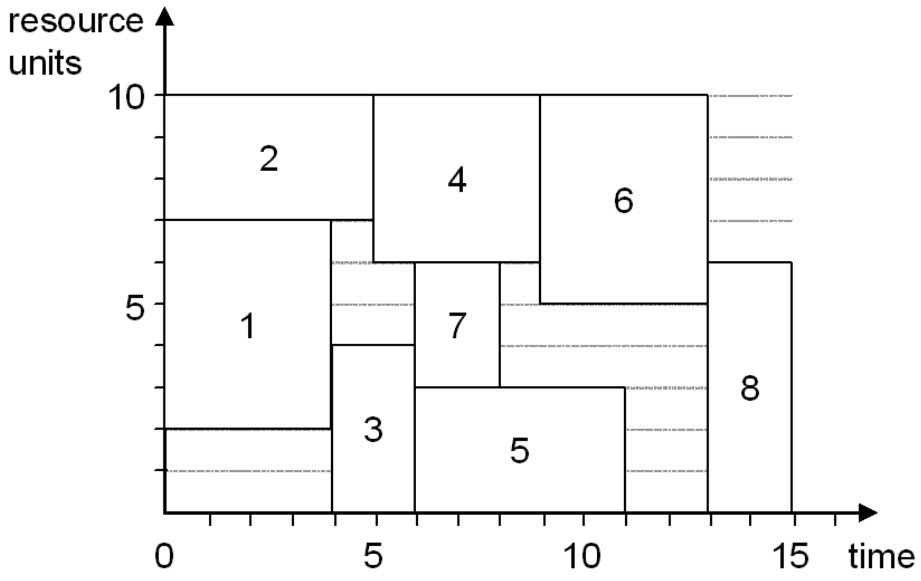


Figure 2-5: Resource profile with resource allocation

2.2. Robustness types and measures

In Chapter 1, the concept of schedule robustness was mentioned as a schedule's insensitivity to disruptions that may occur during project execution. Many different types of robustness have been identified in the literature, calling for rigorous robustness definitions and the use of proper robustness measures. Two often used types of single robustness measures have been distinguished: solution and quality robustness (Sörensen (2001), Herroelen & Leus (2005)). The main difference between quality robustness and solution robustness is that in the former case, it is the quality of the solution that is not allowed to change. This quality is usually measured in terms of makespan or due date performance.

In the latter case, it is the solution itself that is not allowed to change.

2.2.1. Solution robustness or schedule stability

Solution robustness or schedule stability refers to the difference between the baseline schedule and the realized schedule. The difference or distance (S_0, S_T) between the baseline schedule S_0 and the realized schedule S_T for a given execution scenario can be measured as the number of disrupted activities, the difference between the planned and realized activity start times, and other different ways.

For example, the difference can be measured by the weighted sum of the absolute deviation between the planned and realized activity start times: $\Delta \;\; S_0, S_T \;\; = \;\; {}_j w_j \;\; s_j^0 - s_j^T$

where s_j^0 denotes the planned starting time of activity j in the baseline schedule S_0, s_j^T denotes the actual starting time of activity j in the realized schedule S_T , and the weights w_j represent the disruption cost of activity j per time unit, i.e. the non-negative cost per unit time overrun or underrun on the start time of activity j.

In a stochastic environment, the realized activity starting times are stochastic variables $\boldsymbol{s}_j^T$ for which the actual realized values s_j^T for a given execution scenario depend on the disruptions and the applied reactive policy. The objective of the proactive-reactive scheduling procedure is then to minimize: ${}_j w_j E \;\; s_j^0 - \boldsymbol{s}_j^T$

with E denoting the expectation operator, i.e. to minimize the weighted sum of the expected absolute difference between the planned and the realized activity start times.

2.2.2. Quality robustness

Quality robustness refers to the insensitivity of some deterministic objective value of the baseline schedule to distortions. The goal is to generate a solution for which the objective function value does not deteriorate when disruptions occur. Contrary to solution robustness, quality robustness is not concerned with the solution itself, only with its value on the performance metric. It is measured in terms of the value of some objective function z.

In a project setting, commonly used objective functions are project duration (makespan), project earliness and tardiness, project cost, net present value,etc.

When stochastic data are available, quality robustness can be measured by considering the expected value of the objective function, such as the expected makespan $E\ [Cmax]$, the classical objective function used in stochastic resource-constrained project scheduling (Stork 2001).
It is logical to use the service level as a quality robustness measure, i.e. to maximize $P(\boldsymbol{z} \leq z)$, the probability that the objective function value of the realized schedule stays within a certain threshold z. For the makespan objective, we want to maximize the probability that the project completion time does not exceed the project due date $\delta_n,\ i.e.\ P(s_n^T \leq \delta_n$ n), where s_n^T is a stochastic variable that denotes the starting time of the dummy end activity in the realized schedule. We will refer to this measure as the timely project completion probability (TPCP). It should be observed that also the analytic evaluation of this measure is very troublesome in the presence of ample resource availabilities. In the next chapter we introduce a new heuristic approach with the aim to evaluate project completion time with discrete random activity durations.

Chapter 3

Project Scheduling Under Uncertainty Of Networks With Discrete Random Activity Durations

In real projects, defining a good schedule on the basis of deterministic processing times is usually inadequate, because these times are only estimates and are susceptible to unpredictable changes. Deterministic models for project scheduling suffer from the fact that they assume complete information and neglect random influences, that occur during project execution. A typical consequence is the underestimation of the project duration as frequently observed in practice, even in the absence of resource constraints.

In this chapter a method for obtaining relevant information about the project makespan for scheduling models, with dependent random processing time available in the form of scenarios and in the absence of resource constraints is presented.

3.1. State of the Art

As previously discussed, real projects are subject to considerable uncertainty due to a number of possible sources. Resources may become unavailable, activities duration may experience some delay, new activities may be incorporated in the project or other activities may be even deleted. Amongst the full range of sources of significant uncertainty associated with any given project (Atkinson et al. 2006), an obvious aspect of uncertainty concerns estimates of potential variability of activity duration. In this context, our choice involves modeling processing times of activities as random variables. A very traditional issue with respect to stochastic networks is the derivation of the distribution or quantiles of the project completion time. This issue may be valuable in project management, particularly at the time of bidding, and has been the subject of investigation within both academia and industry. A great deal of research has been carried out on methodologies for estimating project time distributions. Due to the inherent difficulty of this task, two main distinct methodologies have been applied, that is the simulation approach (Van Slyke (1963), Sullivan & Hayya (1980), Herrera (2006), Shih (2005)) and the analytical approach (Dodin (1985), Hagstrom, (1990)).

Other approaches focus on approximating either the expected completion time value (Fulkerson 1962) or the probability for completing the project within a given deadline (Soroush 1994) as tightly as possible. A comprehensive review of most of the earliest references is presented in (Elmaghraby 1989). For more recent results the reader is referred to Yao and Chu (2007) and references therein. Hagstrom (1988) showed that the problem of computing the probability that a project finishes by a given time, when activities durations are discrete, independent random variables, is NP-complete.

3.2. Model Proposal

We now present an efficient method to find quantiles of the distribution function when activities durations are dependent discrete random variables, or when a suitable discretization in the form of scenarios is available for continuous dependent random variables. The output can be used by a contractor to assess its capabilities to meet the contractual requirement before bidding and to quantify the risks involved in the schedule.

In this case, the specification of a project is assumed to be given in activity-on-arc (AoA) notation by a connected, directed acyclic graph $\mathrm{G(N,A)}$ (referred to as project network), in which N is the set of nodes, representing network “events”, and A is the set of arcs, representing network activities.

We assume that there is a single start node 0 and a single terminal node n.

When the durations of all the activities are constants, project managers may easily calculate the project completion time by the well-known critical path method. Let denote by

$$\Pi = \pi_k, k = 1, \dots, K$$

the set of paths from the node 0 to the node n in $\mathrm{G(N,A)}$. The project makespan can obviously be defined as

$$L = \max_{\pi_k \in \Pi} L_k$$

where L_k represents the length of the k^{th} path from node 0 to node n. Suppose now that we want to estimate the project duration that will not be exceed with probability at least α, that is, we want to estimate the α-quantile of the makespan distribution function. This information can be obtained through the solution of the following problem:

$$\min s_n \tag{1}$$

$$P \quad s_n \geq s_0 + L_k, \forall \pi_k \; 0, n \; \in \Pi \quad \geq \alpha \tag{2}$$

$$s_n \geq 0 \tag{3}$$

where s_i represents the start time of event i and L_k is the random variable associated to the length of the k^{th} path from the start node 0 to the terminal node n. We consider the starting time of node 0 equal to zero.

The α- quantiles of the makespan represents a project duration that, with a probability α, will not be exceeded. In fact, the joint probabilistic constraints assures that

$$L_k^* \geq L_k, \forall \pi_k \in \Pi$$

with probability at least α.

This definition allows to address the decision-makers risk aversion, ensuring that the project’s operations are unaffected by major delays with a high level of probability.

It is well known that the expectation criterion of the classical PERT model (Malcolm et al. 1959) is most appropriate for a risk-neutral decision maker. We trivially observe that reasoning on the basis of averages always results in an underestimate of the expected duration of the project and leads to a probability of exceeding the due date of the project near to 50%. Criticism against the use of averages has been raised in (Elmaghraby 2005), where a demonstration that gross errors can be made using the average as optimization criterion is reported.
In order to account for possible risk averseness we use the joint probabilistic constraints (2). Such conservatism should be invoked when large potential gains and losses are associated with individual decisions (Schuyler 2001).
It is worth while observing that the use of individual probabilistic constraints is not suitable for this problem, since $L_k,$ will normally be dependent even if the random arc weights are independent, because of common arcs.
Therefore, we consider a formulation with joint probabilistic constraints involving dependent random right-hand side variables. This probabilistic constrained framework is well suited for this kind of problem and it is particularly useful when the penalty for the project to be completed late is very high or simply is not easily quantifiable. The use of chance constrained programming methods to examine some statistical properties of stochastic networks, is not completely new.
Charnes et al. (1964) considered the following chance constrained model to characterize the distribution of total project completion time:

$$\min s_n \tag{4}$$

$$P\left(s_j \geq s_i + d_{ij}\right) \geq \alpha, \forall (i,j) \in A \tag{5}$$

$$s_j \geq 0, \forall j \in N \tag{6}$$

However we notice that the chance constraints paradigm is used, regardless the dependency among activities and paths, and in addition, since the stochastic precedence constraints are written in terms of the starting time of the preceding events, a dynamic problem is established, for which the use of a static stochastic programming problem should be prevented.

The developed model differs from other approaches proposed in the literature in several aspects.

• The methodology does not require any hypothesis on the distribution function of the activities durations. We only assume that a probability density function has been specified for each activity time and that a suitable discretization is available. Different distributions can be used to model different activity times.
• Our model overcomes the limitation of stochastic independence among activities times. Often activities durations are correlated trough the common usage of resources or precedence relations.
• Our model overcomes the limitation of stochastic independence among the paths of the network. Paths are in fact correlated via common activities and the correlation is treated explicitly in the use of joint probabilistic constraints
• Unlike sensitivity analysis, we can account for the effects of simultaneous changes to multiple activity times.
• In the proposed model, by varying the reliability threshold, decision makers might acquire information about different alternatives and might choose the maximum probability of violating the project due date which is allowed to be tolerated.
Finally, we would like to remark that the probabilistic paradigm is one of the most powerful prescriptive methodologies for decision makers, in that represents the global probability of violation of the constraints. Clearly, the most appropriate paradigm for project scheduling models under uncertainty is highly dependent on the specific project at hand.

3.3. Solution Methods

We consider a finite set of scenarios $S = \{1, . . . , |S|\}$, with associated probability p_s, $s = 1, . . . , |S|$. Let us suppose that each edge (i, j) $i,j\ \forall\ i,j\ \in A$ (i, j) has a vector of weights realizations $d^1_{i,j}, \dots, d^{|S|}_{i,j}$. This not only renders the probabilistic problem tractable, but also allows the original data to be used without manipulations, since the variability in activity durations are indeed most often discrete.

Relevant information based on past experience may be useful in this context. In other words, the actual probability distribution that applies during project execution is not known beforehand, and the discrete input scenarios form the best approximation available. Discrete scenarios have been used with similar motivations in (Herroelen & Leus, 2004 B).
Problem (1-3) can be reformulated as follows:

$$\min s_n \tag{7}$$

$$s_n \geq s_0 + L_k\ s\ * \Theta\ s\ , \forall\, \pi_k \in \Pi, \forall\, s\ \in S \tag{8}$$

$$_{s \in S}\ p_s * \theta\ s\ \geq \alpha \tag{9}$$

$$s_n \geq 0 \tag{10}$$

$$\theta\ s\ \ \in \{0,1\} \tag{11}$$

where the notation $L_k\ s$ stands for the length of the k^{th} path from node 0 to node n in scenario s and $\theta\ s$ is an indicator variable, which forces the constraints (8) to be satisfied if $\theta\ s = 1$ and allows the violation if $\theta\ s = 0$. Constraint (9) ensures the fulfillment of the constraints (8) for those scenarios, whose cumulative probability is greater than α.
We observe that, given the particular structure of the problem, it can be rewritten as:

$$\min s_n \tag{12}$$

$$s_n \geq s_0 + max_{\pi_k \in \Pi} L_k\ s\ * \Theta\ s\ ,\ \forall\, s\ \in S \tag{13}$$

$$_{s \in S}\ p_s * \theta\ s\ \geq \alpha \tag{14}$$

$$s_n \geq 0 \tag{15}$$

$$\theta\ s\ \ \in \{0,1\} \tag{16}$$

The length of the maximum path can thus be viewed as a one-dimensional random variable z, with probability distribution function

$$\mathrm{F\ z\ = P\ \ z} < v\ \mathrm{, v\ \in\ R^n}$$

for which there is only one α-efficient point defined by

$$\mathrm{F^{-1}\ \ p\ \ = min\{v{:}\ F\ \ v\ \ \geq \alpha\}}$$

Prékopa (1995).
In the foregoing, we highlight how this model can be tackled from a computational point of view. The algorithm briefly described above, in the sequel referred to as Scenario Longest Path Algorithm (SLPA, for short),

entails two main stages. In the first stage, for each scenario $s = 1 \ldots |S|$, the longest path $\mathrm{Lmax}[s]$ from node 0 to node n in the project network $G(N,A)$ is determined.

Let $D = \mathrm{Lmax}\ s : s \in S$ the $|S|$ dimensional set of the longest paths in each scenario. Afterwards, the solution vector is arranged in increasing order of length. Let $D = \mathrm{Lmax}\ i_1\ , \ldots, \mathrm{Lmax}[i_{\ S}\ ]\}$ with $\mathrm{Lmax}[i_l] \leq [i_{l+1}], l = 1, \ldots, |S| - 1$ be such ordered set. The optimal solution is $\mathrm{Lmax}[i_l^*]$ where i_l^* is the smallest index such that

$_{i=i_1,\ldots,i_l^*} p_i \geq \alpha.$

From the discussion above, it is easy to verify that the computational complexity of the overall procedure is $O(|S||N\ |) + O(|S|\log|S|) + O(|S|)$.

It can be seen that if we consider edge weights independent random variables, the number of scenarios increases exponentially with the number of activities in the project network. In real contexts, however, the activities durations are often correlated leading to a reduction in the number of realizations to be considered. When the number of scenarios is huge, it is likely that the procedure would become cumbersome.

Thus, we present a second algorithm, namely the All Paths Enumeration Algorithm (AllPEA, for short), based on the explicit enumeration of all the paths in the network. After having determined all the paths in the network, the following mixed integer linear programming problem has to be solved:

$$\min s_n \tag{17}$$

$$s_n \geq \quad _{i,j\ \in \pi_k} d_{i,j}^s * \theta_s, \forall\, \pi_k \in K, \forall\, s\ \in S \tag{18}$$

$$_{s \in S}\, p_s * \theta\ s\ \ \geq \alpha \tag{19}$$

$$s_n \geq 0 \tag{20}$$

$$\theta\ s\ \ \in \{0,1\} \tag{21}$$

The problem (17)-(21) has $|S|$ integer variables and a number of constraints equal to the number of paths in the network times the number of scenarios, plus one (i.e., the knapsack constraint (19)).

The path enumeration approach relies on the definition of a search tree.

In particular, branches refer to the decisions of extending a given partial path, whereas nodes refer to partial paths $\Gamma(k)$. We denote with T the set of partial paths to be further extended.

At the top of the search tree (level 0) there is only one partial path composed by the start node. Thus, nodes at level 1 refer to at most $|\mathrm{N}|-1$ partial paths, each defined as the extension of the initial partial path with a node adjacent to the start node. It is worthwhile noting that as a natural consequence of the topological order of project networks the search tree is generated in a way to guarantee that partial paths corresponding to nodes are all different.
In the following, the basic scheme of the proposed approach is reported.

Step 0 (Initialization Phase). Set $\mathrm{T} = \{0\}$, where 0 denotes the start node of the network and $\mathrm{k} = 0$.
Step 1 (Termination Check). Check the list T. If it is empty, STOP. Otherwise, extract a partial path $\Gamma(k)$, delete it from T and go to Step 2.
Step 2 (Path Extension Phase). Let $j_k \in N$ be the last node of the partial path $\Gamma(k)$. For all nodes $i \in \ N\ j_k, i \ \in A\}$, extend the partial path and store it in a newly created partial path. Set $\mathrm{k} := \mathrm{k} + 1$ and insert $\Gamma(k)$ in T. Go to Step 1.

All paths $\Gamma(k)$ in a precedence constrained directed acyclic graph can only use nodes between 0 and j_k since nodes are topologically numbered and an arc $\ i,j\ \in A$ might exist only if $i < j$. Therefore, the number of paths ending in j, denoted as P[j] is bounded by $\ _{i \in V:(i<j)} P[i]$. At node n, in the worst case $P\ n\ = \ _{i \in V:(i \neq n)} P[i]$.

Theorem 3.1 The proposed algorithm finds all the paths in a project network in a finite number of iterations, and this number is bounded above by $2^{\ V\ -1}$.
Proof. Since the project network is acyclic, the number of paths generated and explored in the search tree is finite.
The number of paths in the considered graph can be computed exactly by dynamic programming through the following recursive formula:

$$P\ j\ = \qquad \underset{i \in V:\ i,j\ \in A}{} \qquad P\ i\ ;$$

where *P[j]* is the number of distinct paths, connecting the start node *0* to *j* and *P[1]* is assumed to be equal to one.

In a precedence directed acyclic graph, all paths connecting node 0 to node j, can only includes nodes between 0 and j, since nodes are topologically numbered and an arc $i, j \;\; \in A$ might exist only if $i < j$. Therefore, the value of $P[j]$ is bounded by

$$_{i \in V:\ i<j} \; P \;\; i \;\; ;$$

At node n, in the worst case

$$P \;\; n \;\; = \quad _{i \in V:(i \neq n)} P[i].$$

Exploiting the recursive dependence of each $\mathrm{P[i]}$ on nodes which are prior to i (in topological order), and with some mathematical manipulations (reported in the Appendix A), we come up with the following bound: $P \;\; n \;\; = 2^{\;N \;-1}$. Clearly, if the network is complete in a topological order sense (no arcs are allowed if they destroy the topological order), this bound gives the exact number of paths in the network.

The choice between the two solution methods is likely to depend on the number of scenarios considered and on the network complexity. Several considerations can be taken into account such as the network density and the number of arcs and nodes.

A trivial observation could be that the path enumeration, required by the second solution method, can be overwhelming from a computational point of view for medium sized project networks. It is interesting to see that its practical performance is indeed very good, and, surprisingly, the most time consuming part of the algorithm is the solution of the mixed integer programming problem (17)-(21).

On the contrary, when the number of scenarios becomes huge, the evaluation of the longest path in each scenario may result computationally cumbersome.

We will illustrate the practical behavior of both SLPA and AllPEA in the next section.

Test Problem	N° of Nodes	N° of Arcs	OS
j309–7	30	47	0.11
j306–10	30	47	0.11
j301–1	30	48	0.11
j3037–4	30	67	0.15
j601–1	60	93	0.05
j607–10	60	93	0.05
j6048–9	60	131	0.07
j6030–10	60	112	0.06
j9047–7	90	138	0.03
j909–7	90	137	0.03
j901–1	90	137	0.03
j9035–9	90	194	0.05
j1209–8	120	183	0.03
j12042–10	120	258	0.04
j1201–1	120	183	0.03
j12033–9	120	220	0.03

Table 3-1: Test Problem Characteristics

3.4. Computational Result

To assess the performance of the proposed solution approaches, described in Section 4, computational experiments have been carried out on a set of benchmark problems randomly selected from the project scheduling problem library PSPLIB (Kolisch & Sprecher 1997) and on two large-size instances (N1 : |N | = 200; |A| = 400, N2 : |N | = 300; |A| = 600), randomly generated by using the forward network generator recently proposed in (Guerriero & Talarico 2007).

The characteristics of the test problems, taken from PSPLIB, are shown in Table 1, in which for each instance the number of nodes, the number of arcs and the order strength (i.e., OS) are reported. The order strength measures the number of precedence relationships relative to the size of a project. According to Cooper (1976) the order strength of a project is defined as the number of precedence relationships divided by the maximum number of possible precedence relationships in a project (OS = $2|N|/(|A|(|A|-1))$).

The set of problems consists of four problem types including 30, 60, 90 and 120 nodes. Activity durations are chosen as realizations of a discrete uniform random variable over the range [1, 10]. The number of scenarios S was varied in the set {20, 30, 50, 75, 100, 200, 300, 400, 500, 600, 700, 800, 900, 1000}.

Finally, each instance has been solved for different values of the probability (i.e., reliability) level: 0.8, 0.85, 0.9, 0.95, 0.975, 0.99. The computational experiments have been carried out on a AMD Athlon processor at 1.79 GHz.

The algorithms were coded in AIMMS 3.7 (Bisschop & Roelofs 2007) and CPLEX 10.1 (Ilog, 2006) was used as ILP solver.

We have carried out a large number of experiments: about 10000 instances of the model have been solved, reaching in all the cases the optimal solution in an exact way. A detailed accounting of the numerical results is available in (Bruni et al. 2007).

In what follows, we report only the experiments useful to illustrate the validity of the proposed model and the effectiveness of the developed solution approaches.

A first set of experiments has been carried out with the aim of assessing the variation of the project makespan with respect to the probability/reliability level. In particular, in Figure 3.1, we report the makespan values of four test problems (i.e., j309 − 7, j601 − 1, j901 − 1 and j1209 − 8) with 20 scenarios, versus the probability level. We observe that the makespan increases when α increases.

This is an expected behaviour because with increasing values of α we adopt a more conservative point of view hedging against more disruptions scenarios.

We notice that the worst case makespan can be worked out by considering a reliability level $\alpha=1$. In practice, risk-averse project managers may consider to limit the probability of the project of being late to a small value allowing a risk probability around $1-\alpha = 20\%$.

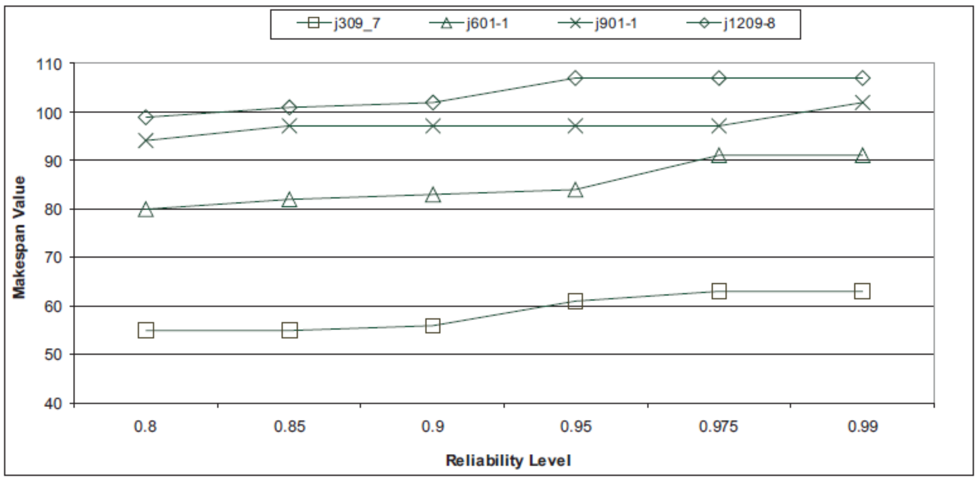


Figure 3-1: Makespan-Reliability Trade-Off

We observe that, very often, little variations in project completion date correspond to relevant gains in terms of reliability. For instance, for the test problem j901 − 1 considering a project due date of 97 rather than 94 would ensure an increase in the reliability level from 0.8 to 0.95. That is, the due date could be delayed with a probability at most equal to 0.05. Similar trends have been observed for all other test problems. As evident, the proposed approach may assist project managers in searching for schedules with acceptable makespan performance experimenting the most appropriate α value.

