ON CONSTRUCTION OF STABLE PROJECT SCHEDULES

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

BARAN GÖRMEZ

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
INDUSTRIAL ENGINEERING

DECEMBER 2005

Approval of the Graduate School of Natural and Applied Sciences

Prof. Dr. Canan Özgen
Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of Master of Science.

Prof. Dr. Çağlar Güven
Head of Department

This is to certify that we have read this thesis and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science.

Assoc. Prof. Dr. Canan Sepil
Supervisor

Examining Committee Members

Assoc. Prof. Dr. Yasemin Serin (METU, IE)

Assoc. Prof. Dr. Canan Sepil (METU, IE)

Assoc. Prof. Dr. Sedef Meral (METU, IE)

Assoc. Prof. Dr. Gülser Köksal (METU, IE)

Assoc. Prof. Dr. İrem Dikmen (METU, CE)

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last name : Baran GÖRMEZ

Signature :

ABSTRACT

ON CONSTRUCTION OF STABLE PROJECT SCHEDULES

Görmez, Baran
M. Sc. Department of Industrial Engineering
Supervisor: Assoc. Prof. Dr. A. Canan Sepil

September 2005, 101 pages

It is a well-known fact that project activities are subject to considerable uncertainty, which may lead to multiple schedule disruptions during project execution. As a result, the random nature of activity durations has been the subject of numerous research efforts since the introduction of the initial PERT. A common problem which arises in project management is the fact that the planned schedule is often disrupted by several uncontrollable factors like weather conditions, other environmental factors, additional time that might be required for rework and correction of detected defects. As a result, project managers are often unable to meet the promised completion dates. It is therefore vital to take into account such possible disruptions and their potential negative consequences at the project schedule design stage. Hence, the ability of the pre-schedule to absorb disruptions may be very important in such settings. At this point two new criteria are used in modern scheduling literature: "robustness" and "stability". In this thesis, we propose several stability measures. These measures are embedded in a tabu search algorithm to generate stable schedules in a multi resource environment subject to random disruptions.

Keywords: Project Scheduling, robustness, stability, Tabu Search.

ÖZ

KARARLI PROJE PLANLARI OLUŞTURULMASI

Görmez, Baran
Yüksek Lisans, Endüstri Mühendisliği Bölümü
Tez Yöneticisi: Doç. Dr. A. Canan Sepil

Eylül 2005, 101 sayfa

Proje aktivitelerinin, proje uygulamaları sırasında pek çok program aksamalarına neden olabilecek önemli derecede belirsizliğe maruz kalmakta olduğu iyi bilinen bir gerçektir. Sonuç olarak, PERT'in ilk sunumundan bu yana aktivite sürelerinin rastlantısallığı üzerine pek çok araştırma yapılmıştır. Proje yönetiminde sıklıkla karşılaşılan bir problem de planlanan programın fark edilen hataların düzeltilmesi ve yeniden yapılması için ekstra çalışma zamanı gereksinimi, iklim etkisi, çeşitli dışsal etkenler gibi birçok kontrol dışı faktör tarafından sık sık aksatılmasıdır. Sonuç olarak, Proje yöneticileri genellikle vaat edilen tamamlama süresini karşılayamaz. Bu sebeple proje planlamasında bu tür aksaklıkları ve neden olabileceği olumsuzlukları dikkate almak gereklidir. Dolayısıyla ön-programlamanın aksaklıkları absorbe etme kabiliyeti çok önemlidir. Bu noktada, modern programlama literatüründe iki yeni kriter kullanılmaktadır: “sağlamlık ve kararlılık”. Bu tezde, birkaç kararlılık ölçüsü önerilmektedir. Bu ölçüler rastlantısal aksamalara maruz kalan çok kaynaklı bir ortamda kararlı programlar üretebilmek için Tabu Arama algoritması içinde kullanılmıştır.

Anahtar kelimeler: Proje Programlanması, Sağlamlık, Kararlılık, Tabu Arama.

To My Family

And Also To İREMİM

ACKNOWLEDGEMENTS

I wish to express my deepest gratitude to my supervisor Assoc. Prof. Dr. Canan Sepil for her guidance, advices, criticisms and encouragements and insights throughout the research.

I would like to express my special appreciation to my family for believing in me and for their both financial and moral support during this study.

I also wish to thank to my dear friends İrem Önoğlu, Deniz Genç, H. Günalp Aşan, Ayça Yetere, Y. Bahadır Duygulu, R. Kaya Göktaş, Ebru Harmancı, Güray Doğan, Fatma Öztürk, Mehmetcan Dinçer, Oğuz Solyalı, Barış Bekki and Işıl Öztürk for their helps, hospitality and friendships throughout my study.

Finally, I am very thankful to İREMİM (İrem Önoğlu) for her love, for her encouragement and for her existence in my life throughout my study.

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

Timely and cost effective production is becoming increasingly important in today's global competitive markets. Scheduling is an integral part of production systems planning since the schedule serves as an overall plan upon which many other shop activities are based (e.g. short-term labor planning, ordering and preparation of raw-material, and planning for tooling and set-up activities) and it serves to maximize performance of the manufacturing facility. Effective production scheduling is particularly important in the large, complex manufacturing systems as in high-technology industries that the simple manual techniques are inadequate to obtain good results. By properly planning the timing of shop-floor activities, performance criteria such as makespan, flow time, or tardiness can be optimized.

Project scheduling deals with the allocation of scarce resources to a set of interrelated activities that are usually directed toward some major output and require a significant period of time to perform. Since the early days of operations research, project scheduling has attracted an ever growing attention. This interest is largely motivated by its great practical importance in diverse industries for designing new production processes, developing new products, setting up new facilities, and implementing new information systems, to cite just a few applications. The two well-known network techniques CPM and PERT, were both developed in the late 1950s to determine the minimum completion time of a project in such a way that the precedence relationships among activities are satisfied. However, in most practical situations the activities of a project require several additional resources. These resources may include manpower, machines, equipment, and capital budget.

Ignoring the resource constraints may result in an infeasible schedule with several resource conflicts. During the last 20 years, Resource Constrained Project Scheduling Problems (RCPSPs) have generated a great amount of research in the operations research literature. Solving a RCPSP means providing an optimal project schedule that not only satisfies the precedence constraints but also provide a feasible allocation of the required resources (i.e., resolve resource conflicts).

It is a well-known fact that project activities are subject to considerable uncertainty, which may lead to multiple schedule disruptions during project execution. As a result, the random nature of activity durations has been the subject of numerous research efforts since the introduction of the initial PERT. The issues of project management under uncertainty and risk management have recently received growing attention. Nevertheless, the development of a pre-computed baseline schedule (pre-schedule) with the objective of assuring stability in the start times of the activities, rather than the minimization of the expected project duration or some other regular objective function, has been mostly overlooked so far.

A common problem which arises in project management is the fact that the planned schedule is often disrupted by several uncontrollable factors like weather conditions, other environmental factors, additional time that might be required for rework and correction of detected defects. As a result, project managers are often unable to meet the promised completion dates. It is therefore vital to take into account such possible disruptions and their potential negative consequences at the project schedule design stage. In a multi-project environment, it may be necessary to make advantage bookings of key staff or equipment to guarantee their availability. Hence, the ability of the pre-schedule to absorb disruptions may be very important in such settings. Other sources of the need for such stability can be hard delivery dates of suppliers or subcontractors, or in a larger sense, a hard due date for intermediate or final deliverables, in other words any time restriction that is external to the project itself.

As a result, the performance of the schedule will degrade. It is therefore desirable to generate schedules that are "robust" or “stable” within a reasonable range of disruptions such as machine breakdowns and processing time variability. By robust it is meant that the performance of the schedule remains high in the presence of disruptions. Robust schedules are insensitive to disruptions. A stable schedule is the schedule that the realized schedule is not deviated from the planned schedule. The performance of the realized schedule is more important than the performance of the planned initial schedule since it is reality not an anticipation as the other.

In this study, we consider project stability problem in a resource constrained project environment. Moreover, we assume that there is a time window corresponding to a time period in which there might be an anticipated disturbance affecting the duration of a set of activities scheduling during that time window. We consider several stability measures and use them in a Tabu Search algorithm that generates stable schedules.

There are many studies related with robust and stable schedules in scheduling theory. Wu et. al (1993) used pairwise swapping and Genetic Algorithm to optimize the makespan and stability of the new schedule obtained after rescheduling a single machine system subjected to disruption. Leon et. al. (1994) employed average slack as a surrogate measure for expected deviation. They tried to find a preschedule that optimizes the expected makespan and deviation from the preschedule under disruptions. Daniels and Kouvelis (1995) aimed to find the least performance measure degradation in worst possible scenario. Mehta and Uzsoy (1998) studied on generating stable schedule. Their performance measure was maximum lateness. O’Donovan et al. (1999) also worked on generating stable schedules. They used total tardiness as performance measure. Sabuncuoğlu and Bayız (2000) study the reactive scheduling problem. Their performance criteria are makespan and mean tardiness.

There are very limited number of studies in the literature considering project stability. Herreolen et al. (2002, 2004) considered the construction of stable project schedules with no resource constraints with the assumption that exactly one anticipated disturbance will occur for the project, resulting with an increase in the duration of a single activity.

The following parts of this study are organized as follows: In the next chapter a literature review on the subject is given. Then we provide problem definition and notations in Chapter 3. In Chapter 4 experimental studies are conducted on networks generated by RanGen with the measures and algorithms explained in the previous chapters. Finally concluding remarks and future research directions will be given in the last chapter.

CHAPTER 2

LITERATURE REVIEW

There are a number of studies on robust and stable scheduling problems. Leon et. al. (1994), developed a definition of schedule robustness which comprises post-disturbance makespan and post-disturbance makespan variability. They have developed robustness measures and robust scheduling methods for the case where a "right-shift" control policy is used, which is on occurrence of a disruption; unfinished jobs are delayed as much as necessary to accommodate the disruption. They derived an exact measure of schedule robustness for the case in which only a single disruption occurs within the planning horizon. A surrogate measure (Average Slack) is developed for the more complex case in which multiple disruptions may occur and then employed in a genetic algorithm to generate robust schedules for job-shops. The experiments show that the robust schedule becomes superior only when processing - time variability is sufficiently large.

Wu et. al. (1999), developed an alternative approach to the scheduling problem by developing a decomposition for job shop scheduling problems based on an ordered assignment of job operations to subsets. First, a critical subset of scheduling decisions is determined and this subset is solved at the beginning of planning horizon while the rest of the scheduling decisions are relegated to future points in time. The method is called as “preprocess first schedule later (PFSL)” scheme. The crucial decisions (preprocessing) are made through a branch and bound algorithm and remaining scheduling decisions are made dynamically. They compared the model with ATC heuristic. Computational experiments show that PFSL scheme provides more robust performance under a wide range of disturbances as compared to traditional methods.

Daniels and Kouvelis (1995) formalized the concept of schedule robustness for production environments with uncertain processing times. They study a single machine problem where the performance criterion is the total flow time. They defined two measures of robustness that focus on a given schedules' worst case deviation from optimality and developed a branch-and-bound approach for the single-machine robust scheduling problem. The computational experiments indicated that the schedules found by their algorithms are more robust than the sequences that are found by SEPT (shortest expected processing time) heuristic.

Mehta and Uzsoy (1998) presented a predictable scheduling approach where the objective is to absorb disruptions without affecting planned external activities such as material procurement and preventive maintenance while maintaining high shop performance. Scheduling with such dual objectives are called as predictable scheduling (PS) and the scheduling which aimed to improve the shop performance without considering the effects of disruptions, and react them as they occur, are called as predictive –reactive scheduling (PRS) by the authors . They used the available information on known uncertainties to generate a predictive schedule. The effect of disruptions is measured by the difference between planned and realized completion times and additional idle times are inserted into the predictive schedule to reduce the completion time deviations. The experimental results show that significant improvement provided in predicting by predictable scheduling with very little degradation in realized schedule.

O'Donovan et. al. (1999) also present a predictable scheduling approach where the effects of disruptions are measured by the deviations of job completion times as in the study of Mehta and Uzsoy (1998). They aimed to minimize total tardiness on a single machine with stochastic machine failures. They inserted idle times into the schedule to absorb the impacts of breakdowns. After computational experiments it is shown that it is better to consider predictability when rescheduling as opposed to just optimizing the primary performance measure.

Bowers (1995) described a number of measures of an activity's importance in a network and compared in an application to an aircraft development. A quantitative comparison of the measures is developed based on simulation of the process of management identifying the key activities and directing their control efforts.
He used the resource constrained (RC) floats as scheduling algorithm which is summarized below.

A conventional resource is employed first, using a minimum latest start (MINLS) heuristic to decide priorities. During the construction of the resource constrained schedule, the sources of each activity's resources are noted and this information is then used to create additional links explicitly representing the resource dependencies in the network. A reverse pass of the network, incorporating the resource links then provides a set of resource constrained latest start and finish times (LS and LF). The earliest start and finish times (ES and EF) were derived as part of conventional analysis and hence the resource constrained (RC) floats may be deduced.

RC float is just one possible measure of an activity's importance and it is worthwhile considering some alternative measures of importance such as conventional float, based on a non-RC network analysis or resource equivalent duration (RED), which is summarizing an activity's demand for the key resources. Also combination of conventional float and RED and the correlation of an activity's duration with the project's duration derived from a perturbation analysis of the network could be considered. However, the output from the resource constraint criticality analysis seems to show more information about project and can be used in variable roles in project planning and implementation.

Tavares et al. (1997) developed a model which deals with both the cost and the duration of each activity and they studied the problem of project scheduling in terms of the project's discounted cost and of the risk of not meeting its completion time. They stated that there have been a lot of research on uncertainty that has focused on the issue of duration but not enough attention is given to the randomness of the cost. They defined the project's risk as a function of the uncertainty of the duration and the cost of each activity and in terms of the adopted schedule which is considered the major decision of this problem. They bring up the term "float factor" in order to decrease the large set of variables into a single decision variable. The model is studied the risk in term of float factor (risk function). The risk can easily be estimated in terms of the float factor and its optimal value can therefore be computed.

Jensen (2001) aimed to achieve schedule robustness and flexibility for job shop problems. He used two methods for these purpose. The first way is the neighborhood-based robustness measure technique. The second way is a simpler idea applicable to tardiness problems; by minimizing a measure of lateness instead of tardiness, the slack in the schedules can be increased, which may improve the rescheduling performance of the schedules. The schedules were produced with a Genetic Algorithm which optimizes cost, neighborhood-based robustness measures and lateness robustness measures for the tardiness problem. After the computational experiments it is found that the neighborhood-based robustness measures improve schedule robustness for all performance measures tried in the study most of the time. Schedule flexibility seems to be improved for maximum tardiness problems and loose summed tardiness problems, however it does not seem to be improved for tight summed tardiness problems and total flow-time problems.

Tavares et. al. (1997), states that the uncertainty of project networks has been mainly considered as the randomness of duration of the activities. However, another major problem for project managers is the uncertainty due to the randomness of the amount of resources required by each activity which can be expressed by the randomness of its cost. Such randomness can seriously affect the discounted cost of the project and it may be strongly correlated with the duration of the activity. A model considering the randomness of both the cost and the duration of each activity is introduced and the problem of project scheduling is studied in terms of the project's discounted cost and of the risk of not meeting its completion time. The adoption of the earliest (latest) starting time for each activity decreases (increases) the risk of delays but increases (decreases) the discounted cost of the project. Therefore, an optimal compromise has to be achieved. This problem of optimization is studied in terms of the probability of the duration and of the discounted cost of the project fal1ing outside the acceptable domain (Risk function) using the concept of float factor as major decision variable.

Sabuncuoğlu and Bayiz (2000) study the reactive scheduling problems in a stochastic manufacturing environment. They measured the effect of system size and type of work allocation (uniform and bottleneck) on the performance of scheduling methods (off-line and on-line). Their performance criteria are mean tardiness and makespan. The authors also investigated a partial scheduling scheme under both deterministic and stochastic environments for several system configurations.

It is seen that the system performance is more affected by the system load and not significantly affected from the system size. Distribution of the load in the system seems to have significant impact on the performance of the scheduling methods e.g. the optimization based off-line scheduling method performs better than on-line dispatching rules when the loads across the machines are not uniform. It is observed that when there are stochastic disturbances such as machine breakdowns, the off-line scheduling method is affected more than the on-line dispatching mechanisms.

Tsai and Gemmill (1998) proposed Tabu Search, which is a heuristic procedure for solving optimization problems which decreases the likelihood of becoming trapped at a local optimum solution, to obtain solutions to resource-constrained, randomized activity duration project scheduling problems. Tabu search employs a set of moves that transform one solution state into another and provides an evaluation function for measuring the attractiveness of these moves. The form of the tabu search procedure is highly flexible and often motivates the creation of new types of moves and evaluation function criteria for different problems. They used multiple tabu lists, randomized short-term memory, and multiple starting schedules for obtaining search diversification. After the experiments, it is seen that tabu search is an efficient way to find good solutions to both the deterministic and stochastic problems.

Verhoven (1998) proposed a tabu search algorithm for a resource-constrained scheduling problem (RCSP) that generalizes the job shop scheduling problem. In this scheduling problem machines are capable of processing several operations simultaneously, and operations may require multiple resources for processing with different processing times on each resource. The neighborhood used in this algorithm is based on reinserting operations on possibly different resource sets such that feasibility is maintained. The algorithm is tested on two types of instances, multiple capacitated job shop instances and job shop instances with resource sets alternatives, which are derived from well-known job shop instances and gave good-quality results.

Artigues et.al. (2003) studied a model for the static resource-constrained project scheduling problem. They proposed static and dynamic scheduling methods, based on a new polynomial insertion algorithm taking advantage on the flow structure. They also used the polynomial heuristic called INSHEUR included in Tabu Search Method. The computational experiments show that the algorithm is of great interest for robust rescheduling in a dynamic environment. The use of this algorithm has been shown to be efficient, especially for highly cumulative instances compared to single pass priority-rule based heuristics and for the largest instances of the literature compared to recent sampling methods. The authors also believe that the tabu search method could be highly improved by analyzing the connectivity of its neighborhood and by adapting to the RCPSP the concept of blocks of critical activities that have been shown to be useful for the generalized job-shop problem.

Al-Fawzan and Haouari (2004), addressed the issue of designing a project schedule which is not only short in time, but also less vulnerable to disruptions due to reworks and other undesirable conditions. To that aim they introduced the concept of schedule robustness and they developed a bi-objective resource-constrained project scheduling model. They considered the objectives of robustness maximization along with makespan minimization. They developed a tabu search algorithm in order to generate an approximate set of efficient solutions. The robustness of the schedule is defined as the total sum of the free slacks. The objective of their model which was called as Bi-objective Resource Constraint Project Scheduling Problem (BRCPSP) was to maximize the robustness and minimize the makespan. A multi-objective tabu search algorithm has been devised for solving the bi-objective resource constrained project scheduling problem. Several variants of the algorithm were compared using statistical design of experiments. The best variant was thoroughly tested on a large sample of 480 benchmark problems and it proved to be robust and to generate in most cases a moderately sized set of efficient solutions. Moreover, its performance on the single objective makespan minimization is comparable to special purpose tabu search algorithms.

Herroelen and Leus (2002), proposes a model for resource allocation for projects with variable activity durations. The allocation is required to be compatible with a deterministic pre-schedule and the objective is to guarantee stability in activity starting times compared with the pre-schedule. Constraint propagation is applied during the search to accelerate the algorithm. They concentrated on the case of a single resource type, since this environment maps into a single resource flow network.

Gören (2002) defined several robustness and stability measures and policies. He also developed two new surrogate measures since the exact measures are difficult to calculate. He used these measures in a tabu search algorithm to generate robust and stable schedules for a single machine subject to random machine breakdowns.

Herroelen and Leus (2004) examined various procedures for the development of a stable pre-schedule, which is unlikely to undergo major changes when it needs to be repaired as a reaction to activity duration disruptions. They developed a mathematical programming model to minimize the expected weighted deviation in activity start times. They assumed that activities do not start before their pre-schedule starting time and the model anticipates a single activity duration to be disrupted and works with discrete disruption scenarios.

CHAPTER 3

PROBLEM DEFINITION AND RESEARCH METHODOLOGY

In project scheduling, project activities are subject to considerable uncertainty, which results in various schedule disruptions during project execution. As a result, several researches were done on the subject of the random nature of activity durations. A great quantity of the studies can be classified in two groups. "Proactive baseline schedules" which are the schedules which are stable or robust that will change little or not when uncertain events occur and, "reactive scheduling" which revises or re-optimizes the baseline schedule when unexpected events occur.

A pre-schedule is robust if it is built with protection against the variabilities during its execution. The term stability refers to the situation where there is little deviation between the pre-schedule and the executed schedule. Reactive scheduling can be done by schedule repairing, rescheduling and contingent scheduling.

Gören (2002) defined some robustness and stability measures; by defining S_1 through S_m as **m** feasible schedule alternatives and f(S) as the value of the performance measure for schedule S. If there are **q** scenarios corresponding to possible disruptions in the course of the project, when the j^{th} scenario occurs with probability a_j, schedule S_j would change to S_{ij} under j^{th} scenario, i.e. S_{ij} is the realized schedule of the initial schedule S_i, if the disruption in scenario j occurs. Let S_j^* be the optimal solution with the used performance measure. The robustness measures that can be used are:

- *Robustness Policies and Measures:*

1. Schedule S_i that minimizes the expected realized performance, $E[f(S_i^r)]$, is selected where $E[f(S_i^r)] = \sum_{j=0}^{q} a_j f(S_{ij})$, for i=1,2,…,m. This is a risk neutral approach which selects the schedule whose performance measure is the best on the average. (Wu et al., 1999)

2. Select S_i that minimizes the worst-case scenario performance. That is select S_i such that $\max_j\{f(S_{ij})\}$ is minimized over all i (i.e. $\min_i\left\{\max_j\{f(S_{ij})\}\right\}$). This policy is a risk-averse approach that selects the schedule whose worst-case performance measure is better than all others' are.

3. If the worst case scenario can be identified, say W_k, k Є{1,2… q}, selecting the schedule that performs the best in W_k can be an appropriate policy. Thus, this policy selects S_i with $\min_i\{f(S_{ik})\}$, where W_k is the worst case scenario.

4. Similarly, if the most probable scenario can be identified, that is the scenario W_m where $m = \arg\max_i\{a_i\}$, for i = 1, 2, . . . q, then selecting the schedule whose performance measure is the best under W_m can be an appropriate policy. This policy selects S_i with $\min_i\{f(S_{im})\}$ where W_m is the most probable scenario.

5. Another policy is to select the schedule such that the expected deviation of the realized schedule's performance from the initial anticipated performance is minimized. That is, select S_i with minimum $\sigma_i = \left|f(S_{i0}) - E\left[f\left(S_i^r\right)\right]\right|$, for i = 1,2, . . . m. This policy emphasizes the definition "robust schedule is the schedule whose performance does not degrade much in the face of disruptions".

6. Another policy is to select the schedule that minimizes a measure that is a convex combination of the aforementioned measures. For example, selecting the schedule which minimizes$\left(r \text{ x } E\left[f\left(S_i^r\right)\right] + (1-r) \text{ x } \sigma_i\right)$, where r is a real number between 0 and 1. The first part of the above measure emphasizes that a robust

schedule should perform well in the face of disruptions. The second part emphasizes that a robust schedule's performance should not degrade much in the face of disruptions. By varying r between 0 and 1, different weights can be given to these two points of view of robustness (Leon et al., 1994) .

Similarly the stability measures are defined by Gören (2002) as:

- *<u>Stability Policies and Measures:</u>*

Completion time differences are used to account the impact of schedule change (i. e stability). Let C_k^S be the completion time of job k under schedule S. Let N be the number of jobs. Construct the completion time differences matrix DM = [dm_{ij}] where $dm_{ij} = \sum_{k=1}^{N} \left| C_k^{S_{ij}} - C_k^{S_{i0}} \right|$.

1. Select schedule S_i that minimizes the expectation of differences, E[dm_i], where $E[dm_i] = \sum_{j=1}^{N} a_j dm_{ij}$. This method selects a schedule whose stability is the best on the average. This is a risk neutral approach. (Mehta and Uzsoy(1998), Mehta and Uzsoy(1999), O'Donovan, Uzsoy, and McKay(1999)).

2. Select S_i that minimizes the worst case scenario completion time differences, that is select S_i such that $\max_j \{dm_{ij}\}$ is minimized over all i. This policy selects a schedule whose worst case stability is greater than all others. This is a risk-averse approach.

3. If the worst-case scenario can be identified, say W_k, k ε {1,2… n}, selecting the scenario whose stability is the best in W_k can be an appropriate policy. Thus, this policy selects S_i with $\min_i \{dm_{ik}\}$, where W_k is the worst case scenario.

4. If the most probable scenario may be clearly identified, that is the scenario W_m where $m = \arg\max_i \{a_i\}$, for i = 1, 2, . . . n, then selecting the scenario whose

stability is the best under W_m can be an appropriate policy. Thus, this policy selects S_i with, $\min_i \{dm_{im}\}$ where W_m is the most probable scenario.

In defining stable project schedules, Herroelen and Leus (2004) considered a case where exactly one anticipated disturbance will occur in the network, due to an increase in the duration of a single activity. They develop a mathematical programming model that aims at minimizing the expected weighted deviation of the actual from the planned activity start times when exactly one activity duration disruption is anticipated.

3.1. Problem Definition and Methodology

If it is considered to design a stable pre-schedule, it is important to obtain an initial schedule which the realized schedule would deviate minimally from this planned schedule. So it can be said that the stability measure is related to the realized schedule, therefore it cannot be known at the beginning. So, the realized schedule can only be predicted and by using surrogate measures its performance can be estimated. Average or Total Free Slack method is used in most of the studies in literature to generate stable schedules.

Total Free Slack measure is a measure of robustness used by Al-Fawzan and Haouari (2004). The free slack si is defined as the amount of time that an activity i can slip without delaying the start of the very next activity while maintaining resource feasibility. Thus the robustness of the schedule is expressed as total sum of free slacks and this value is tried to be maximized. TFS (Total Free Slack) = $\sum_{i=0}^{n} s_i$ (where n is the number of activities in schedule).

In this study the problem that is analyzed may be defined as follows: a project consists of a set **N** of **n** interrelated activities. There are precedence constraints which specify that activity i cannot be performed until all its predecessors defined by the set $\mathbf{p_i}$ have been finished. There are **K** renewable resources. A constant amount of $\mathbf{CAP_k}$ units of resource k (k=1,…,K) is continuously available from time zero onwards. The duration of activity i is $\mathbf{d_i}$ units of time. During this time period a constant amount of $\mathbf{r_{ik}}$ units of resource k (k=1,…,K) is occupied. Preemption is not allowed. The objective is to determine starting times for the activities in such a way that a performance criterion is optimized, the precedence constraints are satisfied, and at each unit of time the total resource requirement does not exceed the resource availability for each resource type. Also a deadline **DL** exists which the schedules are not allowed to exceed.