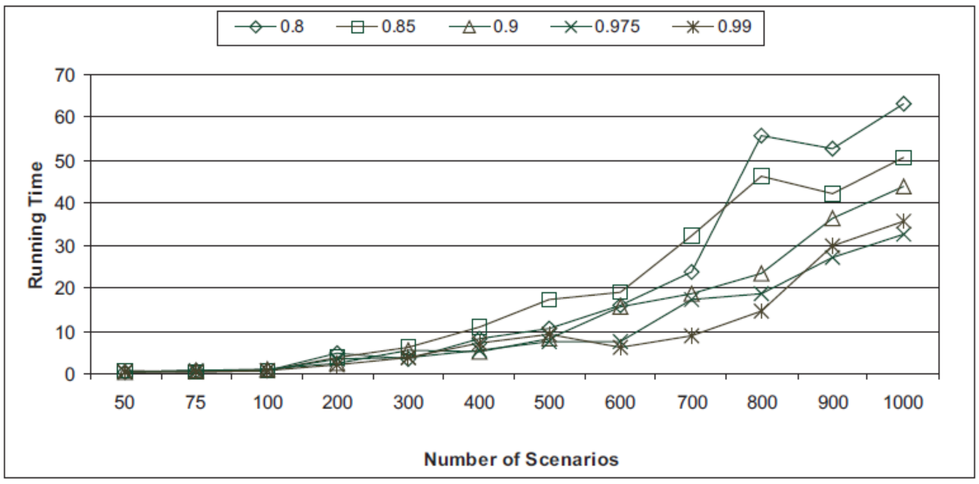


Figure 3-2: Running time (in sec.) of AllPEA for the test problem j306 −10 as a function of the number of scenarios

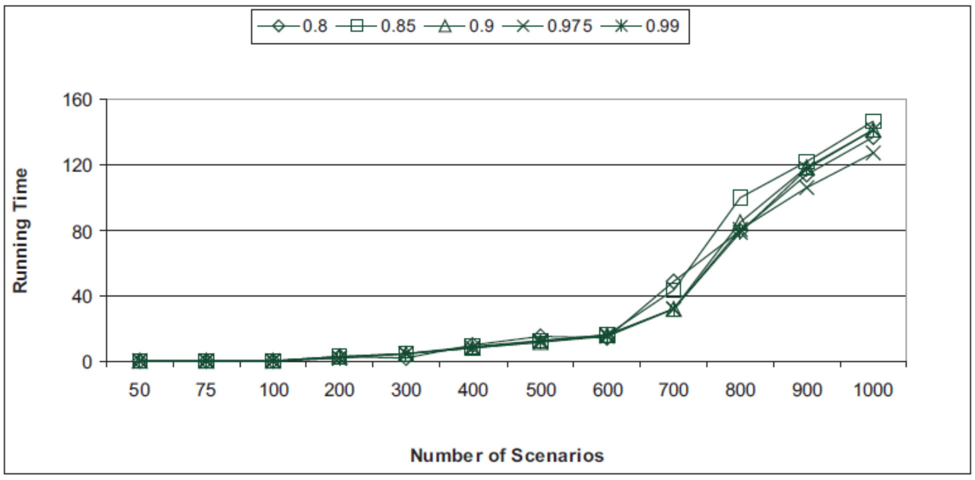


Figure 3-3: Running time (in sec.) of SLPA for the test problem j306 −10 as a function of the number of scenarios

As far as the computational effort is concerned, we have investigated the running time sensibility to various parameters, considered in the experimental phase.

In particular, in Figure 3.2, we report the execution time (in seconds) of All-PEA when solving the test problem j306 − 10 as a function of the number of scenarios for different reliability levels. Similarly, Figure 3.3 shows the computational time required by SLPA on solving the same instance.

It is worth noting that the path enumeration phase requires a constant amount of time, for all the considered scenario cardinalities. Thus, the exponential behaviour of the running time of AllPEA (see Figure 3.2) is mainly due to the computational effort required to solve the mixed integer problem (17)- (21). As far as the running overhead of SLPA is concerned, the computational results indicate also in this case an exponential trend, but with less variability amongst different reliability levels (see Figure 3.3).

In order to assess the influence of the network OS on the running time of the proposed algorithms, we have compared the execution time for the test problems j301 − 1,OS = 0.11 and j3037 − 4,OS = 0.15, both with 30 nodes, for a reliability level of α= 0.8.

Figures 3.4 and 3.5 highlight the related results for AllPEA and SLPA, respectively. From these two figures, it is evident that lower OS levels make the problems more difficult to solve, especially for increasing number of scenarios.

We would like to remark that this trend is more evident for AllPEA. Indeed, a project with a lower order strength has more precedence restrictions among its activities and therefore, the number of paths to be enumerated is larger.

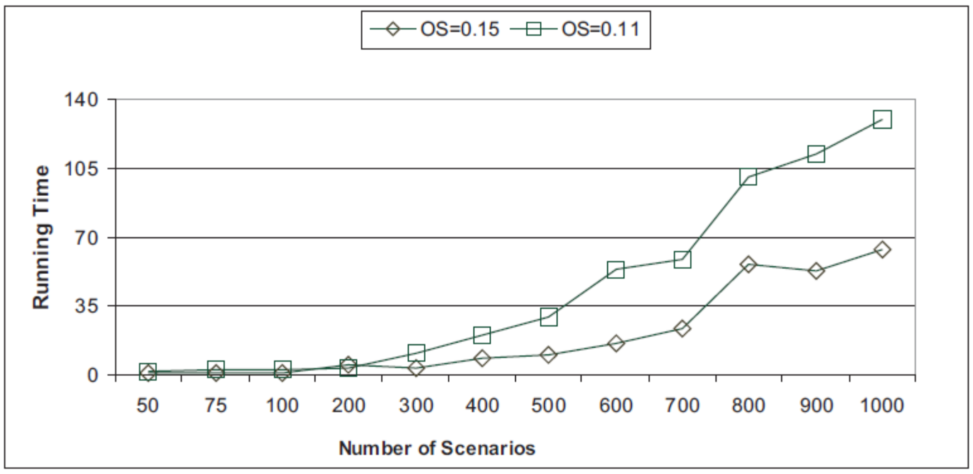


Figure 3-4: Influence of the order strength on AllPEA running time

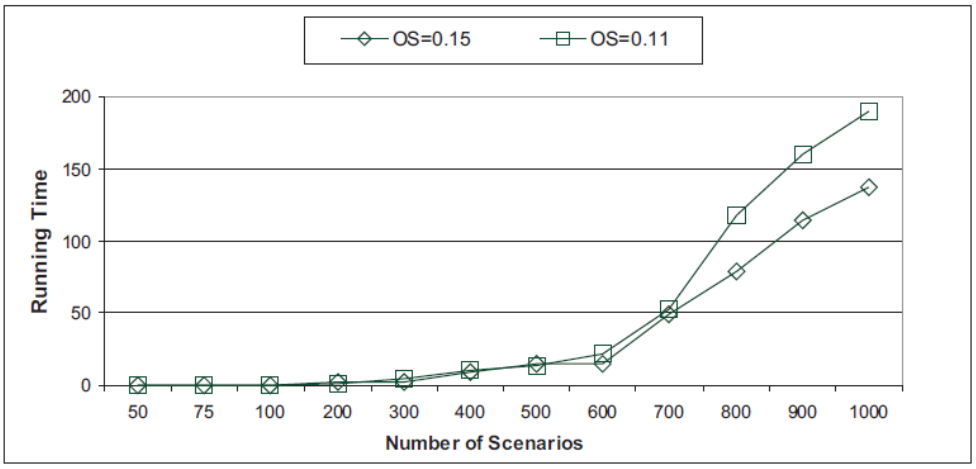


Figure 3-5: Influence of the order strength on SLPA running time

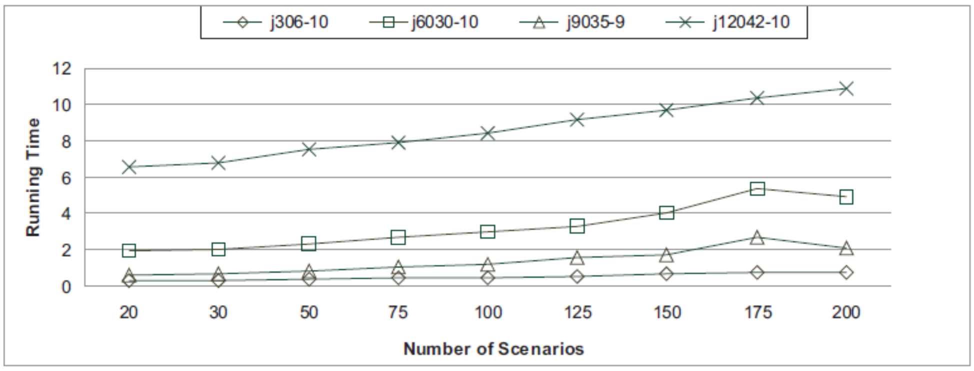


Figure 3-6: Computational time of AllPEA on different test problems

Clearly, the network size (i.e., the number of nodes) plays also a crucial role in the practical efficiency of the proposed solution approaches. In order to illustrate this aspect, in Figure 3.6 we report the running time of AllPEA, for a probability level of 0.9 and a number of scenarios less than 200, when solving the test problems j306 − 10, j6030 − 10, j901 − 1 and j12042 − 10. The computational results of Figure 3.6 underline that the execution time of the algorithm increases with the number of nodes of the project network, as expected. However, we notice that the time needed to solve all but one of the instances are comparable. This may indicate that a threshold size for the problem to become substantially more difficult is 120 nodes. We would like to remark that even for this instance, the solution time does not exceed 12 seconds. This leaves room for application on even larger instances.

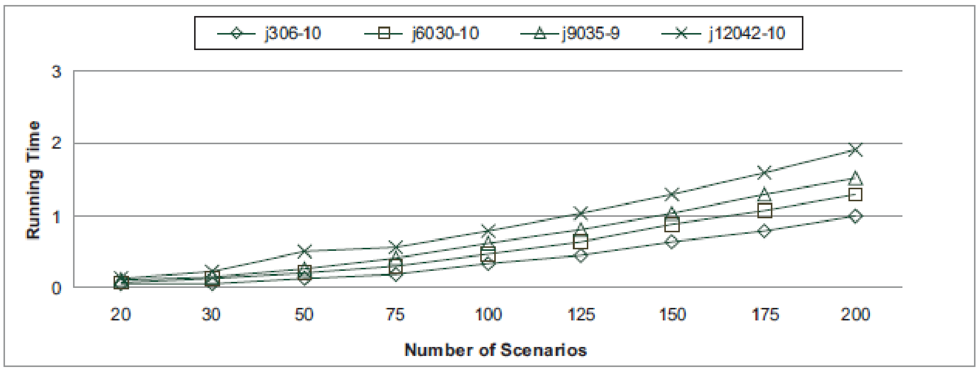


Figure 3-7: Computational time of SLPA on different test problems

Figure 3.7 shows the SLPA computational time for the same test problems. Interestingly, this procedure seems to be less sensitive to the project network dimension. As already observed, a relevant part of the computational time spent by AllPEA is required by the search of all paths of the network, which is exponential in the number of nodes. Finally, we observe that, depending on the project network characteristics, one method may outperform the other. The choice of the most efficient solution approach depends on several factors. To have an idea, let us to consider the two test problems j9047−4 and j909−7. Even though the number of nodes is fixed to 90, and the arcs cardinalities are comparable, the number of paths in the two networks is very different (321 versus 58). Hence, as expected in this case the procedure AllPEA will be computationally more demanding when solving j9047−4. In order to conclude which is the most efficient algorithm, we should compare AllPEA and SLPA over a significant range of complexity measures. Unfortunately, complexity measures are not always useful to explain and predict the time to solve the problem optimally. To support this observation, we remark that the number of paths actually present in a network drastically affect the solution time. This is evident considering that AllPEA running times for the biggest networks N1 and N2 (with 260 and 380 paths, respectively) are comparable with those of test problems j9047−4 and j9035−9 (with 321 and 381 paths, respectively).

Despite these warnings, some conclusions can be drawn on the basis of the numerical results collected. With respect to the comparison between the two solution methods, there is some evidence on the superiority of SLPA over AllPEA at least when the number of scenarios is limited. Indeed, when the number of scenarios is low enough, SLPA outperforms AllPEA; the opposite situation is observed when the scenario cardinality exceeds a certain threshold. This threshold depends on the problem at hand.

For instance, Figure 3.8 shows the trade-off between the two procedures for the test problem j1201−1 for α = 0.9. We observe that for a scenario cardinality below 100, SLPA is more efficient than AllPEA. When the number of scenarios rises above 100, an opposite behaviour emerges, as AllPEA becomes more efficient than SLPA.

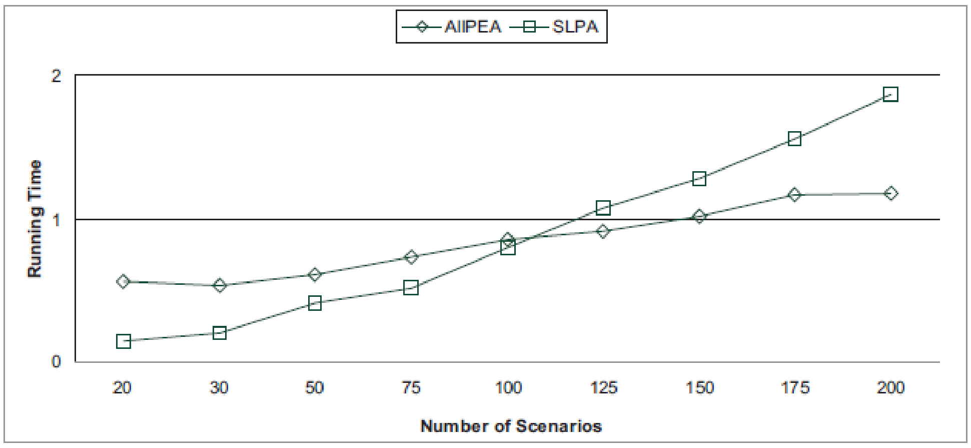


Figure 3-8: Computational time trade-off between AllPEA and SLPA for test problem j1201 − 1

Figure 3.9 is constructed in a similar way as Figure 3.8 for a different test with 60 nodes (i.e., test problem j601−1), but now the threshold scenario cardinality is around 600. Thus, it seems that for smaller networks SLPA performs better notwithstanding the quite high number of scenarios. Nonetheless, it is worth noting that the solution time for the two procedures is quite similar, at least for a scenarios cardinality below the threshold level.

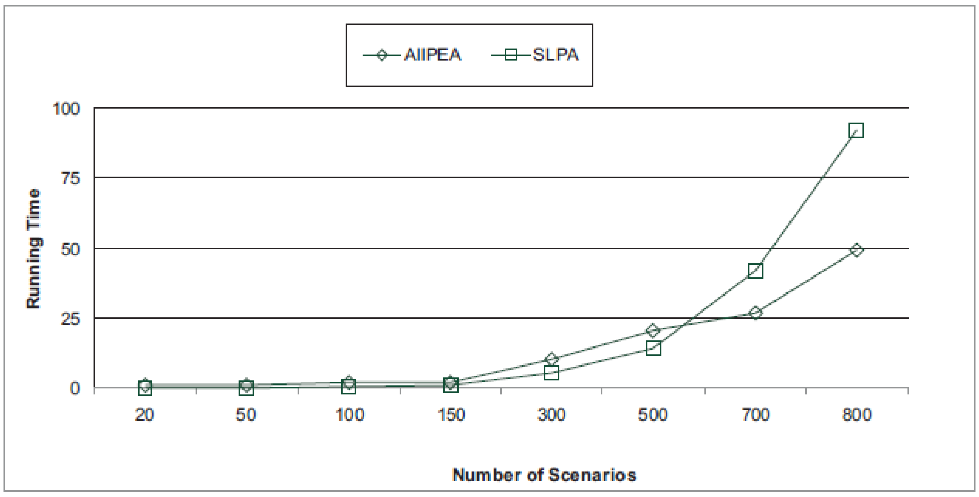


Figure 3-9: Computational time trade-off between AllPEA and SLPA for test problem j601 – 1

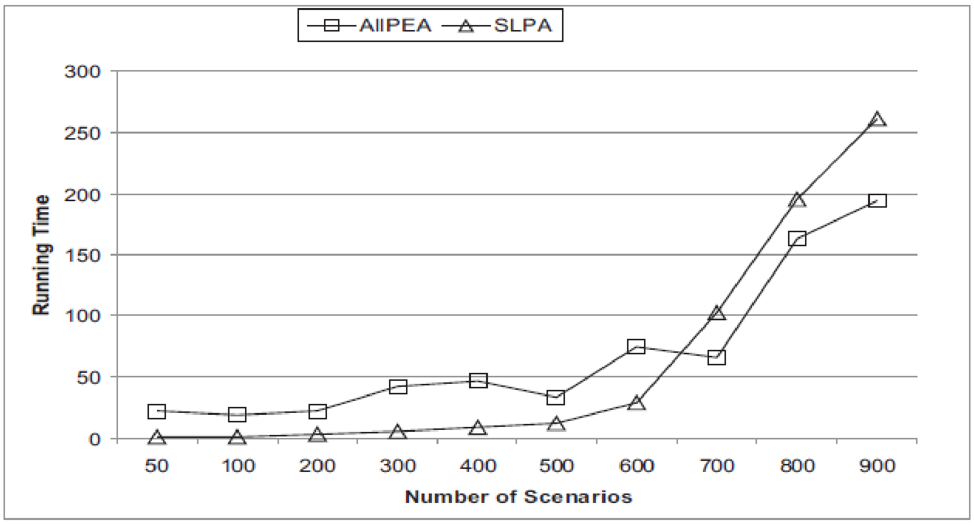


Figure 3-10: Computational time trade-off between AllPEA and SLPA for test problem N2, α = 0.99.

We observe that also for the network N2 with a substantially larger number of nodes, the threshold on the number of scenarios is around 650, similarly to the network with 60 nodes (see 3.10). We may regard this indicator as somewhat misleading. However, we should note that this problem instance has a very limited number of paths, which makes the path enumeration phase very efficient in practice. It is worth noting that for some of the instances examined, it is not evident a superiority of one solution method against the other, at least for the number of scenarios considered. For the largest size instances, we report the computational results in Fig. 3.11-3.14. It is worthwhile to remark that these problem instances can be considered quite suitable in order to simulate a real world situation and validate the behaviour of the proposed model. In fact, a cardinality of 600 for project activities is meaningful related to the typical dimension of a medium term project. Our experiments showed that AllPEA is robust in relation to the number of activities, but, conversely, is highly dependent on the number of paths in the networks. We notice, in fact, that AllPEA running times are higher for the test N1 which has less activities, but more paths than N2. It is worth observing that the computational efforts of the proposed solution methods are not very high, solution times varying over the range [0−400] seconds. Henceforth, the computational results indicate that the procedures are effective even for networks with hundred of activities. On the other hand, the running times drastically increase with the number of scenarios. In this respect, it is worth noting that our model is robust in relation to the number of scenarios used. In fact, in almost all the cases, the makespan found using only 20 scenarios is only a bit different (2% on average) from the makespan evaluated over 1000 scenarios. Nevertheless, we observe that in the case of considering thousand of scenarios, parallel computing could play a crucial role, and this represents the main goal for future development.

Chapter 4

Resource Constrained Project Scheduling Under Uncertainty

In this chapter, we study the resource constrained project scheduling problem under uncertainty. Project activities are assumed to have known deterministic renewable resource requirements and uncertain durations, described by random variables with a known probability distribution function. We propose a joint chance constraints programming approach to tackle the problem under study, presenting a heuristic algorithm in which the buffering mechanism is guided by probabilistic information.

4.1. Overview of the problem

The resource constrained project scheduling problem (RCPSP) consists in minimizing the duration of a project, subject to zero-lag finish-start precedence and resource constraints. In its deterministic version, the RCPSP assumes complete information on the resource usage and activities duration, and determines a feasible baseline schedule, i.e. a list of activity starting times, minimizing the makespan value. A solution for this problem is a baseline schedule which specifies, for each activity, the planned starting times. Notwithstanding its importance, the planned baseline schedule may have little, if some value, in real contexts since project execution may be subject to severe uncertainty and then may undergo several types of disruptions as described in the previous paragraphs. Extensions of the RCPSP, involving the minimization of the expected makespan of a project with stochastic activity durations, have been investigated within the stochastic project scheduling literature. The methodologies for stochastic project scheduling basically view the project scheduling problem as a multi-stage decision process, in which the objective is to minimize the expected project duration subject to zero-lag finish-start precedence and renewable resource constraints. Since the problem is rather involved and an optimal solution is unlikely to be found, scheduling policies (Igelmund & Radermacher (1983), Mohring & Stork (2000), Stork (2000)) and heuristic procedures (Ballestin (2007), Golenko-Ginzburg & Gonik (1998), Golenko-Ginzburg & Gonik (1997), Tsai & Gemmil (1998)) have been used for defining which activities to start at random decision points through time, based on the observed past and the a-priori knowledge about the processing time distributions.

Beside this important research area, the field of proactive (robust) project scheduling literature has received outstanding attention in the last years. It entails to incorporate some knowledge of the uncertainty in the decision-making stage, with the aim to generate predictive schedules that are in some sense robust (i.e. insensitive) to future adverse events.

Van De Vonder et al. (2005), (2006) propose the so-called resource flow-dependent float factor heuristic (RFDFF) to obtain a precedence and resource feasible schedule, using information coming from the resource flow network (Artigues et al., 2003) in the calculation of the so called activity dependent float factor (Leus (2003)). In Van de Vonder et al. (2007), several predictive reactive resource constrained project scheduling procedures are evaluated under the composite objective of maximizing both the schedule stability and the timely project completion probability. For an extensive review of research in this field, the reader is referred to Herroelen and Leus (2004b), (2005). Within this research stream, and when abstraction of resource usage is made, we mention the works Herroelen and Leus (2004a), Rabbani et al. (2007), Tavares et al. (1998).

When resource availability constraints are considered, Leus and Herroelen (2004), Deblaere et al. (2007) and Lambrechts et al.(2007) and (2008) assuming the availability of a feasible baseline schedule, proposed exact and approximate formulations of the robust resource allocation problem.

Within the stochastic programming context, a two-stage integer linear stochastic model has been proposed in Zhu et al. (2007). Target times are determined in the first stage followed by the development of a detailed project schedule in the second stage. The two-stage stochastic model aims at minimizing the cost of project completion and expected penalty incurred by deviating from the specified values. Only one non-renewable resource (the budget) is constrained in the model. A path based two-stage integer programming approach together with a tailored solution methodology based on decomposition has been recently proposed by Klerides & Hadjiconstantinou (2010) for the stochastic discrete time-cost trade-off problem.

The more difficult case involving multiple renewable resources has not been investigated yet in the stochastic programming setting.

This chapter addresses the case of RCPSP with renewable resources and uncertain activities durations represented by independent random variables with known cumulative probability distribution function. The objective is to build a precedence and resource feasible baseline schedule with minimum makespan able to tolerate a certain degree of uncertainty during execution and to absorb dynamic variations in activities durations (we shall refer in the following to this capability as stability or robustness). Our ultimate aim is to develop a project scheduling procedure capable of combining schedule stability and makespan performance. There are a number of different metrics for assessing robustness and stability of a schedule in literature. We adopt as a measure of stability the probability that schedule decisions do not change during execution. In particular, through the use of joint probabilistic constraints, we try to find a schedule that is expected to be respected with a high level of probability. The use of joint probabilistic constraints within the stochastic scheduling problem represents an innovative element of our approach. In effect, at the best of our knowledge, none of the methods proposed in the literature considers joint probabilistic constraints. Indeed, very few research papers explicitly consider probabilistic information in solution methods. We should mention here, the work of Van de Vonder et al. (2008) where the virtual activity duration extension (VADE) heuristic and the starting time criticality (STC) heuristic are introduced to include time buffers in a given schedule while a predefined project due date remains respected. While VADE heuristic relies on the standard deviation of the duration of an activity in order to compute a modified duration, STC heuristic tries to combine information on activity weights and on the probability that activity cannot be started at its scheduled starting time.