The main assumptions for the problem are;

1. Activity durations are known, deterministic and integer.
2. All the predecessors of the activity must be completed before the activity can be started.
3. Once an activity started, it must be completed within the specified duration. No preemption is allowed.
4. There may be one or more types of resources available and each type of resource has fixed amount of availability during the project duration.
5. The objective is to minimize (or maximize) the stability measure selected subject to the activity relationships and resource constraints.

Under these assumptions, the mathematical formulation of the problem with the objective of stability measure minimization (or maximization) can be given as:

Minimize or Maximize (Measure of Stability)

Subject to

1. $0 \leq X_{it} \leq 1$ (and by integer programming techniques, x is constrained to equal either 0 or 1)

2. Activities will be performed

$$\sum_{t=1}^{T} X_{it} = d_i \qquad i = 1, \ldots, n$$

3. Resource capacities will not be exceeded

$$\sum_{i=1}^{n} X_{it} \times r_{ik} \leq CAP_k \qquad k = 1,2,\ldots, K$$

$$t = 0,1,\ldots, T$$

4. No activity will be started before its predecessors are completed

$$d_i X_{jt} \leq \sum_{l=1}^{t-1} X_{jl} \qquad \text{for all } i \in p_j$$

$$t = 0,1,\ldots, T$$

$$j = 1, \ldots, n$$

5. No activity will be split:

$$d_i X_{it} - d_j X_{j(t+1)} + \sum_{l=t+2}^{T} X_{jl} \leq d_j \qquad j =, 1, \ldots, n$$

$$t = 0,1,\ldots, T$$

6. Deadline should not be passed:

$$\sum_{i=1}^{n} X_{it} \leq 0 \qquad t = DL, \ldots, T$$

Where

X_{it} : activity of job i on time t, constrained to the integer values 1 (if job i is active) or 0 (if activity i is inactive)

In our problem we have possible disruptions during the schedule generation process. There is a time window (which may be related to an event like weather conditions or contract negotiation times etc.) that the activities are under risk of disruption with a certain probability. If the event occurs (with probability α) then may be the activities extended at a certain percentage of their parts remained in the window. When the event occurs, an activity may be delayed (with probability β) or not delayed (with probability 1-β).

3.2. Stability Measures

Three different stability measures are considered in this study. These are total free slacks (TFS), expected total delay in the start time of activities (ETD), and total delay in the start time of activities (TD). The details of the measures are explained below;

a) Total Free Slacks (TFS);

   This is the measure used by, Al-Fawzan and Haouari (2004). The free slack s_i is defined as the amount of time that an activity i can slip without delaying the start of the very next activity while maintaining resource feasibility.

   TFS (total free slack) = $\sum_{i=0}^{n} s_i$

b) Expected Total Delay in start times of activities (ETD);

The stability measure selected is the total expected delays in the start times of the activities after disruption from their original starting times.

$$TD_k = \sum_{i=0}^{n} \left(STd_i - ST_i \right)$$

E(Total delay in start time of activities) :

$$ETD = \alpha \times \sum_{k=0}^{SS} P_k \times TD_k$$

Where;

TD_k = Total delay in start times in scenario k of disruption scenarios.

ST_i = Start time of activity i in normal (undisrupted) schedule.

STd_i = start time of activity i in disrupted schedule

P_k = Probability of occurrence of scenario k.

α = Probability of disruption event to occur.

β = Probability of disruption of an activity in disruption time window.

SS = Number of disruption scenarios

An example for computing P_k.

Let's say **β** chosen as 0,2 and there are 3 activities named a, b, c in the time window in the selected schedule;

Then the possible disruption scenarios will be:

_ no activities will be disrupted,

_ only act a disrupted

_ only act b disrupted

_ only act c disrupted

_ a and b disrupted

_ a and c disrupted

_ b and c disrupted

_ all will be disrupted,

The probabilities of these scenarios to occur are calculated as;

p_{none} = (0.8*0.8*0.8)	= 0.512
p_a = (0.2*0.8*0.8)	= 0.128
p_b = (0.8*0.2*0.8)	= 0.128
p_c = (0.8*0.8*0.2)	= 0.128
$p_{a,b}$ = (0.2*0.2*0.8)	= 0.032
$p_{a,c}$ = (0.2*0.8*0.2)	= 0.032
$p_{b,c}$ = (0.8*0.2*0.2)	= 0.032
p_{all} = (0.2*0.2*0.2)	= 0.008
	1

c) Total Delay in the Start time of the Activities (TD);

This measure can be defined as

$$TD = \sum_{i=0}^{n} \left(STd_i - ST_i \right)$$

In dealing with these stability measures; four policies are also tested:

1. Find a schedule that maximizes TFS

   TFS (total free slack) = $\sum_{i=0}^{n} s_i$

2. Find a schedule that minimizes ETD

   ETD = $(\boldsymbol{\alpha} * \sum_{k=0}^{SS} \mathrm{P_k} \text{ x } \mathrm{TD_k})$

3. Find a schedule that minimizes TD

$$\mathrm{TD} = \sum_{i=0}^{n} \left(STd_i - ST_i \right)$$

4. Another policy that is tested is the right shift of the activities in the time window (RS). It is assumed that no work will be done in the disruption time window when disruption occurs and all the activities corresponding to the disruption window are right shifted in case of disruption.

3.3. A Numerical Example

As an example, a project consists of a set N of 15 interrelated activities. There are precedence constraints which are shown in Figure 3.1. There is one renewable resource with a constant amount of capacity 7 units that is continuously available during the implementation of project. The durations of activities and amount of resource occupied by activities are given in Table 3.1. (Note that activities 0 and 16 are dummy activities).

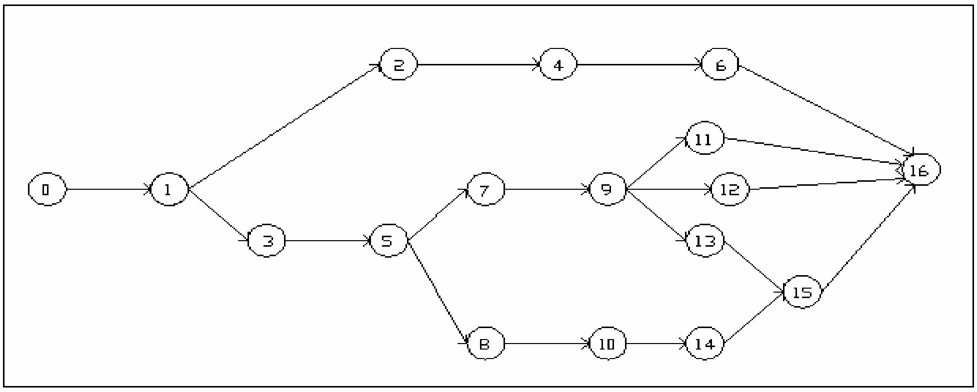

Figure 3.1. Precedence relations of activities

Table 3.1. Durations and resource requirements of activities

Activity	Duration (Weeks)	Resource Requirement
0	0	0
1	8	2
2	8	3
3	4	4
4	14	4
5	4	3
6	12	2
7	4	3
8	12	2
9	8	3
10	13	4
11	5	3
12	4	3
13	4	3
14	4	2
15	3	3
16	0	0

There is a deadline for the completion of the project (chosen as 65 weeks in this case) and the schedules completed after deadline are not considered as feasible.

There is a probability of disruption occurrence in the time window (chosen as between weeks 40-45 in this case) which is α (0.3 in this case) and for the activities in the window all disruption probabilities are considered as having the same probability to occur β (0.2 in this case). The disruption lengths of the activities are taken as a certain percentage (35% in this case) of their parts remained in the window (The values are rounded to nearest integer value to simplify the calculations).

The initial schedule found is given in Figure 3.2.

Number of activities in window = 3

Activities in window are: 6, 10, 11,

In case of disruption new duration of activities will be:

Act_6 = 13 Act_10 = 15 Act_11 = 6

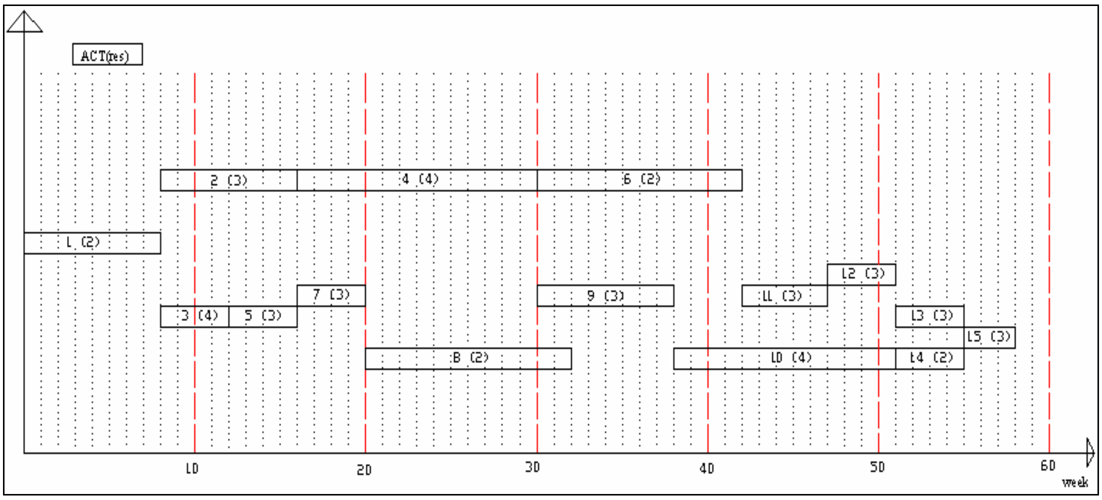

Figure 3.2. Gannt Chart of the Starting schedule

For this schedule;

a) Total free slack of starting schedule is = 6

S_8 =6

b) Expected total deviation in starting schedule = 0.852

For each scenario obtained values of start times and deviations in start times are given in Table A.1 in Appendix A.

So, the expected total deviations in the start times for the starting schedule is computed as,

$$ETD = \boldsymbol{\alpha} * \sum_{k=0}^{SS} (P_k \text{ x } TD_k)$$

c) TD = total delays in start times of activities in starting schedule at average conditions = 11

- When we use the objective of maximizing the TFS of this problem the schedule shown in Figure 3.3. is obtained.

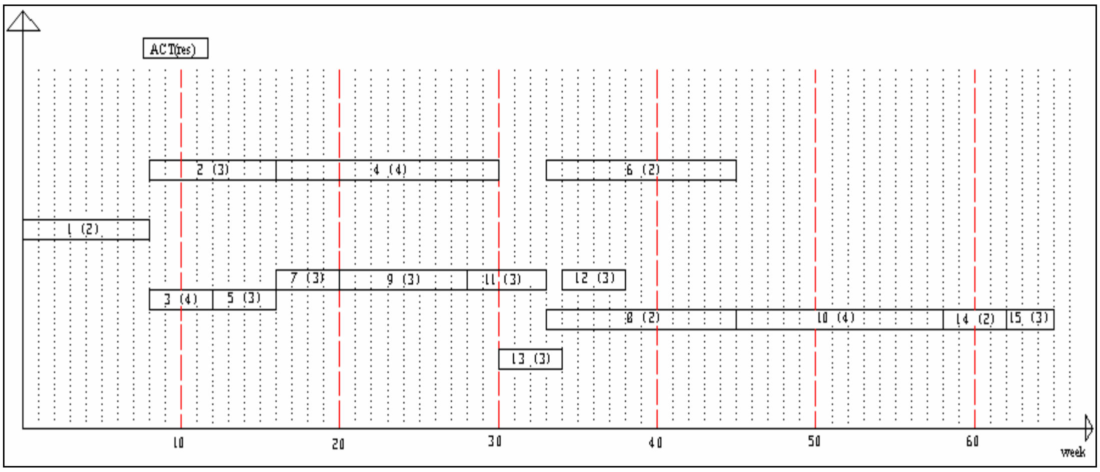

Figure 3.3. Gannt Chart of the schedule found by TFS Policy

Three objective function values are also calculated for this schedule as;

TFS = 47

ETD = 0.48

TD = 8

For each scenario, values in Table A.2. in Appendix A are obtained for start times and deviations in start times.

- When we use minimizing the ETD objective for this problem the schedule shown in Figure 3.4. is obtained.

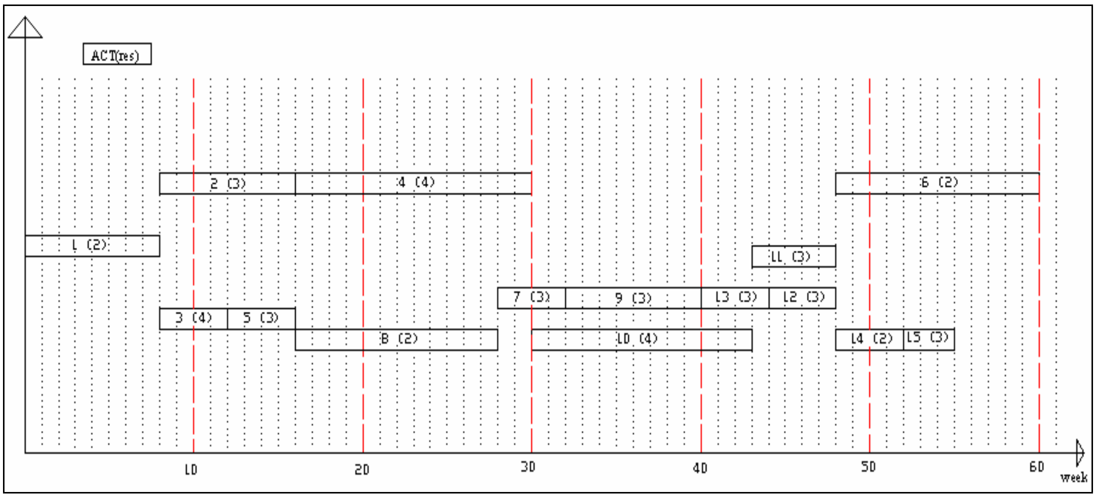

Figure 3.4. Gannt Chart of the schedule found by ETD Policy

There are 4 activities (10, 11, 12 and 13) in disruption window and therefore there are 2^4 = 16 different possible scenarios of disruption and at the moment they have equal probabilities to occur.

For each scenario values of start times and deviations in start times are obtained is shown in Table A.3. (in Appendix A) .

Three objective function values are also calculated for this schedule as;

TFS = 13

ETD = 0.21

TD = 6

- When we use minimizing the TD objective for this problem the schedule shown in Figure 3.5. is obtained.

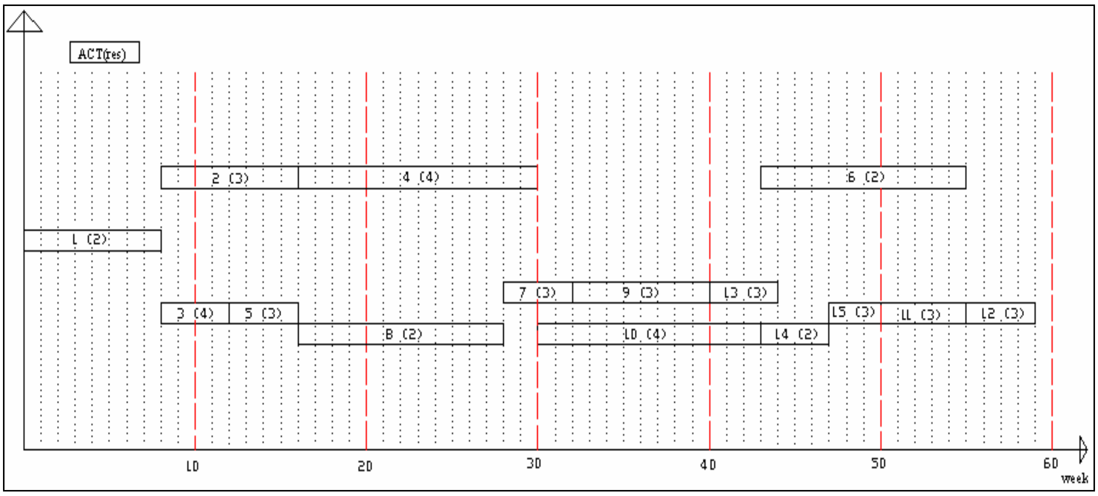

Figure 3.5. Gannt Chart of the schedule found by TD Policy

There are 4 activities (6, 10, 13 and 14) in disruption window as in the above schedule. For each scenario values of start times and deviations in start times are obtained are given in Table A.4. (in Appendix A) .

Three objective function values are also calculated for this schedule as;

TFS = 11

ETD = 0.24

TD = 4

- When we use minimizing the RS objective for this problem the schedule shown in Figure 3.6. is obtained.

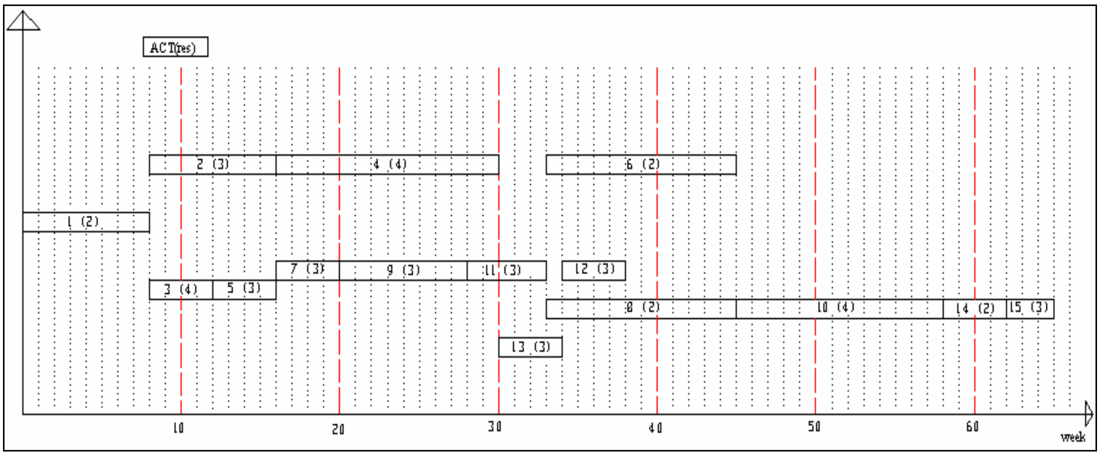

Figure 3.6. Gannt Chart of the schedule found by RS Policy

Three objective function values are also calculated for this schedule as;

TFS = 15

ETD = 0.27

TD = 7

The stability measures for the best schedules of all four policies are given in the below table.

Table 3.2. Stability measure values for best schedules

RESULTS		OBJECTIVE FUNCTION VALUES		
		TFS value	**ETD Value**	**TD Value**
POLICIES	**TFS**	47	0.48	8
	ETD	13	0.21	6
	TD	11	0.24	4
	RS	15	0.27	7

After discussing the stability measures and policies, we decided to continue with;

_ Find a schedule that minimizes Σ(ETD); to reflect the average conditions and compare this policy with widely used policies TFS and RS under the average conditions (objective ETD) and worst case conditions (objective TD).

Therefore the model to be solved will be:

(P1) **Min (α *** $\sum_{k=0}^{SS} P_k \text{ x } TD_k$ **)**

S. T.

$$0 \leq X_{it} \leq 1$$

$$\sum_{t=1}^{T} X_{it} = d_i \qquad i =, 1, \ldots, n$$

$$\sum_{i=1}^{n} X_{it} \text{ x } r_{ik} \leq CAP_k \qquad k = 1,2,\ldots, K \qquad t = 0,1,\ldots, T$$

$$d_i X_{jt} \leq \sum_{l=1}^{t-1} X_{jl} \qquad \text{for all } i \in p_j \qquad t = 0,1,\ldots, T \quad j =, 1, \ldots, n$$

$$d_i X_{it} - d_j X_{j(t+1)} + \sum_{l=t+2}^{T} X_{jl} \leq d_j \qquad j =, 1, \ldots, n \quad t = 0,1,\ldots, T$$

3.4. Proposed Methods

A number of exact methods have been proposed for scheduling a resource constrained project, however, they become computationally impractical for problem of a large size. Since obtaining the solution is restricted to small size projects, it is gained more importance to develop heuristic procedures to obtain near optimum solutions especially for the large size realistic projects. In this study we use a Tabu Search methodology to solve P1.

The algorithm starts with an initial schedule which is generated as earliest start schedule and use tabu search to find new feasible schedules. At each iteration, the performances of several alternative schedules are compared and the best is selected as current solution. The details of the tabu search method and the algorithm used are given in following sections.

3.5. Introduction of Tabu Search Algorithm

Tabu Search method was developed by Glover (1997), it has a wide application area that includes scheduling, production, inventory and investment, routing, layout design and location. Tabu Search (TS) is based on the premise that problem solving, in order to qualify as intelligent, must incorporate adaptive memory and responsive exploration (Glover and Laguna, 1997).

The Tabu Search is a metaheuristic that can be used to solve combinatorial optimization problems. It is different from the well-known hill-climbing local search techniques in the sense that it does not become trapped in local optimal solutions, i.e. the tabu search allows moves out of a current solution that makes the objective function worse in the hope that it eventually will achieve a better solution. New regions of a problems solution space are investigated with the goal of avoiding local minimum and finding the global minimum

In Tabu Search, tabu status of forbidden elements shift according to time and circumstance based on an evolving memory. TS uses adaptive (flexible) memory since it can be overruled for a preferable alternative. TS also uses responsive exploration, i.e. exploitation of good solutions and exploration of new promising regions.

The tabu search requires the following basic elements to be defined:

- *Configuration* is a solution or an assignment of values to variables.
- *A move* characterizes the process of generating a feasible solution to the combinatorial problem that is related to the current solution (i.e. a move is a procedure by which a new (trial) solution is generated from the current one).
- *Set of candidate moves* is the set of all possible moves out of a current configuration. If this set is too large, one could operate with a subset of this set.
- *Tabu restrictions*: These are certain conditions imposed on moves which make some of them forbidden. These forbidden moves are known as tabu. It is done by forming a list of a certain size that records these forbidden moves. This is called the *tabu list*.
- *Aspiration criteria*: These are rules that override tabu restrictions, i.e. if a certain move is forbidden by tabu restriction, then the aspiration criterion, when satisfied, can make this move allowable.

Given the above basic elements, the tabu search scheme can be described as follows: start with a certain (current) configuration, evaluate the criterion function for that configuration. Then, follow a certain set of candidate moves. If the best of these moves is not tabu or if the best is tabu, but satisfies the aspiration criterion, then pick that move and consider it to be the new current configuration; otherwise, pick the best move that is not tabu and consider it to be the new current configuration. Repeat the procedure for a certain number of iterations. On termination, the best solution obtained so far is the solution obtained by the algorithm. Note that the move that is picked at a certain iteration is put in the tabu list so that it is not allowed to be reversed in the next iterations. The tabu list has a certain size, and when the length of the tabu reaches that size and a new move enters that list, then the first move on the tabu list is freed from being tabu and the process continues. The aspiration criterion could reflect the value of the criterion (objective) function, i.e. if the tabu move results in a value of the criterion function that is better than the best known so far, then the aspiration criterion is satisfied and the tabu restriction is overridden by this (Al-Sultan, 1995).

In Tabu search the important parameters are the tabu list size TL, Itermax , the number of iterations before stopping, the neighborhood size NS; and the aspiration criteria.

- In this study we use swap moves to find a schedule in the neighborhood of the current schedule. So, the neighborhood of a schedule with n jobs consists of $\frac{n(n-1)}{2}$ schedules. Only precedence feasible moves are considered.

- Since a move consists of selecting a feasible pair, then the last performed moves are stored in the tabu list. In order to override the tabu list when there is a good move, an aspiration criterion is implemented: a tabu move is accepted if it produces a solution which outperforms the best solution obtained so far. Other aspiration criteria are possible, e.g. setting the tabu list size shorter for better solutions.

3.6. Implementation of Tabu Search Algorithm:

Schedule representation:

A schedule Φ is represented as an ordered list $L = \{o_1, o_2, \ldots, o_n\}$ of the n activities. For the sake of simplicity, only precedence feasible lists are considered. Given a precedence feasible list L; a corresponding precedence and resource feasible schedule can be built by using a Schedule Generation Scheme (SGS). This algorithm starts from scratch and constructs in n steps a feasible schedule, so that at step i (i = 1,…,n) activity o_i is selected and scheduled at the earliest precedence and resource feasible start time. For that purpose, the following Forward Recursion Procedure is used.

Notation

U : set of unscheduled activities

{0} : dummy activity with zero processing time, no resource requirement and precedes all activities.

K : number of resources

T : number of time periods

R_{kt} : amount of resource k used at time t

p_{ij} : set of immediate predecessors of activity i

pp_{ij} : set of predecessors of activity i

ST_i : earliest start time of activity i

C_i^e : earliest completion time of activity i

EC : { C_i^e : j$\notin U$ }

d_i : duration of activity i

X_{it} : 1 if t Є [S_i, C_i^e)

0 otherwise

r_{ik} : resource k requirement of activity i

$[C_q]_{q=0,ACT}$: list of the elements of EC ranked in non-decreasing order

CAP_k : constant resource capacity

The SGS forward recursion algorithm for building schedules is given in part Appendix B. This algorithm provides the earliest start and completion times for the given list. The makespan of the project is C_n^e (Completion time of last activity). In order to calculate the free slack times of activities, we need to compute the latest feasible completion times C_j^1 (j = 1,..., n). Hence, the free slack of an activity j is

$s_j = C_j^1 - C_j^e$ (j = 1,..., n).

While latest completion times are computed for each activity j (j=1,…, n) the following properties hold:

- During the time interval [ST_j, C_j^1] the K resources are continuously available with an amount ≥ r_{ik}
- It is possible to complete activity j at time C_j^1 without delaying the start of the very next activity (i.e. each successor could start at its earliest start time that was calculated by the forward recursion).

In order to compute the latest finish times, the earliest completion times that were calculated with the forward procedure are considered and an ordered list of the n activities L = {u_1, u_2, …, u_n} are constructed where activities are ranked in a non-decreasing order of their earliest completion time. A backward procedure is used so that at step i (i = 1,..., n) activity u_i is included in the partial schedule and scheduled at the latest precedence and resource feasible completion time while satisfying the properties above. In addition to the previous notation below notations are added for backward recursion procedure:

ST_j^1 : latest start time of activity j

$Succ_j$: set of immediate successors of activity j

{n+1}: dummy activity with zero processing time, no resource consumption, which is preceded by all the activities

ES : {$ST_j \notin U$}

$[ST_p]_{p=1,ACT+1}$: list of the elements of ES ranked in non-increasing order

The SGS backward recursion algorithm for computing the latest finish times of activities is given in part Appendix B.