Our work differs from the cited paper in some important aspects. First of all, we consider the stochastic programming framework and, in particular, the probabilistic paradigm in the form of joint probabilistic constraints. This powerful tool allows us to relax the assumption, common in the literature, that only one activity at a time disturbs the starting time of a successor activity, rather limiting the joint probability of disruption of the preceding activities to a given probability level. In addition, we do not start from an initial deterministic unbuffered schedule in which to insert time buffers, although starting from an unbuffered schedule is a very common practice amongst practitioners and researchers. Last, but not least, our point of view is rather new in the literature on predictive stable scheduling procedures where the objective function commonly used (see for instance Leus, 2003) is the so called stability cost function, defined as the weighted sum of the expected absolute deviation between the actually realized activity start times and the planned activity start times. We observe that this objective function is not known a priori, unless a range of execution scenarios (referred to as the training set) are simulated by drawing different actual activities durations from the described distribution functions. In addition, it is not difficult to see that the exact quantification of the deviation between planned and actual starting times heavily depends on the reactive procedure adopted.

The remainder of the chapter is organized as follows. In paragraph 4.2 we describe our scheduling methodology for generating robust baseline schedules.

Paragraph 4.3 is devoted to the presentation of computational experiments and conclusions are given in paragraph 4.4.

4.2. Stochastic project scheduling with robustness constraints

4.2.1. Notation and problem description

We assume that the activity network of a resource-constrained project in activity-on-node representation is given by a directed acyclic graph $G =(N,$

A). Each node in the set $N = \{0, . . . , n + 1\}$ corresponds to a single project activity and each arc in the set A corresponds to a precedence relation between each pair of activities (Wiest & Levy, 1977). The activities 0 and n+1 are the dummy start and the dummy end activity, respectively.

For each activity $j \in N$, $FS(j)$ denotes the set of successor activities of j LS_j and LF_j indicate the latest starting time and the latest finish time of j, respectively.

We assume the presence of a set of K renewable resources with a per-period availability a_k. Each activity $j \in N$ has to be processed without interruptions, requiring a constant amount of resource r_{jk} for each renewable resource type k, $k = 1 . . . ,K$. We assume that the duration of an activity is represented by a stochastic variable and that the vector of durations $d \ \omega$ is distributed according to a joint probability distribution that follows a known distribution defined on a given probability space Ω equipped with an algebra F and with a probability measure $\mathbb{R}$. Then, the random vector of starting times can be denoted with $s \ \omega = s_0 \ \omega , \ldots , s_{n+1} \ \omega$ and the associated random vector of completion times with $f \ \omega = (f_0 \ \omega , \ldots , f_{n+1}(\omega))$. In uncertain environments, especially from a practical point of view, project managers are mainly interested in the generation of a proactive schedule (i.e a vector of proactive starting times $\sigma = (\sigma_{0,\ldots,}\sigma_{n+1})$ and finish times $\phi = (\phi_{0,\ldots,}\phi_{n+1})$ with a quality that does not degrade during execution with respect to future perturbations. The vector σ can be interpreted as the mapping of random activity durations into a vector of resource and precedence feasible starting times performed according to a function $\Pi : \Omega \rightarrow \mathrm{R}^{+}_{\mathrm{n+1}}$ referred as policy. If we introduce an additional variable

$$\Delta_j \ \omega = \begin{matrix} s_j \ \omega \ - \sigma_j \ if \ s_j \ \omega \ - \sigma_{j>0} \\ 0 \ otherwise \end{matrix}$$

representing deviation between the actual and the planned starting time for activity j, a natural question is how to construct an anticipative project execution policy and a vector of predictive starting and completion times that attempt to limit the risk of such deviation. Two classical approaches are used in stochastic programming to deal with deviations. Unit penalty costs can be assigned for each individual deviation, and the resulting expected penalty cost can be minimized, or alternatively, one may specify a model in which we accept deviations with a certain probability.

In project scheduling problems, when it is not easy to quantify the penalty associated with a schedule disruption, a risk based perspective could be preferred. Therefore, in this chapter, we adopt this point of view.
From a mathematical standpoint risk, averse constraints can be formulated using the theory of joint probabilistic constraints. More formally, in our setting a schedule is deemed robust if it fulfills the following constraint:
$\mathbb{P} = \Delta_j \geq 0, \forall\, j\ \in N) \leq \epsilon$ = with a low value of the risk parameter ϵ. Stating it differently, a schedule is robust if it exhibits a low probability of disruption. It is easy to recognize that the topic addressed in this chapter is closely related to the topic of solution stability addressed in Van de Vonder et al. (2005), also refereed as solution robustness. Whilst robustness has been considered in the literature mainly as an objective function to be minimized, in our work we introduce a robustness constraint, in the form of probabilistic constraints.
The stochastic RCPSP, as investigated in the present chapter, can be formally stated as follows: for a given resource-constrained project, with known activity duration distributions, construct a proactive schedule, with predictive starting times σ satisfying (1), that attempts to minimize the project makespan $C_{max}\ \Pi\ = \sigma_{n+1}$
It is worth noting that the model tries to handle what is called the stability/makespan trade-off, by accounting for solution stability through the use of joint probabilistic constraints, whereas the objective function mathematically translates the makespan minimization. In order to tackle this complex combinatorial stochastic problem, we develop a heuristic procedure that we shall present hereafter.

4.2.2. The heuristic procedure

As mentioned before, we are interested in determining a policy Π and a vector of predictive starting times σ such that the makespan of the schedule is minimized and the risk of disruption is limited from above in probability by the parameter ϵ.
A scheduling policy under uncertainty may be seen as a stochastic dynamic decision process that defines which activities to start at certain decision points t, based on the knowledge of the observed past up to t and the statistical distributions of activities durations. A decision point occurs either

at the beginning of the project, or when at least one of the running activities is completed, until the last activity is scheduled.

Our heuristic approach is based on a stage-wise approximation of the full complex stochastic dynamic problem relying on the decoupling of the dynamic from the stochastic aspect of the problem.

In particular, the dynamicity of the problem is treated at a higher level, viewing the project as a sequence of decisions on resources allocation whereas the probabilistic aspect is tackled at decision points. At each decision point, a resource feasible partial schedule is built and suitable proactive starting and completion times are set by means of an anticipative stochastic model that accounts for future uncertainty. Since the policy we use can be viewed as a stochastic dynamic version of the parallel schedule generation scheme, it is easy to verify that the partial schedule constructed is feasible with respect to precedence and resource constraints.

A detailed description of the proposed stochastic dynamic generation scheme (SDGS) heuristic is given in what follows. Let denote with: g the iteration counter;

- t_g the decision time associated to the iteration g;
- $A_g$the set of activities which are active at t_g;
- E_g the set of activities whose predecessors at time t_g have been completed;
- S_g a subset of E_g comprising activities that will start at time t_g;
- $R_k(t_g)$ the residual resource availability at time t_g;
- ρ a priority rule.

An algorithmic description of the SDGS heuristic is given below.

Scheme of the SDGS heuristic

```
Initialization
Set $g := 0, E_g := \{0\}, A_g := \{\emptyset\}, S_g := \{\emptyset\}, t_g := 0, R_k(0) := a_k, k = 1,\dots,K$
Choice a priority rule $\rho$
Repeat until all the activities have been scheduled:
    • Compute $E_g$ Repeat until it is not possible to select activities:
    Step 1. Use the priority rule $\rho$ to select a new activity $j \in E_g$ to be
    scheduled
    $E_g \coloneqq E_g\{j\}$
    Step 2. If j is such that $R_k \; t_g \; - r_{jk\_}$ 0 then
    $S_g := S_g \cup \{j\}$
    $A_g := A_g \cup \{j\}$
    $R_k(t_g) := a_k - \; _{j \in A_g} r_{jk}$
    $\sigma_j = t_g$
    Determine the proactive completion times
        $\phi_j \, \forall \, j \in S_g$
    taking into account constraint (1)
    Else
    Go to Step 1.
    End If
End Repeat
• Set the next decision point as follows:
    $t_{g+1} := min_{j \in A_g} \phi_j$
    $g := g + 1$
    End Repeat
```

4.2.3. Generating activities completion times

While the dynamic aspect of the problem is tackled at decision points by the scheduling policy presented above, the stochasticity should be taken into account in the determination of proactive starting and completion times. In our problem, predictive starting and completion times should fulfill at each decision point the stability constraint (1).
We recall that at each decision point t_g, our decisions concern the appropriate proactive completion times of activities in Sg since the starting times of activities j ∈ S_g are set as $\sigma_j = t_g$, where t_g represents a completion time of one previously scheduled activity. As a reminder we note that these decisions should ensure the satisfaction of the stability constraint (1). We now observe that potentially any activity in A_g could cause a disruption among its successor, as far as its completion time represents the new decision point at which an unscheduled activity *j* should set its starting time σ_j.

We further notice that at the time we take decisions we do not know which activities would be scheduled in the next stage, nor what will be the next decision point.

Therefore, at a generic instant $t_g \leq t \leq t_{g+1}$ the probability of not causing a disruption in the schedule in the future is the probability that for any activity $j \in A_g$ currently under execution the condition $\phi_i \geq f_i \omega$ is verified.

Now, we may conclude that the problem boils down in appropriately setting completion times by solving at each decision point t_g the following problem with joint chance constraints:

$\min \mathrm{Mpar}$ 2

$\mathrm{Mpar} \geq \phi_i \, \forall \, i \in S_g$ (3)

$P \; \phi_i \geq t_g + d_i \; \omega \;\; \forall \, i \in S_g \; \geq 1 - \epsilon$ (4)

Where *Mpar* represents the makespan of the partial schedule built considering activities $i \in S_g$ and $t_g + d_i \; \omega = f_i \; \omega \;\; \forall i \in S_g$. Here joint chance constraints are imposed to set the completion time of the activities, at each decision point, in such a way that the probability of not disrupting the schedule in the future is at least $(1 - \epsilon)$ (i.e. the risk of disruption is at most ϵ). We emphasize that completion times of the activities in $A_g \setminus S_g$ have been set appropriately at a previous decision point.

In order to show how the heuristic works, we consider the toy example reported in Figure 4.1, with only five activities plus the two dummy activities

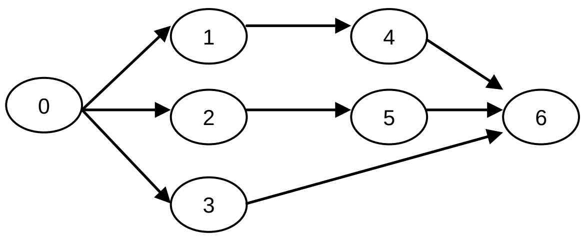


Figure 4-1: Toy example

0 and 6. It is assumed that only one resource is required to execute the activities (*i.e., $k = 1$*) and the resource consumptions are: $r_{11} = 2, r_{21} = 2, r_{31} = 1, r_{41} = 1, r_{51} = 2$ *and* $a_1 = 5$. In addition, activities durations follow a Poisson distribution with mean *1* for activities *1, 2, 3* and *0.5* for activities *4* and *5*.

The activities have been ordered by the rule ρ as follows: $< 1, 2, 3, 4, 5 >$.

After the dummy activity *0* has been scheduled activities *1, 2, 3* can be scheduled at time $t_0 = 0$ ($S_0 = \{1,2,3\}$ since they do not cause any resource conflict. Therefore $\sigma_1 = \sigma_2 = \sigma_3 = 0$). Problem (2)-(4) is then invoked and completion times ϕ_1, ϕ_2 and ϕ_3 are appropriately set. The next decision point $t_1 = 2$. At that time, activity *4* is an eligible activity and there are sufficient resources units available, so it is started at time t_1. For activity *5* no sufficient resource units are available. Therefore $S_0 = \{4\}$ and problem (2)-(4) reduces to a problem with a single chance constraint. The next decision point $t_2 = 3$. At this time activity *5* is started.
The resulting feasible schedule is depicted in Figure 4.2.

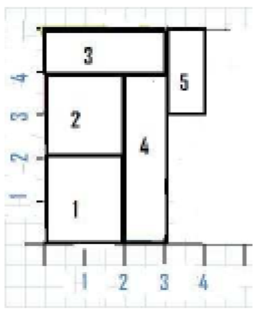


Figure 4-2: Resulting schedule

disruption probability of activity *4* depends on the disruption probability of activities *1* and *2*, whereas it is not influenced by the completion time of activity *3*. By imposing a threshold risk parameter of $\epsilon = 0{,}2$ our heuristic set completion times of activities *1, 2, 3* in such a way that:

$$\mathbb{P} \begin{array}{l} \phi_1 \geq f_1\ \omega \\ \phi_2 \geq f_2\ \omega \\ \phi_3 \geq f_3\ \omega \end{array} \geq 1 - 0.2$$

thus limiting the disruption probability of activity *4*. We should point out that if, at least in principle, separate chance constraints can be used to deal with uncertain durations, the solution provided by the corresponding model may in some context be considered inappropriate. In fact, imposing a small probability of disruption for each activity $j \in S_g$ does not assure a small joint probability for all $j \in S_g$.

In our example, the probability that activity 4 cannot be started at its scheduled starting time $t_1 = 2$ depends on the probability of the event that predecessors activities 1 and 2 disturb its planned starting time.

If chance constraints would have been used with ϵ = 0.2, the probability of disrupting activity 4 would have been $1-(1-0.2)*(1-0.2)=0.36$, notwithstanding the fact that the probability of having each activity disrupted is less than 0.2.

It is worth observing that, although at each decision point we accept the risk of a disruption with probability ϵ, we cannot impose a limit on the probability of not completing the whole project on time. However, we may express the timely project completion probability as a function of the number of decision points performed by the algorithm. A crude lower bound for the probability of project to be completed on time is $1 - \epsilon^{\mathrm{G}}$ where G is an upper bound on the number of iterations.

4.2.4. Solving the joint probabilistically constrained problem

The SDGS heuristic involves the repeated solution of model (2)-(4). In the following, we show how to derive a deterministic equivalent formulation, in the case of independent random variables. Under the independence assumption among the random variables $\mathrm{d_i}\ \omega$, the probabilistic constraints (4) can be rewritten as

$$_{i \in S_g}\mathbb{P}(\phi_i \geq t_g + \mathrm{d_i}\,\omega\,) \geq (1 - \epsilon) \qquad (5).$$

Denoting with $F_{\mathrm{d_i}\ \omega}$) the marginal probability distribution function of the random variable $\mathrm{d_i}\ \omega$, and with a variable substitution $\psi_i = \phi_i - t_g$, constraints (5) can be stated equivalently as

$$\underset{i \in S_g}{} F_{\mathrm{d_i}\,\omega}(\psi_i) \geq (1 - \epsilon)$$

and by taking logarithms:

$$\underset{j \in S_g}{} \ln F_{\mathrm{d_i}\,\omega}(\psi_i) \geq \ln(1 - \epsilon)$$

(Jagannathan, 1974; Miller & Wagner, 1965). Since the logarithm is an increasing function and $0 \leq F_{\mathrm{d_i}\ \omega} \leq 1$, this transformation is legitimate. Furthermore, for log–concave distribution functions, convexity of the constraints is preserved. The class of log–concave random variables includes several commonly used probability distributions as for example the Uniform, Normal, Exponential and many others (see Prékopa, 1995, Dentcheva et al. 1998). We observe that also in the case of discrete distributions, problems with joint probabilistic constraints can be reduced to deterministic equivalent problems. For more details, the interested readers are referred to Dentcheva et al. (1998) (In Appendix B the transformation is detailed for illustrative purposes).

Therefore, depending on the nature continuous or discrete of the random variables involved in the problem at hand, the deterministic equivalent problem takes the form of a nonlinear continuous problem or a linear integer problem.

4.3. Computational Experiments

This section is devoted to the presentation and the discussion of the computational experiments carried out with the SDGS heuristic. The numerical tests have been designed to evaluate the performance of the SDGS heuristic in comparison with a set of benchmark heuristics that we shall present in the next paragraph.

4.3.1. Benchmark approaches

In order to assess the performance of the proposed SDGS heuristic, we have considered for comparison a set of scheduling procedures based on the use of separate chance constraints. More specifically, both parallel schedule generation schemes (PSGS) and serial schedule generation schemes (SSGS) have been designed by replacing deterministic durations by their $1-\epsilon$ -quantile counterparts $F^{-1}_{\mathrm{d_i}\ \omega}\ 1-\epsilon$. We should remark that this is equivalent of using separate chance constraints within classical schedule generation heuristics for the deterministic RCPSP. The following static priority rules for generating the priority list have been tested.

• (MaxC): The MaxC rule orders the activities $j \in N$ by decreasing value of their total resource requirement $r_j = \ {}_{k=1}^{K} r_{ik}$.

• (MinC): The MinC rule orders the activities$j \in N$ by increasing value of their total resource requirement $r_j = \ {}_{k=1}^{K} r_{ik}$.

• (MaxD*C) The MaxD*C rule orders the activities by decreasing value of $F^{-1}_{\mathrm{d_i}\ \omega}\ \ 1-\epsilon\ \ \ *r_j$ with ψ_j defined as above.

• (MinD) The MinD rule orders the activities by increasing value of $F^{-1}_{\mathrm{d_i}\ \omega}\ \ 1-\epsilon$.

• (LST) The LST rule orders the activities by increasing value of their latest starting time LS_j as described in Kolisch & Hartmann (1999).

• (LFT) The LFT orders the activities by increasing value of their latest finish time LF_j as described in Davis & Patterson (1975).

• (MTS) The MTS orders the activities by decreasing value of the number of their successors, that is $|FS_j|$ as described in Alvarez-Valdes & Tamarit (1989).

Whilst the last three rules have been taken from the literature (Kolisch & Hartmann, 1999), the other rules have been proposed by the authors.
Moreover, we have considered the STC (Van de Vonder et al., 2008) and the RFDFF heuristic (Van de Vonder et al., 2006).

4.3.2. Computational results

The computational experiments have been performed on a PC Pentium III, 667 MHz, 256 MB of RAM. All procedures were coded in AIMMS language (Bisschop & Roelofs, 2007) and the subproblems solved with Cplex 10.1 (ILOG CPLEX 6.5: Users Manual, 1999) and Conopt (Drud, 1996). Algorithms 1-4 are the PSGS with the first four priority rules, algorithms 5-8 are the SSGS with the same priority rules, algorithms from 9 to 11 are PSGS with the rules LST, LFT and MTS, respectively. The SDGS heuristic procedure has been executed considering the four priority rules MaxC, MinC, MaxD*C and MinD (algorithms A-D) since they behave the best. The STC and RFDFF heuristics algorithms consider a deterministic project due date and start from a minimum makespan schedule in which time buffers are inserted in order to protect against anticipated disruptions. The unbuffered baseline schedule required by the STC and RFDFF heuristics has been obtained by applying procedure 4 (PSGS heuristic with MinD rule) with different ϙ values (this choice has been motivated by the fact that SDGS heuristic is a stochastic version of the PSGS) whereas the project due date has been set equal to the average makespan obtained by procedures A-D. With this setting, the comparison turns out to be fair in terms of resulting makespan.

The instability weights w_j have been considered equal to one for all the activities, except for the final one for which the weight has been set equal to as suggested in Van de Vonder et al., 2008. This particular setting reflects the fact that in our heuristic the probabilistic constraints are imposed on all the activities with the same value for the risk parameter ϵ which, in turn implies that all the activities are considered equally important. Furthermore, since the objective of our heuristic is the makespan minimization, the weight of the last activity has been set to 10 in order to give more emphasis on the project makespan (in this respect it could be beneficial to recall that the weight of the dummy end activity denotes the cost of delaying the project completion beyond a predefined deterministic project due date). For the STC heuristic the stability cost improvement has been evaluated on a training set of 100 scenarios.

All the scheduling procedures have been executed for 5 values of ϵ namely $\{0.2, 0.15, 0.1, 0.05, 0.01\}$.

The computational experiments have been carried out on a set of benchmark problems selected from the project scheduling problem library PSPLIB (Kolisch and Sprecher, 1997), available at http ://129.187.106.231/psplib/,including 30, 60 and 90 nodes, leading to a total of 2550 runs.

For all the instances, two types of distribution have been tested in order to assess the effectiveness of the proposed approach with both continuous and discrete distributions.

In particular, for the continuous case, we have assumed that real activity duration is a uniform random variable U(0.75d, 2.85d), where d has been set equal to the deterministic duration and for the discrete case, we have considered a Poisson distribution with mean d. Activity durations are assumed to be independent.

Extensive simulation has been used to evaluate all procedures on robustness measures and computational efficiency. For every network instance, 1000 scenarios have been simulated by drawing different actual activity durations from the described distribution functions. Using these simulated activity durations, the realized schedule is constructed by applying the following reactive procedure. An activity list is obtained by

ordering the activities in increasing order of their starting times in the proactive schedule.
Ties are broken by increasing activity number. Relying on this activity list, a parallel schedule generation scheme builds a schedule based on the actual activity durations. We opted for the railway execution mode (Deblaere et al., 2007) never starting activities earlier than their prescheduled start time in the baseline schedule. Actually, this type of constraint is inherent to course scheduling, sports timetabling, railway and airline scheduling, or when activity execution cannot start before the necessary resources have been delivered.

4.3.3. Analysis of results

Rather than showing the complete set of the numerical results, fully reported in Beraldi et al. 2007, we give in Tables 4.1-4.6, for each procedure, the average results calculated over all networks and executions. The quality has been evaluated by the following a posteriori measures of stability: average tardiness (Tavg), average timely project completion probability (TPCP), average disruption probability over all networks and executions (Davg). Also the predictive makespan (Mak) has been reported enabling a fair comparison amongst the different algorithms.
In the next subsection we will present our results for the discrete and the continuous case. Here we shall briefly comment on the computational times since they are rather low and do not constitute a bottleneck for the algorithms execution. In particular, the CPU time is very limited for algorithms 1-11, given the simple schedule construction procedures based on the parallel and serial schedule generation scheme. Also procedures A-D are competitive in terms of timely performance, with CPU times varying from 0 to a couple of seconds. The execution times of procedures A-D, is slightly higher than the computational time of STC and RFDFF only for 90 nodes networks and discrete distribution. This is due to the extra effort required for solving, at each decision point, the integer linear deterministic equivalent of the probabilistic model. For the continuous case, the computational requirements for SDGS and the STC and RFDFF heuristics are roughly similar, although the model to be solved within the SDGS heuristic is a nonlinear continuous model.

In order to give an idea of the size of the probabilistic problems solved within the SDGS heuristic, Table 7 reports the average, minimum and maximum number of variables and constraints involved in the solution of problem (2)-(4) at each iteration of the SDGS heuristic for both the discrete and the continuous case. For the sake of completeness also the average number of iterations performed is reported. As evident a higher number of variables and constraints characterizes the integer deterministic equivalent problems related to discrete random variables.

4.3.3.1. Discrete distribution

In this section, we comment on computational results obtained for the discrete case. A detailed accounting of the numerical results is reported in Tables 4.1-4.3.

A first set of numerical experiments has been carried out with the aim of assessing the variation of the performance measures of our SDGS algorithm as a function of the risk level (measured by ϵ).

We report in Figures 4.3 and 4.4 the Tavg and the TPCP for different ϵ values, for the 30 nodes test problems. As we can observe in Figure 1, the average tardiness decreases with ϵ. This is an expected result since for decreasing value of ϵ we impose a more prudent project manager's position imposing a higher risk aversion level. Mathematically speaking, as the value of ϵ decreases, probabilistic constraints are somehow more binding and the schedule is more robust, since it is less exposed to disruptions. The opposite trend can be observed in Figure 4.4 for the TPCP which increases for decreasing ϵ values.

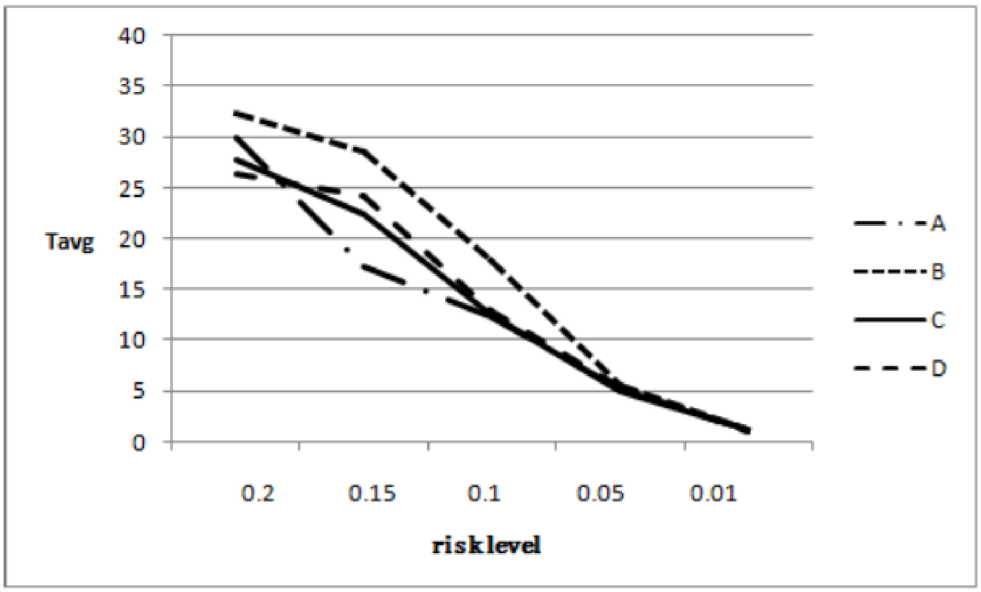


Figure 4-3: ϵ values versus Tavg trade-off

A second set of experiment has been carried out to compare the performance of the SDGS with respect to the benchmark approaches presented in Section 4.3.1. In particular, we shall present hereafter a graphical comparison on the basis of the expected makespan EXPMAK, (obtained as the sum of the predictive makespan Mak plus the expected tardiness Tavg) and the average probability of disruption Davg.

Figures 4.5, 4.6 and 4.7 show the EXPMAK for the 30, 60 and 90 nodes networks respectively. Average values have been reported for the procedures A-D (named SDGS) and 1-11 (named OTHERS). The STC heuristic has been included in the graph whereas the RFDFF heuristic has been excluded from the comparison since it is always largely outperformed, as evident from Tables 4.1-4.3.

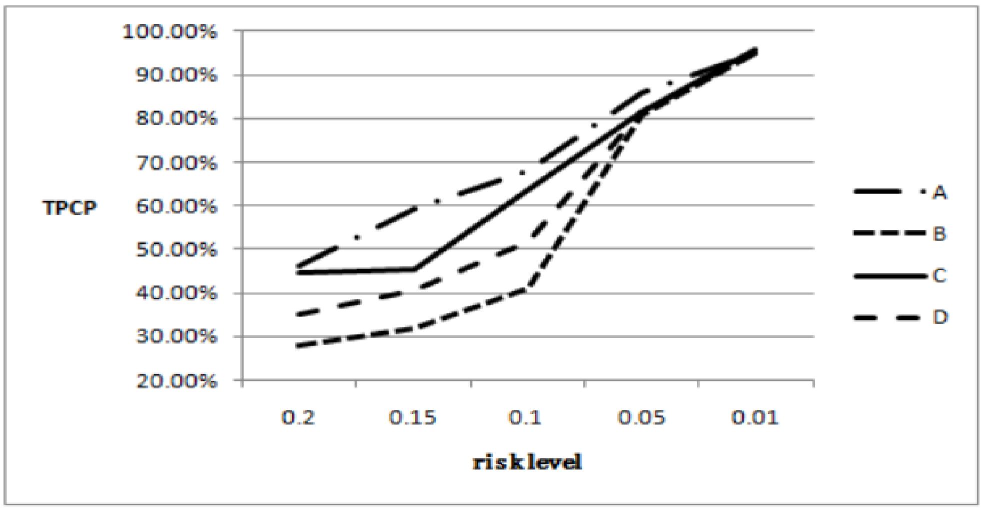


Figure 4-4: ϵ values versus TPCP trade-off

The reader may notice a seemingly strange trend in the results. In effect, the expected makespan seems to have a non-monotone behaviour, with a decreasing slope up to the minimum and an increasing or constant slope afterwards. This behaviour is more evident for the 30 nodes networks, and in general it is relevant for algorithms from 1 to 11 and for the STC in the case with 90 nodes. This unforeseen descendant behaviour of the expected makespan is due to the influence of two opposing forces that are in effect. As depicted in Figure 4.8, on the one hand there is a predictive makespan (Mak) whose value increases as ϵ decreases, and on the other hand the expected tardiness (Tavg) that drastically reduces as long as the risk we are willing to bear decreases.

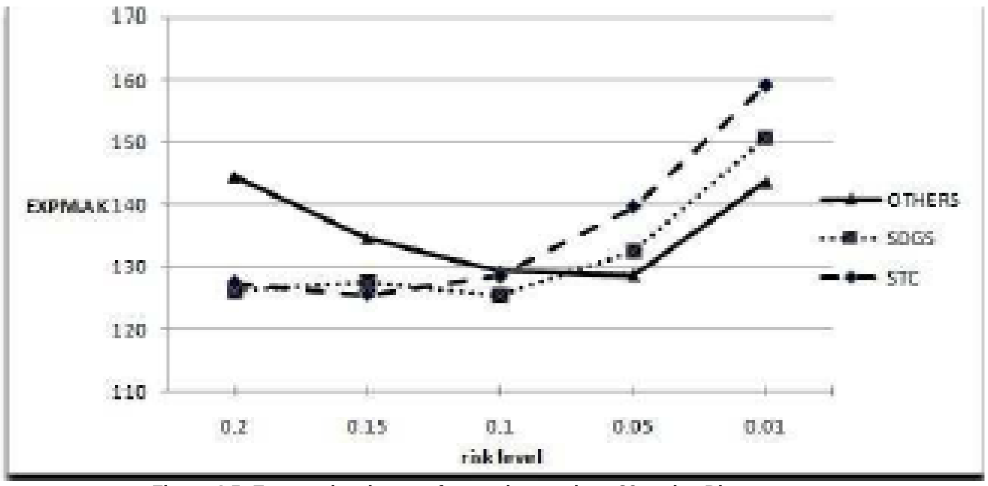


Figure 4-5: Expected makespan for varying ϵ values-30 nodes-Discrete case

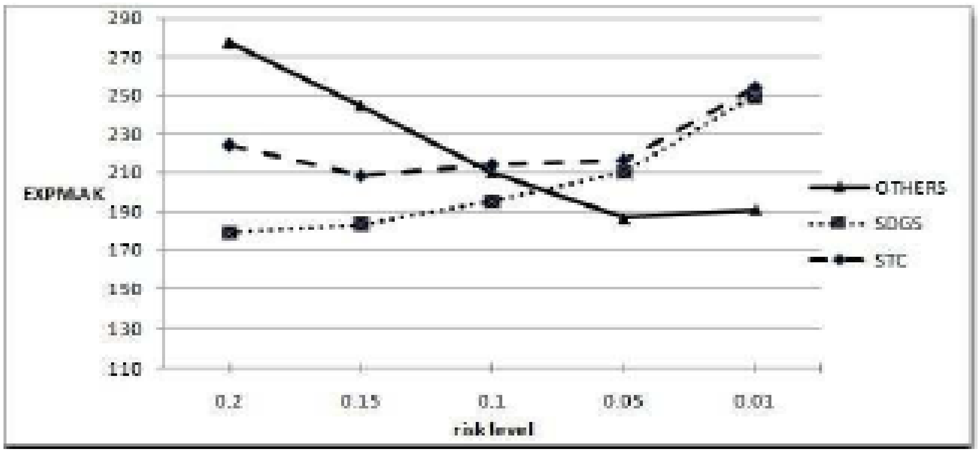


Figure 4-6: Expected makespan for varying ϵ values-60 nodes-Discrete case

It is immediately clear from Figures 4.5, 4.6 and 4.7 that the expected makespan of SDGS is in general smaller than that of STC, but the same is not thoroughly true for procedures 1-11. It is worth observing, in fact, that there is an intersection between the continuous line of procedures 1-11 and the dashed line of SDGS for between 0.05 and 0.1. Therefore, for relevant risk levels (ϵ between [0.05, 0.2]) SDGS should be preferred in terms of EXPMAK.

An opposite behaviour emerges for lower risk levels. In practice, risk-averse project managers, for budget restrictions, may accept to bear some risk to avoid unnecessary extra costs. Therefore, the range [0.05, 0.2] constitutes a meaningful choice for moderately risk averse project managers.

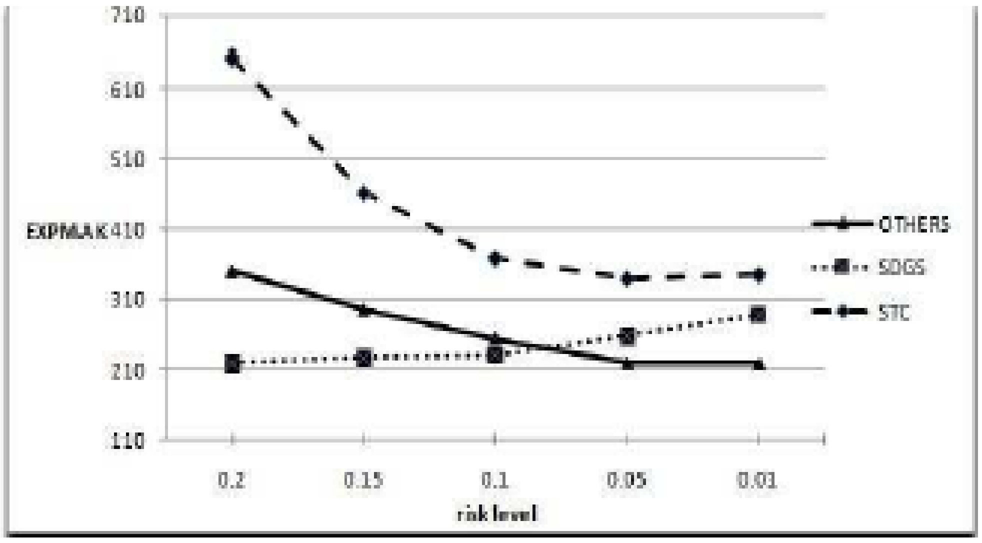


Figure 4-7: Expected makespan for varying ϵ values-90 nodes-Discrete case

Figures 4.9, 4.10 and 4.11 investigate the comparative performance of the algorithms in terms of Davg. We may observe that SDGS exhibits the best performance with very low Davg, especially for large networks, as evident in Figure 9. This claim is supported by the consideration that the Davg gap between SDGS and STC algorithms increases with the dimension of the network. With respect to the comparison between this two algorithms, there is some evidence on the superiority of SDGS over STC for the stability measures considered up to this point. This superiority is also supported by the Tavg values which can be unacceptably high for both the STC heuristic and the others benchmark heuristics considered. For instance in Table 4.3 we can observe that the Tavg of the benchmark heuristics is on average more than one order of magnitude higher than the Tavg of SDGS.

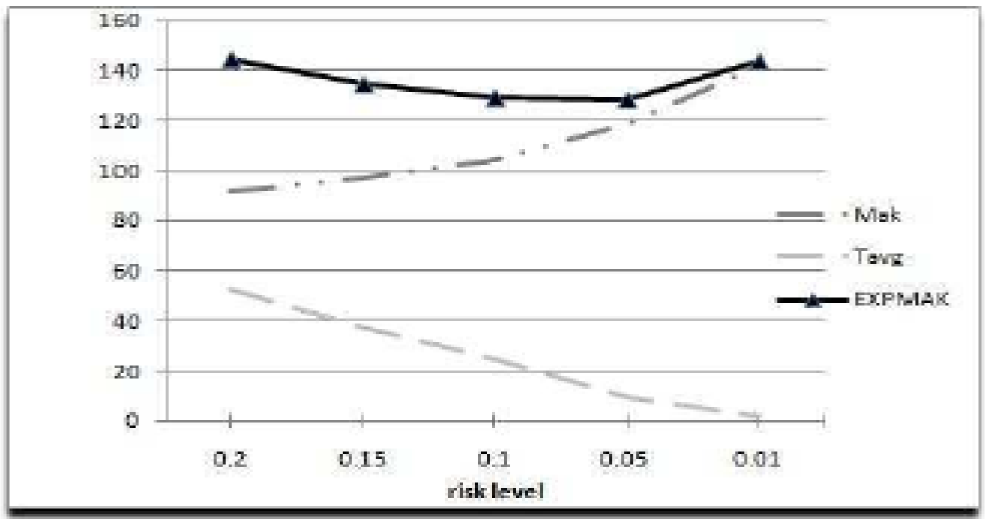


Figure 4.8: Expected makespan components for varying ϵ values-30 nodes-Discrete case

In effect, an apposite behaviour can be observed for the TPCP for which it is evident the superiority of the STC over all the algorithms considered. We would like to remark that in the worst case, the TPCP of the STC heuristic doubles the TPCP of the SDGS.

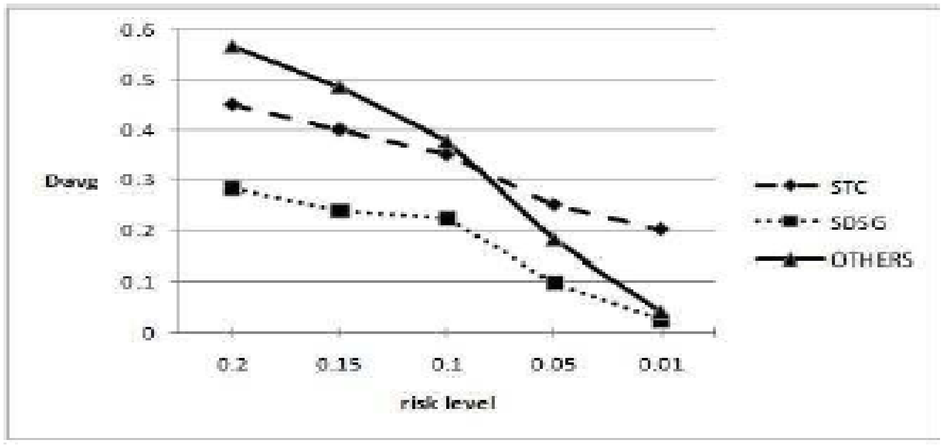


Figure 4.9: Davg for varying ϵ values-30 nodes-Discrete case

Nonetheless, we observe that there is no unitary evidence on the superiority of STC over procedures 1-11. In fact, we observe that for ϵ values higher than 0.075 (which represents approximately the intersection point in Figures 4.9 and 4.10) STC behaves worse than procedures 1-11, whereas the opposite is true for ϵ values below 0.075. This suggests that there is a golden value for the risk parameter that could guide the manager in the choice of the appropriate heuristic to use.

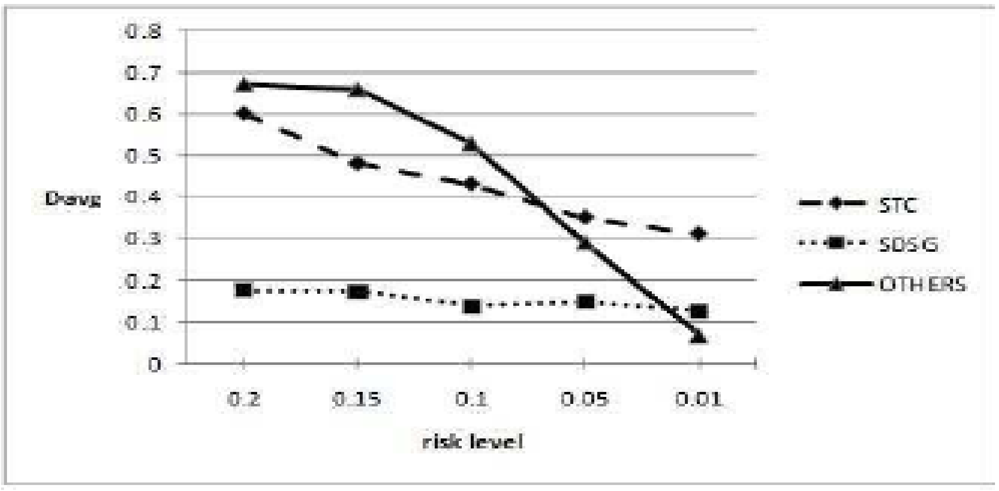


Figure 4.10: Davg for varying ϵ values-60 nodes-Discrete case

If solution stability is deemed of utmost importance, the best choice seems to be the SDGS heuristic. This heuristic guarantees very good stability performance in terms of disruption probability. If, on the contrary, the sensitivity of the schedule performance in terms of the objective value is the criterion to pursuit, we observe that nice results are obtained for $\epsilon \geq 0.01$ by the STC heuristic with high TPCP and also acceptable stability indicators.

When the project manager is very conservative and risk averse ($\epsilon \leq 0.05$) an attractive alternative especially for large instances can be constituted by procedures 1-11 that offer a good comprise between computational time and solution quality. However, above this risk level they fall inevitably in solutions of substantially lower quality.

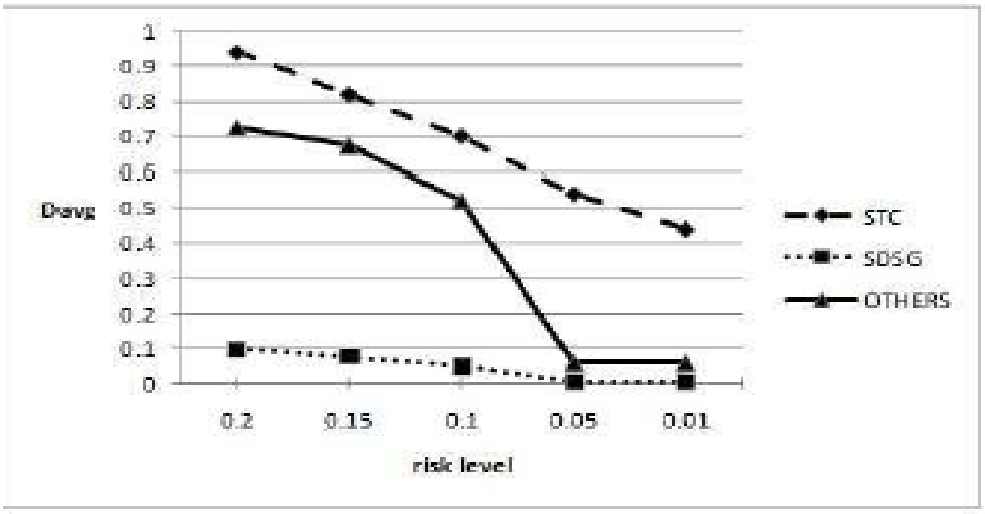


Figure 4.11: Davg for varying ϵ values-90 nodes-Discrete case

As a marginal note, we point out that the performances of the algorithms 1-11 are barely indistinguishable and depend on the ordering criterion adopted. Unfortunately, there is no unitary evidence of one criterion over the others.

As far as the RFDFF is concerned, we observe that notwithstanding the unbuffered schedule fed into RFDFF depends on the ϵ value considered, the results obtained are almost the same whatever the risk aversion of the decision maker is. This behaviour can be due to the right-justification mechanism, which insert buffers in front of the activities in order to make the schedule solution robust.

4.3.3.2. Continuous distribution

Tables 4.4, 4.5, and 4.6 summarise the results for the continuous distribution function. Some conclusions can be drawn on the basis of the numerical results collected.

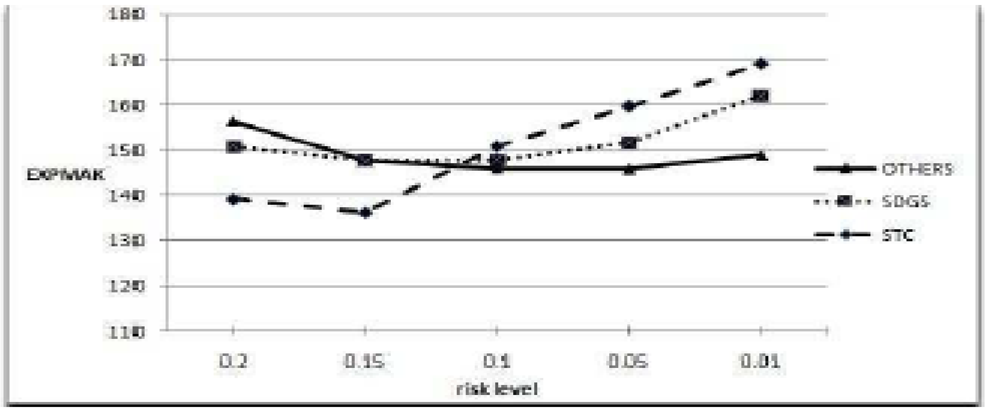


Figure 4.12: Expected makespan for varying ϵ values-30 nodes-Continuous case

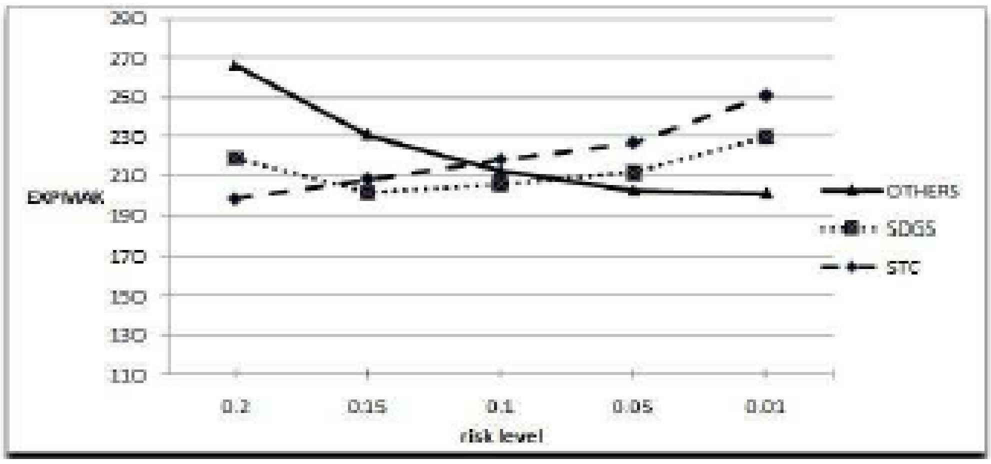


Figure 4.13: Expected makespan for varying ϵ values-60 nodes-Continuous case

As before, we show in Figures 4.12, 4.13 and 4.14 the EXPMAK for the 30, 60 and 90 nodes networks respectively. We observe that the general trend is similar to the one observed for the discrete case, albeit with some differences. We notice that the performance in terms of EXPMAK of the benchmark heuristics (excluding as before the RFDFF) is now comparable to the performance of the SDGS, at least for the 30 and 60 nodes networks.

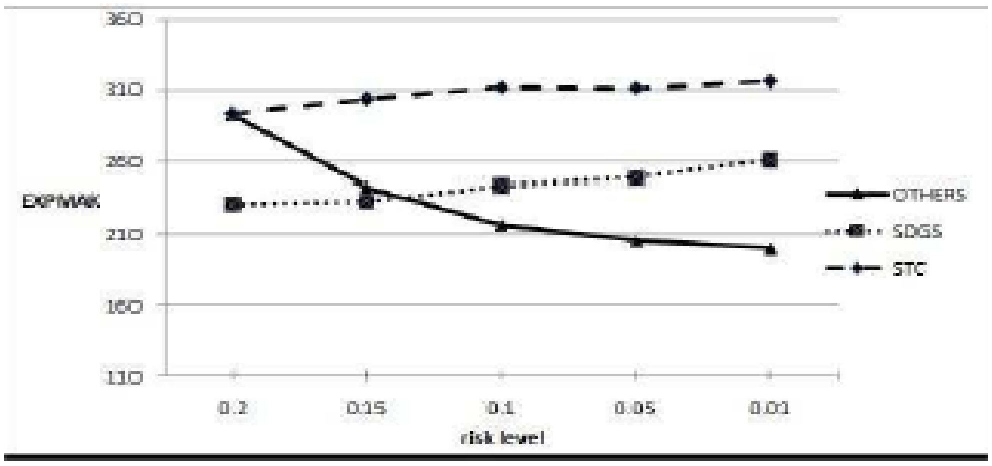


Figure 4.14: Expected makespan for varying ϵ values-90 nodes-Continuous case

As already observed in the discrete case, also in this case procedures 1-11 outperforms SDGS and STC in the expected makespan for ϵ values between 0.1 and 0.15.