In order to compute the ETD and TD measure values the activities corresponding to the disruption time window and the durations of these activities in case of disruption must be calculated. After that the start times of activities in case of each disruption scenario could be computed.

Neighborhood structure:

The same neighborhood structure as in Al-Fawzan and Haouari is applied. At a given iteration, let Φ_n be the current feasible schedule with a corresponding precedence feasible priority list $L = \{o_1, o_2, \ldots, o_n\}$. N neighbors generated by selecting feasible moves. A move requires two activities to be swapped in the list L. Only precedence feasible moves are considered. Obviously, considering only lists which are compatible with precedence constraints reduce the computational burden, since in the Forward Recursion Procedure the activities are simply scheduled according to the ordering that is specified by the corresponding list.

Selection Strategy:

A precedence- feasible list of the n activities L_0 is built and used to generate an initial schedule. Assume that Φ_n is the current feasible schedule. For each neighbor schedule compute the stability (objective function) and makespan. If the stability value of the selected neighbor is smaller than all other neighbor selections this neighbor is selected as the best neighbor in this iteration. If two or more neighbors have the same stability value then the makespan value is considered and the neighbor with the smallest makespan is selected.

Tabu list attributes and aspiration criterion:

Since a move consists of selecting a feasible pair; then in the tabu list the last performed moves are stored. In order to override the tabu list when there is a good move, the following aspiration criterion is implemented: a tabu move is accepted if it produces a solution which outperforms the best solution obtained so far.

Tabu list size:

The tabu list embodies one of the functions of the short term memory, which enables the algorithm to have some memory of the history. In this case, it tells the algorithm not to reverse the last TL moves, where TL is the size of the tabu list.

Synthesis of the TS algorithm

Initialization

Construct a precedence-feasible list L and generate an initial schedule Φ_0

TL = $\varnothing$

For v =0 to $Iter_{max}$ do

Repeat

Select N moves (non-tabu moves and/or satisfying the aspiration criterion)

Construct the set of neighbors NS (Φ_0) = { Φ^u : u = 1, …, N}

Compute for each Φ^u obj^u (measure value) & ms^u (makespan)

Select Φ^* such that

$Min_{u=1,\dots,N}\ obj^u$

If there is equality $Min_{u=\text{ equal neighbours}}\ ms^u$

$\Phi_0 = \Phi^*$

Update the tabu list TL

CHAPTER 4

EXPERIMENTAL DESIGN

We will compare the ETD policy with the two widely used policies TFS and RS under two conditions:

_ Average conditions and

_ Worst case conditions.

For this reason, we conducted an experimental design that determines the performances on different policies on different problem settings. Experimental studies were conducted on networks generated by RanGen, which is a problem generator developed by Demeulemeester et al. (2003) with the measures and algorithms explained in above chapters.

RanGen is a random network generator for activity on the node (AoN) networks for different classes of project scheduling problems. The objective of generator is to construct random networks with prespecified parameters in order to control the relation between the hardness of a problem instance and the logic of the underlying network. Both parameters which are related to the network topology and resource-related parameters are implemented. The network generator meets the shortcomings of former network generators since the generator employs a wide range of different parameters which have clearly demonstrated in former studies to have a relation with the hardness of different project scheduling problems.

4.1. Factor and Factor Levels

An experimental design is established in order to observe and identify corresponding changes in the output response after making changes to the input variables of a process. The independent variables (quantitative and qualitative) that are related to the response variable are called factors. The effect of a factor is defined as the change in response produced by a change in the level of the factor, which is the intensity setting of a factor (i.e. the value assumed by a factor in an experiment). A treatment is a particular combination of levels of the factors involved in an experiment.

The number of activities and number of resources are estimated to affect the objection function value and in order to see these effects the number of activities and the number of resources are defined as factors of the design. Both of them are examined in two levels such as small and large.

The OS (Order Strength) value is defined as the number of precedence relations (including the transitive ones but not including the arcs connecting the dummy start or end activity) divided by the theoretical maximum number of precedence relations and it is also referred as *restrictiveness*. Restrictiveness, which is a measure of network complexity for the resource constrained project scheduling problem is therefore estimated to affect the objection function value and defined as a factor of the design. It is examined in two levels.

Resource factor reflects the average proportion of resource types requested per activity and concequently measures the density of the resource matrix. The *resource strength* is used for determining the total avaibility of renawable resource for each resource type. Therefore, *resource factor* and *resource strength* are taken as factors and both are examined in two levels.

It is obvious that the selected α and β values may have an effect on the output response. So, they are taken as factors too and they are examined in three levels.

Finally the policies considered for schedule selection are taken as a factor. All the factors and their levels are given at Table 4.1.

Table 4.1. Factors and factor levels of the problem

FACTOR	Test Levels		
	Level 1	Level 2	Level 3
Number of Activities	30	60	-
Number of Resources	1	4	-
Resource Factor	0.5	0.75	-
Resource Strength	0.5	0.7	-
Restrictiveness (OS)	0.5	0.65	-
α	0.2	0.4	0.8
β	0.2	0.4	0.8
Schedule selection policy (SSP)	TFS	RS	ETD

There are 2 x 2 x 2 x2 x 2 x 3 x 3 x 3 = 864 treatments in the design. In each case, by using the first 5 factors, five networks were generated by RanGen. So 864 x 5 =4320 problems are solved and analyzed in this experiment.

4.2. Response Variable

We are looking for the stability of the planned schedules for average conditions and worst case conditions. Therefore, our response variables for this experiment will be ETD and TD measure values for the schedules created by three policies explained in Chapter 3.

The relative performance of the policies for average and worst case conditions are obtained using the below formula:

$$\%\ \text{Deviation from the best OFV} = \frac{\text{Best OFV for the condition} - \text{OFV for this policy}}{\text{Best OFV for the condition}} \times 100$$

4.3. Problem Parameters

The time window that the activities are under risk of disruption is selected as the 1 10^{th} period of the starting schedule which is placed at the centre (which means the window starts at the time when 45% of the starting schedule elapsed and ends when 55% of the starting schedule elapsed). Also a deadline is put for the neighbor schedules which is %30 longer than the makespan of starting schedule. The amount of increase in the part of the activity that remained in disruption time window is selected as %35.

4.4. Fine Tuning Algorithm Parameters

The important parameters for TS algorithm used in this study are the Tabu List Size TL, the number of iterations before stopping $iter_{max}$ and the neighborhood size NS.

All the neighbors of the schedules are taken into consideration, so a neighborhood size is not defined. Some pilot runs were conducted to determine the levels for the other two parameters of the scheduling algorithm. For the first 5 problems generated by RanGen (30 act, 1 res, RF=0.5, RS=0.5, OS=0.5, α and β = 0.2) five levels of TL and 3 levels of $Iter_{max}$ are conducted for both measures. The results are given in Tables C.1 – C.5 in Appendix C.

As it is seen from the above tables the variation of the parameters TL and $iter_{max}$ does not play any significant role. Hence we fixed the Tabu List Size TL = 5, and the number of $Iter_{max}$ = 100.

4.5. Design of Experimental Model

When there are several factors of interest in an experiment, a factorial design should be used. By factorial experiment it is meant that in each complete trial or replicate of the experiment all possible combinations of the levels of the factors are investigated. The experiment that has every possible combination of multiple factors over each test level is called a Full Factorial Experiment

In our experiment, there are eight factors. Five of the factors (number of activities, number of resources, resource factor, resource strength and restrictiveness) have two levels and remaining three factors (probability of disruption event to occur, probability of disruption of an activity in time window and the schedule selection policy) have three levels, and each replicate contains 25 x 33 (864) possible combinations. The complete factorial experiment is the one that includes all possible combinations of the levels of the factors as treatments. For each combination 5 problems generated so, 864 x 5 = 4320 problem instances generated by RanGen using the control parameters.

Statistical analysis has two main steps, factorial design to consider the factor effects, and residual analysis to check the adequacy of the model.

The linear model for a factorial design includes the main effects (terms for each of the factors in the experiment) and terms for the interactions. In general, a regression model that is used to summarize the results of the experiment, is in an equation of the form

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + ... + \beta_k x_k + \varepsilon$$

where y is the response variable and the x's are a set of predictor variables, β values are the regression coefficients, and ε is the usual NID $(0,\sigma^2)$ random error term.

As explained above, in our model there are eight factors. The linear statistical model for our design is described as:

$$\begin{aligned} y_{\text{ijklmnop}} = {} & \mu + \beta_i + \delta_j + \gamma_k + \sigma_l + \rho_m + \omega_n + \lambda_o + \psi_p + \\ & (\beta\delta)_{ij} + (\beta\gamma)_{ik} + (\beta\sigma)_{il} + (\beta\rho)_{im} + (\beta\omega)_{in} + (\beta\lambda)_{io} + (\beta\psi)_{ip} + \\ & (\delta\gamma)_{jk} + (\delta\sigma)_{jl} + (\delta\rho)_{jm} + (\delta\omega)_{jn} + (\delta\lambda)_{jo} + (\delta\psi)_{jp} + \\ & (\gamma\sigma)_{kl} + (\gamma\rho)_{km} + (\gamma\omega)_{kn} + (\gamma\lambda)_{ko} + (\gamma\psi)_{kp} + \\ & (\sigma\rho)_{lm} + (\sigma\omega)_{\text{ln}} + (\sigma\lambda)_{lo} + (\sigma\psi)_{lp} + \\ & (\rho\omega)_{mn} + (\rho\lambda)_{mo} + (\rho\psi)_{mp} + \\ & (\omega\lambda)_{no} + (\omega\psi)_{np} + \\ & (\lambda\psi)_{pq} + \varepsilon_{\text{ijklmnopq}} \end{aligned}$$

where

μ : overall mean

β_i : effect of the number of activities factor i=1,2

δ_j : effect of the number of resources factor j=1,2

γ_k : effect of resource factor factor k=1,2

σ_l : effect of resource strength factor l=1,2

ρ_m : effect of restrictiveness factor m=1,2

ω_n : effect of probability of dis. event factor o=1,2,3

λ_o : effect of probability of dis. of an activity factor p=1,2,3

ψ_p : effect of schedule selection criteria factor q=1,2,3

and

$\varepsilon_{ijklmnopq}$: the usual NID $(0,\sigma^2)$ random error term.

4.6. Model Adequacy Checking

For testing the hypothesis, the model errors are assumed to be normally and independently distributed random variables with mean zero and variance σ^2. The variance σ^2 is assumed to be constant for all levels of factors. If these assumptions are valid, the Analysis of Variance procedure is an exact test of the hypothesis of no difference in the factor levels.

Generally these assumptions do not hold exactly. Therefore, it is not true to rely on the Analysis of Variance until validity of these assumptions has been checked. By examination of residuals, violation of basic assumptions can be easily investigated. The residuals for the model can be described as;

$$\text{Residual} = \text{response} - \text{fitted value}$$

If the model is adequate, the residuals should be structureless; which means that they should not contain obvious patterns. In the following sections model adequacy checking is shown for worst case and average condition values.

4.6.1. Worst Case conditions

Before adopting ANOVA to the problem, the adequacy of the model is checked considering the data for the worst case conditions that is objective TD (Obj_TD).

The normal probability plot of residuals and histograms of residuals are given in Figure 4.1. and Figure 4.2., respectively. Figure 4.3 shows the plot of residuals versus fitted values.

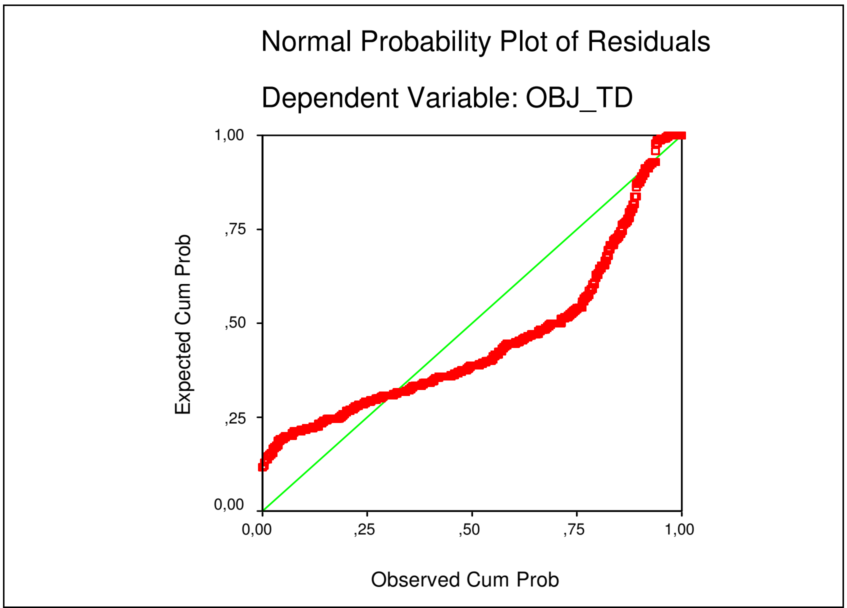

Figure 4.1. Normal Probability plot of Residuals for obj_TD

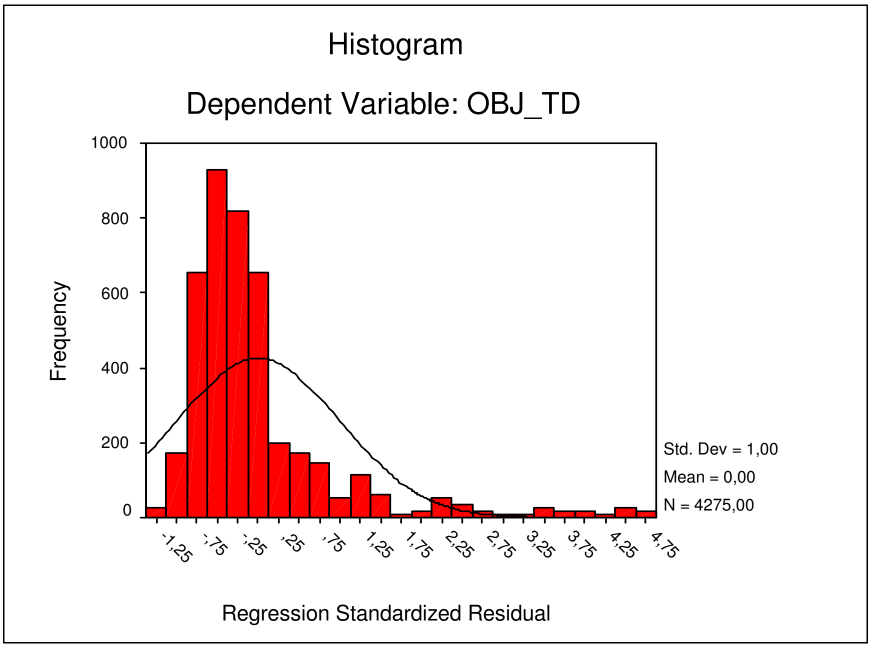

Figure 4.2. Histogram of Residuals for obj_TD

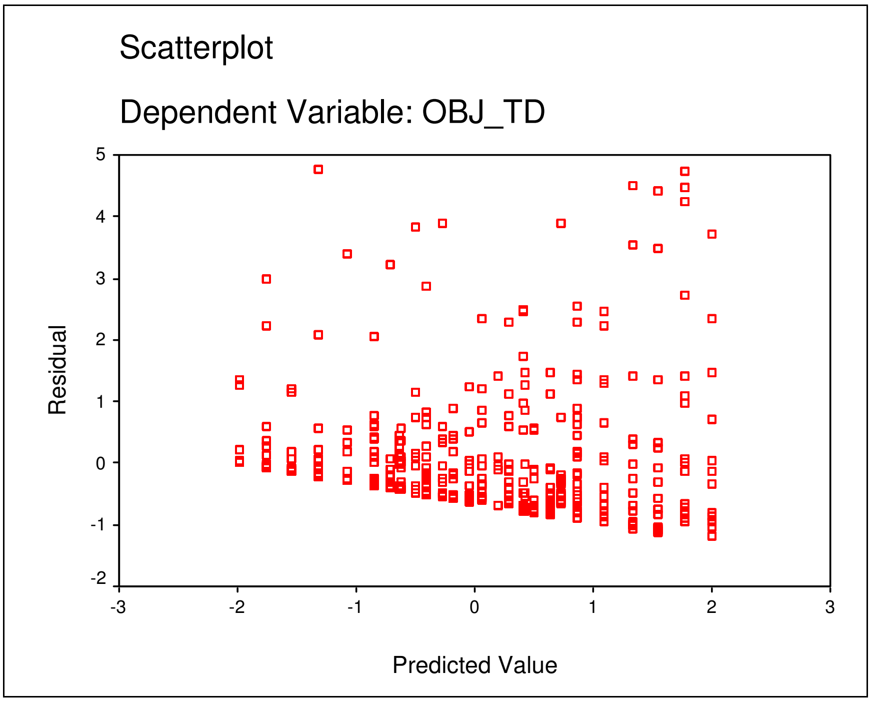

Figure 4.3. Residuals vs. Fitted values for obj_TD

From the figures it is seen that the residuals are not normally distributed. In order to check the constant variance assumption residual versus each factor are plotted and they are given in Appendix D in Figures D.1 – D.7. From the figures it is seen that for the “number of activity factor” and “Resource factor” factor the variances differs for each level. Therefore, it is necessary to make transformation on the obj_TD response.

The log-it transformation can be useful for correcting both nonnormality in process data and subgroup process variation that is related to the subgroup mean.

The log-it transformation is a particularly useful family of transformations. It is defined as:

$$y' = \ln\left(\frac{y}{1-y}\right)$$

where y is the response variable.

Given a particular transformation such as the log-it transformation defined above, it is helpful to define a measure of the normality of the resulting transformation. One measure is to compute the correlation coefficient of a normal probability plot. The correlation is computed between the vertical and horizontal axis variables of the probability plot and is a convenient measure of the linearity of the probability plot (the more linear the probability plot, the better a normal distribution fits the data).

Another way considerwed for normalization of residuals was taking the average values of the responses for 5 replications. However results obtained by this method didn’t satisfy a normal distribution. Therefore these calculations are not shown in details.

After transformation, normal probability of residuals, histogram of residuals and residuals vs. fitted values plot are given in Figure 4.4., Figure 4.5. and Figure 4.6., respectively. The plots of factors vs response after transformation are also given in Appendix D in Figures D.8. - D.14.

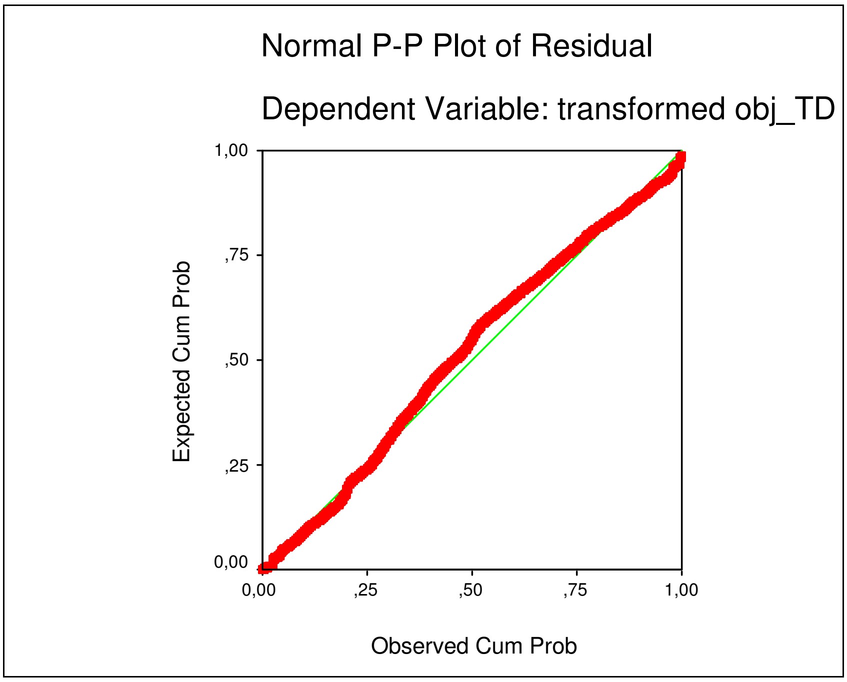

Figure 4.4. Normal Probability plot of Residuals for Transformed obj_TD

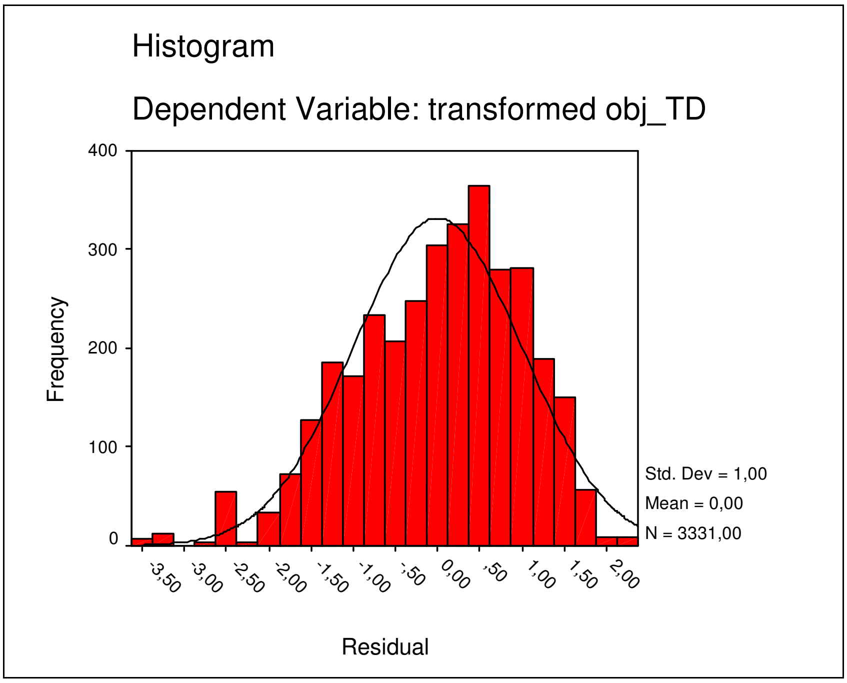

Figure 4.5. Histogram of Residuals for Transformed obj_TD

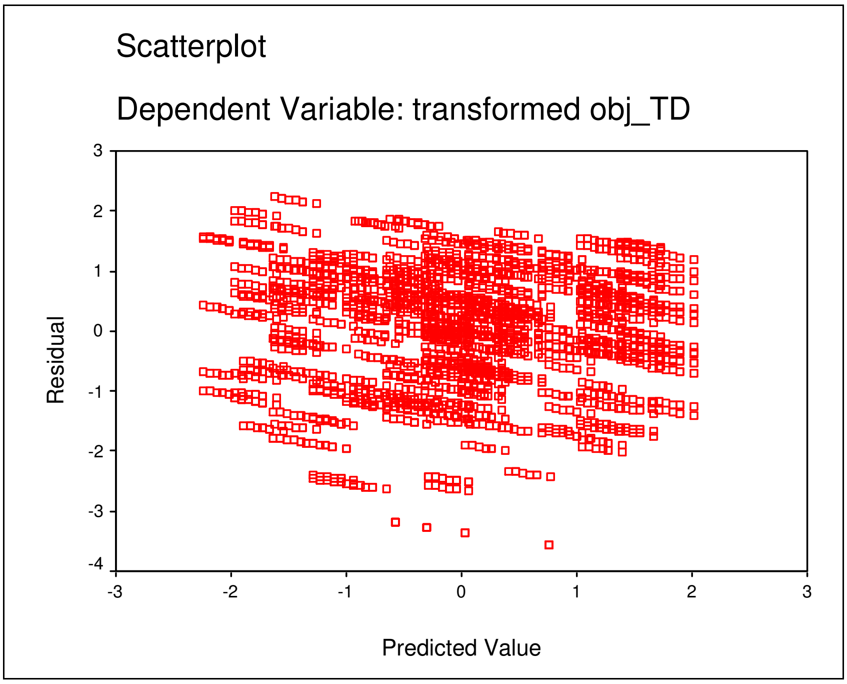

Figure 4.6. Residuals vs. Fitted values for Transformed obj_TD

ANOVA table for the transformed obj_TD responses is the Table 4.2. All the main effects of factors and all 2-way interactions of the factors are evaluated at 0,05 significance level. It is seen that all the factors except for alpha and beta factors have significant influences in obj_TD measure. From ANOVA table it is clearly seen that the interactions between factors number of activities, number of resources, resource factor, resource strength and restrictiveness are significant except the case number of resource-resource strength pair. Also, alpha and beta factors are not interacted with any other factor.

Table 4.2. ANOVA Table of Transformed obj_TD Values

General Linear Model: transformed Obj_TD versus Factors

Factor	Type	Levels	Values
SSP	fixed	3	TFS_Sch; ETD_Sch; RS_Sch
no of act	fixed	2	30; 60
no of res	fixed	2	1; 3
RF	fixed	2	0,50; 0,75
RS	fixed	2	0,5; 0,7
OS	fixed	2	0,50; 0,65
alpha	fixed	3	0,2; 0,4; 0,8
beta	fixed	3	0,2; 0,4; 0,8

Analysis of Variance for transformed OBJ_TD, using Adjusted SS for Tests

Source	DF	Seq SS	Adj SS	Adj MS	F	P
SSP	2	12204,80	12116,04	6058,02	2572,75	0,000
no of act	1	2638,14	2614,66	2614,66	1110,40	0,000
no of res	1	658,58	664,07	664,07	282,02	0,000
RF	1	845,73	818,66	818,66	347,67	0,000
RS	1	561,03	589,36	589,36	250,29	0,000
OS	1	739,69	731,80	731,80	310,78	0,000
alpha	2	0,00	0,00	0,00	0,00	1,000
beta	2	0,00	0,00	0,00	0,00	1,000
2-WAY INTERACTIONS						
SSP*no of act	2	840,27	825,24	412,62	175,23	0,000
SSP*no of res	2	210,34	213,92	106,96	45,42	0,000
SSP*RF	2	54,31	56,68	28,34	12,04	0,000
SSP*RS	2	138,24	144,02	72,01	30,58	0,000
SSP*OS	2	114,59	115,59	57,79	24,54	0,000
SSP*alpha	4	0,00	0,00	0,00	0,00	1,000
SSP*beta	4	0,00	0,00	0,00	0,00	1,000
no of act*no of res	1	256,93	261,58	261,58	111,09	0,000
no of act*RF	1	54,22	50,79	50,79	21,57	0,000
no of act*RS	1	26,17	25,35	25,35	10,76	0,001
no of act*OS	1	381,34	389,29	389,29	165,33	0,000
no of act*alpha	2	0,00	0,00	0,00	0,00	1,000
no of act*beta	2	0,00	0,00	0,00	0,00	1,000
no of res*RF	1	23,73	24,17	24,17	10,26	0,001
no of res*RS	1	1,13	1,49	1,49	0,63	0,426
no of res*OS	1	28,41	27,76	27,76	11,79	0,001
no of res*alpha	2	0,00	0,00	0,00	0,00	1,000
no of res*beta	2	0,00	0,00	0,00	0,00	1,000
RF*RS	1	233,99	235,36	235,36	99,96	0,000
RF*OS	1	64,37	63,36	63,36	26,91	0,000
RF*alpha	2	0,00	0,00	0,00	0,00	1,000
RF*beta	2	0,00	0,00	0,00	0,00	1,000
RS*OS	1	87,74	87,74	87,74	37,26	0,000
RS*alpha	2	0,00	0,00	0,00	0,00	1,000
RS*beta	2	0,00	0,00	0,00	0,00	1,000
OS*alpha	2	0,00	0,00	0,00	0,00	1,000
OS*beta	2	0,00	0,00	0,00	0,00	1,000
alpha*beta	4	0,00	0,00	0,00	0,00	1,000
Error	4211	9915,60	9915,60	2,35		
Total	4274	30079,33				

Mean of each factor level by using obj-TD responses before transformation is given in Table 4.3. and Main Effects Plots and Interaction Plots of data means of each factor for obj_TD are given in Appendix E.