We further observe that in this case, STC outperforms SDGS for values above 0.1 for the 30 nodes network and above 0.15 for the 60 nodes network.

The EXPMAK of the STC for the network with 90 nodes is on the contrary quite high. This worsening in the EXPMAK is compensated by an higher TPCP for the SDGS, as evident from Figure 4.15. Indeed, also in the others test problems considered, the SDGS heuristic shows TPCP value closer to the STC values than in the discrete case (see Tables 4.4, 4.5 and 4.6).

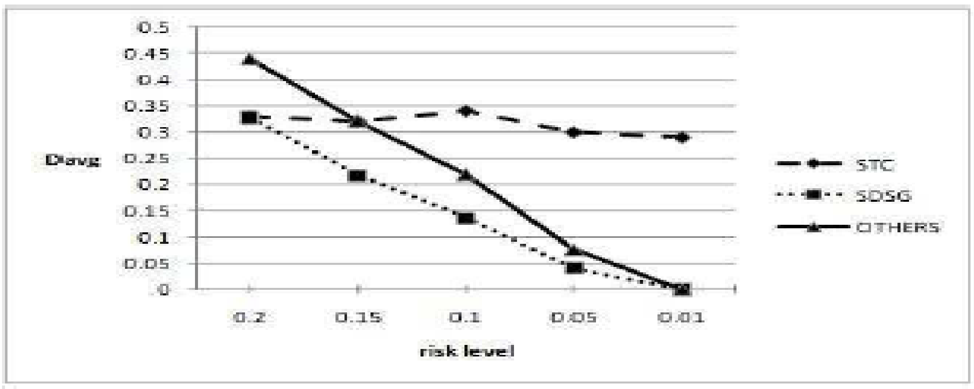


Figure 4.15: TPCP for varying ϵ values-60 nodes-Continuous case

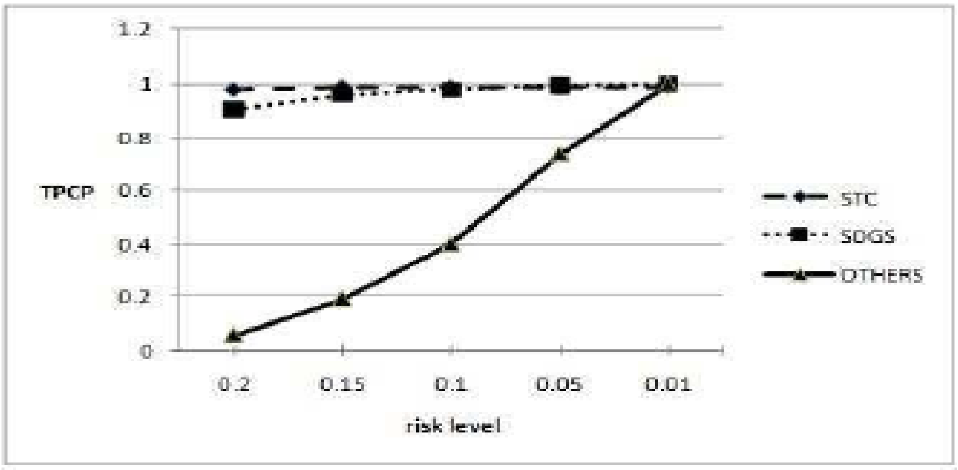


Figure 4.16: Davg for varying ϵ values-30 nodes-Continuous case

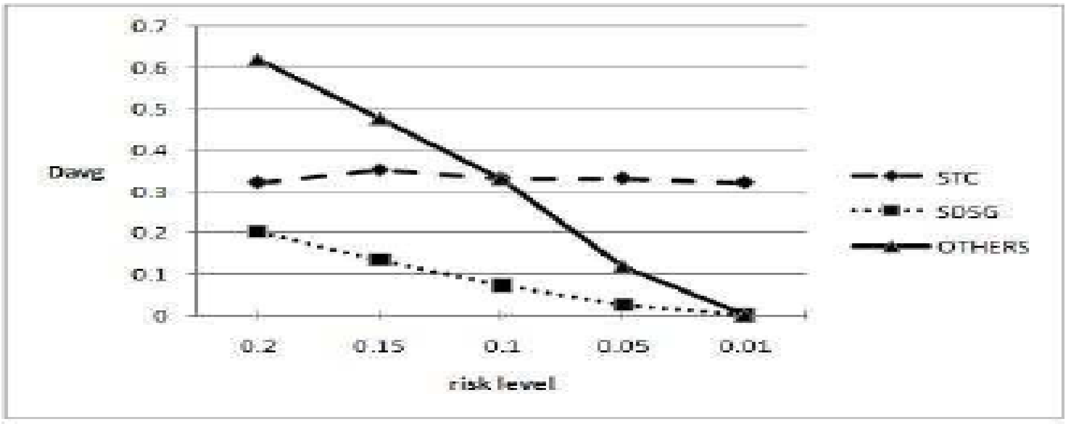


Figure 4.17: Davg for varying ϵ values-60 nodes-Continuous case

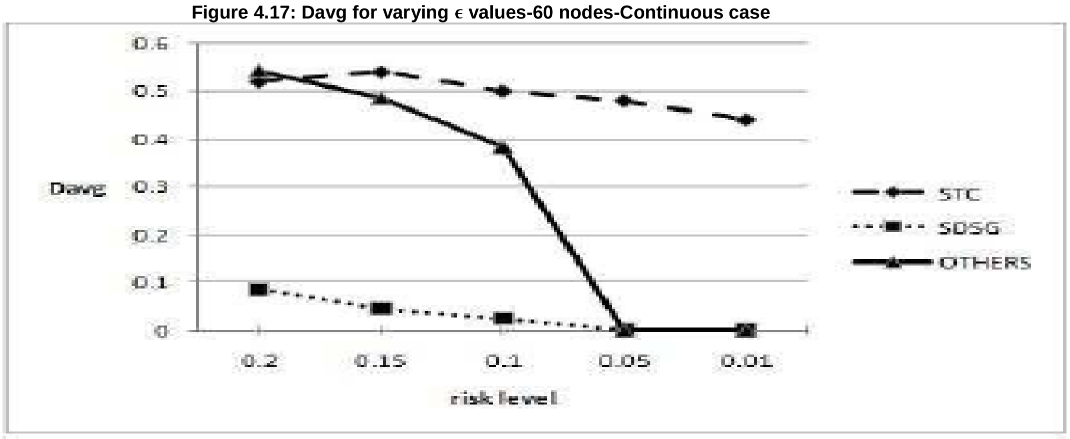
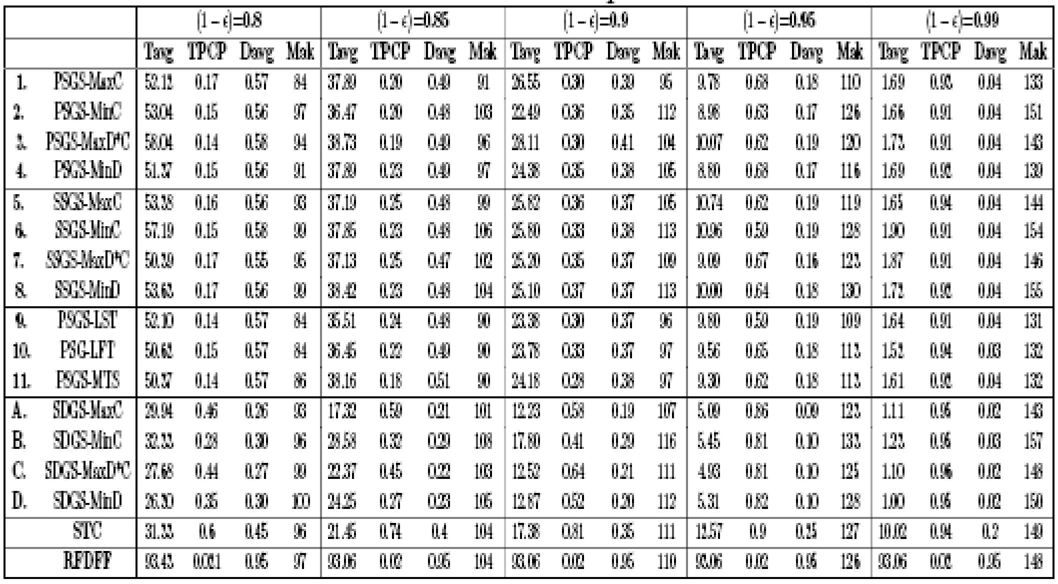


Figure 4.18: Davg for varying ϵ values-90 nodes-Continuous case

As a byproduct, we observe that the STC heuristic seems to be less sensible to the variation of the risk value, with Davg quite high, especially if compared with the SDGS values. This behaviour can be observed in Figures 4.16, 4.17 and 4.18. It is also worth noting that the Davg of the SDGS heuristic is very low, falling down to zero for small ϵ values.

4.4. Conclusions

We proposed a joint chance constrained model for project scheduling problem with robustness constraints and developed a heuristic procedure for its solution. The heuristic exploits probabilistic information on random activities duration within the framework of joint probabilistic constraints.

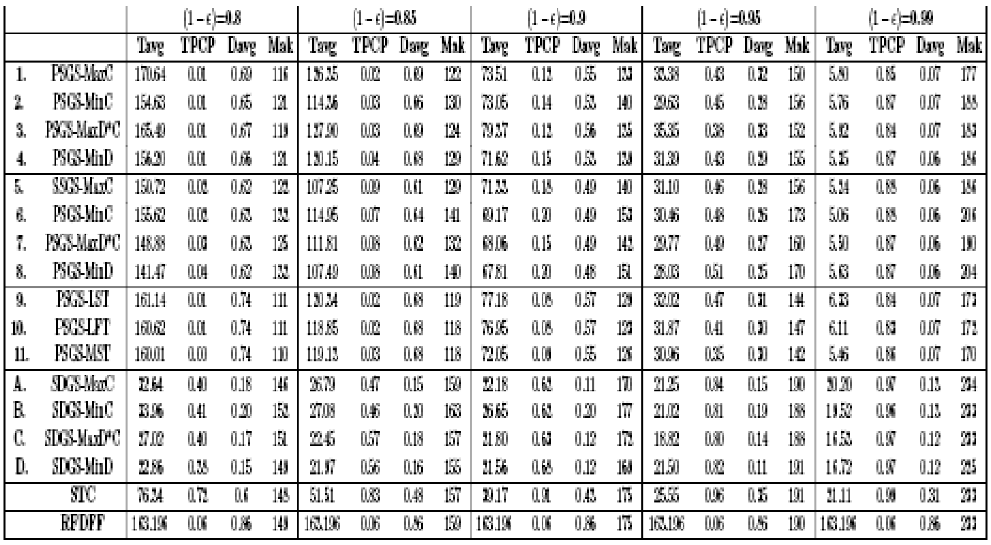


		$(1-\epsilon)=0.8$				$(1-\epsilon)=0.85$				$(1-\epsilon)=0.9$				$(1-\epsilon)=0.95$				$(1-\epsilon)=0.99$			
		Tavg	TPCP	Davg	Mak	Tavg	TPCP	Davg	Mak	Tavg	TPCP	Davg	Mak	Tavg	TPCP	Davg	Mak	Tavg	TPCP	Davg	Mak
1.	PSGS-MaxC	52.13	0.17	0.57	84	37.80	0.20	0.40	91	26.55	0.30	0.30	95	9.78	0.68	0.18	110	1.69	0.93	0.04	133
2.	PSGS-MinC	53.04	0.15	0.56	97	36.47	0.20	0.48	103	22.49	0.36	0.35	112	8.96	0.63	0.17	126	1.66	0.91	0.04	151
3.	PSGS-MaxD*C	58.04	0.14	0.58	94	38.73	0.19	0.40	96	28.11	0.30	0.41	104	10.07	0.62	0.19	120	1.73	0.91	0.04	143
4.	PSGS-MinD	51.37	0.15	0.56	91	37.80	0.23	0.40	97	24.38	0.35	0.38	105	8.80	0.68	0.17	116	1.69	0.92	0.04	130
5.	SSGS-MaxC	53.38	0.16	0.56	93	37.19	0.25	0.48	99	25.82	0.36	0.37	105	10.74	0.62	0.19	119	1.65	0.94	0.04	144
6.	SSGS-MinC	57.19	0.15	0.58	90	37.85	0.23	0.48	106	25.80	0.33	0.38	113	10.96	0.50	0.19	128	1.90	0.91	0.04	154
7.	SSGS-MaxD*C	50.39	0.17	0.55	95	37.13	0.25	0.47	102	25.20	0.35	0.37	100	9.09	0.67	0.16	123	1.87	0.91	0.04	146
8.	SSGS-MinD	53.63	0.17	0.56	90	38.42	0.23	0.48	104	25.10	0.37	0.37	113	10.00	0.64	0.18	130	1.72	0.92	0.04	155
9.	PSGS-LST	52.10	0.14	0.57	84	35.51	0.24	0.48	90	23.38	0.30	0.37	96	9.80	0.50	0.19	109	1.64	0.91	0.04	131
10.	PSG-LFT	50.62	0.15	0.57	84	36.45	0.22	0.40	90	23.78	0.33	0.37	97	9.56	0.65	0.18	113	1.52	0.94	0.03	132
11.	PSGS-MTS	50.37	0.14	0.57	86	38.16	0.18	0.51	90	24.18	0.28	0.38	97	9.30	0.62	0.18	113	1.61	0.92	0.04	132
A.	SDGS-MaxC	29.94	0.46	0.26	93	17.32	0.59	0.21	101	12.23	0.58	0.19	107	5.09	0.86	0.09	123	1.11	0.96	0.02	143
B.	SDGS-MinC	32.33	0.28	0.30	96	28.58	0.32	0.29	108	17.80	0.41	0.29	116	5.45	0.81	0.10	133	1.23	0.96	0.03	157
C.	SDGS-MaxD*C	27.68	0.44	0.27	90	22.37	0.45	0.22	103	12.52	0.64	0.21	111	4.93	0.81	0.10	125	1.10	0.96	0.02	148
D.	SDGS-MinD	26.30	0.35	0.30	100	24.25	0.27	0.23	105	12.87	0.52	0.20	112	5.31	0.82	0.10	128	1.00	0.96	0.02	150
	STC	31.33	0.6	0.45	96	21.45	0.74	0.4	104	17.38	0.81	0.35	111	12.57	0.9	0.25	127	10.02	0.94	0.2	149
	RFDFF	93.43	0.001	0.95	97	93.06	0.02	0.95	104	93.06	0.02	0.95	110	93.06	0.02	0.96	126	93.06	0.02	0.95	148

Table 4-1: Results on 30 nodes test problems with discrete duration variability

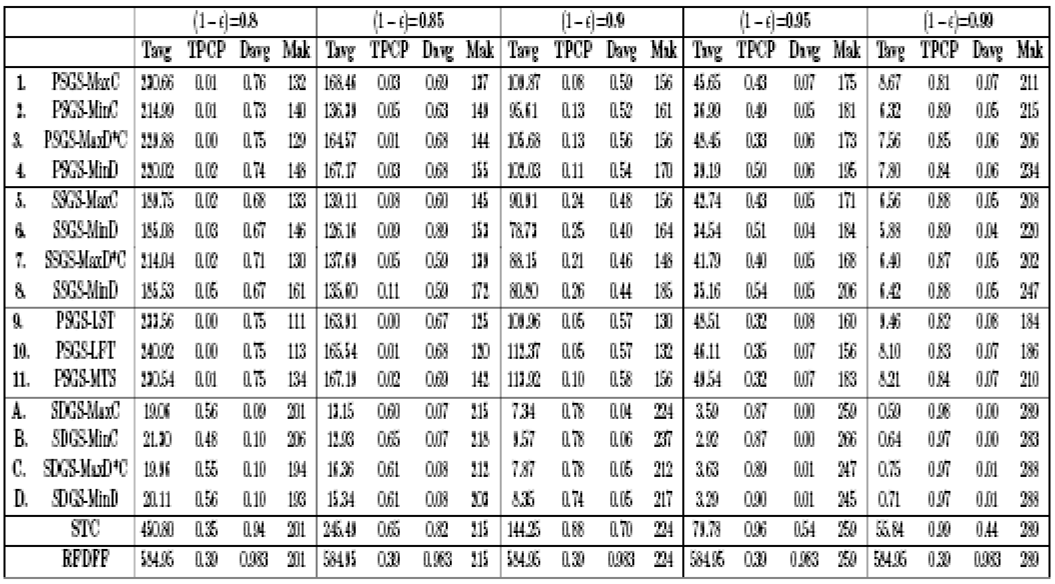
In the proposed algorithm, the temporal aspect of the problem is treated at a higher level, whereas the probabilistic aspect is tackled at decision points, when activities are supplied by available resources. The scheduling approach can be tailored to reflect the level of risk that an individual decision maker is willing to bear in uncertain environments.

We illustrated the favorable performance of the model and demonstrated that a rigorous treatment of uncertainty might lead to better uncertainty hedging.



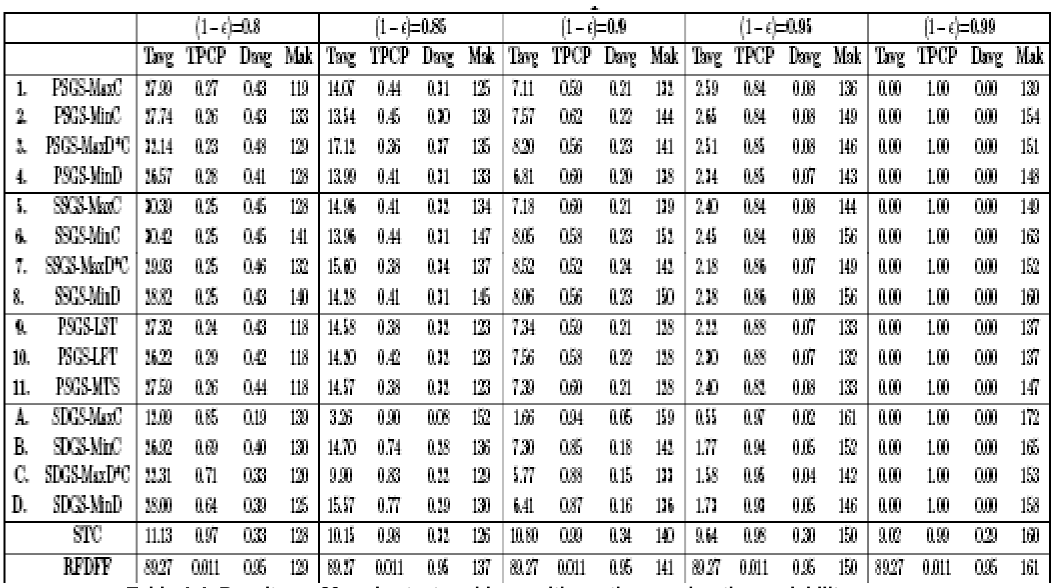
		$(1-\epsilon)$=0.8				$(1-\epsilon)$=0.85				$(1-\epsilon)$=0.9				$(1-\epsilon)$=0.95				$(1-\epsilon)$=0.99			
		Tavg	TPCP	Davg	Mak	Tavg	TPCP	Davg	Mak	Tavg	TPCP	Davg	Mak	Tavg	TPCP	Davg	Mak	Tavg	TPCP	Davg	Mak
1.	PSGS-MaxC	170.64	0.01	0.69	116	136.25	0.02	0.69	122	73.51	0.12	0.55	133	35.38	0.43	0.32	150	5.80	0.85	0.07	177
2.	PSGS-MinC	154.63	0.01	0.65	121	114.26	0.03	0.66	130	73.05	0.14	0.53	140	29.63	0.45	0.28	156	5.76	0.87	0.07	188
3.	PSGS-MaxD*C	165.40	0.01	0.67	119	137.90	0.03	0.69	124	79.27	0.12	0.56	135	35.35	0.38	0.33	152	5.82	0.84	0.07	183
4.	PSGS-MinD	156.20	0.01	0.66	121	130.15	0.04	0.68	129	71.62	0.15	0.53	138	31.30	0.43	0.29	155	5.35	0.87	0.06	186
5.	SSGS-MaxC	150.72	0.02	0.62	122	107.25	0.09	0.61	129	71.23	0.18	0.49	140	31.10	0.46	0.28	156	5.24	0.88	0.06	186
6.	PSGS-MinC	155.62	0.02	0.63	132	114.95	0.07	0.64	141	69.17	0.20	0.49	153	30.46	0.48	0.26	173	5.06	0.88	0.06	206
7.	PSGS-MaxD*C	148.88	0.03	0.63	125	111.81	0.08	0.62	132	68.06	0.15	0.49	142	29.77	0.49	0.27	160	5.50	0.87	0.06	190
8.	PSGS-MinD	141.47	0.04	0.62	132	107.49	0.08	0.61	140	67.81	0.20	0.48	151	28.03	0.51	0.25	170	5.63	0.87	0.06	204
9.	PSGS-LST	161.14	0.01	0.74	111	120.24	0.02	0.68	119	77.18	0.08	0.57	128	32.02	0.47	0.31	144	6.23	0.84	0.07	173
10.	PSGS-LFT	160.62	0.01	0.74	111	118.85	0.02	0.68	118	76.95	0.08	0.57	123	31.87	0.41	0.30	147	6.11	0.83	0.07	172
11.	PSGS-MST	160.01	0.00	0.74	110	119.13	0.03	0.68	118	72.05	0.09	0.55	126	30.96	0.35	0.30	142	5.46	0.86	0.07	170
A.	SDGS-MaxC	22.64	0.40	0.18	146	26.79	0.47	0.15	150	22.18	0.62	0.11	170	21.25	0.84	0.15	190	20.20	0.97	0.13	204
B.	SDGS-MinC	23.06	0.41	0.20	152	27.08	0.46	0.20	163	26.65	0.62	0.20	177	21.02	0.81	0.19	188	19.52	0.96	0.13	203
C.	SDGS-MaxD*C	27.02	0.40	0.17	151	22.45	0.57	0.18	157	21.80	0.63	0.12	172	18.82	0.80	0.14	188	16.53	0.97	0.12	203
D.	SDGS-MinD	22.86	0.38	0.15	149	21.97	0.56	0.16	155	21.56	0.68	0.12	160	21.50	0.82	0.11	191	16.72	0.97	0.12	205
	STC	76.24	0.72	0.6	148	51.51	0.83	0.48	157	20.17	0.91	0.43	175	25.55	0.96	0.35	191	21.11	0.98	0.31	203
	RFDFF	163.196	0.06	0.86	149	163.196	0.06	0.86	150	163.196	0.06	0.86	175	163.196	0.06	0.86	190	163.196	0.06	0.86	203