Table 4.3 Factor Level Means for TD Objective

FACTORS	LEVELS	SELECTION POLICY			MEANS
		TFS	RS	ETD	
Number of Activities	30	410	47	13	157,75
	60	637	114	98	285,58
Number of Resources	1	505	71	52	209,6
	3	541	91	59	233,7
Resource Factor	0,5	370	46	34	150,33
	0,75	678	116	77	293,64
Resource Strength	0,5	564	103	69	246,88
	0,7	483	59	41	196,07
Restrictiveness (OS)	0,5	622	110	82	272,99
	0,65	425	52	28	169,86
α	0,2	523,5	81	55	221,53
	0,4	523,5	81	55	221,53
	0,8	523,5	81	55	221,53
β	0,2	523,5	81	55	221,53
	0,4	523,5	81	55	221,53
	0,8	523,5	81	55	221,53
Total Mean for Obj_TD		**523,5**	**81**	**55**	**221,53**

The experimentation revealed the fact that the two commonly used stability policies stated in the literature, the total free slack policy and the right shift policy produced poorer results compared to ETD policy in the case when we considered the worst case condition, i. e. all the activities in the time window are delayed.

α and β having no significance is an expected condition for this case since we assume that all possible delays are realized.

On the average, ETD policy gave the best results, right-shift policy is the policy that gave better results compared to the total free slack policy.

As the number of activities, the number of resources and the resource factor are increased the policies produced poorer results; whereas as the resource strength and restrictiveness are increased the policies produced better results.

4.6.2. Average Conditions

Before adopting ANOVA to the problem, the adequacy of the model is checked considering the data for average conditions that is objective ETD (obj_ETD).

The normal probability plot of residuals and histograms of residuals are given in Figure 4.7. and Figure 4.8., respectively. Figure 4.9 shows the plot of residuals versus fitted values.

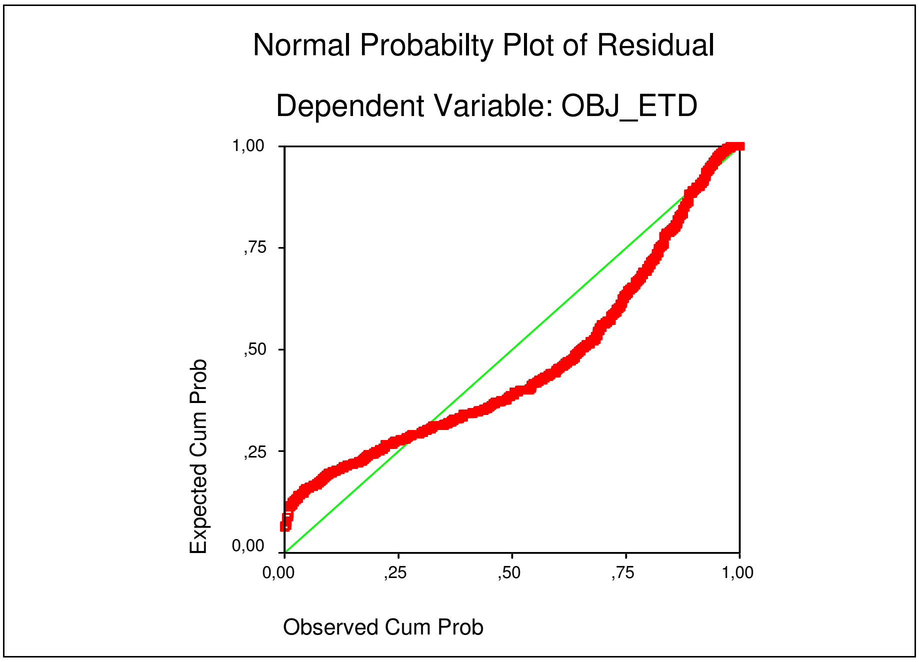

Figure 4.7. Normal Probability plot of Residuals for obj_ETD response

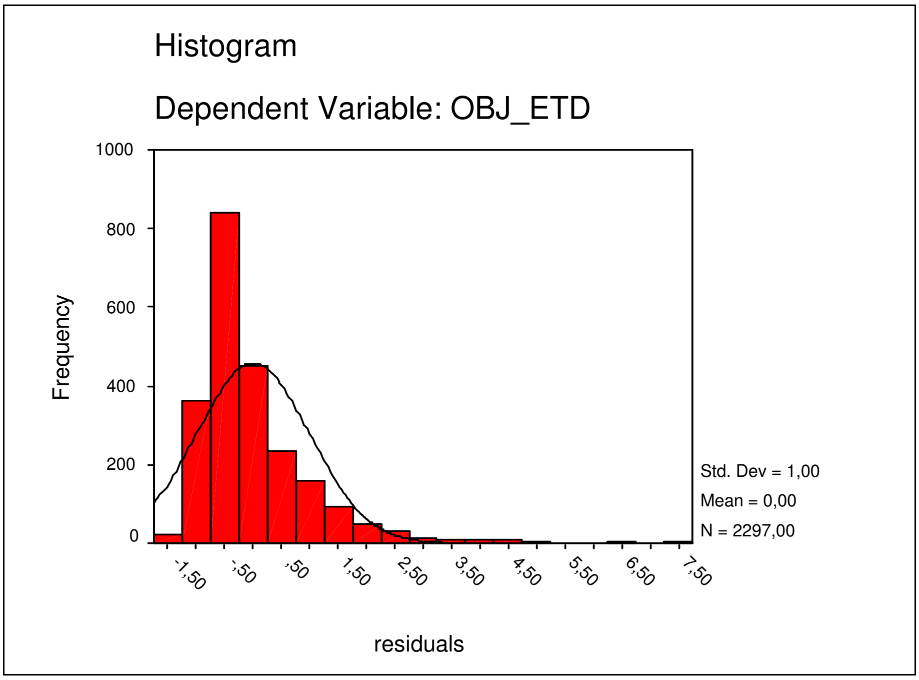

Figure 4.8. Histogram of Residuals for obj_ETD response

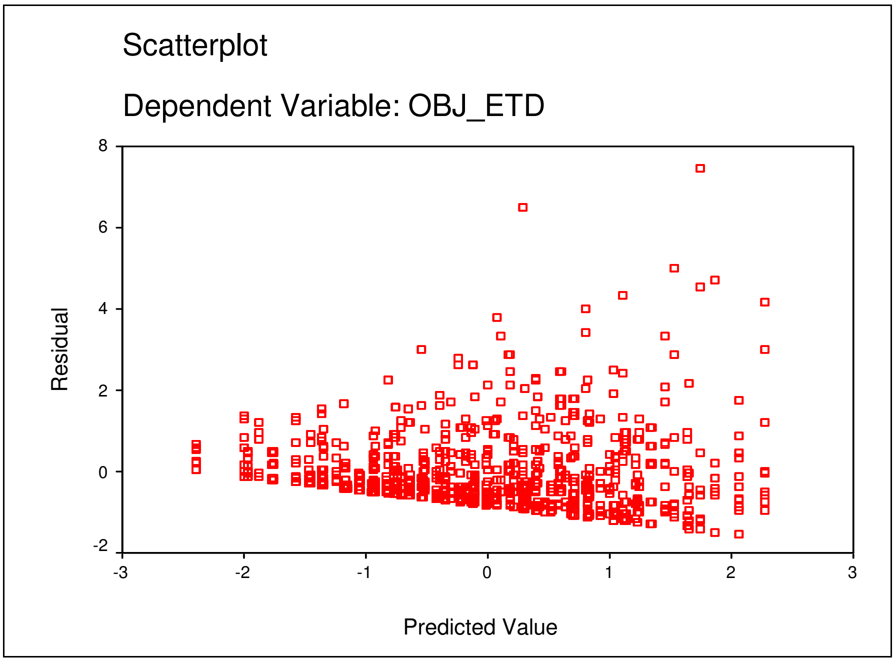

Figure 4.9. Residuals vs. Fitted values for obj_ETD response

From the figures it is seen that the residuals are not normally distributed. In order to check the constant variance assumption residual versus each factor are plotted and they are given in Appendix D in Figures D.15 – D.21. From the figures it is seen that for all factors except for alpha factor the constant variance assumption fails. Therefore, it is necessary to make transformation on the obj_ETD response.

The log-it transformation is applied for the responses in average conditions also. After transformation, normal probability of residuals, histogram of residuals and residuals vs. fitted values plot are given in Figure 4.10., Figure 4.11. and Figure 4.12., respectively. The plots of factors vs response after transformation are also given in Appendix D in Figures D.22. - D.28.

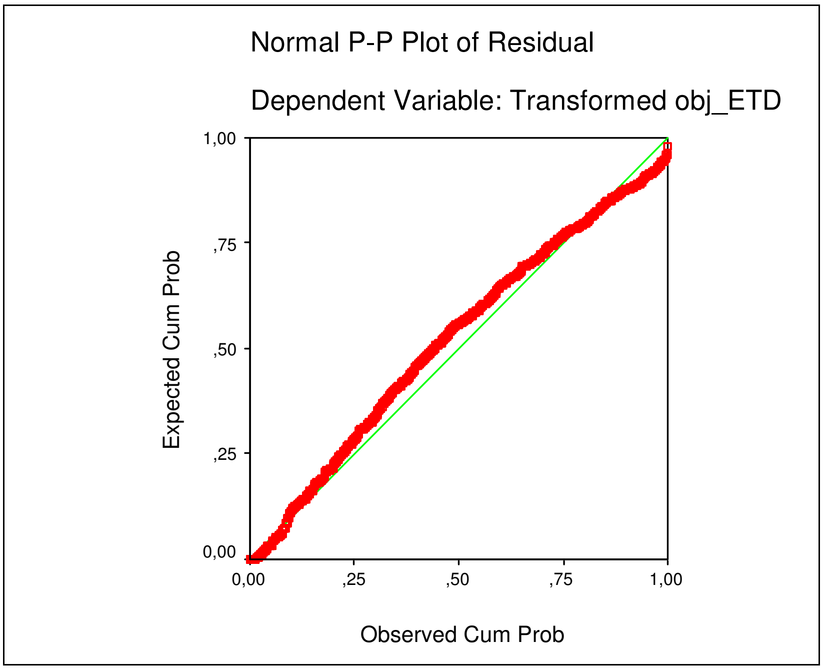

Figure 4.10. Normal Probability plot of Residuals for Trans. obj_ETD response

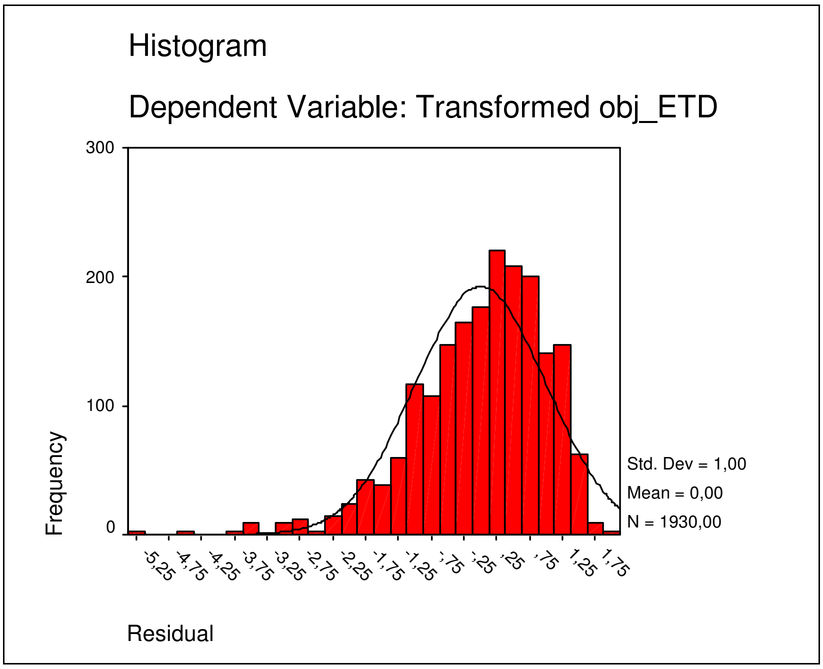

Figure 4.11. Histogram of Residuals for Trans. obj_ETD response

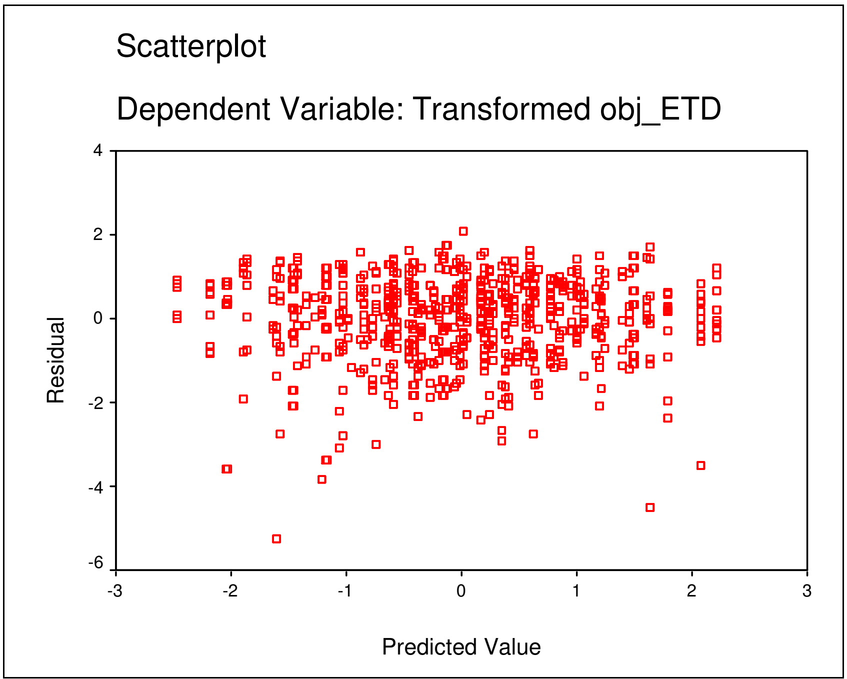

Figure 4.12. Residuals vs. Fitted values for Trans. obj_ETD response

ANOVA table for the transformed obj_ETD responses is the Table 4.4. All the main effects of factors and all 2-way interactions of the factors are evaluated at 0,05 significance level. It is seen that all the factors except for alpha factor have significant influences in obj_ETD measure. From ANOVA table it is clearly seen that the interactions between factors "schedule selection policy - number of activities", "schedule selection policy – resource strength", "number of activities-number of resources", "number of activities-restrictiveness", "number of resources-resource factor", "resource factor-restrictiveness" and "resource strength-restrictiveness" are significant.

Table 4.4. ANOVA Table of Transformed obj_ETD Values

General Linear Model: Transformed obj_ETD versus Factors

Factor	Type	Levels	Values
SSP	fixed	2	TFS_Sch; RS_Sch
no of act	fixed	2	30; 60
no of res	fixed	2	1; 3
RF	fixed	2	0,50; 0,75
RS	fixed	2	0,5; 0,7
OS	fixed	2	0,50; 0,65
alpha	fixed	3	0,2; 0,4; 0,8
beta	fixed	3	0,2; 0,4; 0,8

Analysis of Variance for Transformed obj_ETD, using Adj. SS for Tests

Source	DF	Seq SS	Adj SS	Adj MS	F	P
SSP	1	235,755	250,846	250,846	89,01	0,000
no of act	1	29,975	5,103	5,103	1,81	0,000
no of res	1	761,517	705,455	705,455	250,34	0,000
RF	1	510,692	456,116	456,116	161,86	0,000
RS	1	152,891	189,100	189,100	67,10	0,000
OS	1	58,765	71,424	71,424	25,35	0,000
alpha	2	0,011	0,014	0,007	0,00	0,997
beta	2	43,177	40,083	20,041	7,11	0,001
2-WAY INTERACTIONS						
SSP*no of act	1	35,755	40,212	40,212	14,27	0,000
SSP*no of res	1	10,554	4,113	4,113	1,46	0,227
SSP*RF	1	1,954	0,135	0,135	0,05	0,827
SSP*RS	1	20,169	22,636	22,636	8,03	0,005
SSP*OS	1	1,747	0,191	0,191	0,07	0,795
SSP*alpha	2	0,006	0,013	0,006	0,00	0,998
SSP*beta	2	2,977	2,391	1,195	0,42	0,654
no of act*no of res	1	342,542	327,232	327,232	116,12	0,000
no of act*RF	1	6,019	5,089	5,089	1,81	0,179
no of act*RS	1	3,342	1,705	1,705	0,60	0,437
no of act*OS	1	25,698	31,949	31,949	11,34	0,001
no of act*alpha	2	0,025	0,032	0,016	0,01	0,994
no of act*beta	2	4,113	5,219	2,610	0,93	0,396
no of res*RF	1	68,342	64,416	64,416	22,86	0,000
no of res*RS	1	0,836	1,208	1,208	0,43	0,513
no of res*OS	1	2,201	3,106	3,106	1,10	0,294
no of res*alpha	2	0,009	0,012	0,006	0,00	0,998
no of res*beta	2	0,478	0,615	0,307	0,11	0,897
RF*RS	1	4,854	6,055	6,055	2,15	0,143
RF*OS	1	108,128	103,653	103,653	36,78	0,000
RF*alpha	2	0,015	0,015	0,008	0,00	0,997
RF*beta	2	2,094	2,367	1,184	0,42	0,657
RS*OS	1	6,265	6,781	6,781	2,41	0,121
RS*alpha	2	0,013	0,012	0,006	0,00	0,998
RS*beta	2	6,401	6,258	3,129	1,11	0,330
OS*alpha	2	0,003	0,004	0,002	0,00	0,999
OS*beta	2	0,457	0,457	0,229	0,08	0,922
alpha*beta	4	0,011	0,011	0,003	0,00	1,000
Error	2243	6320,867	6320,867	2,818		
Total	2296	8768,659				

Mean of each factor level by using obj_ETD responses before transformation is given in Table 4.5. and Main Effects Plots and Interaction Plots of data means of each factor for obj_ETD are given in Appendix E.

Table 4.5 Factor Level Means for ETD Objective

FACTORS	LEVELS	SELECTION POLICY		MEANS
		TFS	**RS**	
Number of Activities	30	66,4	50	58,50
	60	49,7	33,3	52,60
Number of Resources	1	45,4	27	40,00
	3	70,4	57,5	64,51
Resource Factor	0,5	44,8	30,6	38,50
	0,75	69,8	51,8	61,37
Resource Strength	0,5	61,8	53,2	57,95
	0,7	54	32	43,78
Restrictiveness (OS)	0,5	64,8	45	55,49
	0,65	51	39	45,68
α	0,2	58	42,2	50,71
	0,4	58	42,2	50,69
	0,8	58	42,2	50,66
β	0,2	69	49,7	59,80
	0,4	54	40	47,61
	0,8	51,5	37	45,07
Total Mean for Obj_ETD		**57,8**	**42**	**50,68**

The experimentation revealed the fact that the two commonly used stability policies stated in the literature, the total free slack policy and the right shift policy do not produce results as good as the ETD policy which tries to find a schedule that minimizes the expected total delay in the start time of the activities for the average conditions. Among the two policies, right-shift policy, in general, produced better results than the total free slack policy.

As the number of activities, resource strength and restrictiveness increased the two policies produced better results, whereas as the number of resources are increased, the two policies produced poorer results.

The result of α and β values being not significant in the performance of these policies is not intuitive. It may be explained as, as long as there is a delay, two policies which does not take into account the deviations from the scheduled start times, provides the same solutions.

CHAPTER 5

CONCLUSION

Project scheduling deals with the allocation of scarce resources to a set of interrelated activities that are usually directed toward some major output and require a significant period of time to perform. The two well-known network techniques CPM and PERT, were both developed in the late 1950's to determine the minimum completion time of a project in such a way that the precedence relationships among activities are satisfied. However, project activities are subject to considerable uncertainty, which may lead to multiple schedule disruptions during project execution.

In this study, we consider project stability problem in a resource constrained project environment and also there is a time window corresponding to a time period in which there might be an anticipated disturbance affecting the duration of a set of activities scheduling during that time window. We consider three stability measures and three policies. Since finding the exact solution to the problems in real life projects which consist of a large number of activities is not possible, we use Tabu Search heuristic to generate stable schedules.

Three policies (TFS, TD and ETD) are used to obtain pre-schedules and two stability measures (TD and ETD) are used to compare the stability values of these schedules. 4320 problems are generated by RanGen algorithm by using eight different factors. Factors used in the study are chosen by mainly two criteria. These factors reflect the characteristics of the networks with respect to the activities, resource constrainedness and precedence relations. These factors and factor levels are the ones that are largely used in literature for the project scheduling problems. Number of activities, number of resources, resource factor, resource strength and restrictiveness are the factors that are used in the phase of generating networks by RanGen algorithm. Other three parameters probability of disruption event to occur, probability of disruption of an activity in time window and schedule selection policy are used in solving the problems and they are related with two separate objectives.

As a result of the experimental design, it is seen that probability of disruption event to occur does not have a significant effect on the results. Mean deviations from the best schedules are the lowest when ETD policy is used as the selection policy and the highest when TFS policy is used as the selection policy. All factors except for alpha and beta have influence on the worst case conditions objective and all factors except for alpha have influence on the average conditions objective.

As mentioned in section 4.2 the percent deviation from the best solution is considered as our response variable. Since alpha is a multiplier in the objective function formula (Section 3.2) and it is placed in both numerator and denominator of the fraction of the formula used for computing the response variable, the effect of alpha factor is terminated. If we had considered the stability measure values as response variables we could be able to see the effects of alpha factor on the results.

In future, as an extention of this study, problem solutions with other heuristics and methods can be tried. Also some other stability measures or schedule selection criterias can be tried.

REFERENCES

1- Al-Fawzan, M. A., Haouari, M., 2004. A bi-objective model for robust resource-constrained project scheduling. International Journal of Production Economics.

2- Al-Sultan, K. S., 1995. A tabu search approach to the clustering problem. Pattern Recognition, Vol. 28, No 9, pp. 1443-1451.

3- Artigues, C., Michelon P., Reusser S., 2003. Insertion Techniques for Static and Dynamic Resource-Constrained Project Scheduling. European Journal of Operational Research 149, 249-267.

4- Blazewicz, J., Cellary, W., Slowinski, R., Weglarz, J., 1986. Scheduling under resource constraints–Deterministic models. Annals of Operations Research 7.

5- Bowers, J. A., 1995. Criticality in Resource Constraint Networks. Journal of the Operational Research Society, 46, 80-91.

6- Buey, G., 1989, Production Scheduling: Practice and Theory. European Journal of Operational Research 39, 17-31.

7- Cavalcante, C.C.B., Souza, C. C., Savelsbergh, M. W. P., Wang Y., Wolsey, L. A., 2001, Scheduling Projects with Labor Constrains. Discrete Applied Mathematics 112, 27-52.

8- Daniels, R. L., Kouvelis, P., 1995, Robust Scheduling to Hedge against Processing Time Uncertainty in Single-stage Production. Vol. 41, No. 2.

9- De Reyck, B., Herroelen, W., 1999. The Multi-Mode Resource-Constrained Project Scheduling Problem with Generalized Precedence Relations. European Journal of Operational Research 119. 538-556.

10- Demeuleemester, E., Vanhoucke, M., Herroelen, W., 2003, RanGen: A Random Network Generator For Activity-on-the-Node Networks. Journal of Scheduling 6. 17-38.

11- Dodin, B., Elimam, A. A., Rolland, E., 1998, Tabu Search in Audit Scheduling. European Journal of Operational Research 106. 373-392.

12- Dooley, K. J., Mahmoodi, F., 1992, Identification of Robust Scheduling Heuristics: Application of Taguchi Methods in Simulation Studies, Computers Ind. Engng , Vol.22, No. 4, 359-368.

13- Glover, F., Laguna, M., 1997. Tabu Search. Kluwer Academic Publishers, USA.

14- Gören, S., 2002. Robustness and Stability Measures for Scheduling Policies in a Single Machine Environment. A Thesis Submitted to the Department of Industrial Engineering and the Institute of Engineering and Sciences of Bilkent University.

15- Herroelen, W., Leus, R., 2002. Stability and Resource Allocation in Project Baseline Scheduling. Research Report, Department of Applied Economics, KULeuven.

16- Herroelen, W., Leus, R., 2004. The Construction of Stable Baseline Projects. European Journal of Operational Research 156. 550– 565

17- Herroelen, W., Leus, R., 2005. The Complexity of Machine Scheduling For Stability with a Single Disrupted Job. Operations Research Letters 33. 151 – 156

18- Herroelen, W., Leus, R., 2005. Project Scheduling Under Uncertainty: Survey and Research Potentials. European Journal of Operational Research Vol. 165, Issue 2, 289– 306.

19- Honkomp, S. J., Mockus, L., Reklaitis, G. V., 1997. Robust Scheduling with Processing Time Uncertainty. Computers Chemical Engineering, Vol. 21, Suppl, S1055-S1060.

20- Jensen, M., T., 2001. Improving Robustness and Flexibility of Tardiness and Total Flow-Time Job Shops Using Robustness Measures. Applied Soft. Computing, 35-52.

21- Kolisch, R., Sprecher, A., Drexl, A., 1995. Characterization and Generation of a General Class of Resource-Constrained Project Scheduling Problems. Manag. Sci., 41, 1693-1703.

22- Kouvelis, P., Yu, G., 1996. Robust Discrete Optimization and Its Applications. Kluwer Academic Publishers, Netherlands.

23- Kusiak, A., He, D. W., 1998. Design for Agility a Scheduling Perspective. Robotics and Computer- Integrated Manufacturing 14, 415-427.

24- Leon, V. J., Wu, S. D., Storer, R. H., 1994. Robustness Measures and Robust Scheduling for Job Shops. IIE Transactions, Vol. 26, No. 5.

25- Mackworth, A.K., 1997. Consistency in Networks of Relations. Artificial Intelligence 8, 99-118.

26- Mehta, S., V., Uzsoy, R. M., 1998. Predictable Scheduling of a Job Shop Subject to Breakdowns. IEEE Transactions on Robotics and Automation. Vol. 14, No. 3.

27- Mendenhall, W., Terry, S., 1996. A Second Course in Statistics: Regression Analysis. 5th Ed., Prentice Hall, Inc, USA.

28- O'Donovan, R., Uzsoy, R., McKay, K. N., 1999. Predictable Scheduling of a Single Machine with Breakdown and Sensitive Jobs. International Journal of Production Research, Vol. 37, No. 18, 4217-4233.

29- Ortaç, N., 1994. Comparisons of Heuristics for Resource-Constrained Project Scheduling with Discounted Cash Flows and Progress Payments. A Master's Thesis for The Degree of Master of Science in Industrial Engineering, METU, Ankara.

30- Özay, B., 2002. Project Scheduling with Progress Payments in Generalized Project Networks. Industrial Engineering, METU, Ankara.

31- Sabuncuoğlu, I., Bayız, M., 2000. Analysis of Reactive Scheduling Problems in a Job Shop Environment. European Journal of Operational Research 126. 567-586.

32- Schumacher, J., Lukszo, Z. V., Weijnen, M. P. C., 1999. Disturbances and Their Impact on Scheduling, Computers & Industrial Engineering, 37, 75-79.

33- Sepil, C., Ortaç, N., 1997. Performance of the Heuristic Procedures for Constrained Projects with Progress Payments. Journal of the Operational Research Society, 48, 1123-1130.

34- Tavares, L. V., Ferreira, J. A. A, Coelho, J. S., 1997. On The Optimal Management of Project Risk. European Journal of Operational Research 107 451-469

35- Tsai, Y., Gemmill, D. D., 1998. Using Tabu Search to Schedule Activities of Stochastic Resource-Constrained Projects. European Journal of Operational Research 111. 129-141

36- Valls V., Quintanilla S., Ballestin F., 2003. Resource-constrained Project Scheduling: A critical activity reordering heuristic. European Journal of Operational Research 149. 282-301.

37- Verhoeven, M.G.A., 1998. Tabu Search for Resource-constrained Scheduling. European Journal of Operational Research 106. 266-276.

38- Wiest, J. D., Levy, F. K., 1969. A Management Guide to PERT/CPM. Prentice-Hall, New Jersey.

39- Wu, S. D., Byeon, E., Storer, R., H., A Graph-Theoretic Decomposition of the Job Shop Scheduling Problem to Achieve Scheduling Robustness. Operations Research, Vol. 47, No.1

40- Xu, J., Chiu, S. Y., Glover, F., 1998. Fine-tuning a Tabu Search Algorithm with Statistical Tests. Int. Trans. Opl. Res., Vol.5, No.3, 233-244.

41- Zawack, D., Balas, E., Adams, J., 1998, The Shifting Bottleneck Procedure for Job Shop Scheduling. Management Science, Vol. 34, No. 3.

APPENDIX A

TABLES FOR NUMERICAL EXAMPLE

Table A.1. Values of start times and deviations in start times in starting schedule

f=1	f=2	f=3	f=4	f=5	f=6	f=7	f=8
Sd0 = 0	Sd0 = 0	Sd0 = 0	Sd0 = 0	Sd0 = 0	Sd_0 = 0	Sd_0 = 0	Sd_0 = 0
Sd1 = 0	Sd1 = 0	Sd1 = 0	Sd1 = 0	Sd1 = 0	Sd_1 = 0	Sd_1 = 0	Sd_1 = 0
Sd2 = 8	Sd2 = 8	Sd2 = 8	Sd2 = 8	Sd2 = 8	Sd_2 = 8	Sd_2 = 8	Sd_2 = 8
Sd3 = 8	Sd3 = 8	Sd3 = 8	Sd3 = 8	Sd3 = 8	Sd_3 = 8	Sd_3 = 8	Sd_3 = 8
Sd4 = 16	Sd4 = 16	Sd4 = 16	Sd4 = 16	Sd4 = 16	Sd_4 = 16	Sd_4 = 16	Sd_4 = 16
Sd5 = 12	Sd5 = 12	Sd5 = 12	Sd5 = 12	Sd5 = 12	Sd_5 = 12	Sd_5 = 12	Sd_5 = 12
Sd6 = 30	Sd6 = 30	Sd6 = 30	Sd6 = 30	Sd6 = 30	Sd_6 = 30	Sd_6 = 30	Sd_6 = 30
Sd7 = 16	Sd7 = 16	Sd7 = 16	Sd7 = 16	Sd7 = 16	Sd_7 = 16	Sd_7 = 16	Sd_7 = 16
Sd8 = 20	Sd8 = 20	Sd8 = 20	Sd8 = 20	Sd8 = 20	Sd_8 = 20	Sd_8 = 20	Sd_8 = 20
Sd9 = 30	Sd9 = 30	Sd9 = 30	Sd9 = 30	Sd9 = 30	Sd_9 = 30	Sd_9 = 30	Sd_9 = 30
Sd10 = 38	Sd10 = 38	Sd10 = 38	Sd10 = 38	Sd10 = 38	Sd_10 = 38	Sd_10 = 38	Sd_10 = 38
Sd11 = 42	Sd11 = 43	Sd11 = 42	Sd11 = 43	Sd11 = 43	Sd_11 = 42	Sd_11 = 42	Sd_11 = 43
Sd12 = 47	Sd12 = 48	Sd12 = 47	Sd12 = 48	Sd12 = 49	Sd_12 = 48	Sd_12 = 48	Sd_12 = 49
Sd13 = 51	Sd13 = 51	Sd13 = 51	Sd13 = 52	Sd13 = 51	Sd_13 = 52	Sd_13 = 51	Sd_13 = 53
Sd14 = 51	Sd14 = 52	Sd14 = 53	Sd14 = 53	Sd14 = 53	Sd_14 = 53	Sd_14 = 52	Sd_14 = 53
Sd15 = 55	Sd15 = 56	Sd15 = 57	Sd15 = 57	Sd15 = 57	Sd_15 = 57	Sd_15 = 56	Sd_15 = 57
Sd16 = 58	Sd16 = 59	Sd16 = 60	Sd16 = 60	Sd16 = 60	Sd_16 = 60	Sd_16 = 59	Sd_16 = 60
dev0 = 0	dev0 = 0	dev0 = 0	dev0 = 0	dev0 = 0	dev_0 = 0	dev_0 = 0	dev_0 = 0
dev1 = 0	dev1 = 0	dev1 = 0	dev1 = 0	dev1 = 0	dev_1 = 0	dev_1 = 0	dev_1 = 0
dev2 = 0	dev2 = 0	dev2 = 0	dev2 = 0	dev2 = 0	dev_2 = 0	dev_2 = 0	dev_2 = 0
dev3 = 0	dev3 = 0	dev3 = 0	dev3 = 0	dev3 = 0	dev_3 = 0	dev_3 = 0	dev_3 = 0
dev4 = 0	dev4 = 0	dev4 = 0	dev4 = 0	dev4 = 0	dev_4 = 0	dev_4 = 0	dev_4 = 0
dev5 = 0	dev5 = 0	dev5 = 0	dev5 = 0	dev5 = 0	dev_5 = 0	dev_5 = 0	dev_5 = 0
dev6 = 0	dev6 = 0	dev6 = 0	dev6 = 0	dev6 = 0	dev_6 = 0	dev_6 = 0	dev_6 = 0
dev7 = 0	dev7 = 0	dev7 = 0	dev7 = 0	dev7 = 0	dev_7 = 0	dev_7 = 0	dev_7 = 0
dev8 = 0	dev8 = 0	dev8 = 0	dev8 = 0	dev8 = 0	dev_8 = 0	dev_8 = 0	dev_8 = 0
dev9 = 0	dev9 = 0	dev9 = 0	dev9 = 0	dev9 = 0	dev_9 = 0	dev_9 = 0	dev_9 = 0
dev10 = 0	dev10 = 0	dev10 = 0	dev10 = 0	dev10 = 0	dev_10 = 0	dev_10 = 0	dev_10 = 0
dev11 = 0	dev11 = 1	dev11 = 0	dev11 = 1	dev11 = 1	dev_11 = 0	dev_11 = 0	dev_11 = 1
dev12 = 0	dev12 = 1	dev12 = 0	dev12 = 1	dev12 = 2	dev_12 = 1	dev_12 = 1	dev_12 = 2
dev13 = 0	dev13 = 0	dev13 = 0	dev13 = 1	dev13 = 0	dev_13 = 1	dev_13 = 0	dev_13 = 2
dev14 = 0	dev14 = 1	dev14 = 2	dev14 = 2	dev14 = 2	dev_14 = 2	dev_14 = 1	dev_14 = 2
dev15 = 0	dev15 = 1	dev15 = 2	dev15 = 2	dev15 = 2	dev_15 = 2	dev_15 = 1	dev_15 = 2
dev16 = 0	dev16 = 1	dev16 = 2	dev16 = 2	dev16 = 2	dev_16 = 2	dev_16 = 1	dev_16 = 2
total = 0	total = 5	total = 6	total = 9	total = 9	total = 8	total = 4	total = 11

Where,

f= number of disruption scenario

Sd = Start times of activities after disruption

dev = deviation in start times of activities

Table A.2. Values of start times and deviations in start times in schedule

f=1	f=2	f=3	f=4
Sd0 = 0 Sd1 = 0 Sd2 = 8 Sd3 = 8 Sd4 = 16 Sd5 = 12 Sd6 = 33 Sd7 = 16 Sd8 = 33 Sd9 = 20 Sd10 = 45 Sd11 = 28 Sd12 = 34 Sd13 = 30 Sd14 = 58 Sd15 = 62 Sd16 = 65	Sd0 = 0 Sd1 = 0 Sd2 = 8 Sd3 = 8 Sd4 = 16 Sd5 = 12 Sd6 = 33 Sd7 = 16 Sd8 = 33 Sd9 = 20 Sd10 = 45 Sd11 = 28 Sd12 = 34 Sd13 = 30 Sd14 = 58 Sd15 = 62 Sd16 = 65	Sd0 = 0 Sd1 = 0 Sd2 = 8 Sd3 = 8 Sd4 = 16 Sd5 = 12 Sd6 = 33 Sd7 = 16 Sd8 = 33 Sd9 = 20 Sd10 = 47 Sd11 = 28 Sd12 = 34 Sd13 = 30 Sd14 = 60 Sd15 = 64 Sd16 = 67	Sd0 = 0 Sd1 = 0 Sd2 = 8 Sd3 = 8 Sd4 = 16 Sd5 = 12 Sd6 = 33 Sd7 = 16 Sd8 = 33 Sd9 = 20 Sd10 = 47 Sd11 = 28 Sd12 = 34 Sd13 = 30 Sd14 = 60 Sd15 = 64 Sd16 = 67
dev0 = 0 dev1 = 0 dev2 = 0 dev3 = 0 dev4 = 0 dev5 = 0 dev6 = 0 dev7 = 0 dev8 = 0 dev9 = 0 dev10 = 0 dev11 = 0 dev12 = 0 dev13 = 0 dev14 = 0 dev15 = 0 dev16 = 0	dev0 = 0 dev1 = 0 dev2 = 0 dev3 = 0 dev4 = 0 dev5 = 0 dev6 = 0 dev7 = 0 dev8 = 0 dev9 = 0 dev10 = 0 dev11 = 0 dev12 = 0 dev13 = 0 dev14 = 0 dev15 = 0 dev16 = 0	dev0 = 0 dev1 = 0 dev2 = 0 dev3 = 0 dev4 = 0 dev5 = 0 dev6 = 0 dev7 = 0 dev8 = 0 dev9 = 0 dev10 = 2 dev11 = 0 dev12 = 0 dev13 = 0 dev14 = 2 dev15 = 2 dev16 = 2	dev0 = 0 dev1 = 0 dev2 = 0 dev3 = 0 dev4 = 0 dev5 = 0 dev6 = 0 dev7 = 0 dev8 = 0 dev9 = 0 dev10 = 2 dev11 = 0 dev12 = 0 dev13 = 0 dev14 = 2 dev15 = 2 dev16 = 2
total = 0	total = 0	total = 8	total = 8

Table A.3. Values of start times and deviations in start times in schedule

f=1	f=2	f=3	f=4	f=5	f=6	f=7	f=8
Sd0 = 0	Sd0 = 0	Sd0 = 0	Sd0 = 0	Sd0 = 0	Sd0 = 0	Sd0 = 0	Sd0 = 0
Sd1 = 0	Sd1 = 0	Sd1 = 0	Sd1 = 0	Sd1 = 0	Sd1 = 0	Sd1 = 0	Sd1 = 0
Sd2 = 8	Sd2 = 8	Sd2 = 8	Sd2 = 8	Sd2 = 8	Sd2 = 8	Sd2 = 8	Sd2 = 8
Sd3 = 8	Sd3 = 8	Sd3 = 8	Sd3 = 8	Sd3 = 8	Sd3 = 8	Sd3 = 8	Sd3 = 8
Sd4 = 16	Sd4 = 16	Sd4 = 16	Sd4 = 16	Sd4 = 16	Sd4 = 16	Sd4 = 16	Sd4 = 16
Sd5 = 12	Sd5 = 12	Sd5 = 12	Sd5 = 12	Sd5 = 12	Sd5 = 12	Sd5 = 12	Sd5 = 12
Sd6 = 48	Sd6 = 48	Sd6 = 48	Sd6 = 48	Sd6 = 48	Sd6 = 48	Sd6 = 48	Sd6 = 48
Sd7 = 28	Sd7 = 28	Sd7 = 28	Sd7 = 28	Sd7 = 28	Sd7 = 28	Sd7 = 28	Sd7 = 28
Sd8 = 16	Sd8 = 16	Sd8 = 16	Sd8 = 16	Sd8 = 16	Sd8 = 16	Sd8 = 16	Sd8 = 16
Sd9 = 32	Sd9 = 32	Sd9 = 32	Sd9 = 32	Sd9 = 32	Sd9 = 32	Sd9 = 32	Sd9 = 32
Sd10 = 30	Sd10 = 30	Sd10 = 30	Sd10 = 30	Sd10 = 30	Sd10 = 30	Sd10 = 30	Sd10 = 30
Sd11 = 43	Sd11 = 44	Sd11 = 43	Sd11 = 44	Sd11 = 44	Sd11 = 43	Sd11 = 43	Sd11 = 44
Sd12 = 44	Sd12 = 44	Sd12 = 44	Sd12 = 44	Sd12 = 44	Sd12 = 44	Sd12 = 44	Sd12 = 44
Sd13 = 40	Sd13 = 40	Sd13 = 40	Sd13 = 40	Sd13 = 40	Sd13 = 40	Sd13 = 40	Sd13 = 40
Sd14 = 48	Sd14 = 48	Sd14 = 48	Sd14 = 48	Sd14 = 48	Sd14 = 48	Sd14 = 48	Sd14 = 48
Sd15 = 52	Sd15 = 52	Sd15 = 52	Sd15 = 52	Sd15 = 52	Sd15 = 52	Sd15 = 52	Sd15 = 52
Sd16 = 60	Sd16 = 60	Sd16 = 60	Sd16 = 60	Sd16 = 60	Sd16 = 60	Sd16 = 60	Sd16 = 60
dev0 = 0	dev0 = 0	dev0 = 0	dev0 = 0	dev0 = 0	dev0 = 0	dev0 = 0	dev0 = 0
dev1 = 0	dev1 = 0	dev1 = 0	dev1 = 0	dev1 = 0	dev1 = 0	dev1 = 0	dev1 = 0
dev2 = 0	dev2 = 0	dev2 = 0	dev2 = 0	dev2 = 0	dev2 = 0	dev2 = 0	dev2 = 0
dev3 = 0	dev3 = 0	dev3 = 0	dev3 = 0	dev3 = 0	dev3 = 0	dev3 = 0	dev3 = 0
dev4 = 0	dev4 = 0	dev4 = 0	dev4 = 0	dev4 = 0	dev4 = 0	dev4 = 0	dev4 = 0
dev5 = 0	dev5 = 0	dev5 = 0	dev5 = 0	dev5 = 0	dev5 = 0	dev5 = 0	dev5 = 0
dev6 = 0	dev6 = 0	dev6 = 0	dev6 = 0	dev6 = 0	dev6 = 0	dev6 = 0	dev6 = 0
dev7 = 0	dev7 = 0	dev7 = 0	dev7 = 0	dev7 = 0	dev7 = 0	dev7 = 0	dev7 = 0
dev8 = 0	dev8 = 0	dev8 = 0	dev8 = 0	dev8 = 0	dev8 = 0	dev8 = 0	dev8 = 0
dev9 = 0	dev9 = 0	dev9 = 0	dev9 = 0	dev9 = 0	dev9 = 0	dev9 = 0	dev9 = 0
dev10 = 0	dev10 = 0	dev10 = 0	dev10 = 0	dev10 = 0	dev10 = 0	dev10 = 0	dev10 = 0
dev11 = 0	dev11 = 1	dev11 = 0	dev11 = 1	dev11 = 1	dev11 = 0	dev11 = 0	dev11 = 1
dev12 = 0	dev12 = 0	dev12 = 0	dev12 = 0	dev12 = 0	dev12 = 0	dev12 = 0	dev12 = 0
dev13 = 0	dev13 = 0	dev13 = 0	dev13 = 0	dev13 = 0	dev13 = 0	dev13 = 0	dev13 = 0
dev14 = 0	dev14 = 0	dev14 = 0	dev14 = 0	dev14 = 0	dev14 = 0	dev14 = 0	dev14 = 0
dev15 = 0	dev15 = 0	dev15 = 0	dev15 = 0	dev15 = 0	dev15 = 0	dev15 = 0	dev15 = 0
dev16 = 0	dev16 = 0	dev16 = 0	dev16 = 0	dev16 = 0	dev16 = 0	dev16 = 0	dev16 = 0
total = 0	total = 1	total = 0	total = 1	total = 1	total = 0	total = 0	total = 1

Table A.3. (Continued)

f=9	f=10	f=11	f=12	f=13	f=14	f=15	f=16
Sd0 = 0	Sd0 = 0	Sd0 = 0	Sd0 = 0	Sd0 = 0	Sd0 = 0	Sd0 = 0	Sd0 = 0
Sd1 = 0	Sd1 = 0	Sd1 = 0	Sd1 = 0	Sd1 = 0	Sd1 = 0	Sd1 = 0	Sd1 = 0
Sd2 = 8	Sd2 = 8	Sd2 = 8	Sd2 = 8	Sd2 = 8	Sd2 = 8	Sd2 = 8	Sd2 = 8
Sd3 = 8	Sd3 = 8	Sd3 = 8	Sd3 = 8	Sd3 = 8	Sd3 = 8	Sd3 = 8	Sd3 = 8
Sd4 = 16	Sd4 = 16	Sd4 = 16	Sd4 = 16	Sd4 = 16	Sd4 = 16	Sd4 = 16	Sd4 = 16
Sd5 = 12	Sd5 = 12	Sd5 = 12	Sd5 = 12	Sd5 = 12	Sd5 = 12	Sd5 = 12	Sd5 = 12
Sd6 = 48	Sd6 = 49	Sd6 = 49	Sd6 = 49	Sd6 = 49	Sd6 = 49	Sd6 = 48	Sd6 = 49
Sd7 = 28	Sd7 = 28	Sd7 = 28	Sd7 = 28	Sd7 = 28	Sd7 = 28	Sd7 = 28	Sd7 = 28
Sd8 = 16	Sd8 = 16	Sd8 = 16	Sd8 = 16	Sd8 = 16	Sd8 = 16	Sd8 = 16	Sd8 = 16
Sd9 = 32	Sd9 = 32	Sd9 = 32	Sd9 = 32	Sd9 = 32	Sd9 = 32	Sd9 = 32	Sd9 = 32
Sd10 = 30	Sd10 = 30	Sd10 = 30	Sd10 = 30	Sd10 = 30	Sd10 = 30	Sd10 = 30	Sd10 = 30
Sd11 = 43	Sd11 = 44	Sd11 = 43	Sd11 = 44	Sd11 = 44	Sd11 = 43	Sd11 = 43	Sd11 = 44
Sd12 = 45	Sd12 = 45	Sd12 = 45	Sd12 = 45	Sd12 = 45	Sd12 = 45	Sd12 = 45	Sd12 = 45
Sd13 = 40	Sd13 = 40	Sd13 = 40	Sd13 = 40	Sd13 = 40	Sd13 = 40	Sd13 = 40	Sd13 = 40
Sd14 = 48	Sd14 = 49	Sd14 = 49	Sd14 = 49	Sd14 = 49	Sd14 = 49	Sd14 = 48	Sd14 = 49
Sd15 = 52	Sd15 = 53	Sd15 = 53	Sd15 = 53	Sd15 = 53	Sd15 = 53	Sd15 = 52	Sd15 = 53
Sd16 = 60	Sd16 = 61	Sd16 = 61	Sd16 = 61	Sd16 = 61	Sd16 = 61	Sd16 = 60	Sd16 = 61
dev0 = 0	dev0 = 0	dev0 = 0	dev0 = 0	dev0 = 0	dev0 = 0	dev0 = 0	dev0 = 0
dev1 = 0	dev1 = 0	dev1 = 0	dev1 = 0	dev1 = 0	dev1 = 0	dev1 = 0	dev1 = 0
dev2 = 0	dev2 = 0	dev2 = 0	dev2 = 0	dev2 = 0	dev2 = 0	dev2 = 0	dev2 = 0
dev3 = 0	dev3 = 0	dev3 = 0	dev3 = 0	dev3 = 0	dev3 = 0	dev3 = 0	dev3 = 0
dev4 = 0	dev4 = 0	dev4 = 0	dev4 = 0	dev4 = 0	dev4 = 0	dev4 = 0	dev4 = 0
dev5 = 0	dev5 = 0	dev5 = 0	dev5 = 0	dev5 = 0	dev5 = 0	dev5 = 0	dev5 = 0
dev6 = 0	dev6 = 1	dev6 = 1	dev6 = 1	dev6 = 1	dev6 = 1	dev6 = 0	dev6 = 1
dev7 = 0	dev7 = 0	dev7 = 0	dev7 = 0	dev7 = 0	dev7 = 0	dev7 = 0	dev7 = 0
dev8 = 0	dev8 = 0	dev8 = 0	dev8 = 0	dev8 = 0	dev8 = 0	dev8 = 0	dev8 = 0
dev9 = 0	dev9 = 0	dev9 = 0	dev9 = 0	dev9 = 0	dev9 = 0	dev9 = 0	dev9 = 0
dev10 = 0	dev10 = 0	dev10 = 0	dev10 = 0	dev10 = 0	dev10 = 0	dev10 = 0	dev10 = 0
dev11 = 0	dev11 = 1	dev11 = 0	dev11 = 1	dev11 = 1	dev11 = 0	dev11 = 0	dev11 = 1
dev12 = 1	dev12 = 1	dev12 = 1	dev12 = 1	dev12 = 1	dev12 = 1	dev12 = 1	dev12 = 1
dev13 = 0	dev13 = 0	dev13 = 0	dev13 = 0	dev13 = 0	dev13 = 0	dev13 = 0	dev13 = 0
dev14 = 0	dev14 = 1	dev14 = 1	dev14 = 1	dev14 = 1	dev14 = 1	dev14 = 0	dev14 = 1
dev15 = 0	dev15 = 1	dev15 = 1	dev15 = 1	dev15 = 1	dev15 = 1	dev15 = 0	dev15 = 1
dev16 = 0	dev16 = 1	dev16 = 1	dev16 = 1	dev16 = 1	dev16 = 1	dev16 = 0	dev16 = 1
total = 1	total = 6	total = 5	total = 6	total = 6	total = 5	total = 1	total = 6