Table 4-2: Results on 60 nodes test problems with discrete duration variability



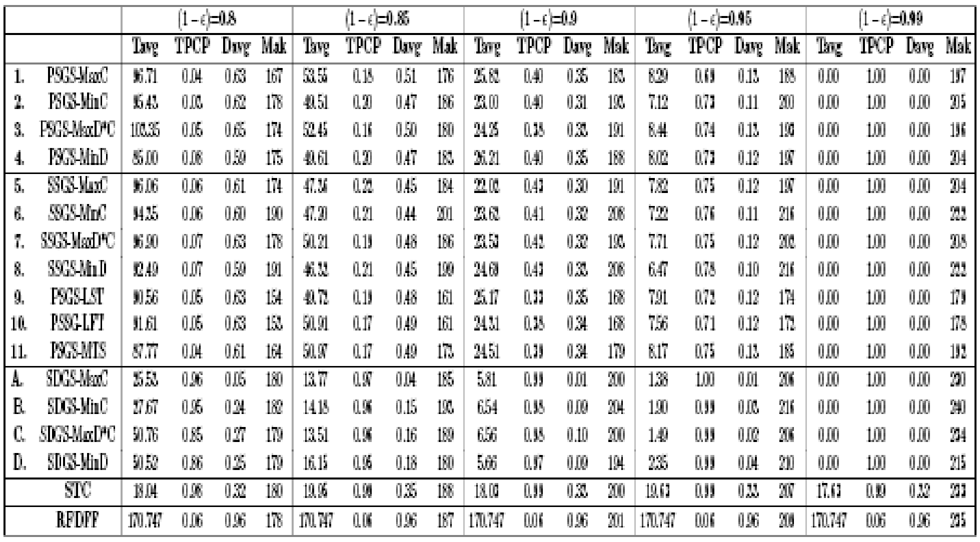
		$(1-\epsilon)=0.8$				$(1-\epsilon)=0.85$				$(1-\epsilon)=0.9$				$(1-\epsilon)=0.95$				$(1-\epsilon)=0.99$			
		Tavg	TPCP	Davg	Mak	Tavg	TPCP	Davg	Mak	Tavg	TPCP	Davg	Mak	Tavg	TPCP	Davg	Mak	Tavg	TPCP	Davg	Mak
1.	PSGS-MaxC	220.66	0.01	0.76	132	168.46	0.03	0.69	137	108.87	0.08	0.59	156	45.65	0.43	0.07	175	8.67	0.81	0.07	211
2.	PSGS-MinC	214.90	0.01	0.73	140	136.39	0.05	0.63	149	95.61	0.13	0.52	161	36.90	0.49	0.05	181	6.32	0.89	0.05	215
3.	PSGS-MaxD*C	229.88	0.00	0.75	129	164.57	0.01	0.68	144	106.68	0.13	0.56	156	48.45	0.33	0.06	173	7.56	0.85	0.06	206
4.	PSGS-MinD	220.02	0.02	0.74	148	167.17	0.03	0.68	155	102.03	0.11	0.54	170	39.19	0.50	0.06	195	7.80	0.84	0.06	234
5.	SSGS-MaxC	189.75	0.02	0.68	133	130.11	0.08	0.60	145	90.91	0.24	0.48	156	42.74	0.43	0.05	171	6.56	0.88	0.05	208
6.	SSGS-MinD	185.08	0.03	0.67	146	126.16	0.09	0.89	153	78.73	0.25	0.40	164	34.54	0.51	0.04	184	5.88	0.89	0.04	220
7.	SSGS-MaxD*C	214.04	0.02	0.71	130	137.69	0.05	0.59	139	88.15	0.21	0.46	148	41.79	0.40	0.05	168	6.40	0.87	0.05	202
8.	SSGS-MinD	185.53	0.05	0.67	161	135.60	0.11	0.59	172	80.80	0.26	0.44	185	35.16	0.54	0.05	206	6.42	0.88	0.05	247
9.	PSGS-LST	233.56	0.00	0.75	111	163.91	0.00	0.67	125	108.96	0.05	0.57	130	48.51	0.32	0.08	160	9.46	0.82	0.08	184
10.	PSGS-LFT	240.92	0.00	0.75	113	165.54	0.01	0.68	120	112.37	0.05	0.57	132	46.11	0.35	0.07	156	8.10	0.83	0.07	186
11.	PSGS-MTS	220.54	0.01	0.75	134	167.19	0.02	0.69	142	113.92	0.10	0.58	156	49.54	0.32	0.07	183	8.21	0.84	0.07	210
A.	SDGS-MaxC	19.06	0.56	0.09	201	13.15	0.60	0.07	215	7.34	0.78	0.04	224	3.50	0.87	0.00	250	0.50	0.98	0.00	280
B.	SDGS-MinC	21.30	0.48	0.10	206	12.93	0.65	0.07	218	9.57	0.78	0.06	237	2.92	0.87	0.00	266	0.64	0.97	0.00	283
C.	SDGS-MaxD*C	19.96	0.55	0.10	194	16.36	0.61	0.08	212	7.87	0.78	0.05	212	3.63	0.89	0.01	247	0.75	0.97	0.01	288
D.	SDGS-MinD	20.11	0.56	0.10	193	15.34	0.61	0.08	203	8.35	0.74	0.05	217	3.29	0.90	0.01	245	0.71	0.97	0.01	288
	STC	450.80	0.35	0.94	201	245.49	0.65	0.82	215	144.25	0.88	0.70	224	79.78	0.96	0.54	250	55.84	0.99	0.44	280
	RFDFF	584.96	0.39	0.983	201	584.95	0.39	0.963	215	584.95	0.39	0.983	224	584.95	0.39	0.963	250	584.95	0.39	0.983	280

Table 4-3: Results on 90 nodes test problems with discrete duration variability



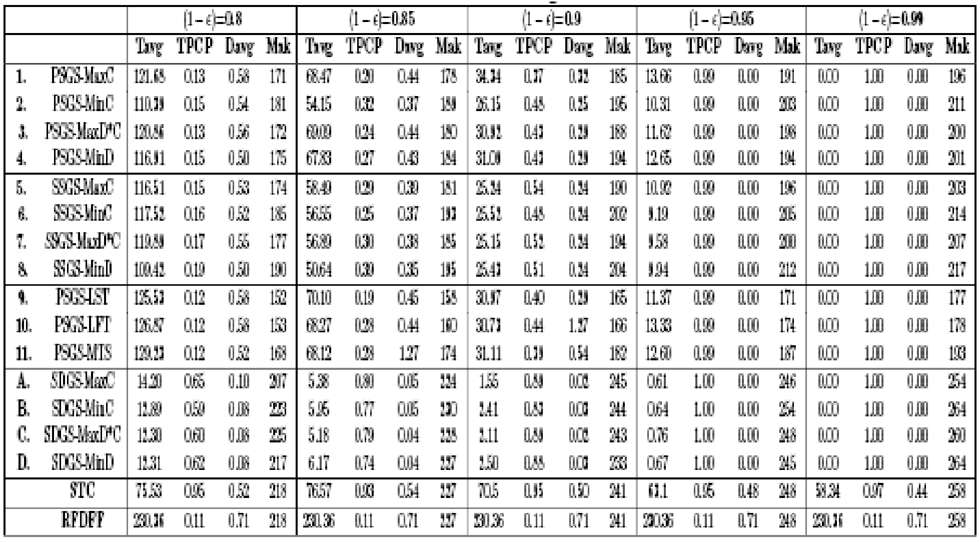
		$(1-\epsilon)=0.8$				$(1-\epsilon)=0.85$				$(1-\epsilon)=0.9$				$(1-\epsilon)=0.95$				$(1-\epsilon)=0.99$			
		Tavg	TPCP	Davg	Mak	Tavg	TPCP	Davg	Mak	Tavg	TPCP	Davg	Mak	Tavg	TPCP	Davg	Mak	Tavg	TPCP	Davg	Mak
1.	PSGS-MaxC	27.90	0.27	0.43	119	14.07	0.44	0.31	125	7.11	0.50	0.21	132	2.59	0.84	0.08	136	0.00	1.00	0.00	139
2.	PSGS-MinC	27.74	0.26	0.43	133	13.54	0.45	0.30	139	7.57	0.62	0.22	144	2.65	0.84	0.08	149	0.00	1.00	0.00	154
3.	PSGS-MaxD*C	32.14	0.23	0.48	129	17.12	0.36	0.37	135	8.20	0.56	0.23	141	2.51	0.85	0.08	146	0.00	1.00	0.00	151
4.	PSGS-MinD	26.57	0.28	0.41	128	13.90	0.41	0.31	133	6.81	0.60	0.20	138	2.34	0.85	0.07	143	0.00	1.00	0.00	148
5.	SSGS-MaxC	30.30	0.25	0.45	128	14.96	0.41	0.32	134	7.18	0.60	0.21	139	2.40	0.84	0.08	144	0.00	1.00	0.00	140
6.	SSGS-MinC	30.42	0.25	0.45	141	13.96	0.44	0.31	147	8.05	0.58	0.23	152	2.45	0.84	0.08	156	0.00	1.00	0.00	163
7.	SSGS-MaxD*C	29.03	0.25	0.46	132	15.60	0.38	0.34	137	8.52	0.52	0.24	142	2.18	0.86	0.07	149	0.00	1.00	0.00	152
8.	SSGS-MinD	28.82	0.25	0.43	140	14.28	0.41	0.31	145	8.06	0.56	0.23	150	2.38	0.86	0.08	156	0.00	1.00	0.00	160
9.	PSGS-LST	27.32	0.24	0.43	118	14.58	0.38	0.32	123	7.34	0.59	0.21	128	2.22	0.88	0.07	133	0.00	1.00	0.00	137
10.	PSGS-LFT	26.22	0.29	0.42	118	14.30	0.42	0.32	123	7.56	0.58	0.22	128	2.30	0.88	0.07	132	0.00	1.00	0.00	137
11.	PSGS-MTS	27.50	0.26	0.44	118	14.57	0.38	0.32	123	7.30	0.60	0.21	128	2.40	0.82	0.08	133	0.00	1.00	0.00	147
A.	SDGS-MaxC	12.09	0.85	0.19	130	3.26	0.90	0.08	152	1.66	0.94	0.05	159	0.55	0.97	0.02	161	0.00	1.00	0.00	172
B.	SDGS-MinC	26.02	0.69	0.40	130	14.70	0.74	0.28	136	7.30	0.85	0.18	142	1.77	0.94	0.05	152	0.00	1.00	0.00	165
C.	SDGS-MaxD*C	22.31	0.71	0.33	120	9.90	0.83	0.22	129	5.77	0.88	0.15	133	1.58	0.96	0.04	142	0.00	1.00	0.00	153
D.	SDGS-MinD	28.00	0.64	0.39	125	15.57	0.77	0.29	130	6.41	0.87	0.16	136	1.73	0.93	0.05	146	0.00	1.00	0.00	158
	STC	11.13	0.97	0.33	128	10.15	0.98	0.32	126	10.80	0.99	0.34	140	9.64	0.98	0.30	150	9.02	0.99	0.29	160
	RFDFF	89.27	0.011	0.95	129	89.27	0.011	0.96	137	80.27	0.011	0.95	141	80.27	0.011	0.95	150	89.27	0.011	0.95	161

Table 4-4: Results on 30 nodes test problems with continuous duration variability



		$(1-\epsilon)$=0.8				$(1-\epsilon)$=0.85				$(1-\epsilon)$=0.9				$(1-\epsilon)$=0.95				$(1-\epsilon)$=0.99			
		Tavg	TPCP	Davg	Mak	Tavg	TPCP	Davg	Mak	Tavg	TPCP	Davg	Mak	Tavg	TPCP	Davg	Mak	Tavg	TPCP	Davg	Mak
1.	PSGS-MaxC	96.71	0.04	0.63	167	53.55	0.18	0.51	176	25.82	0.40	0.35	183	8.29	0.69	0.13	188	0.00	1.00	0.00	197
2.	PSGS-MinC	95.43	0.03	0.62	178	40.51	0.20	0.47	186	23.00	0.40	0.31	193	7.12	0.73	0.11	200	0.00	1.00	0.00	205
3.	PSGS-MaxD*C	103.35	0.05	0.65	174	52.45	0.16	0.50	180	24.25	0.38	0.33	191	8.44	0.74	0.13	193	0.00	1.00	0.00	196
4.	PSGS-MinD	85.00	0.08	0.59	175	40.61	0.20	0.47	183	26.21	0.40	0.35	188	8.02	0.73	0.12	197	0.00	1.00	0.00	204
5.	SSGS-MaxC	96.06	0.06	0.61	174	47.34	0.22	0.45	184	22.02	0.43	0.30	191	7.82	0.75	0.12	197	0.00	1.00	0.00	204
6.	SSGS-MinC	94.25	0.06	0.60	190	47.20	0.21	0.44	201	23.62	0.41	0.32	208	7.22	0.76	0.11	216	0.00	1.00	0.00	222
7.	SSGS-MaxD*C	96.90	0.07	0.63	178	50.21	0.19	0.48	186	23.53	0.42	0.32	193	7.71	0.75	0.12	202	0.00	1.00	0.00	208
8.	SSGS-MinD	92.49	0.07	0.59	191	46.32	0.21	0.45	199	24.69	0.43	0.33	208	6.47	0.78	0.10	216	0.00	1.00	0.00	222
9.	PSGS-LST	90.56	0.05	0.63	154	40.72	0.19	0.48	161	25.17	0.33	0.35	168	7.91	0.72	0.12	174	0.00	1.00	0.00	179
10.	PSSG-LFT	91.61	0.05	0.63	153	50.91	0.17	0.49	161	24.31	0.38	0.34	168	7.56	0.71	0.12	172	0.00	1.00	0.00	178
11.	PSGS-MTS	87.77	0.04	0.61	164	50.97	0.17	0.49	173	24.51	0.39	0.34	179	8.17	0.75	0.13	185	0.00	1.00	0.00	192
A.	SDGS-MaxC	25.53	0.96	0.05	180	13.77	0.97	0.04	185	5.81	0.99	0.01	200	1.38	1.00	0.01	206	0.00	1.00	0.00	230
B.	SDGS-MinC	27.67	0.95	0.24	182	14.18	0.96	0.15	193	6.54	0.98	0.09	204	1.90	0.99	0.03	216	0.00	1.00	0.00	240
C.	SDGS-MaxD*C	50.76	0.85	0.27	179	13.51	0.96	0.16	189	6.56	0.98	0.10	200	1.49	0.99	0.02	206	0.00	1.00	0.00	234
D.	SDGS-MinD	50.52	0.86	0.25	179	16.15	0.95	0.18	180	5.66	0.97	0.09	194	2.35	0.99	0.04	210	0.00	1.00	0.00	215
	STC	18.04	0.98	0.32	180	19.96	0.98	0.35	188	18.03	0.99	0.33	200	19.63	0.99	0.23	207	17.63	0.89	0.32	233
	RFDFF	170.747	0.06	0.96	178	170.747	0.06	0.96	187	170.747	0.06	0.96	201	170.747	0.06	0.96	208	170.747	0.06	0.96	225

Table 4-5: Results on 60 nodes test problems with continuous duration variability



		$(1-\epsilon)$=0.8				$(1-\epsilon)$=0.85				$(1-\epsilon)$=0.9				$(1-\epsilon)$=0.95				$(1-\epsilon)$=0.99			
		Tavg	TPCP	Davg	Mak	Tavg	TPCP	Davg	Mak	Tavg	TPCP	Davg	Mak	Tavg	TPCP	Davg	Mak	Tavg	TPCP	Davg	Mak
1.	PSGS-MaxC	121.68	0.13	0.58	171	68.47	0.20	0.44	178	34.34	0.37	0.32	185	13.66	0.99	0.00	191	0.00	1.00	0.00	196
2.	PSGS-MinC	110.39	0.15	0.54	181	54.15	0.32	0.37	189	26.15	0.48	0.25	195	10.31	0.99	0.00	203	0.00	1.00	0.00	211
3.	PSGS-MaxD*C	120.86	0.13	0.56	172	69.09	0.24	0.44	180	30.92	0.43	0.29	188	11.62	0.99	0.00	198	0.00	1.00	0.00	200
4.	PSGS-MinD	116.91	0.15	0.50	175	67.83	0.27	0.43	184	31.09	0.43	0.29	194	12.65	0.99	0.00	194	0.00	1.00	0.00	201
5.	SSGS-MaxC	116.51	0.15	0.53	174	58.40	0.29	0.39	181	25.34	0.54	0.24	190	10.92	0.99	0.00	196	0.00	1.00	0.00	203
6.	SSGS-MinC	117.52	0.16	0.52	185	56.55	0.25	0.37	193	25.52	0.48	0.24	202	9.19	0.99	0.00	205	0.00	1.00	0.00	214
7.	SSGS-MaxD*C	119.89	0.17	0.55	177	56.89	0.30	0.38	185	25.15	0.52	0.24	194	9.58	0.99	0.00	200	0.00	1.00	0.00	207
8.	SSGS-MinD	109.42	0.19	0.50	190	50.64	0.39	0.35	195	25.43	0.51	0.24	204	9.94	0.99	0.00	212	0.00	1.00	0.00	217
9.	PSGS-LST	125.53	0.12	0.58	152	70.10	0.19	0.45	158	30.97	0.40	0.29	165	11.37	0.99	0.00	171	0.00	1.00	0.00	177
10.	PSGS-LFT	126.87	0.12	0.58	153	68.27	0.28	0.44	160	30.73	0.44	1.27	166	13.33	0.99	0.00	174	0.00	1.00	0.00	178
11.	PSGS-MTS	129.33	0.12	0.52	168	68.12	0.28	1.27	174	31.11	0.39	0.54	182	12.60	0.99	0.00	187	0.00	1.00	0.00	193
A.	SDGS-MaxC	14.20	0.65	0.10	207	5.38	0.80	0.05	224	1.55	0.89	0.02	245	0.61	1.00	0.00	246	0.00	1.00	0.00	254
B.	SDGS-MinC	12.89	0.59	0.08	223	5.95	0.77	0.05	230	2.41	0.83	0.03	244	0.64	1.00	0.00	254	0.00	1.00	0.00	264
C.	SDGS-MaxD*C	12.30	0.60	0.08	225	5.18	0.79	0.04	228	2.11	0.89	0.02	243	0.76	1.00	0.00	248	0.00	1.00	0.00	260
D.	SDGS-MinD	12.31	0.62	0.08	217	6.17	0.74	0.04	227	2.50	0.88	0.03	233	0.67	1.00	0.00	245	0.00	1.00	0.00	264
	STC	75.53	0.96	0.52	218	76.57	0.93	0.54	227	70.5	0.95	0.50	241	63.1	0.95	0.48	248	58.34	0.97	0.44	258
	RFDFF	230.36	0.11	0.71	218	230.36	0.11	0.71	227	230.36	0.11	0.71	241	230.36	0.11	0.71	248	230.36	0.11	0.71	258

Table 4-6: Results on 90 nodes test problems with continuous duration variability

			0.8							0.85							0.9							0.95							0.99						
			Number of variables			Number of constraints			It	Number of variables			Number of constraints			It	Number of variables			Number of constraints			It	Number of variables			Number of constraints			It	Number of variables			Number of constraints			It
			max	min	avg	max	min	avg	avg	max	min	avg	max	min	avg	avg	max	min	avg	max	min	avg	avg	max	min	avg	max	min	avg	avg	max	min	avg	max	min	avg	avg
30	A	Discrete	16.71	9.38	12.41	8.25	5.00	5.65	7.63	16.29	9.00	11.49	8.50	5.00	5.75	7.88	15.57	8.13	10.63	8.50	5.00	5.90	7.75	10.86	5.75	7.69	8.25	5.00	5.73	7.38	4.63	3.00	3.34	7.25	4.00	4.68	7.13
		Continuous	3.78	2.00	2.45	4.78	3.00	3.45	11.22	4.00	2.00	2.60	5.00	3.00	3.60	9.90	3.90	2.00	2.50	4.90	3.00	3.50	9.70	3.90	2.00	2.52	4.90	3.00	3.52	9.90	4.00	2.10	2.57	5.00	3.10	3.57	8.60
	B	Discrete	16.14	9.25	12.79	8.50	5.00	6.04	4.63	15.50	9.00	11.34	8.50	5.00	5.87	5.38	15.00	7.63	10.42	8.75	5.00	5.99	5.88	10.71	5.75	7.93	8.75	5.00	6.03	5.13	4.63	3.00	3.51	7.25	4.00	5.03	4.63
		Continuous	4.00	2.00	2.39	5.00	3.00	3.39	8.22	4.00	2.00	2.45	5.00	3.00	3.45	8.49	3.99	2.00	2.49	4.99	3.00	3.49	7.77	3.99	2.00	2.47	4.99	3.00	3.47	8.49	3.80	2.01	2.44	4.80	3.01	3.44	7.86
	C	Discrete	16.50	9.00	12.43	8.25	5.00	5.66	8.00	16.00	8.50	11.98	8.50	5.00	6.04	7.25	14.57	7.75	10.63	8.50	5.00	5.96	7.63	8.49	3.88	5.15	6.72	4.00	4.09	7.72	4.63	3.00	3.49	7.25	4.00	4.98	5.50
		Continuous	3.89	2.00	2.45	4.89	3.00	3.45	9.33	3.90	2.00	2.43	4.90	3.00	3.43	9.85	3.90	2.00	2.48	4.90	3.00	3.48	8.98	3.90	2.00	2.47	4.90	3.00	3.47	8.55	3.78	2.00	2.46	4.78	3.00	3.46	8.09
	D	Discrete	17.14	8.75	12.84	8.25	5.00	5.90	8.00	16.38	8.63	11.36	8.50	5.00	5.76	7.50	14.43	7.38	9.97	8.50	5.00	5.62	8.63	11.00	5.75	7.67	8.25	5.00	5.73	6.88	4.63	3.00	3.46	7.25	4.00	4.91	5.75
		Continuous	4.22	2.00	2.47	5.22	3.00	3.47	9.22	4.19	2.00	2.50	5.19	3.00	3.50	8.48	4.09	2.00	2.57	5.09	3.00	3.57	8.40	3.99	2.00	2.54	4.99	3.00	3.54	8.35	3.98	2.00	2.50	4.98	3.00	3.50	8.61
60	A	Discrete	23.70	8.90	15.30	9.80	5.00	6.73	18.70	23.60	8.50	13.99	10.60	5.00	6.73	18.40	19.89	7.11	11.77	10.78	5.00	6.54	17.22	14.78	5.11	8.77	10.56	5.00	6.47	16.33	5.22	3.00	3.62	8.44	4.00	5.25	17.89
		Continuous	5.40	2.00	3.34	6.40	3.00	4.34	16.10	5.20	2.10	3.31	6.20	3.10	4.31	16.30	5.20	2.00	3.13	6.20	3.00	4.13	16.80	5.00	2.00	3.16	6.00	3.00	4.16	16.70	5.30	2.10	3.18	6.30	3.10	4.18	17.40
	B	Discrete	24.70	9.10	15.17	10.00	5.00	6.68	17.80	23.00	8.40	13.41	10.20	5.00	6.49	17.70	19.11	7.44	11.55	10.11	5.00	6.45	17.22	13.78	5.33	8.45	9.44	5.00	6.24	16.78	5.22	3.00	3.53	8.44	4.00	5.07	16.89
		Continuous	4.90	2.00	3.19	5.90	3.00	4.19	16.00	4.80	2.00	3.14	5.80	3.00	4.14	16.20	5.00	2.00	3.08	6.00	3.00	4.08	16.20	5.00	2.00	3.08	6.00	3.00	4.08	15.40	5.00	2.00	3.13	6.00	3.00	4.13	15.90
	C	Discrete	25.10	8.90	15.67	10.40	5.00	6.92	18.30	22.90	8.30	13.97	10.00	5.00	6.78	18.50	19.56	7.44	12.34	10.11	5.00	6.80	18.44	13.67	5.33	8.62	9.67	5.00	6.36	17.67	5.33	3.00	3.60	8.67	4.00	5.20	18.44
		Continuous	5.40	2.00	3.24	6.40	3.00	4.24	16.50	5.20	2.00	3.32	6.20	3.00	4.32	16.50	4.90	2.00	3.17	5.90	3.00	4.17	16.50	5.30	2.00	3.16	6.30	3.00	4.16	16.40	5.20	2.20	3.28	6.20	3.20	4.28	17.00
	D	Discrete	25.40	8.70	15.73	10.40	5.00	6.91	17.70	23.90	8.40	14.06	10.20	5.00	6.76	19.20	19.00	7.11	12.13	9.89	5.00	6.70	18.56	14.00	5.22	8.70	9.44	5.00	6.39	17.67	5.44	3.00	3.67	8.89	4.00	5.34	16.56
		Continuous	5.10	2.00	3.43	6.10	3.00	4.43	15.10	5.12	2.00	3.26	6.12	3.00	4.25	17.15	4.99	2.10	3.26	5.99	3.10	4.26	16.25	4.93	2.10	3.24	5.93	3.10	4.24	16.74	4.72	2.02	3.22	5.72	3.02	4.22	17.10
90	A	Discrete	40.67	8.67	20.66	16.33	5.00	8.83	22.00	36.11	9.78	18.74	15.89	5.22	8.82	21.33	32.67	7.89	15.81	16.11	5.00	8.51	22.56	19.89	5.78	10.82	13.44	5.00	7.91	22.89	7.11	3.00	4.32	12.22	4.00	6.65	23.89
		Continuous	7.60	2.00	3.80	8.60	3.00	4.80	21.00	8.60	2.00	4.23	9.60	3.00	5.23	20.50	8.80	2.20	4.52	9.80	3.20	5.52	18.90	8.60	2.00	4.58	9.60	3.00	5.58	19.10	8.50	2.00	4.50	9.50	3.00	5.50	19.30
	B	Discrete	38.60	10.70	20.99	14.80	5.00	8.92	21.30	31.56	8.67	17.75	13.67	5.00	8.34	23.44	27.56	7.78	15.13	14.11	5.00	8.18	23.00	21.89	5.00	10.55	15.67	5.00	7.72	24.11	7.44	3.00	4.36	12.89	4.00	6.72	22.89
		Continuous	8.00	2.00	3.84	9.00	3.00	4.84	20.70	8.00	2.00	4.23	9.00	3.00	5.23	19.90	7.90	2.10	4.54	8.90	3.10	5.54	18.50	8.10	2.00	4.56	9.10	3.00	5.56	18.70	8.10	2.20	4.63	9.10	3.20	5.63	18.40
	C	Discrete	38.33	9.22	21.09	14.78	5.00	9.23	20.44	33.33	9.44	18.67	14.78	5.00	8.74	21.89	30.11	8.22	16.25	15.22	5.00	8.75	21.67	19.67	5.44	10.83	13.67	5.00	7.90	23.33	7.11	3.00	4.27	12.22	4.00	6.55	24.44
		Continuous	8.00	2.00	4.04	9.00	3.00	5.04	21.00	7.80	2.00	4.53	8.80	3.00	5.53	19.20	7.90	2.00	4.75	8.90	3.00	5.75	18.20	7.80	2.00	4.58	8.80	3.00	5.58	19.20	7.80	2.00	4.69	8.80	3.00	5.69	18.90
	D	Discrete	37.44	9.67	21.01	15.22	5.00	8.97	21.56	33.22	9.00	18.91	15.00	5.00	8.84	21.89	29.56	8.11	16.38	15.00	5.00	8.78	21.78	19.78	5.56	11.14	14.11	5.00	8.09	22.67	7.11	3.00	4.42	12.22	4.00	6.83	23.33
		Continuous	7.60	2.00	4.19	8.60	3.00	5.19	19.70	8.30	2.20	4.66	9.30	3.20	5.66	18.60	8.00	2.00	4.58	9.00	3.00	5.58	19.30	7.80	2.20	4.69	8.80	3.20	5.69	18.20	7.80	2.10	4.58	8.80	3.10	5.58	19.10

Table 4-7: Average, minimum and maximum number of variables and constraints per iteration

Chapter 5

A real application: Robust Project Scheduling in Construction Industry

This chapter addresses the problem of scheduling under uncertainty in construction projects. The existing methods for determining a project schedule are based on assumption of complete knowledge of project parameters. But in reality there is uncertainty in construction projects, deriving from a multitude of context-dependent sources and often provided as outcome of a risk analysis process. Thus classical deterministic analysis might provide a schedule which is not sufficiently protected against possible disruptions. A quantitative methodology is developed for planning construction projects under uncertainty aimed at determining a reliable resource feasible project schedule by taking into account the available probabilistic information to produce solutions that are less sensitive to perturbations that occur on line. To provide evidence on the potential of the developed approach, a validation phase on a real construction project is carried out. The project behavior under several scenarios is evaluated by using a simulation approach.

5.1. Introduction

Construction projects are usually characterized by high complexity. Several factors determine this feature: a great number of activities has to be performed in order to achieve project completion, a variety of resources, both material and human, are necessary to perform activities, and therefore great capital investments have to be managed. An efficient scheduling phase is crucial in order to ensure that the project is completed on time and within budget. In this respect, a detailed baseline project schedule plays a crucial role: as widely recognized in (Metha & Uzsoy 1998) and in (Möhring & Stork 2000) it supports project managers in monitoring the work progress, facilitating resource allocation and providing a basis for managing external activities, such as relations with contractors.

In construction industry, baseline schedule generation is usually performed by using different scheduling techniques, like, for instance, PERT (Malcolm et al. 1959) and CPM (Wiest & Levy 1977), embedded in most computer software packages developed for construction project management. The main drawback of these time-oriented scheduling techniques is the assumption of unlimited availability of resources for each project activity (Nkasu 1994).

In real construction projects, many problems arise when activities require resources that are available only in limited quantities making resource allocation indispensable in the generation of realistic baseline project schedules (Kim & Garza 2005). As a matter of fact, ignoring resource considerations in the scheduling phase of the project will lead to extremely poor schedule performance (Just & Murphy 1994). Woodworth and Shanahan in (Woodworth & Shanahan 1998) have shown that schedules based on time-oriented networks are exceeded by an average of around 38%.

Moreover, the complex dynamic and uncertain environment in which construction projects have to be performed highlights the need for effective planning and scheduling tools. Since the early sixties better tools and techniques to asses project risks were developed to assist project managers. ((Camps 1996), (Chapman & Ward 1997), (Guildford 1998), (Simister 1994)).

After a proper risk assessment program has quantified the impact of potential risks involved in the project at hand on individual activities duration a risk response method must be set (Zhu et al. 2005). We can distinguish among different approaches to deal with uncertainty in a scheduling environment. Hayes et al. (1986) and Marshall (1988) provide good introductions to the subject. Here we mention the work of Vaziri et al. (2007) that propose a dynamic control policies in the form of planned resource allocation to project activities exploiting also the impact of resource allocation on uncertain durations. Park & Mora (2004), use a simulation-based buffering strategy to generate a robust construction plan that protects against uncertainties. Schatteman et al. (2008) develop an integrated methodology that consists into two phases. First, individual project activities uncertainty is estimated identifying and quantifying the risk by grouping activities with similar risk profiles. Then, in the second phase, this input is used for generating a robust baseline schedule by introducing time buffers in a precedence and resource feasible project schedule.

The underlying assumption in most of these risk management techniques is that the uncertainty can be quantified using statistical analysis, given that past information is available regarding both the probability of undesired events and the effect of such events on the project.

Regrettably, this assumption may also be violated and therefore in that case we cannot resume to probability theory.

When the type of uncertainty encountered in construction projects does not fit the axiomatic basis of probability theory, fuzzy set-based methods (Zeng et al. 2007) or possibility theory (Mohamed & McCowan, 2001) may be used.

Another non-statistical approach for analyzing the risk associated with highly uncertain project scheduling based on the info-gap theory is reported in Ben-Haim (2006) and Regev et al. (2006).

Our work takes place in the former group of models, where probabilistic uncertainty is assumed.

We describe a methodology for planning resource constrained construction projects in real contexts using uncertainty estimation of project activities to generate a baseline schedule which is protected against disruptions .

In particular, under the assumption that activities durations may be represented by random variables, we present an iterative algorithm based on the theory of joint chance constraints embedded within a friendly user interface to generate a reliable schedule that is protected against uncertain events with a certain probability.

The remainder of the chapter is organized as follows. The following paragraph introduces problem assumptions and notations and presents the scheduling methodology for generating baseline schedules in uncertain environments. Paragraph 5.3 is devoted to the presentation of the application of the methodology to a real case study. Results are analyzed in paragraph 5.4 and conclusions are given in paragraph 5.5.

5.2. Dealing with uncertainty in construction projects

After having identified the risks and their potential impact on activities duration, it is possible to define a suitable strategy to face the risk. The project scheduling system we propose try to cope with uncertainty by taking into account the available probabilistic information to produce solutions that are less sensitive to perturbations that occur on line and relies on the generation of a robust project baseline schedule that is sufficiently protected against distortions that may occur during actual project execution. Hereafter, robustness will be referred as the capability to hedge against uncertain events, by starting activities as originally planned, avoiding extra cost and time overrun.

In order to be self-contained and for the sake of clarity, we briefly introduce some notation for the resource-constrained project scheduling problem (RCPSp), consisting in minimizing the duration of a project, subject to the finish-start, zero-lag precedence constraints and the resource constraints. A detailed notation list is reported in the appendix C.

Let us consider a project represented by a directed acyclic graph $G = (N,A)$ characterized by an activities set $N = (1..n)$ and a precedence relations set A= $i,j \;\; i,j \in N\}$ where $\mathrm{i,j} \in A$ if and only if activity j can start only after activity i is finished. Let 0 be the dummy activity representing the project start, n be the dummy activity corresponding to the project conclusion and d_i be the deterministic duration associated with each activity i. Uncertain activities duration are represented by random variables $d_i \; \forall \, i \in N$, with known cumulative probability distribution functions F_{d_i}.

Let us denote by K the set of renewable resources and let a_k be the constant per period availability of resource type k, $k = 1..., K$. We assume that each activity $j \in N$ has to be processed without interruptions requiring a constant amount of resource r_{ik}, for each renewable resource.

The classic optimization problem in project networks is finding a feasible schedule such that the project completion time is minimized.

The resource constraints make the problem substantially more difficult and usually require, to be expressed, the use of binary variables (Pritsker, 1969). A conceptual decision model (Christofides et al., 1987) may be used instead.

The scheduling procedure we propose (in the sequel referred as Resource Allocation Heuristic-RAH for short-) constructs a robust baseline schedule through a stepwise increase of a partial schedule in which a proper resource allocation is performed by solving a joint chance constrained problem. More specifically, at each iteration g of the algorithm is associated a decision point t_g in which scheduling decisions are made by means of an anticipative stochastic model setting proactive starting and completion times. In this way a resource feasible schedule is built incrementally by successive decision points until all the activities have been scheduled.

Let us define for each time point t_g S_g as the set of activities that can start at time point t_g and E_g as the set containing all activities which can be precedence feasibly started at t_g (we call these activities eligible). If E_g contains more than one element a competition has to be arranged to choose the optimal subset of activities that can be supplied by the residual resource availability at time t_g namely $r_k(t_g)$. The most competitive activities are then chosen to start at t_g and therefore included in the set S_g. Since none of the eligible activities can still be started without violating the resource constraints, a new decision point is set at the earliest completion time of the activities that are in progress.

An algorithmic description of RAH is given in Figure 5.1.

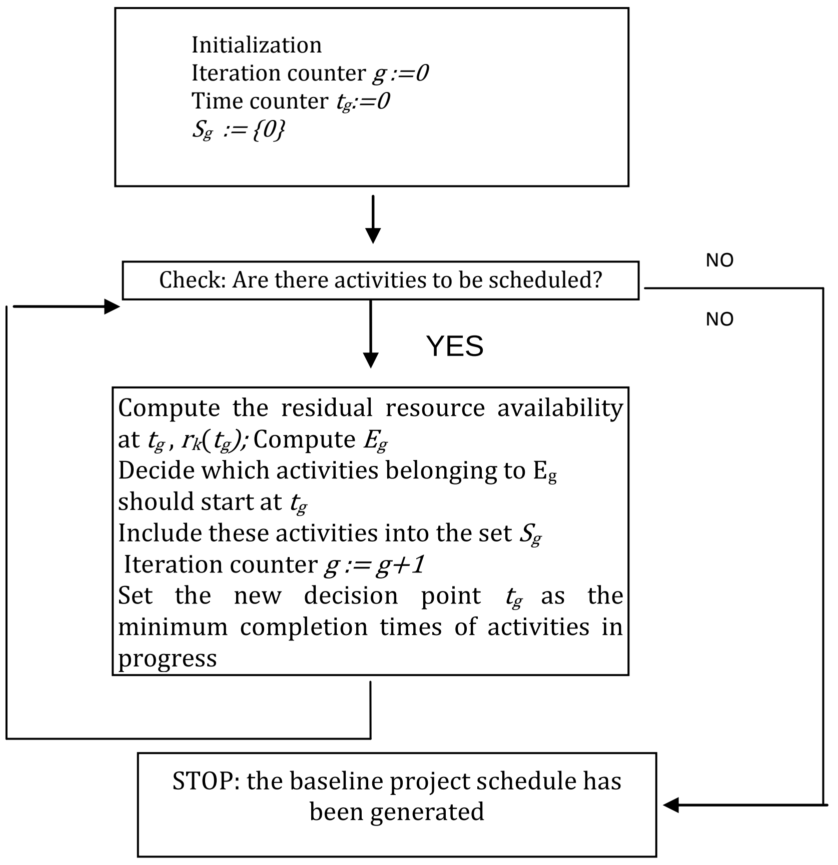

Initialization
Iteration counter $g:=0$
Time counter $t_g:=0$
$S_g := \{0\}$
Check: Are there activities to be scheduled?
NO
NO
YES
Compute the residual resource availability at t_g, $r_k(t_g)$; Compute E_g
Decide which activities belonging to E_g should start at t_g
Include these activities into the set S_g
Iteration counter $g := g+1$
Set the new decision point t_g as the minimum completion times of activities in progress
STOP: the baseline project schedule has been generated

Figure 5.1. Typical RAH iteration

At each decision point $t_g$decisions concerning which activities, belonging to E_g , can start are made on the basis of the solution of the following stochastic problem.

$$\max \quad _{i\in E_g}\beta_i - \gamma C \tag{1}$$

$$C \geq c_i - M \;\; 1-\beta_i \;\; \forall\, i \in E_g \tag{2}$$

$$P(\, c_i + M \;\; 1-\beta_i \;\; \geq t_g + d_i \,\forall\, i \in E_g) \geq \alpha \tag{3}$$

$$_{i\in E_g} r_{ik}\beta_i \geq r_k \;\; t_g \qquad \forall\, k = 1..K \tag{4}$$

$$\beta_i = \;\; 0{,}1 \;\forall\, i \in E_g \tag{5}$$

where the decision variables are completion times $c_i \, \forall \, i \in E_g$ and binary variables $\beta_i \, \forall \, i \in E_g$ which take value one if activity i can be feasibly scheduled and zero otherwise. Only the most competitive activities in term of the objective function are actually scheduled and their associated β variables take value one. We point out that the formulation set the completion time of an activity i only if the activity is included in the set S_g $(i.e. \beta_i = 1)$ through a Big-M formulation with parameter M.

The objective function tries to balance two conflicting objectives weighted through the parameter γ, the resource and the time allocation decisions. Therefore, whilst the first term of the objective function tries to maximize the resource consumption at each decision point, the second term aims at reducing the makespan of the partial schedule built up to moment t_g. The parameter γ, is adjusted dynamically in order to find a good balance between the two conflicting objectives. We should notice that our model assumes, as common in the scheduling literature, that resources will be wasted if not used. We should point out that the subset of activities included in S_g minimizes the waste of resources tanks to the first term of the objective function.

Constraints (2) set the makespan of the sub-schedule built at t_g as the maximum of the completion times of actually scheduled activities. The completion times of the scheduled activities is set by means of constraints (3) in such a way that the joint probability of not delaying subsequent activities is at least α. The use of joint probabilistic constraints (Miller and Wagner, 1965) is rather original in the literature.

Constraints (4) ensure that the subset of activities to start concurrently at time t_g respects the residual resources availability at time t_g.

Problem (1)–(5) has to be solved at each decision point, when at least more than one activity is ready to be operated and the residual available amount of resources is not zero.

The proposed RAH can be viewed as a particular parallel schedule generation scheme (SGS) (Kolisch and Hartmann, 2006) in which, rather than using a priority rule for deciding the set of activities to be included in the partial schedule, the solution of a stochastic problem is used.

The RAH has been then embedded in a user friendly tool for project scheduling under uncertainty. In practical terms, the only action required to managers is to define the project breakdown structure and store project data such as activity number, ID and resource requirement, precedence relations among activities, resource availability. Once the data are uploaded on the system, the baseline schedule is automatically generated, and if desired, the simulation phase performed. A screenshot of the graphical interface developed for our tool is reported in Figure 4.3.

5.3. Empirical illustration of RAH: a real case study

In this section, we document the application of our approach on a real project for construction of students' apartments at the University of Calabria, Italy. Such project consists of 43 activities; the first and the last one are dummy activities representing the starting and the ending time of the project respectively. The project network is reported in Figure 5.2, while id, number, expected duration and labour requirement of activities are reported in Table 5.1.

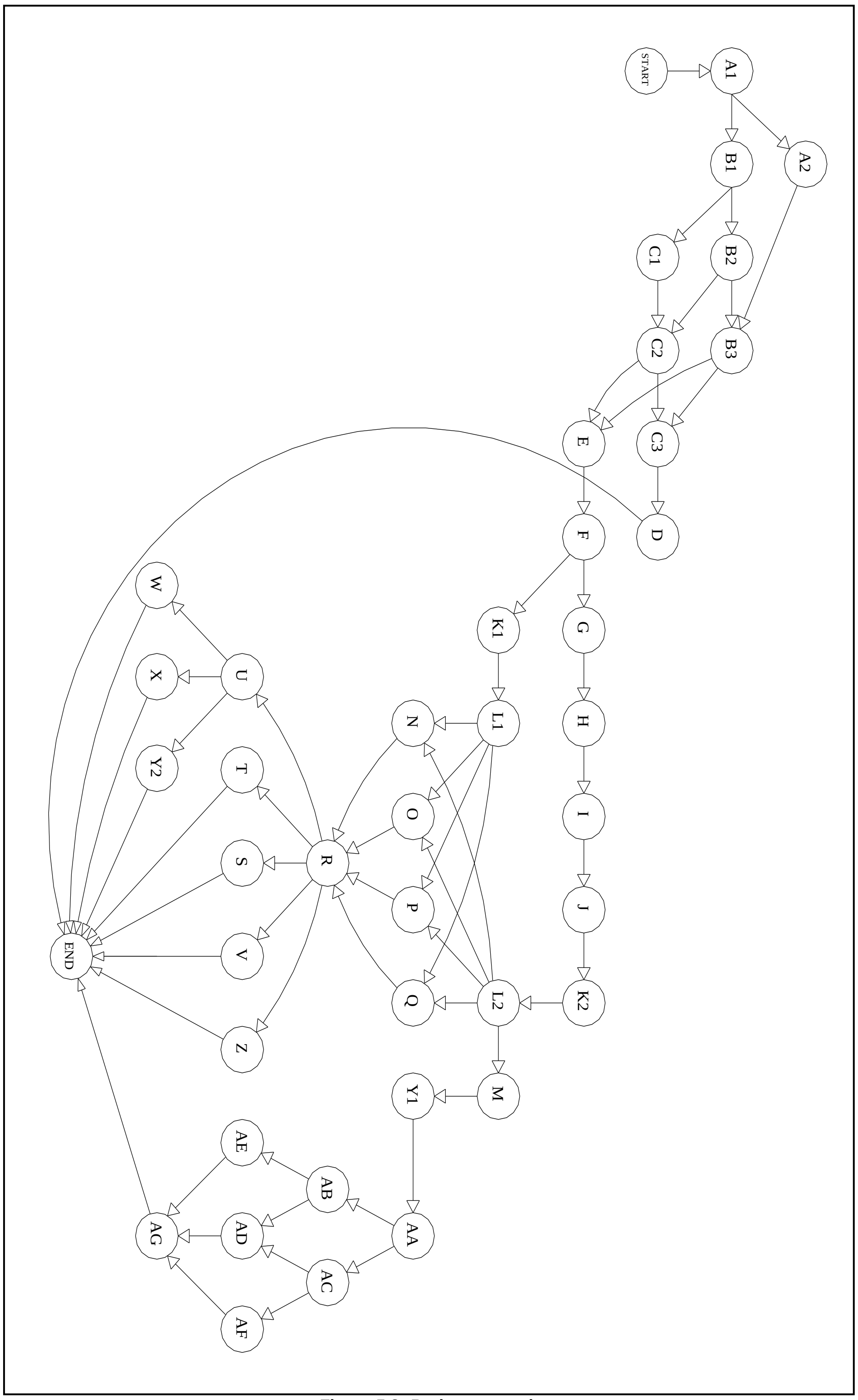


Figura 5.2. Project network

Activities ID	Activities number	Activities description	Expected activities duration (days)	Resource requirement per day
START	1	DUMMY START	0	0
A1	2	Building yard delimitation	10	7
A2	3	Building yard resource preparation	20	7
B1	4	Excavation works	16	5
B2	5	Grading	16	5
B3	6	Site preparation	18	5
C1	7	Basement foundations	16	6
C2	8	Footings	16	6
C3	9	Foundation walls	18	6
D	10	Crawl space	50	6
E	11	First floor	100	6
F	12	Second floor	75	6
G	13	Third floor	75	6
H	14	Fourth floor	50	6
I	15	Fifth floor	50	6
J	16	Roofing	45	23
K1	17	Exterior wall	90	6
K2	18	Exterior wall	30	6
L1	19	Interior wall	70	6

Table 5-1: Activities ID, number, details, duration and resource requirement

We used the initial project network developed by the project team and their activity time estimates as input. Afterwards the project managers considered two risks to be important: errors in execution and poor weather conditions. Estimates of an optimistic, a pessimistic and a most likely estimate of activities duration were obtained from the project management team during an interview session and compared with the historical data of similar construction projects completed at the University of Calabria. A triangular distribution has then been used for characterizing the randomness of activities duration.

5.4. Analysis of Results

In a first experiment we have analyzed the effectiveness of the proposed scheduling procedure in hedging against uncertainty. In particular, we generated two different schedules.

A protected schedule for the UNICAL project constructed by using the scheduling mechanism implemented in the RAH in which the probability level α was set to 0.95, that is, we expect with a probability at least 0.95 that the project delivery date will met (we expect a so called timely project completion probability- TPCP for short-of 0.95) and an unprotected deterministic schedule generated by the project managers on the basis of their own experience and with the support of a quantitative tools for the solution of resource constrained project scheduling problems.

Then, managers estimated project completion time taking into account external/internal critical factors such as weather conditions, manpower and resources availability, most-likely durations of activities.

In order to perform an a posteriori analysis we tested the two schedules in a simulation phase, in which a number of possible project realizations, called scenarios, were simulated and a reactive scheduling procedure was applied for each scenario, opting for never starting activities earlier than their prescheduled start time in the baseline schedule.

The schedule generated by managers sets a completely unrealistic planned project delivery date of about 1250 days, with a probability around 50% to be exceeded. Having a TPCP=*0.5* can be very unsatisfactory especially for construction projects for which very high penalties are usually associated to heavy due date violations and schedule breakages. Such observation underlines the crucial value of an accurate planning phase and, as a byproduct, the inadequacy of traditional scheduling procedures in facing uncertainty.

The schedule generated by the proposed approach imposing in the RAH a reliability level of 0.95 results in a planned project delivery date of 1517 with a TPCP equal to 0.96. A second analysis has been carried out with the aim of comparing the methodology against a total of 14 scheduling procedures. To assess the performance of RAH we used as benchmark heuristics the above mentioned SGSs considering, instead of the deterministic durations, the α-fractiles of activities duration. We shall refer to these heuristics in the following as parallel separate chance-constraints based heuristic (PSCCBH) and serial separate chance-constraints based heuristic (SSCCBH). Regardless the schedule generation scheme applied, the resulting schedule depends on the ordering criterion adopted.

In order to generate the priority list, we used several static priority rules taken from the literature. We tested the 'LST' rule (Kolisch et al. 1995) (activities are ordered by increasing value of their latest starting time), the 'LFT' rule (David and Patterson, 1975) (activities are ordered by increasing value of their latest finish time) and the 'MTS' rule (Alvarez-Valdes and Tamarit, 1989) (activities are ordered by decreasing value of the number of their successors).

Furthermore, we have proposed some new rules for the problem at hand. In particular the 'MinC' rule orders activities by increasing value of their resource requirement; the 'MinD' rule sorts activities by increasing value of their α-fractile, the 'MaxC' rule orders activities by decreasing value of their resource requirement and, the last one, 'MaxD*C' rule orders activities by decreasing value of their α-duration*resource requirement.

We have also compared our algorithm with the approaches present in the scientific literature closer to our work; that is the starting time criticality (STC) (Van de Vonder et al. 2008) and the resource flow dependent float factor (RFDFF) heuristics (Van De Vonder et al. 2006).

We have tested all the algorithms, but the RFDFF and the STC heuristics, for the following α values {0.8; 0.85; 0.9; 0.95; 0.99}. For the sake of clarity, although RFDFF and STC heuristics construct exactly the same schedule whatever the risk averseness of the decision maker (i.e. for all the α values), we have reported the related results for all the probability levels tested. The following measures were used to assess the performance of the baseline schedules obtained: average tardiness (TAVG), average timely project completion probability (TPCP), average number of jobs whose starting time in the actual schedule differs from the baseline schedule (#del) and CPU time in seconds (time) on a PC Pentium III, 667 MHz, 256 MB RAM.

The tardiness of activity i represents the difference between its actual completion time and the planned one in the baseline schedule. It is evident that if a penalty is due for each extra period required to execute an activity, tardiness represents an important measure of performance for scheduling in construction project.

Table 5.2 shows the results collected for each approach.