Table A.4. Values of start times and deviations in start times in schedule

f=1	f=2	f=3	f=4	f=5	f=6	f=7	f=8
Sd0 = 0	Sd0 = 0	Sd0 = 0	Sd0 = 0	Sd0 = 0	Sd0 = 0	Sd0 = 0	Sd0 = 0
Sd1 = 0	Sd1 = 0	Sd1 = 0	Sd1 = 0	Sd1 = 0	Sd1 = 0	Sd1 = 0	Sd1 = 0
Sd2 = 8	Sd2 = 8	Sd2 = 8	Sd2 = 8	Sd2 = 8	Sd2 = 8	Sd2 = 8	Sd2 = 8
Sd3 = 8	Sd3 = 8	Sd3 = 8	Sd3 = 8	Sd3 = 8	Sd3 = 8	Sd3 = 8	Sd3 = 8
Sd4 = 16	Sd4 = 16	Sd4 = 16	Sd4 = 16	Sd4 = 16	Sd4 = 16	Sd4 = 16	Sd4 = 16
Sd5 = 12	Sd5 = 12	Sd5 = 12	Sd5 = 12	Sd5 = 12	Sd5 = 12	Sd5 = 12	Sd5 = 12
Sd6 = 43	Sd6 = 43	Sd6 = 44	Sd6 = 44	Sd6 = 43	Sd6 = 44	Sd6 = 43	Sd6 = 44
Sd7 = 28	Sd7 = 28	Sd7 = 28	Sd7 = 28	Sd7 = 28	Sd7 = 28	Sd7 = 28	Sd7 = 28
Sd8 = 16	Sd8 = 16	Sd8 = 16	Sd8 = 16	Sd8 = 16	Sd8 = 16	Sd8 = 16	Sd8 = 16
Sd9 = 32	Sd9 = 32	Sd9 = 32	Sd9 = 32	Sd9 = 32	Sd9 = 32	Sd9 = 32	Sd9 = 32
Sd10 = 30	Sd10 = 30	Sd10 = 30	Sd10 = 30	Sd10 = 30	Sd10 = 30	Sd10 = 30	Sd10 = 30
Sd11 = 50	Sd11 = 50	Sd11 = 50	Sd11 = 50	Sd11 = 50	Sd11 = 50	Sd11 = 50	Sd11 = 50
Sd12 = 55	Sd12 = 55	Sd12 = 55	Sd12 = 55	Sd12 = 55	Sd12 = 55	Sd12 = 55	Sd12 = 55
Sd13 = 40	Sd13 = 40	Sd13 = 40	Sd13 = 40	Sd13 = 40	Sd13 = 40	Sd13 = 40	Sd13 = 40
Sd14 = 43	Sd14 = 43	Sd14 = 44	Sd14 = 44	Sd14 = 43	Sd14 = 44	Sd14 = 43	Sd14 = 44
Sd15 = 47	Sd15 = 47	Sd15 = 48	Sd15 = 48	Sd15 = 47	Sd15 = 48	Sd15 = 47	Sd15 = 48
Sd16 = 59	Sd16 = 59	Sd16 = 59	Sd16 = 59	Sd16 = 59	Sd16 = 59	Sd16 = 59	Sd16 = 59
dev0 = 0	dev0 = 0	dev0 = 0	dev0 = 0	dev0 = 0	dev0 = 0	dev0 = 0	dev0 = 0
dev1 = 0	dev1 = 0	dev1 = 0	dev1 = 0	dev1 = 0	dev1 = 0	dev1 = 0	dev1 = 0
dev2 = 0	dev2 = 0	dev2 = 0	dev2 = 0	dev2 = 0	dev2 = 0	dev2 = 0	dev2 = 0
dev3 = 0	dev3 = 0	dev3 = 0	dev3 = 0	dev3 = 0	dev3 = 0	dev3 = 0	dev3 = 0
dev4 = 0	dev4 = 0	dev4 = 0	dev4 = 0	dev4 = 0	dev4 = 0	dev4 = 0	dev4 = 0
dev5 = 0	dev5 = 0	dev5 = 0	dev5 = 0	dev5 = 0	dev5 = 0	dev5 = 0	dev5 = 0
dev6 = 0	dev6 = 0	dev6 = 1	dev6 = 1	dev6 = 0	dev6 = 1	dev6 = 0	dev6 = 1
dev7 = 0	dev7 = 0	dev7 = 0	dev7 = 0	dev7 = 0	dev7 = 0	dev7 = 0	dev7 = 0
dev8 = 0	dev8 = 0	dev8 = 0	dev8 = 0	dev8 = 0	dev8 = 0	dev8 = 0	dev8 = 0
dev9 = 0	dev9 = 0	dev9 = 0	dev9 = 0	dev9 = 0	dev9 = 0	dev9 = 0	dev9 = 0
dev10 = 0	dev10 = 0	dev10 = 0	dev10 = 0	dev10 = 0	dev10 = 0	dev10 = 0	dev10 = 0
dev11 = 0	dev11 = 0	dev11 = 0	dev11 = 0	dev11 = 0	dev11 = 0	dev11 = 0	dev11 = 0
dev12 = 0	dev12 = 0	dev12 = 0	dev12 = 0	dev12 = 0	dev12 = 0	dev12 = 0	dev12 = 0
dev13 = 0	dev13 = 0	dev13 = 0	dev13 = 0	dev13 = 0	dev13 = 0	dev13 = 0	dev13 = 0
dev14 = 0	dev14 = 0	dev14 = 1	dev14 = 1	dev14 = 0	dev14 = 1	dev14 = 0	dev14 = 1
dev15 = 0	dev15 = 0	dev15 = 1	dev15 = 1	dev15 = 0	dev15 = 1	dev15 = 0	dev15 = 1
dev16 = 0	dev16 = 0	dev16 = 0	dev16 = 0	dev16 = 0	dev16 = 0	dev16 = 0	dev16 = 0
total = 0	total = 0	total = 3	total = 3	total = 0	total = 3	total = 0	total = 3

Table A.4. (Continued)

f=9	f=10	f=11	f=12	f=13	f=14	f=15	f=16
Sd0 = 0	Sd0 = 0	Sd0 = 0	Sd0 = 0	Sd0 = 0	Sd0 = 0	Sd0 = 0	Sd0 = 0
Sd1 = 0	Sd1 = 0	Sd1 = 0	Sd1 = 0	Sd1 = 0	Sd1 = 0	Sd1 = 0	Sd1 = 0
Sd2 = 8	Sd2 = 8	Sd2 = 8	Sd2 = 8	Sd2 = 8	Sd2 = 8	Sd2 = 8	Sd2 = 8
Sd3 = 8	Sd3 = 8	Sd3 = 8	Sd3 = 8	Sd3 = 8	Sd3 = 8	Sd3 = 8	Sd3 = 8
Sd4 = 16	Sd4 = 16	Sd4 = 16	Sd4 = 16	Sd4 = 16	Sd4 = 16	Sd4 = 16	Sd4 = 16
Sd5 = 12	Sd5 = 12	Sd5 = 12	Sd5 = 12	Sd5 = 12	Sd5 = 12	Sd5 = 12	Sd5 = 12
Sd6 = 43	Sd6 = 43	Sd6 = 44	Sd6 = 44	Sd6 = 43	Sd6 = 44	Sd6 = 43	Sd6 = 44
Sd7 = 28	Sd7 = 28	Sd7 = 28	Sd7 = 28	Sd7 = 28	Sd7 = 28	Sd7 = 28	Sd7 = 28
Sd8 = 16	Sd8 = 16	Sd8 = 16	Sd8 = 16	Sd8 = 16	Sd8 = 16	Sd8 = 16	Sd8 = 16
Sd9 = 32	Sd9 = 32	Sd9 = 32	Sd9 = 32	Sd9 = 32	Sd9 = 32	Sd9 = 32	Sd9 = 32
Sd10 = 30	Sd10 = 30	Sd10 = 30	Sd10 = 30	Sd10 = 30	Sd10 = 30	Sd10 = 30	Sd10 = 30
Sd11 = 50	Sd11 = 50	Sd11 = 50	Sd11 = 50	Sd11 = 50	Sd11 = 50	Sd11 = 50	Sd11 = 50
Sd12 = 55	Sd12 = 55	Sd12 = 55	Sd12 = 55	Sd12 = 55	Sd12 = 55	Sd12 = 55	Sd12 = 55
Sd13 = 40	Sd13 = 40	Sd13 = 40	Sd13 = 40	Sd13 = 40	Sd13 = 40	Sd13 = 40	Sd13 = 40
Sd14 = 43	Sd14 = 43	Sd14 = 44	Sd14 = 44	Sd14 = 43	Sd14 = 44	Sd14 = 43	Sd14 = 44
Sd15 = 48	Sd15 = 48	Sd15 = 49	Sd15 = 49	Sd15 = 48	Sd15 = 49	Sd15 = 48	Sd15 = 49
Sd16 = 59	Sd16 = 59	Sd16 = 59	Sd16 = 59	Sd16 = 59	Sd16 = 59	Sd16 = 59	Sd16 = 59
dev0 = 0	dev0 = 0	dev0 = 0	dev0 = 0	dev0 = 0	dev0 = 0	dev0 = 0	dev0 = 0
dev1 = 0	dev1 = 0	dev1 = 0	dev1 = 0	dev1 = 0	dev1 = 0	dev1 = 0	dev1 = 0
dev2 = 0	dev2 = 0	dev2 = 0	dev2 = 0	dev2 = 0	dev2 = 0	dev2 = 0	dev2 = 0
dev3 = 0	dev3 = 0	dev3 = 0	dev3 = 0	dev3 = 0	dev3 = 0	dev3 = 0	dev3 = 0
dev4 = 0	dev4 = 0	dev4 = 0	dev4 = 0	dev4 = 0	dev4 = 0	dev4 = 0	dev4 = 0
dev5 = 0	dev5 = 0	dev5 = 0	dev5 = 0	dev5 = 0	dev5 = 0	dev5 = 0	dev5 = 0
dev6 = 0	dev6 = 0	dev6 = 1	dev6 = 1	dev6 = 0	dev6 = 1	dev6 = 0	dev6 = 1
dev7 = 0	dev7 = 0	dev7 = 0	dev7 = 0	dev7 = 0	dev7 = 0	dev7 = 0	dev7 = 0
dev8 = 0	dev8 = 0	dev8 = 0	dev8 = 0	dev8 = 0	dev8 = 0	dev8 = 0	dev8 = 0
dev9 = 0	dev9 = 0	dev9 = 0	dev9 = 0	dev9 = 0	dev9 = 0	dev9 = 0	dev9 = 0
dev10 = 0	dev10 = 0	dev10 = 0	dev10 = 0	dev10 = 0	dev10 = 0	dev10 = 0	dev10 = 0
dev11 = 0	dev11 = 0	dev11 = 0	dev11 = 0	dev11 = 0	dev11 = 0	dev11 = 0	dev11 = 0
dev12 = 0	dev12 = 0	dev12 = 0	dev12 = 0	dev12 = 0	dev12 = 0	dev12 = 0	dev12 = 0
dev13 = 0	dev13 = 0	dev13 = 0	dev13 = 0	dev13 = 0	dev13 = 0	dev13 = 0	dev13 = 0
dev14 = 0	dev14 = 0	dev14 = 1	dev14 = 1	dev14 = 0	dev14 = 1	dev14 = 0	dev14 = 1
dev15 = 1	dev15 = 1	dev15 = 2	dev15 = 2	dev15 = 1	dev15 = 2	dev15 = 1	dev15 = 2
dev16 = 0	dev16 = 0	dev16 = 0	dev16 = 0	dev16 = 0	dev16 = 0	dev16 = 0	dev16 = 0
total = 1	total = 1	total = 4	total = 4	total = 1	total = 4	total = 1	total = 4

APPENDIX B

SOLUTION ALGORITHMS

SGS – forward recursion procedure

Initialization: All $S_i = 0$; EC={0}, q = 0, $C_0 = 0$

For i = 1 to n do

Select the activity $j = o_j$

Set U=U\{j}

For all t Є EC do

For k=1…, K do

Compute R_k

$ST_j = Max_{i \in pij}\{ST_i + d_i\}$

Let q be the index with $ST_j = C_q$

While ($\forall$ t$\in \lfloor ST_j, ST_j + d_i \rfloor$, $\forall$ k = 1,..., K, $R_{kt} \geq r_j$) is False do

q := q+1

$ST_j = C_q$

End (While)

C_j^e := $ST_j + d_j$

EC = EC $\cup$ { C_j^e }, Update C

End for i

SGS – backward recursion procedure

Initialization: ES={C_{max}}, q = 1, $ST_1 = C_{max}$

For i = n down to 1 do

    j = u_j

    Set U=U\{j}

    For all t Є ES do

        For k=1…, K do

            Compute R_k

    C_j^1 = Min $h_{i \in Succj}$\{ ST_h^1 \}

    Let q be the index with C_j^1 = ST_q

    While ($\forall$ t $\in \left[ST_j, C_j^1 \right]$, $\forall$ k = 1,..., K, $R_{kt} \geq r_j$) is False do

      q := q+1

      C_j^1 := ST_q

    End (While)

    ST_j^1 := C_j^1 - d_j

    ES = ES $\cup$ \{ST_j\}, Update ST

End for i

Extra Notation for algorithms for finding the best schedules for measures

win_{down} : start of disruption time window

win_{up} : end of disruption time window

h : number of activities that corresponds to the time period which disruption may occur.

D_f : Activities in time period, f = 1,…,h

d_ratio : % increase in the duration of activity for the part remained in dis. Window

d_part_i : Part of activity i remained in disruption window

dd_i : new durations of activities in case of disruption, i = 1,…,h

Sd_i : Starting times of activities after disruption, i = 1,…,n

ss : number subsets for possible disruption probabilities

SS_{ss} : total amount of right shift after disruptions for each subset

Dev_{qi} : deviation at start time of activity i in scenario q

P_q : probability of each scenario to occur

$Devtot_q$: total deviation at start time of activities in scenario q

Obj_metd : objective function value calculated for starting schedule

Algorithm for finding the best schedule with ETD policy

Initialization: h=0, $dd_i = d_i$ (for all i=1,…,n),

Repeat SGS

For i = 1 to n do

If ($win_{down} \leq ST_i < win_{up}$)

h=h+1,

$D_h = i$,

if ($C_i^e \leq win_{up}$)

$d\_part_i = d_i$

else

$d\_part_i = win_{up} - ST_j$;

$dd_i = d_i + (d\_part \times d\_ratio)$

End (if)

If ($win_{down} < C_i^e \leq win_{up}$)

$h=h+1$,

$D_h = i$,

$d\_part = C_i^e - win_{down}$

$dd_i = d_i + (d\_part \times d\_ratio)$

End (if)

If ($win_{down} \geq ST_i$ & $C_i^e \geq win_{up}$)

$h=h+1$,

$D_h = i$,

$d\_part = win_{up} - win_{down}$

$dd_i = d_i + (d\_part \times d\_ratio)$

End (if)

$ss = 2^h$

For q = 1 to ss do (for each q take one subset of all possible disruptions)

Repeat SGS (with dd_i values instead of d_i for D_h in subset q)

$Dev_{qi} = STd_i - ST_i$, i = 1,…,n (activities can not start before their original start times after disruption)

$$devtot_q = \sum_{i=1}^{An} dev_{qi}$$

$$obj\_ETD = \frac{\sum_{q=1}^{ss} (P_q * devtot_q)}{\sum_{q=1}^{ss} P_q}$$

Algorithm for finding the best schedule with TD schedule

Initialization: h=0, $dd_i = d_i$ (for all i=1,…,n),

Repeat SGS

For i = 1 to n do

If ($win_{down} \le ST_i < win_{up}$)

h=h+1,

$D_h = i$,

if ($C_i^e \le win_{up}$)

$d\_part_i = d_i$

else

$d\_part_i = win_{up} - ST_j$;

$dd_i = d_i + (d\_part \times d\_ratio)$

End (if)

If ($win_{down} < C_i^e \le win_{up}$)

h=h+1,

$D_h = i$,

$d\_part = C_i^e - win_{down}$

$dd_i = d_i + (d\_part \times d\_ratio)$

End (if)

If ($win_{down} \ge ST_i$ & $C_i^e \ge win_{up}$)

h=h+1,

$D_h = i$,

$d\_part = win_{up} - win_{down}$

$dd_i = d_i + (d\_part \times d\_ratio)$

End (if)

Repeat SGS (with dd_i values instead of d_i for all D_h)

$dev_{0i} = STd_i - S_i$, i = 1,…,n (activities can not start before their original start times after disruption)

$$\text{obj\_TD} = \sum_{i=1}^{n} dev_{0i}$$

Algorithm for finding the best schedule with RS policy

Initialization: h=0, $dd_i = d_i$ (for all i=1,…,n),

Repeat SGS

For i = 1 to n do

For all activity in window

$dd_i = d_i + (win_{up} - win_{down})$

Repeat SGS (with dd_i values instead of d_i for all D_h)

$dev_{0i} = STd_i - ST_i$, i = 1,…,n (activities can not start before their original start times after disruption)

$$\text{obj\_RS} = \sum_{i=1}^{n} dev_{0i}$$

APPENDIX C

TABLES FOR FINE TUNING ALGORITHMS

Table C.1. Results of Fine Tuning run for problem 1

TLS	$Iter_{max}$	Obj_TD	Obj_ETD
1	50	8	6,20
	100	8	6,20
	200	8	6,20
5	50	8	6,20
	100	8	6,20
	200	8	6,20
7	50	8	6,20
	100	8	6,20
	200	8	6,20
10	50	8	6,20
	100	8	6,20
	200	8	6,20
15	50	8	6,20
	100	8	6,20
	200	8	6,20

Table C.2. Results of Fine Tuning run for problem 2

TLS	$Iter_{max}$	Obj_TD	Obj_ETD
1	50	9	6,64
	100	9	6,64
	200	9	6,64
5	50	9	6,64
	100	9	6,64
	200	9	6,64
7	50	9	6,64
	100	9	6,64
	200	9	6,64
10	50	9	6,64
	100	9	6,64
	200	9	6,64
15	50	9	6,64
	100	9	6,64
	200	9	6,64

Table C.3. Results of Fine Tuning run for problem 3

TLS	$Iter_{max}$	Obj_TD	Obj_ETD
1	50	10	6,64
	100	10	6,64
	200	10	6,64
5	50	10	6,64
	100	10	6,64
	200	10	6,64
7	50	10	6,64
	100	10	6,64
	200	10	6,64
10	50	10	6,64
	100	10	6,64
	200	10	6,64
15	50	10	6,64
	100	10	6,64
	200	10	6,64

Table C.4. Results of Fine Tuning run for problem 4

TLS	$Iter_{max}$	Obj_TD	Obj_ETD
1	50	15	10,83
	100	15	10,83
	200	15	10,83
5	50	15	10,83
	100	15	10,83
	200	15	10,83
7	50	15	10,83
	100	15	10,83
	200	15	10,83
10	50	15	10,83
	100	15	10,83
	200	15	10,83
15	50	15	10,83
	100	15	10,83
	200	15	10,83

Table C.5. Results of Fine Tuning run for problem 5

TLS	$Iter_{max}$	Obj_TD	Obj_ETD
1	50	12	9,19
	100	12	9,19
	200	12	9,19
5	50	12	9,19
	100	12	9,19
	200	12	9,19
7	50	12	9,19
	100	12	9,19
	200	12	9,19
10	50	12	9,19
	100	12	9,19
	200	12	9,19
15	50	12	9,19
	100	12	9,19
	200	12	9,19