As evident RAH ranks best among the heuristics. The performances of the PSCCBH and the SSCCBH are clearly indistinguishable, and depend on the ordering criterion adopted.
Unfortunately, there is no unitary evidence of one criterion over the others. For these heuristics, the number of activities whose actual starting time exceeds the planned starting time (reported in the column #del) is quite satisfactory, especially for increasing α values.
With the aim of assessing the variation of the performances of the various heuristics tested with respect to the probability level α, we show in Figure 5.4 the tardiness, and in Figure 5.5 the TPCP for different α values. In both cases, the average performance over PSCCBHs has been considered for comparison. The tardiness of the STC and RFDFF heuristics has not been reported in Figure 5.4, since it is very high. As evident, the schedule performances get worse with decreasing α values. This result is expected, since there is clearly a correlation between the schedule robustness and the values of α.
As far as the computational effort is concerned, we observe that PSCCBH and SSCCBH are very efficient. This is due to the simple schedule construction procedures, based on the parallel and serial schedule generation schemes. On the contrary, the computational time of RAH is quite huge. This is due to the extra effort required for solving, at each decision point, the probabilistic model.

5.5. Conclusions

This chapter presents an approach for efficient scheduling of construction project problem under uncertainty. We provide a tool equipped with a simulation module for an a posteriori assessment of the schedule performance able to support managers in developing a workable and realistic project schedule. The generated baseline schedule serves in this phase as a guideline for project control and monitoring.
Managers have been provided with a tool with a graphical interface and very easy to use, capable to quantify the risk associated to a baseline schedule and to support their experience in the planning phase of the project. Furthermore, managers can consider a more realistic delivery date when take part in a call in which it is preferable that the starting time of

activities and the ending time of the project will be kept fixed in time as much as possible.

Appendix A

Proof of Theorem 3.1.

$$P\ n\ =\ {}_{i\in V,i\neq n}\ P\ i\ =$$

$$= P\ n-1\ +\ P\ n-2\ +\ P\ n-3\ +,\ldots,P\ 1\ =$$

$$=\ \ P\ n-2\ +\ P\ n-3\ +,\ldots,P\ 1\ \ \ +P\ n-2\ +\ P\ n-3\ +,\ldots,P\ 1$$

$$= 2P[n-3]$$

$$= P\ n-3\ =\ P\ n-4\ +\ P\ n-5\ +,\ldots,P\ 1$$

$$= (P\ n-5\ +\ P\ n-6\ +,\ldots,P\ 1\ =\ 2P\ n-4\ .$$

$Thus\ \ P\ n\ \ *\ P\ n-4\ .$

$Exploiting\ also\ P\ \ n-i\ \ as\ a\ function\ of\ P\ \ n-i-1\ ,P\ \ n-i-2\ ,\ldots,P\ 1$
$we\ come\ up\ with\ the\ following\ formula$:

$$P\ n\ =\ \underbrace{2*\ldots*2}_{n-1}P\ 1$$

Appendix B

Let us consider the following probabilistic constraints to be solved at a generic decision point t_g:

$$\underset{i \in S_g}{} \ln F_{\mathrm{d_i}\ \omega}\ (\psi_i) \geq \ln(1-\epsilon)\,.$$

Since the values ψ_i have to satisfy the above constraint, it is evident that $\psi_i \geq l_i$ where l_i represents the $1-\epsilon$ – quantile of the marginal distribution $F_{\mathrm{d_i}\ \omega}$, that is the smallest integer value such that $F_{\mathrm{d_i}\ \omega}\ (\psi_i) \geq \ln(1-\epsilon)$. In the case of log–concave marginal distribution, it is possible to rewrite ψ_i in a 0 − 1 formulation. If $l_i + k_i$ is a known upper bound, ψ_i can be written as $\psi_i = \ l_j + \ \overset{k_i}{\underset{k=1}{}} \psi_{ik}$.

where ψ_{ik} are binary variables. Therefore, the probabilistic constraints can be rewritten as:

$$r_j = r_j = \qquad \underset{i \in S_g}{} \qquad \overset{k_i}{\underset{k=1}{}} a_{ik}\psi_{ik} \geq p$$

where $a_{ik} = \ln F_{\mathrm{d_i}\ \omega}\ (l_i + k) - \ln F_{\mathrm{d_i}\ \omega}\ (l_i + k - 1)$ and $p = \ln(1-\epsilon) - \ln F(l)$. Let us consider, for instance, a problem of the type (2)-(4) with joint probabilistic constraint involving two activities, namely 1 and 2. Let us suppose that $\mathrm{d}_1\ \ \omega$ and $\mathrm{d}_2\ \ \omega$ follow a Poisson distribution with mean 2 and 1, respectively, and $\epsilon = 0.2$. Let also introduce two integer vectors

$$\psi_1 = \ l_1 + \qquad \overset{k1}{\underset{k=1}{}} \psi_{1k} = 3 + \psi_{11} + \psi_{12} + \psi_{13}$$

and

$$\psi_2 = \ l_2 + \qquad \overset{k2}{\underset{k=1}{}} \psi_{2k} = 2 + \psi_{21} + \psi_{22}$$

where

l_1 s the $1-\epsilon$ –quantile of the distribution function of the random variable $\mathrm{d}_1, \mathrm{F}_{\mathrm{d}_1}$, $\mathrm{k}_1 = 3$ is an upper bound on the value of d_1 (i.e. 1-quantile of the distribution function of the random variable d_1). Analogue meanings have l_2 and k_2.

Therefore, the joint probabilistic constraints can be transformed in the following mixed integer problem.

$\mathrm{minMpar}$

$\mathrm{Mpar} \; \geq \; 3 + \psi_{11} + \psi_{12} + \psi_{13}$

$\mathrm{Mpar} \; \geq \; 2 + \psi_{21} + \psi_{22}$

$a_{11}\psi_{11} + a_{12}\psi_{12} + a_{13}\psi_{13} + a_{21}\psi_{21} + \; a_{22}\psi_{22} \geq \; \ln 0.8 - \; \ln(0.86 * 0.92)$-

$\psi_1 = 3 + \psi_{11} + \psi_{12} + \psi_{13}$

$\psi_2 = 2 + \psi_{21} + \psi_{22}$

$\psi_{11}, \psi_{12}, \psi_{13}, +\psi_{21}, \psi_{22} \;\; \in \{0,1\}$

where, for instance $a_{11} = \ln F_{\mathrm{d_i}} - \ln F_{\mathrm{d_i}} = 0.95 - 0.86$.

If we instead suppose that $\mathrm{d}_1 \;\; \omega$ and $\mathrm{d}_2 \;\; \omega$ follow a Uniform distribution $U_1 \;\; a_1, b_1 \;\; , U_1 \;\; a_2, b_2$, respectively, and $\epsilon = 0.2$ the joint probabilistic constraints in problem can be transformed in the following nonlinear problem.

$\mathrm{minMpar}$

$\mathrm{Mpar} \; \geq \; \phi_\mathrm{i}$

$\mathrm{Mpar} \; \geq \; \phi_2$

$\ln F\mathrm{d}_1 \;\; \omega \;\; (\phi_1) + \ln F\mathrm{d}_2 \;\; \omega \;\; (\phi_2) \; \geq \; \ln 0.8$

$\phi_1, \phi_{2,} \mathrm{Mpar} \; \geq 0$

Appendix C

Notation list

SET AND INDEXES:

$N = (1..n)$ set of activities indexed by i,j.

$A = \{(i, j) : i, j \in N\}$ set of precedence relations.

$G = (N,A)$ compound set representing the project network as a directed acyclic graph.

K set of renewable resources indexed by k.

t time index.

s_g set of activities scheduled at decision point t_g.

E_g set of all eligible activities which can be precedence feasibly started at t_g

PARAMETERS

t_g decision point.

d_i deterministic duration associated to the activity i.

$\hat{d}_i$ random variable representing uncertain duration of activity i.

$F_{\hat{d}_i}$ cumulative probability distribution function of the random variable $\hat{d}_i$.

r_{ik} constant amount of resource k required by activity i.

$r_k(t_g)$ residual resource availability at time t_g for resource k.

M positive big value.

γ parameter adjusted dynamically.

VARIABLES

c_i completion time of the activity i.

β_i binary variable taking value one if activity i can be feasibly scheduled and zero otherwise.

C makespan of the partial schedule built *at* t_g.

Bibliography

Alvarez-Valdes, R. & Tamarit, J.M. 1989. Heuristic algorithms for resource constrained project scheduling: A review and an empirical analysis, in: R. Slowinski, J. Weglarz (eds.), Advances in project scheduling, Elsevier, Amsterdam, 113-134.

Atkinson, R., Crawford, L. & Ward, S. (2006). Fundamental uncertainties in projects and the scope of project management. International Journal of Project Management, 24, pp 687-698.

Aytug, H., Lawley, M., McKay, K., Mohan, S. & Uzsoy, R. (2005). Executing production schedules in the face of uncertainties: A review and some future directions. European Journal of Operational Research, 161(1), pp 86–110.

Artigues, C., Michelon, P. & Reusser, S. (2003). Insertion techniques for static and dynamic resource-constrained project scheduling. European Journal of Operational Research, 149(2), pp 249–267.

Artigues, C. & Roubellat, F. (2000). A polynomial activity insertion algorithm in a multi-resource schedule with cumulative constraints and multiple modes. European Journal of Operational Research, 127, pp 294–316.

Ballestin, F. (2007). When it is worthwhile to work with the stochastic RCPSP?. Journal of Scheduling, 10, pp 153-166.

Beraldi, P., Bruni, M.E., Guerriero, F., Pinto, E. (2007) Heuristic procedures for the robust project scheduling problem, Technical Report, ParCoLab, DEIS, University of Calabria.

Bisschop, J. & Roelofs, M. (2007). AIMMS. 3.7 User's guide. Paragon Decision Technology B.V., The Netherlands.

Blazewicz, J., Cellary, W., Slowinski, R. & Weglarz, J. (1986). Scheduling under resource constraints - deterministic models. Annals of Operations Research, 7, pp 1–359.

Brucker, P., Drexl, A., Möhring, R., Neumann, K. & Pesch, E. (1999). Resource-constrained project scheduling: notation, classification, models and methods. European Journal of Operational Research, 112, pp 3–41.

Bruni, M. E., Guerriero, F. & Pinto, E. (2007). Solution Approaches to Find the α-critical path in Project Networks. Technical Report, ParCoLab, DEIS, University of Calabria.

Camps, J.A, (1996). Simple steps help minimize costs risks in project management, Oil and Gas Journal, 94 (4), pp 32-36.

Chapman, C.B., Ward S. Project Risk Management: Processes, Techniques and Insights, Wiley and Sons (1997).

Charnes, A., Cooper, W.W. & Thompson, G.L. (1964). Critical Path Analyses via Chance Constrained and Stochastic Programming. Operations Research, 12(3), pp 460–470.

Cooper, D.F. (1976). Heuristics for scheduling resource-constrained projects: An experimental investigation. Management Science, 22(11), pp 1186–1194.

CPLEX, ILOG CPLEX 6.5: Users Manual, CPLEX Optimization, Inc., Incline Village, NV, 1999.

Christofides, N., Alvarez-Valdes, R. & J.M. Tamarit (1987). Project scheduling with resource constraints: a branch and bound approach. European Journal of Operational Research, ,29, pp 262–273.

Davenport, A. & Beck, J. (2002). A survey of techniques for scheduling with uncertainty. Unpublished manuscript.

David, E. & Patterson, J. (1975). A comparison of heuristic and optimum solutions in resource-constrained project scheduling. Management Science, 21 (8), pp 944-955.

Deblaere, F., Demeulemeester, E., Herroelen, W. & Van de Vonder, S. (2007). Robust Resource Allocation Decisions in Resource-Constrained Projects. Decision Sciences, 38, pp 1-37.

Demeulemeester, E. & Herroelen, W. (2002). Project scheduling - A research handbook. Vol. 49 of International Series in Operations Research & Management Science. Kluwer Academic Publishers, Boston.

Dentcheva, D., Prékopa, A., Ruszczy´nski, A. (1998). On stochastic integer programming under probabilistic constraints. RUTCOR-Rutgers Center for Operations Research RRR 29-98, Rutgers University, 640 Bartholomew Road Piscataway, New Jersey 08854-8003, USA, 1998

Drezet, L. (2005). Résolution d'un problème de gestion de projets sous contraintes de resources humaines: de l'approche pr´edictive l'approche réactive. PhD thesis. Université Francois Rabelais Tours.

Dodin, B. (1985). Bounding the Project Completion Time of PERT Network.Operations Research, 33, pp 862–865.

Drud, A. S. CONOPT: A System for Large Scale Nonlinear Optimization, Reference Manual for CONOPT Subroutine Library, 69p, ARKI Consulting and Development A/S, Bagsvaerd, Denmark (1996).

Elmaghraby, S.E. (1989). The estimation of some network parameters in PERT model of activity network: Review and critique, in: R Slowinski, J Weglarz (Eds.), Advances in Project Scheduling, Part III, Elsevier, Amsterdam, pp 371- 432.

Elmaghraby, S. (2005). On the fallacy of averages in project risk management. European Journal of Operational Research, 165(2), pp 307–313.

Fulkerson, D.R. (1962). Expected Critical Path Lengths in PERT Networks.Operations Research, 10(6), pp 808–817.

Goel, V., Grossmann, I.E. (2006). A class of stochastic programs with decision dependent uncertainty. Mathematical Programming, 108 (2), pp 355-394.

Golenko-Ginzburg, D. & Gonik, A. (1997). Stochastic network project cheduling with non-consumable limited resources. International Journal of Production Economics, 48, pp 29-37.

Golenko-Ginzburg, D. & Gonik, A. (1998). A heuristic for network project scheduling with random activity durations depending on the resource allocation. International Journal on Production Economics, 55, pp 149-162.

Gordon, J. & Tulip, A. (1997). Resource scheduling. International Journal of Project Management, 15 (6), pp 359-70.

Guerriero, F. & Talarico, L. (2007). A Solution Approach to Find the Critical Path in a Time-Constrained Activity Network. Technical Report, ParCoLab, DEIS, University of Calabria, July 2007.

Guildford, W.S. (1998). Practical Risk Assesment for project management, International journal of project management, 16 (2),pp 130-131.

Hagstrom, J.M. (1988). Computational complexity of PERT problems. Networks, 18, pp 139–147.

Hagstrom, J.M. (1990). Computing the Probability Distribution of Project Duration in a PERT Network. Networks, 20, pp 231-244.

Hayes, R.W., Perry, J.G. Thompson, P.A. & Willmer, G. (1986). Risk management in engineering construction, Thomas Telford, London.

Herrera, L. (2006). Stochastic Critical Path. Proceeding of the 2006 Cristal Ball User Conference.

Herroelen, W., De Reyck, B. & Demeulemeester, E. (1998). Resource constrained scheduling: a survey of recent developments. Computers and Operations Research, 25, pp 279-302.

Herroelen, W. & Leus, R. (2004a). The construction of stable project baseline schedules. European Journal of Operational Research, 156, pp 550-565.

Herroelen, W. & Leus, R. (2004b). Robust and reactive project scheduling: a review and classification of procedures. International Journal of production Research, 42(8), pp 1599-1620.

Herroelen, W., & Leus, R. (2005). Project scheduling under uncertainty—Survey and research potentials. European Journal of Operational Research, 165, pp 289-306.

Igelmund, G. & Radermacher, F.J. (1983). Algorithmic approaches to preselective strategies for stochastic scheduling problems. Networks, 13, pp 29-48.

Jagannathan, R. (1974). Chance-Constrained Programming with Joint Constraints. Operations Research, 22 (2), pp 358–372.

Just, M.R. & Murphy, J.P. (1994). The effect of resource constraints on project schedules, Transaction of AACE International, Morgantown.

Kim, K. & Garza, M.J. (2005). Evaluation of the resource-constrained critical path method algorithms, Journal of Construction Engineering and Management, (131) 5, pp 522-532.

Klerides,e. and Hadjiconstantinou, E. (2010). A decomposition-based stochastic programming approach for the project scheduling problem under time/cost trade-off settings and uncertain durations. Computers and Operations Research, in press DOI: doi:10.1016/j.cor.2010.03.002*.*

Kolisch, R. & Hartmann, S. (1999). Heuristic algorithms for solving the resource-constrained project scheduling problem: classification and computational analysis. Kluwer Academic Publishers. In Weglarz J.: Project scheduling: recent models, algorithms and applications.

Kolisch, R. & Hartmann, S. (2006). Experimental investigation of heuristics for resource-constrained project scheduling: an update., European Journal of Operational Research, 174 (1), pp 23-37.

Kolisch, R. & Padman, R. (1999). An integrated survey of deterministic project scheduling. Omega, 49, pp 249–272.

Kolisch, R., Sprecher, A. & Drexl, A. (1995). Characterization and generation of a general class of resource constrained project scheduling problems. Management Science, 41 (10), pp 1693-1703.

Kolisch, R. & Sprecher, A. (1997). PSPLIB - a project scheduling problem library. European Journal of Operational Research, 96, pp 205-216.

Lambrechts, O., Demeulemeester, E. and Herroelen, W. (2007). A tabu search procedure for developing robust predictive project schedules. International Journal of Production Economics, 111 (2), pp 496–508.

Lambrechts, O., Demeulemeester, E. and Herroelen, W. (2008). Proactive and reactive strategies for resource-constrained project scheduling with uncertain resource availabilities. Journal of Scheduling, 11 (2), 121–136.

Leus, R. (2003). The generation of stable project plans. Ph.D. Thesis, Department of Applied Economics, Katholieke Universiteit Leuven, Belgium.

Leus, R. & Herroelen, W. (2004). Stability and resource allocation in project planning. IIE Transactions, 36, pp 667-682.

Malcolm, D..G, Roseboom, J.H., Clark, C.E. & Fazar, W. (1959.) Application of a Technique for Research and Development Program Evaluation. Operations Research, 7(5), pp 656–669.

Maes, J., Vandoren, C., Sels, L. & Roodhooft, F. (2000). Onderzoek naar oorzaken van faillissementen van kleine en middelgrote bouwondernemingen. Unpublished manuscript, CTEO Leuven.

Marshall, H.E. (1988). Techniques for treating uncertainty and risk in the economic evaluation of building investments, US Department of Commerce, Washington DC.

Mehta, S. & Uzsoy, R. (1998). Predictive scheduling of a job shop subject to breakdowns. IEEE Transactions on Robotics and Automation, 14, pp 365–378.

Miller, B. & Wagner, H. (1965). Chance constrained programming with joint constraints. Operations Research, 13 (6), pp 930-945.

Mohamed, S., & McCowan, A. (2001). Modelling project investment decisions under uncertainty using possibility theory. IJPM, 19, pp 231-241.

Möhring, R., Radermacher, F. & Weiss, G. (1984). Stochastic scheduling problems I - Set strategies. Zeitschrift f¨ur Operations Research, 28, pp 193– 260.

Möhring, R., Radermacher, F. & Weiss, G. (1985). Stochastic scheduling problems II - General strategies. Zeitschrift f¨ur Operations Research, 29, pp 65–104.

Möhring, R.H. & Stork, F. (2000). Linear preselective policies for stochastic project scheduling. Mathematical Methods of Operations Research, 52, pp 501-515.

Nkasu, MM. (1994). COMSARS: a computer-sequencing approach to multiresource-constrained scheduling -part 1: deterministic networks, International Journal of Project Management 12 (3), pp 183-92.

Park M. & Peña-Mora, F. (2004). Reliability Buffering for Construction Projects, Journal of Construction Engineering & Management, 130 (5), pp 626-637.

Prékopa, A. (1995). Stochastic Programming. Kluwer Scientific Publisher, Boston.

Pritsker, A., Watters, L. & Wolfe, P. (1969). Multiproject scheduling with limited resources: A zero–one programming approach. Management Science, 16, pp 93–107.

Rabbani, M., Fatemi Ghomi, S.M.T., Jolai, F. & Lahiji, N.S. (2007). A new heuristic for resource-constrained project scheduling in stochastic networks using critical chain concept. European Journal of Operational Research, 176, pp 794-808.

Regev, S., Shtub, A & Ben-Haim, Y. (2006). Managing project risks as knowledge gaps, Project Management Journal, 37 (5), pp 17-25.

Schuyler, J. (2001). Risk and Decision Analysis in Projects. Project Management Institute.

Schatteman, D. Herroelen, W. & Van de Vonder,S. (2008). A. Boone, A methodology for integrated risk management and proactive scheduling of construction projects, Journal of Construction Engineering & Management, 134 (11), 885-895.

Shih, N.H. (2005). Estimating completion-time distribution in stochastic activity networks. Journal of the Operational Research Society, 56, pp 744-749.

Simister, S.J. (1994). Usage and benefits of project risk analysis and management, International Journal of Project Management, 12 (1) 5-10.

Sörensen, K. (2001). Tabu searching for robust solutions. Proceedings of the 4th Metaheuristics International Conference.

Soroush, H.M. (1994). The Most Critical Path in a PERT Network. The Journal of Operational Research society 45(3), pp 287–300.

Stork, F. (2000). Branch-and-bound algorithms for stochastic resource-constrained project scheduling, Research Report. 702/2000. Technische Universitat Berlin

Stork, F. (2001). Stochastic Resource-Constrained Project Scheduling. PhD thesis. Technical University of Berlin, School of Mathematics and Natural Sciences.

Sullivan, R.H. & Hayya, J.C. (1980). A Comparison of the Method of Bounding Distribution (MBD) and Monte Carlo Simulation for Analyzing Stochastic Acyclic Networks. Operations Research, 28, pp 614–617.

Tavares, L.V., Ferreira, J.A.A. & Coelho, J.S. (1998). On the optimal management of project risk. European Journal of Operational Research 107, pp 451-469.

Tsai, Y.W. & Gemmil, D.D. (1998). Using tabu search to schedule activities of stochastic resource-constrained projects. European Journal of Operational Research 111, pp 129-141.

Van de Vonder, S., Demeulemeester, E., Herroelen, W. & Leus, R. (2005). The use of buffers in project management:The trade-off between stability and makespan. International Journal of Production Economics, 97, pp 227-240.

Van De Vonder, S., Demeulemeester, E., Herroelen,W. & Leus, R. (2006). The trade-off between stability and makespan in resource-constrained project scheduling, International Journal of Production Research, 44(2), pp 215-236.

Van de Vonder,S., Demeulemeester, E. & Herroelen,W. (2007). A classification of predictive-reactive project scheduling procedures. Journal of Scheduling, 10, pp 195-207.

Van de Vonder,S., Demeulemeester, E. & Herroelen, W. (2008). Proactive heuristic procedures for robust project scheduling: An experimental analysis. European Journal of Operational Research, 189, pp 723-733.

Van Slyke, R.M. (1963). Monte Carlo Methods and the PERT Problem. Operations Research, 11, pp 849–860.

Vaziri, K. Carr P. G. & Nozick, L. (2007). Project Planning for Construction under Uncertainty with Limited Resources, Journal of Construction Engineering & Management, 133 (4) ,pp 268-276 .

Wang, J. (2005). Constraint-based schedule repair for product development projects with time-limited constraints. International Journal of Production Economics, 95, 399-414.

Wiest, J.D. & Levy, F. K. (1977). A Management Guide to PERT/CPM with GERT/PDM/DCPM and Other Networks, Prentice-Hall, New Jersey.

Woodworth, B.M. & Shanahan, S. (1998). Identifying the critical sequence in a resource-constrained project, International Journal of Project Management 6 (2), pp 89-96.

Wu, S., Storer, H. & Chang, P.C. (1993). One-machine rescheduling heuristics with efficiency and stability as criteria. Computers and Operations Research, 20, pp 1–14.

Zeng, J. Min A. & Smith, N. J. (2007). Application of a fuzzy based decision making methodology to construction project risk assessment International Journal of Project Management, 25,pp 589-600

Zhu, G., Bard, J.F. & Yu, G. (2005). Disruption management for resource-constrained project scheduling. Journal of the Operational Research Society, 56, pp 365-381.

Zhu, G., Bard, J.F. & Yu, J.F. (2007). A two-stage stochastic programming approach for project planning with uncertain activity durations. Journal of Scheduling 10, pp 167-180.

Yakov Ben-Haim, Info-gap Decision Theory: Decisions Under Severe Uncertainty, 2nd edition, Academic Press, London (2006).

Yang, K.K. (1996). Effects of erroneous estimation of activity durations on scheduling and dispatching a single project. Decision Sciences, 27(2), pp 255–290.

Yao, M.J. & Chu, W. M. (2007). A new approximation algorithm for obtaining the probability distribution function for project completion time. Computers and Mathematics with Applications, 54, pp 282–295.