APPENDIX D

RESIDUAL VS FACTOR LEVEL GRAPHS

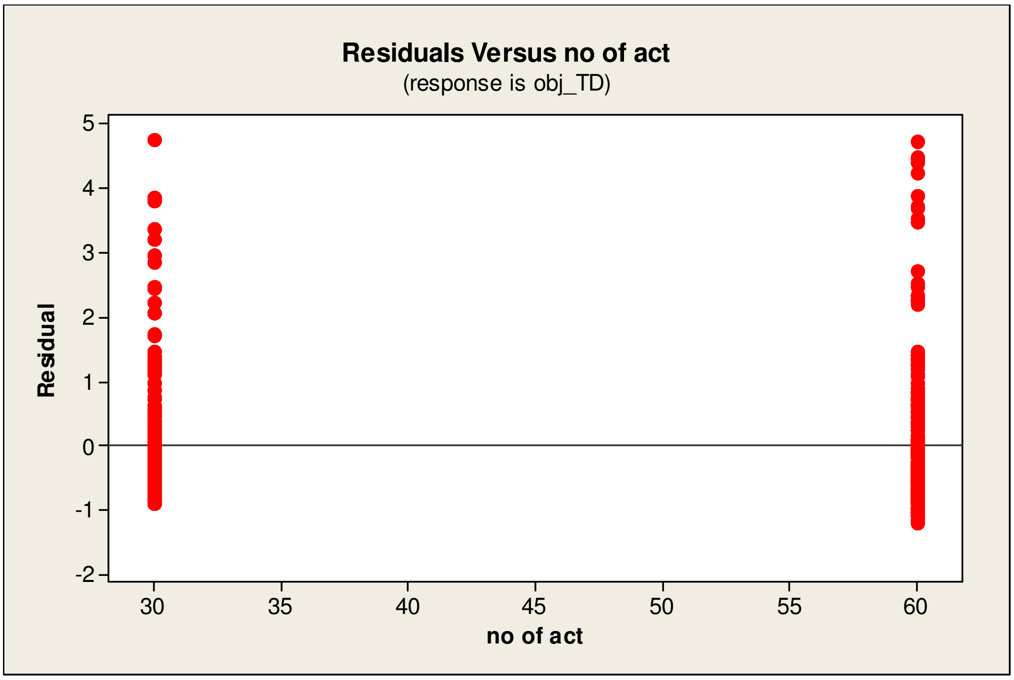

Figure D.1. Residuals vs. # of Activities for obj_TD response

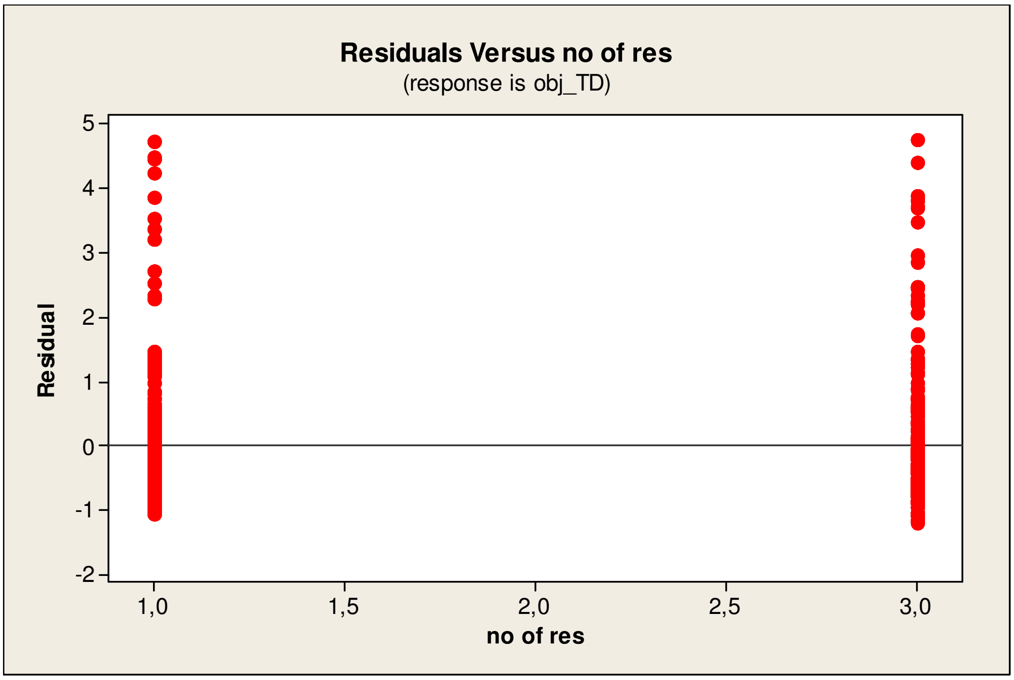

Figure D.2. Residuals vs. # of Resources for obj_TD response

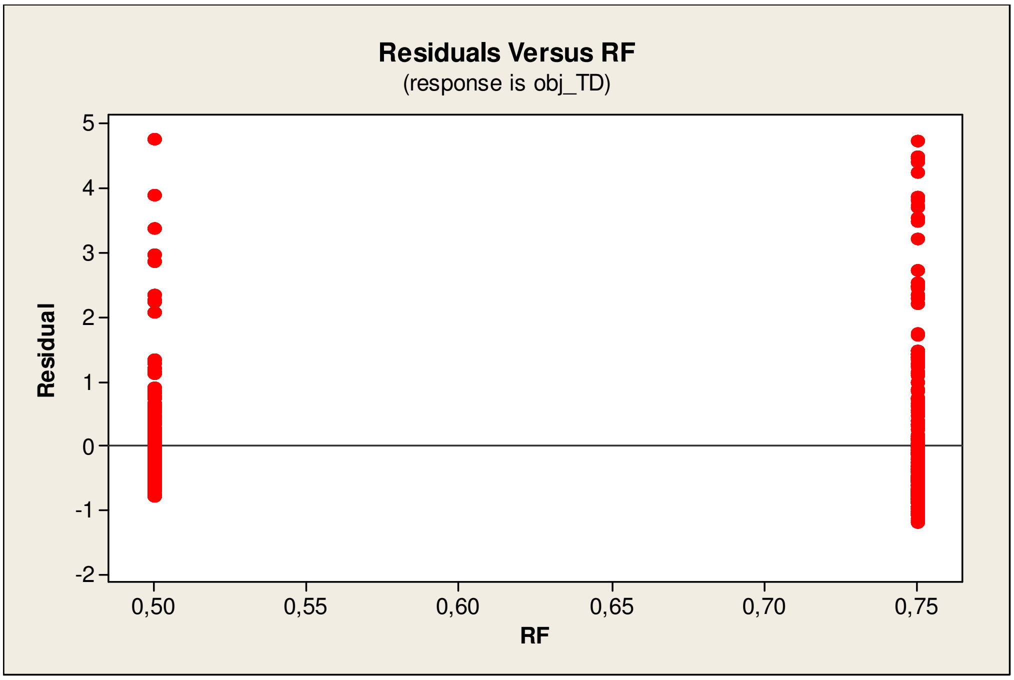

Figure D.3. Residuals vs. Resource Factor for obj_TD response

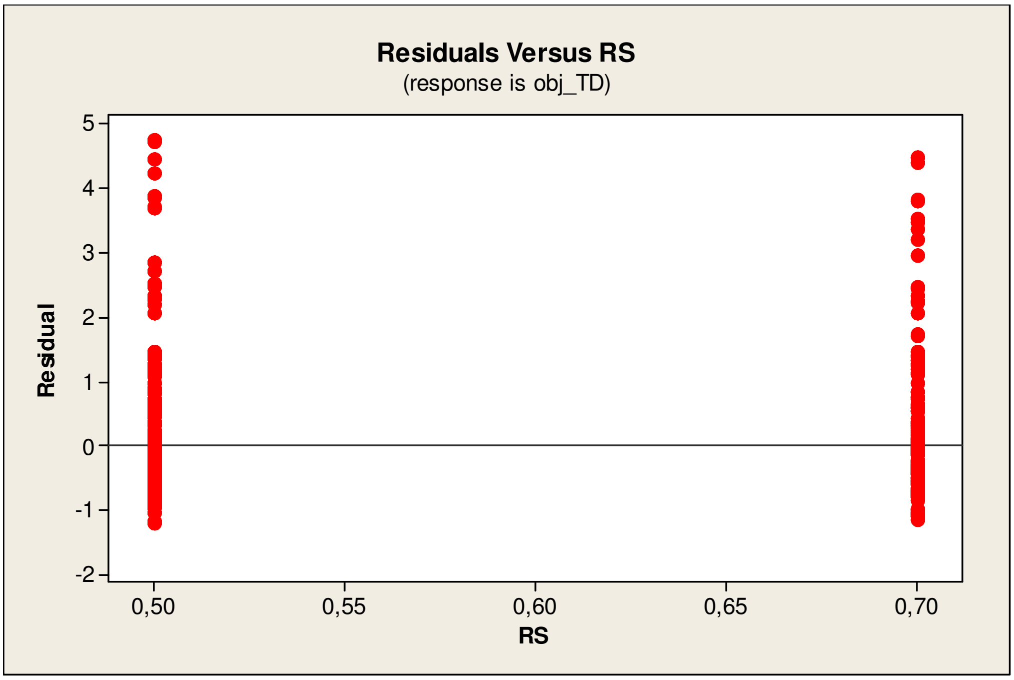

Figure D.4. Residuals vs. Resource Strength for obj_TD response

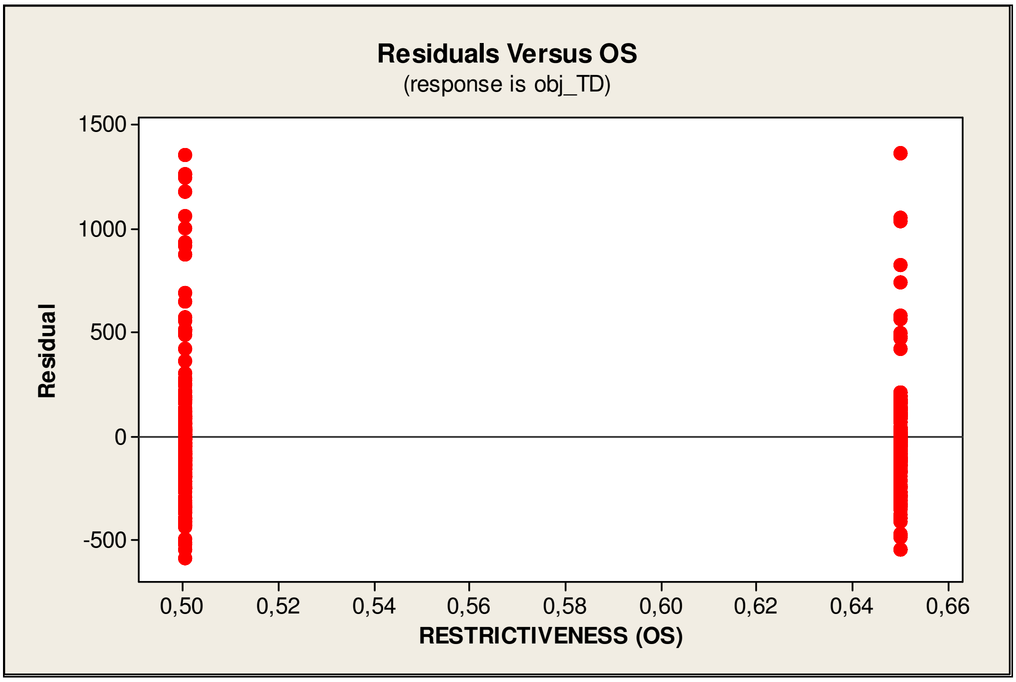

Figure D.5. Residuals vs. Restrictiveness for obj_TD response

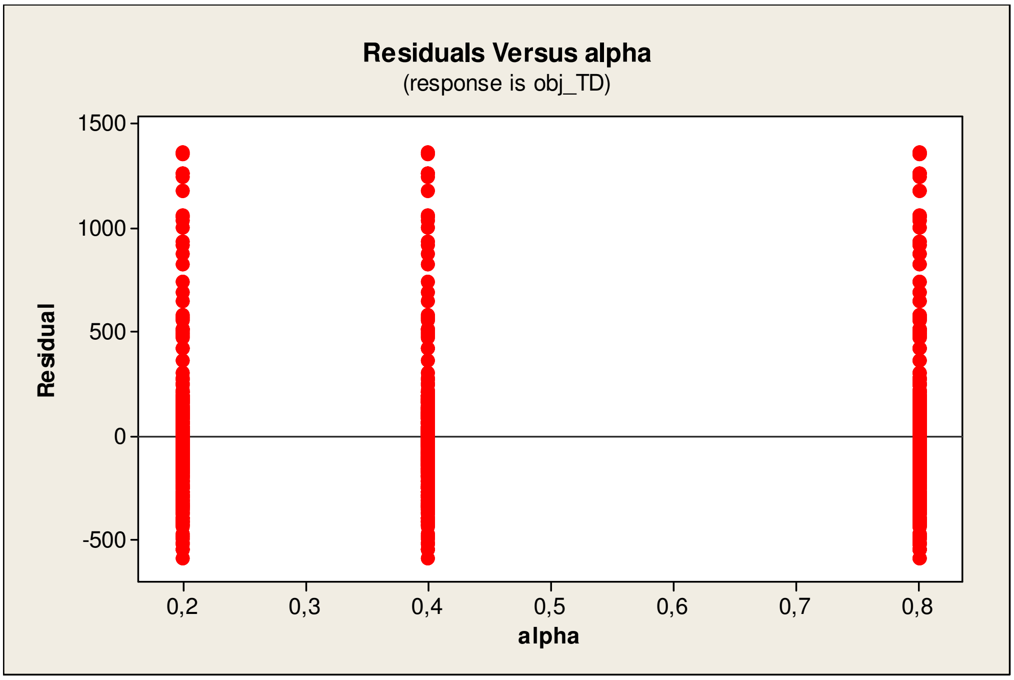

Figure D.6. Residuals vs. Alpha for obj_TD response

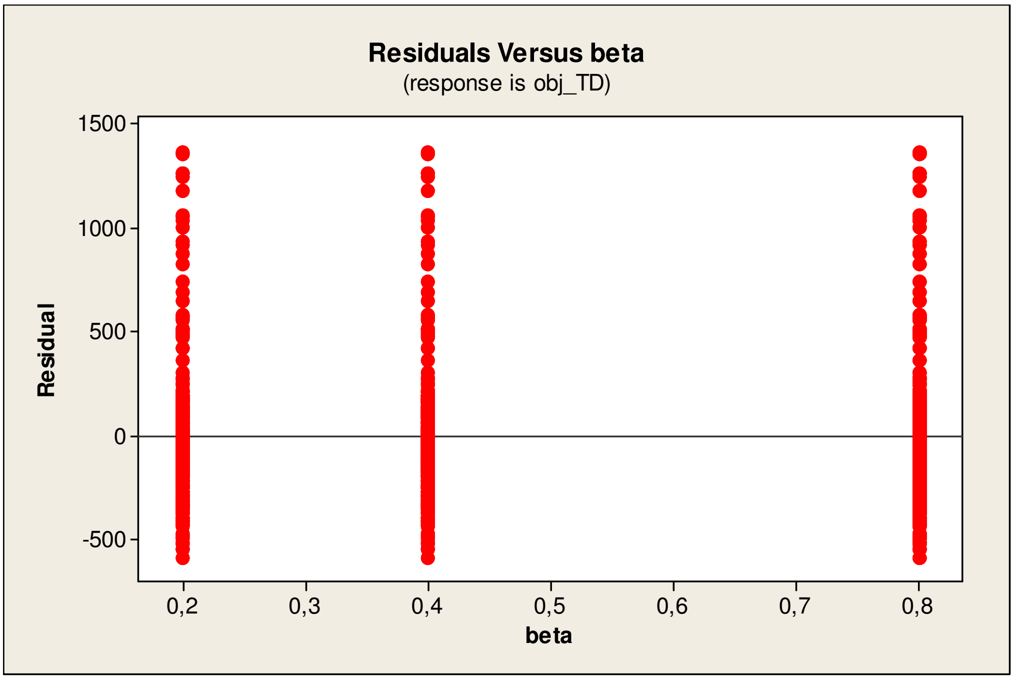

Figure D.7. Residuals vs. Beta for obj_TD response

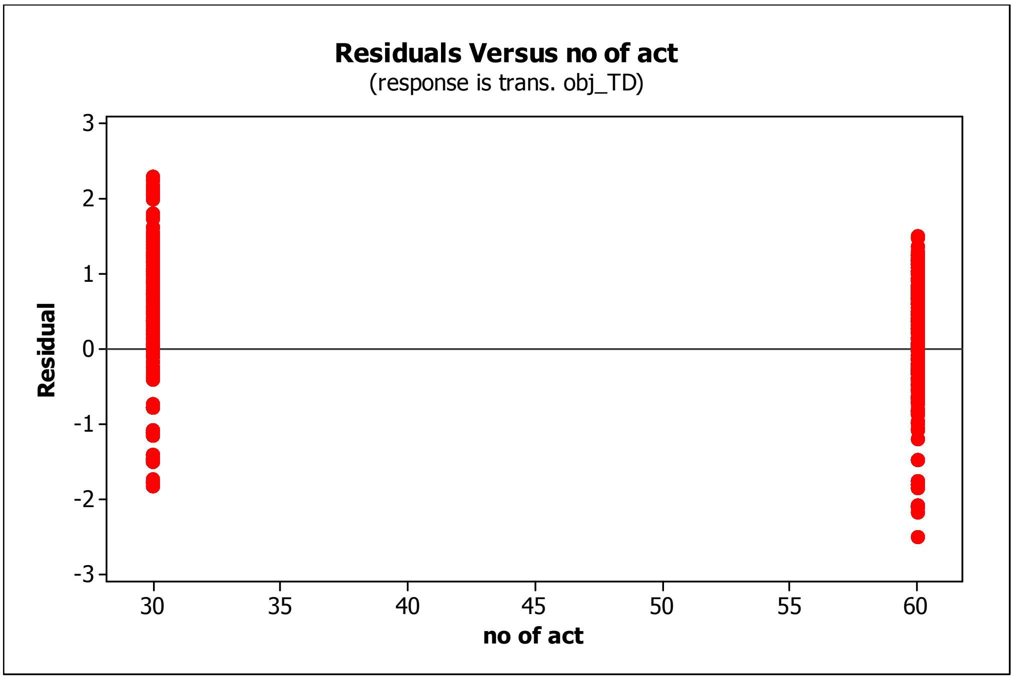

Figure D.8. Residuals vs. # of Activities for Transformed obj_TD response

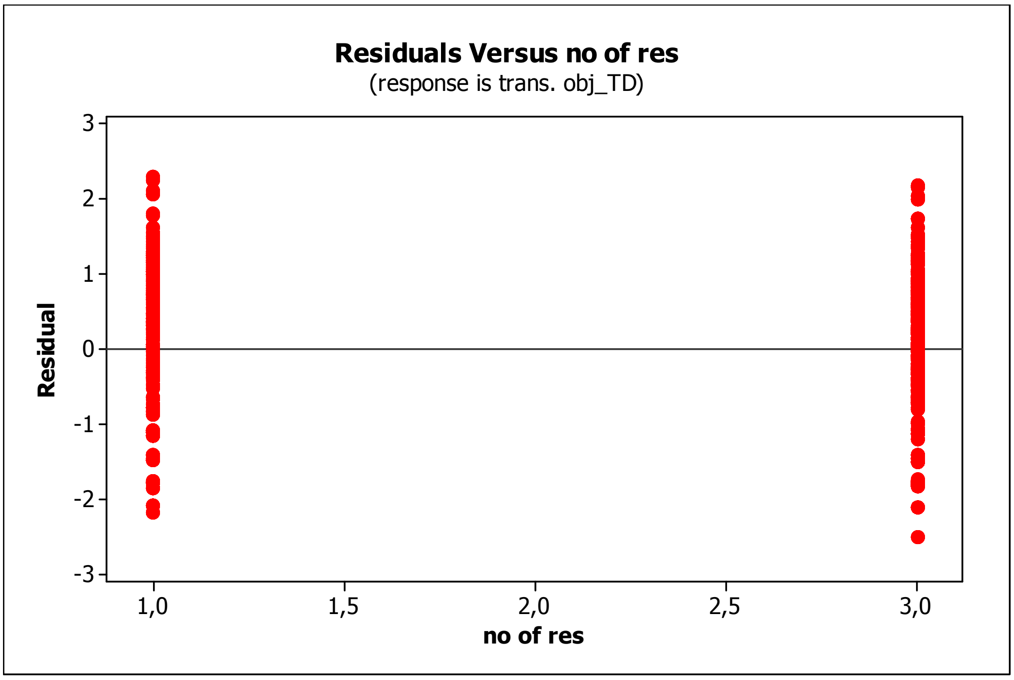

Figure D.9. Residuals vs. # of Resources for Transformed obj_TD response

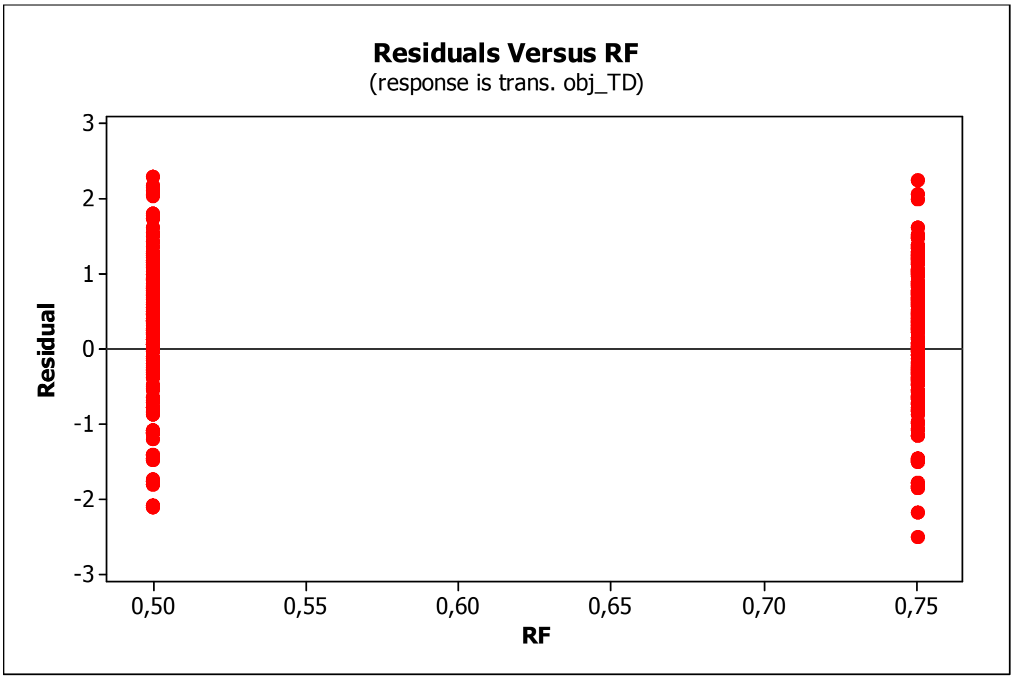

Figure D.10. Residuals vs. Resource Factor for Transformed obj_TD response

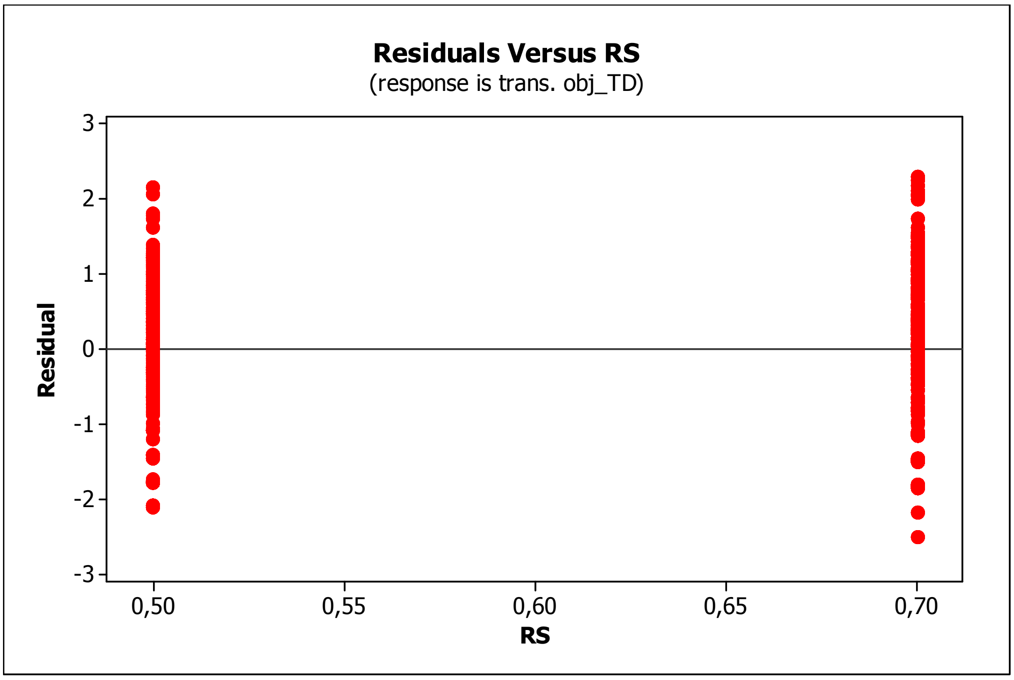

Figure D.11. Residuals vs. Resource Strength for Transformed obj_TD response

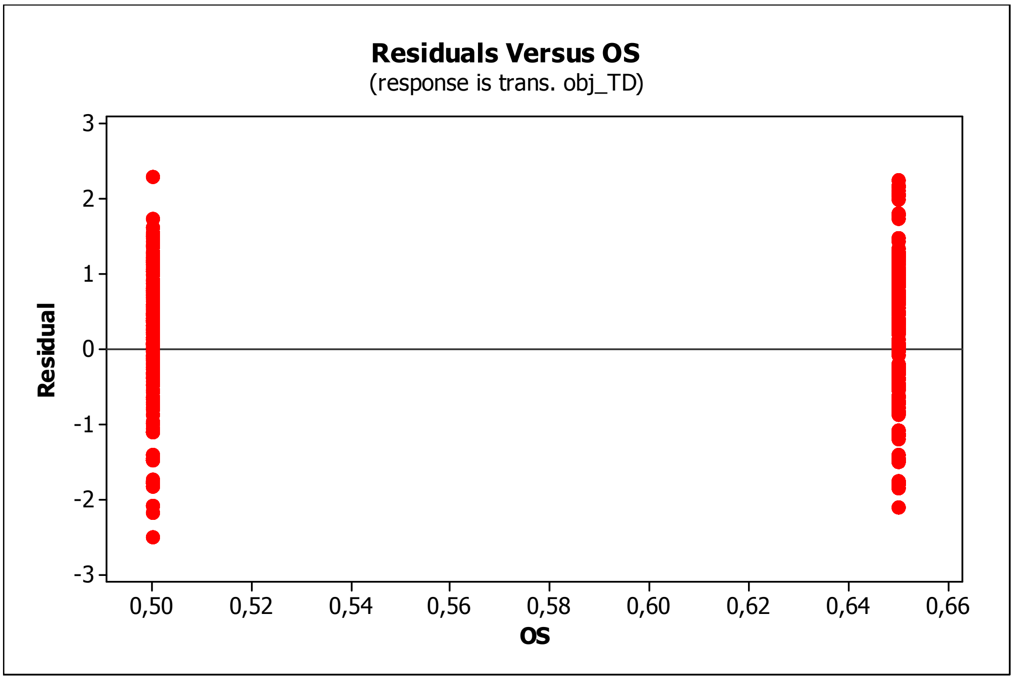

Figure D.12. Residuals vs. Restrictiveness for Transformed obj_TD response

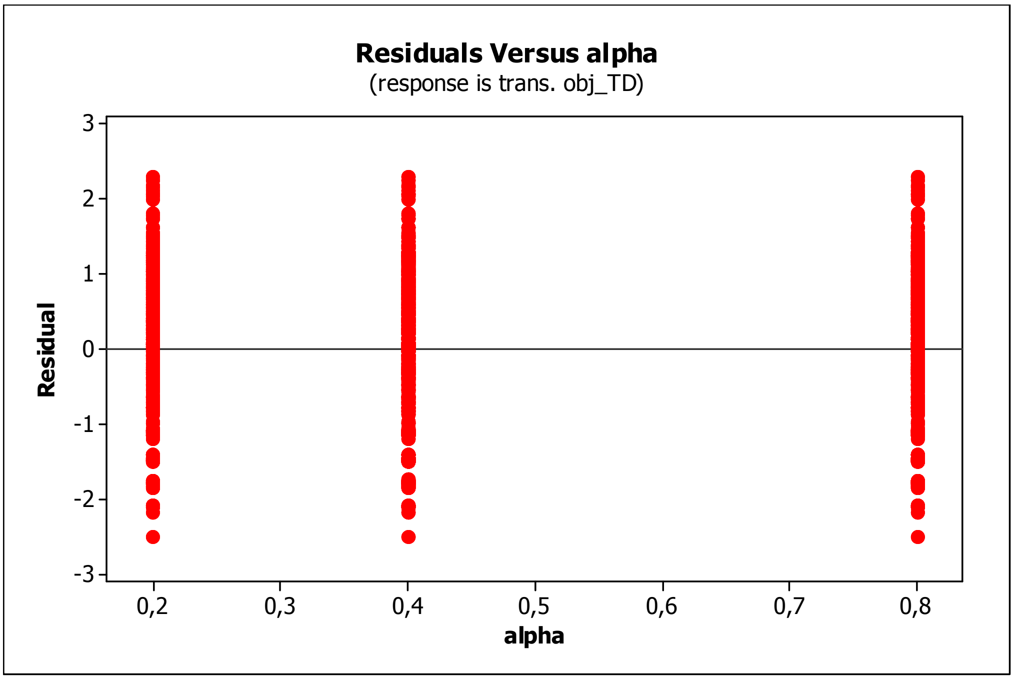

Figure D.13. Residuals vs. Alpha for Transformed obj_TD response

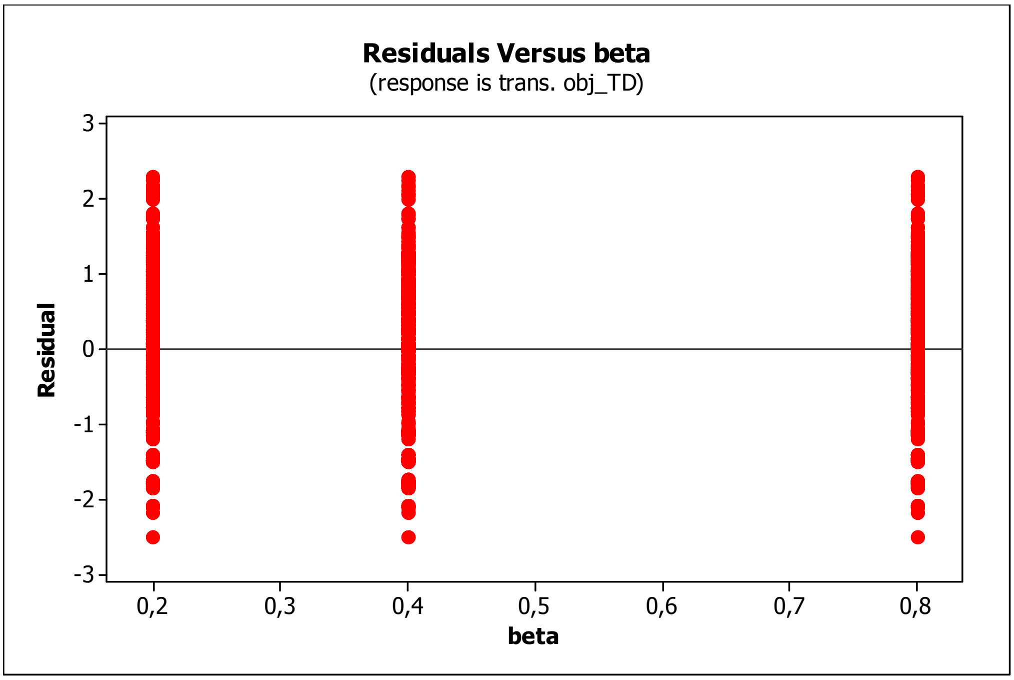

Figure D.14. Residuals vs. Beta for Transformed obj_TD response

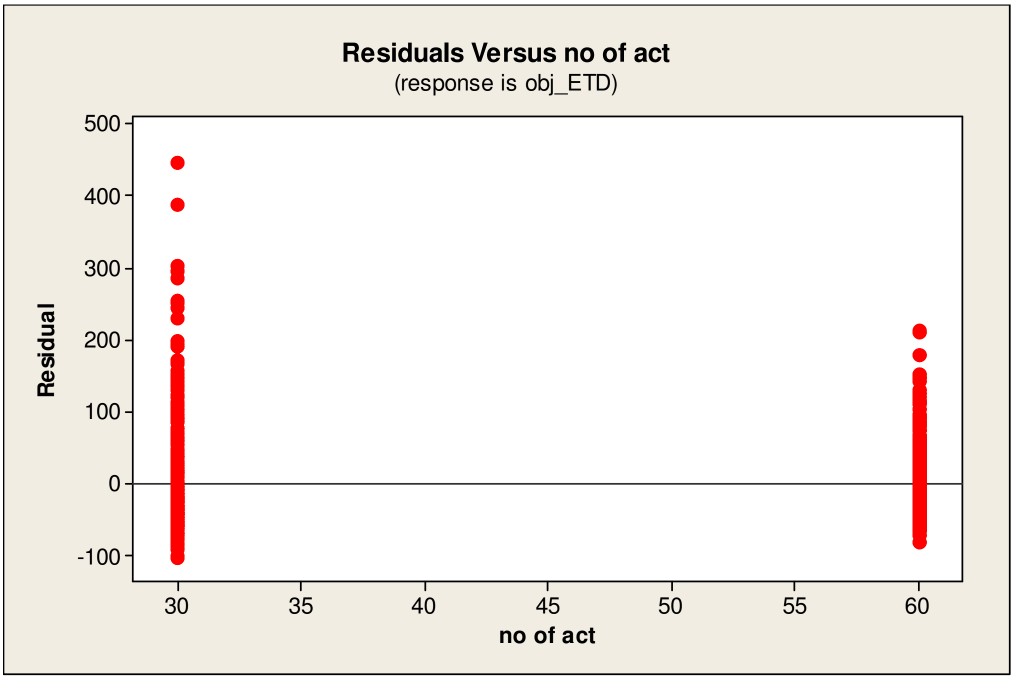

Figure D.15. Residuals vs. # of Activities for obj_ETD response

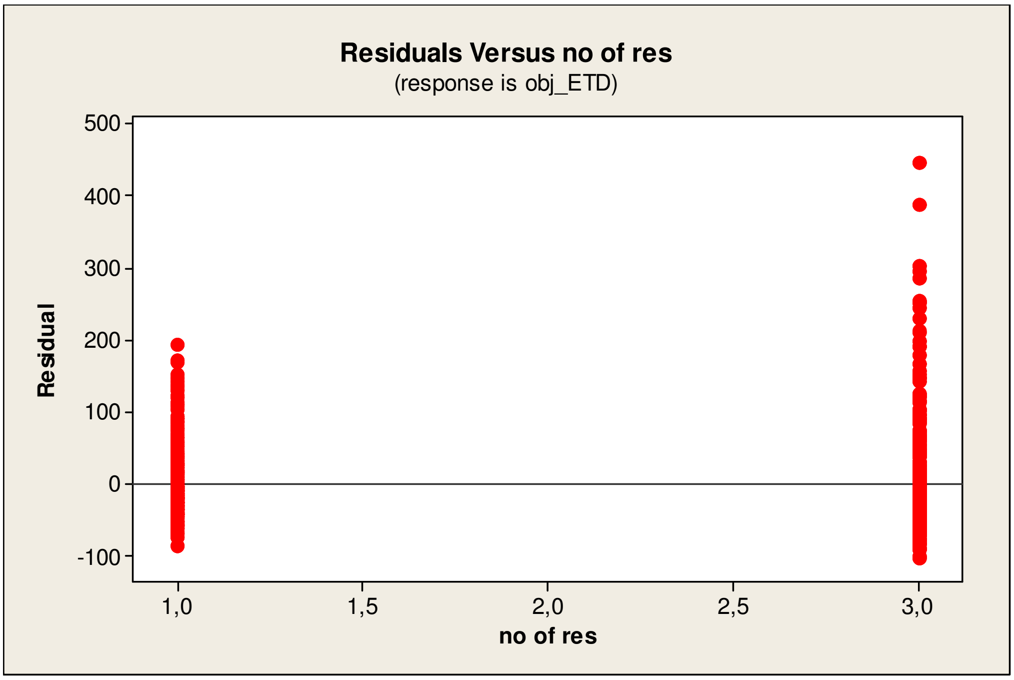

Figure D.16. Residuals vs. # of Resources for obj_ETD response

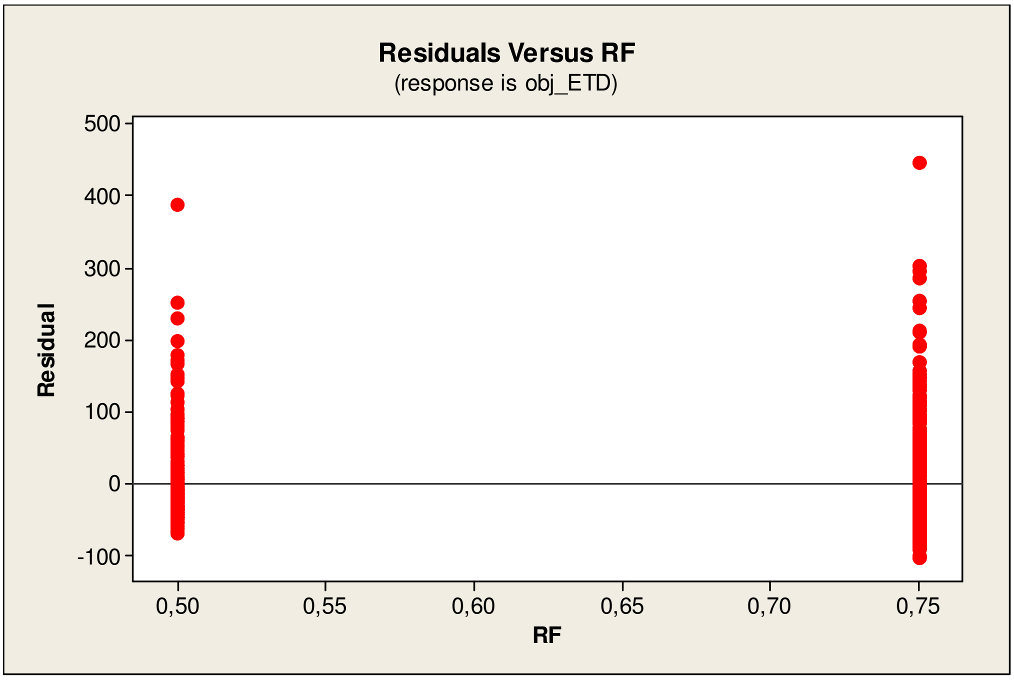

Figure D.17. Residuals vs. Resource Factor for obj_ETD response

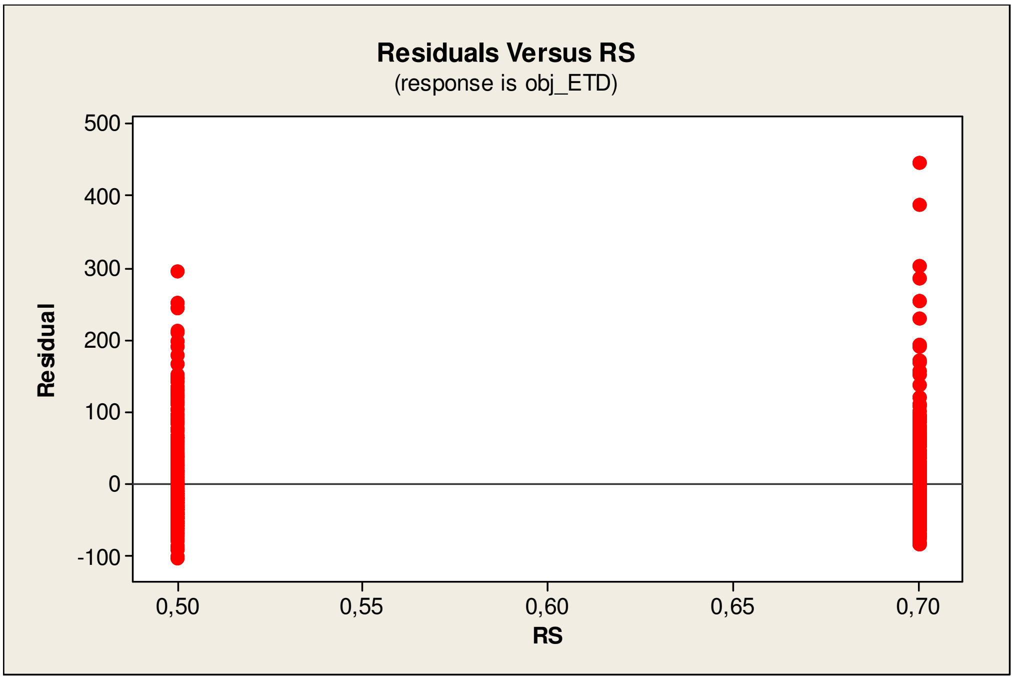

Figure D.18. Residuals vs. Resource Strength for obj_ETD response

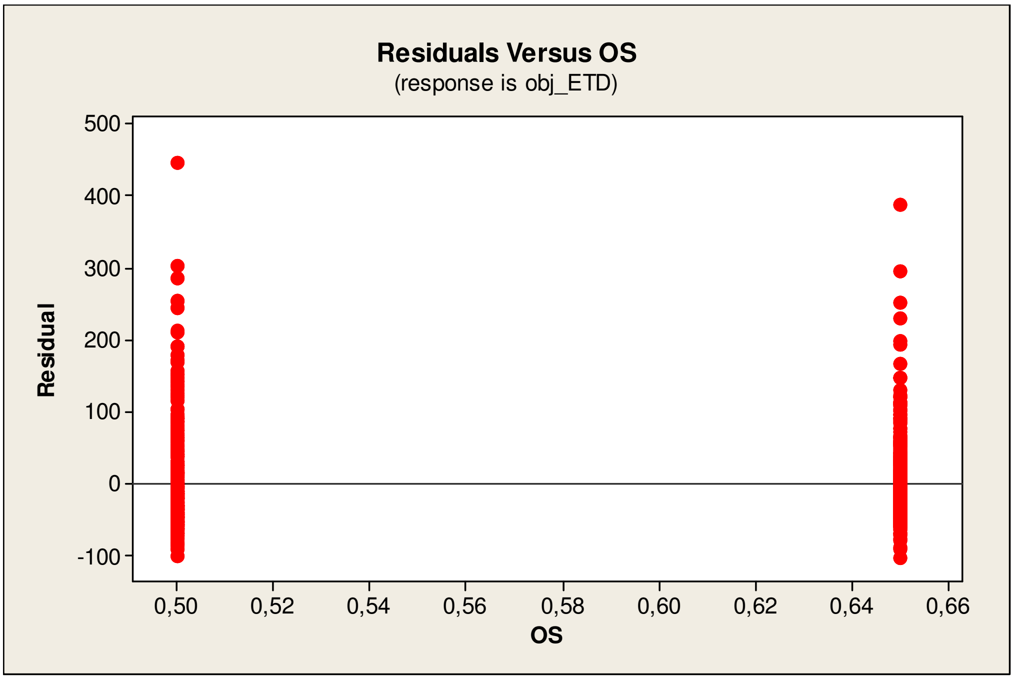

Figure D.19. Residuals vs. Restrictiveness for obj_ETD response

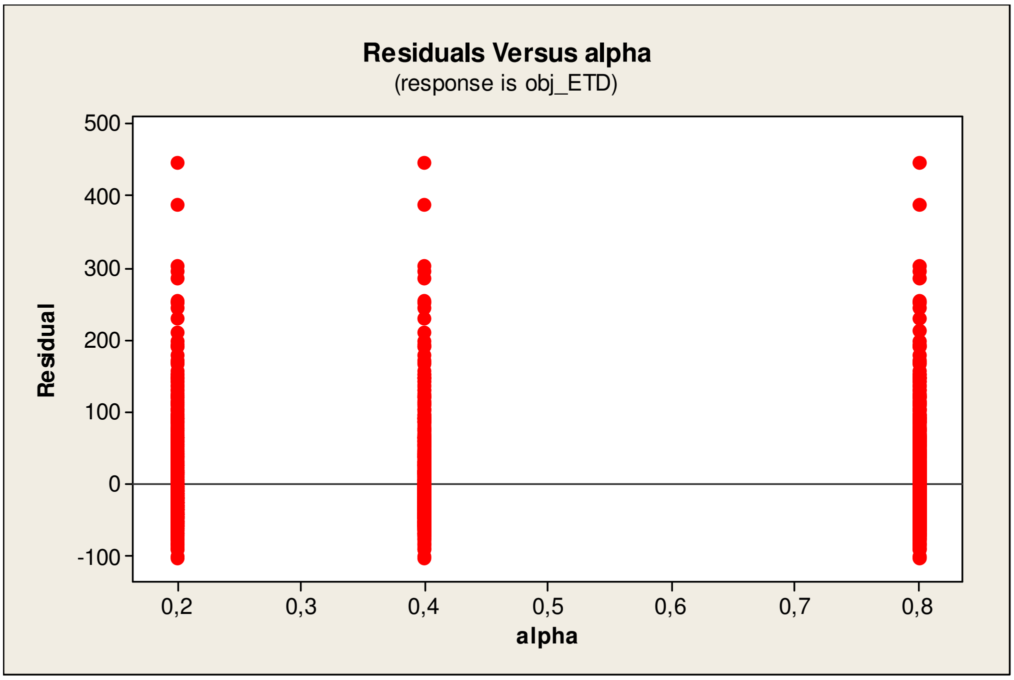

Figure D.20. Residuals vs. Alpha for obj_ETD response

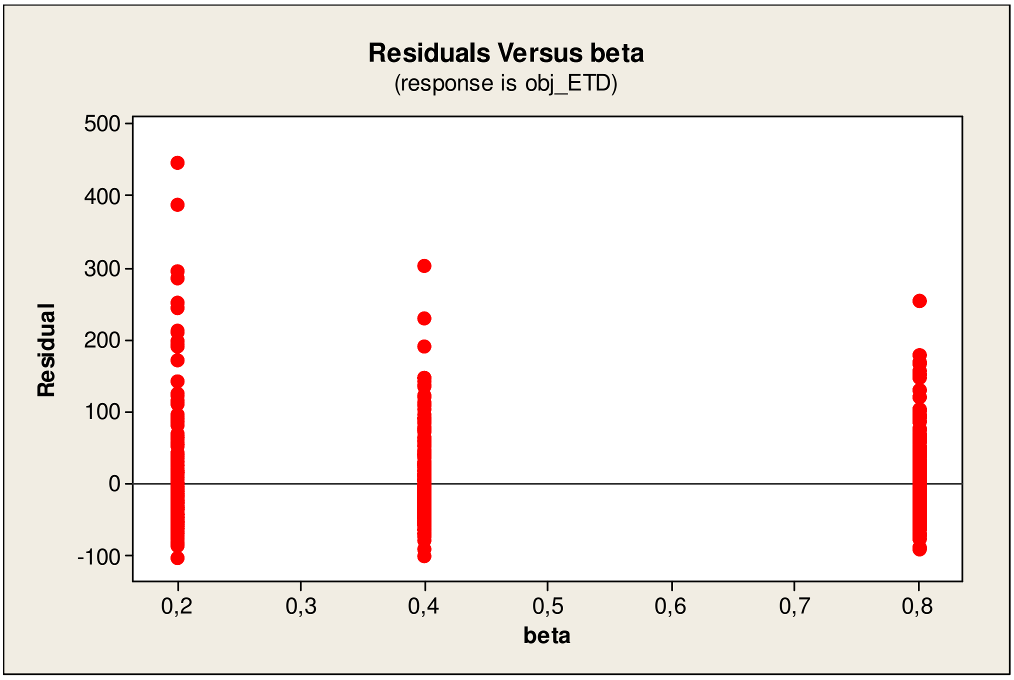

Figure D.21. Residuals vs. Beta for obj_ETD response

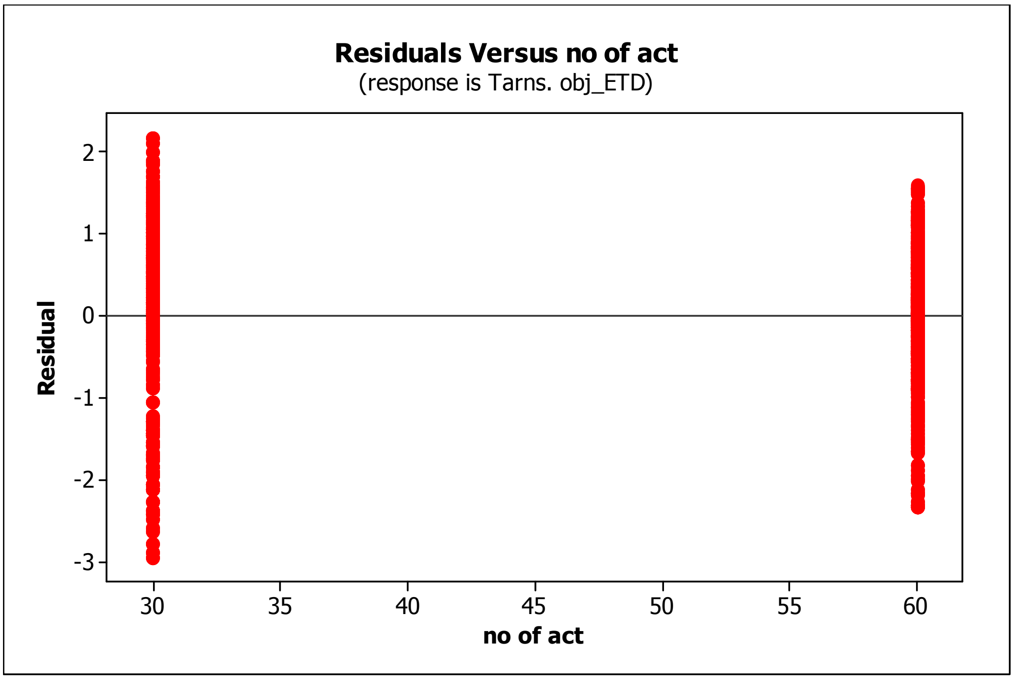

Figure D.22. Residuals vs. # of Activities for Transformed obj_ETD response

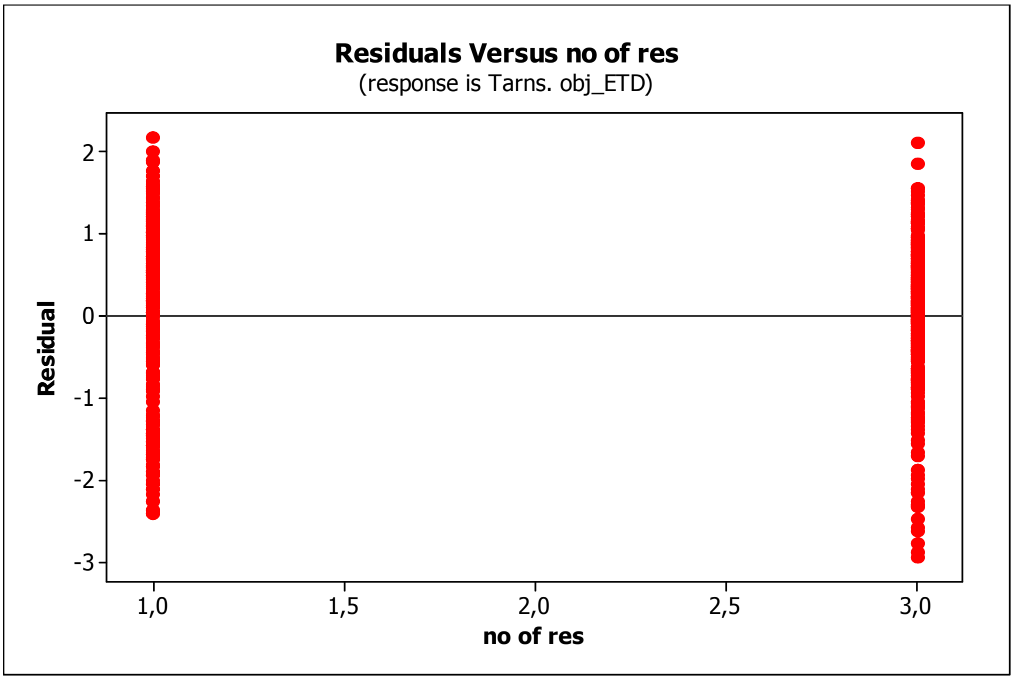

Figure D.23. Residuals vs. # of Resources for Transformed obj_ETD response

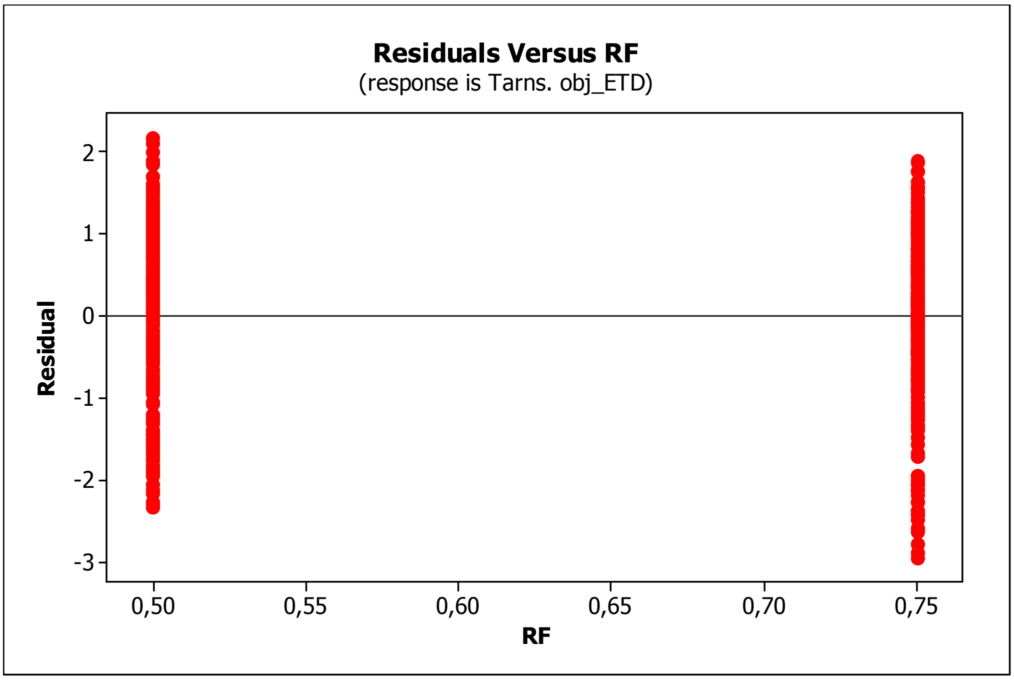

Figure D.24. Residuals vs. Resource Factor for Transformed obj_ETD response

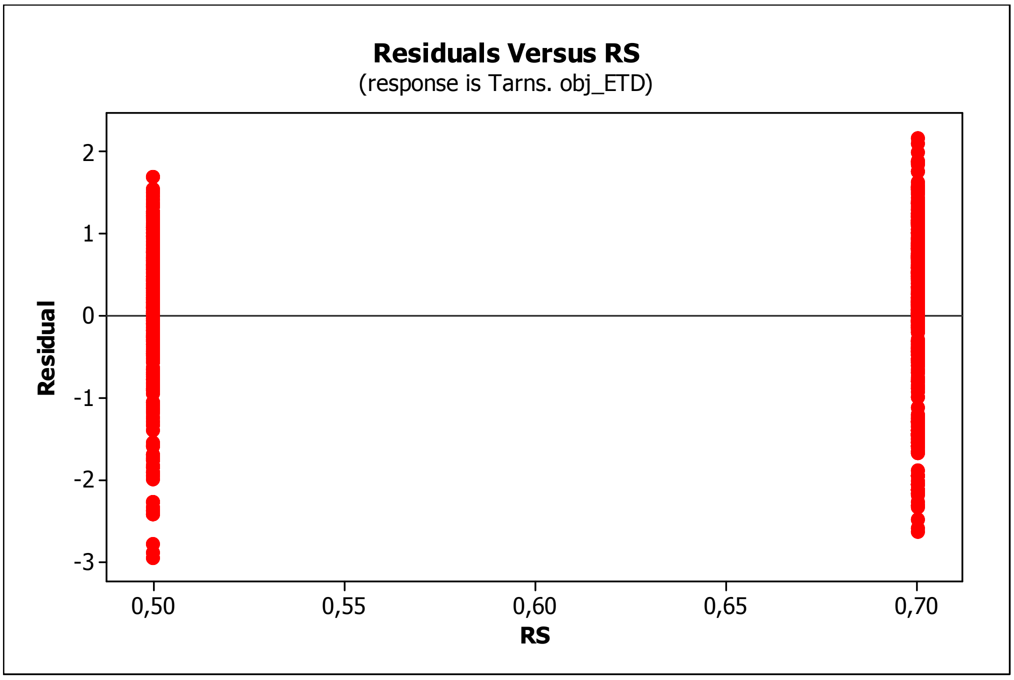

Figure D.25. Residuals vs. Resource Strength for Transformed obj_ETD response

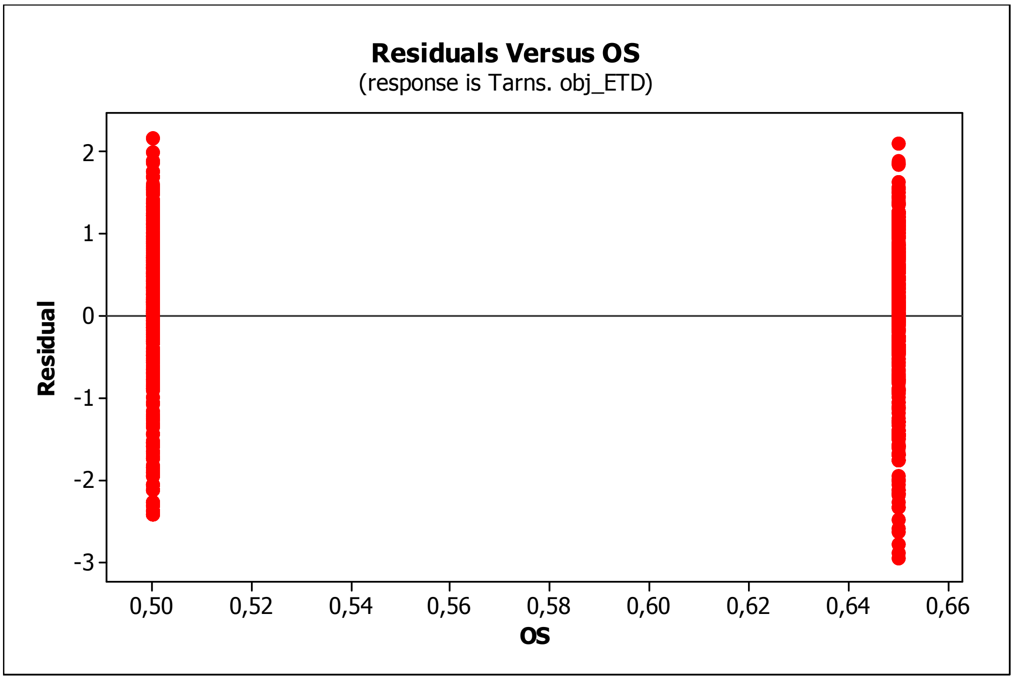

Figure D.26. Residuals vs. Restrictiveness for Transformed obj_ETD response

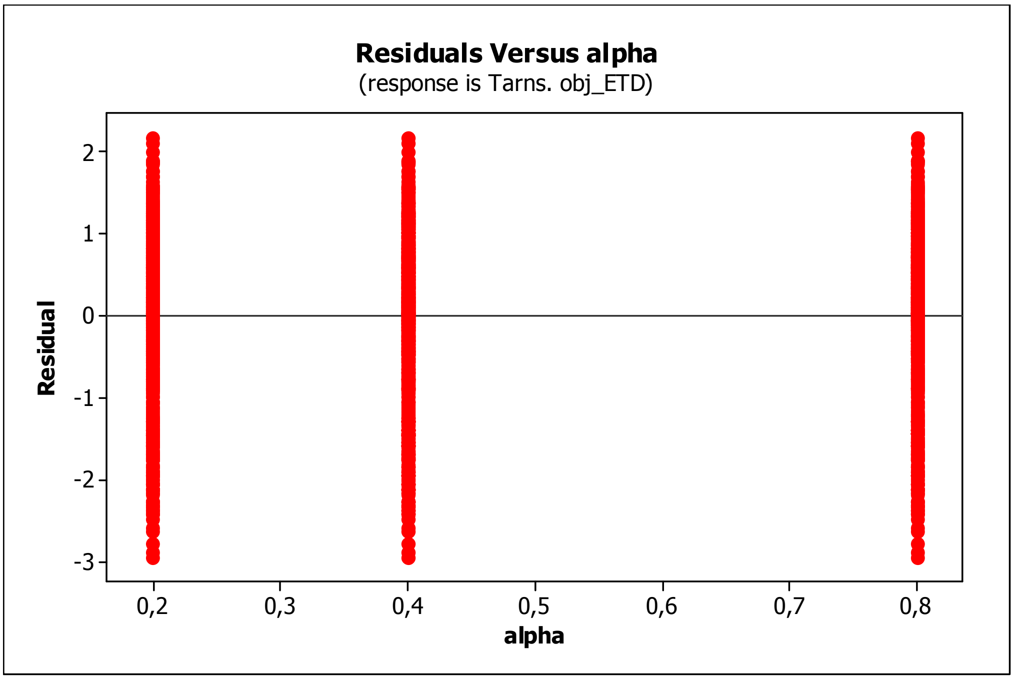

Figure D.27. Residuals vs. Alpha for Transformed obj_ETD response

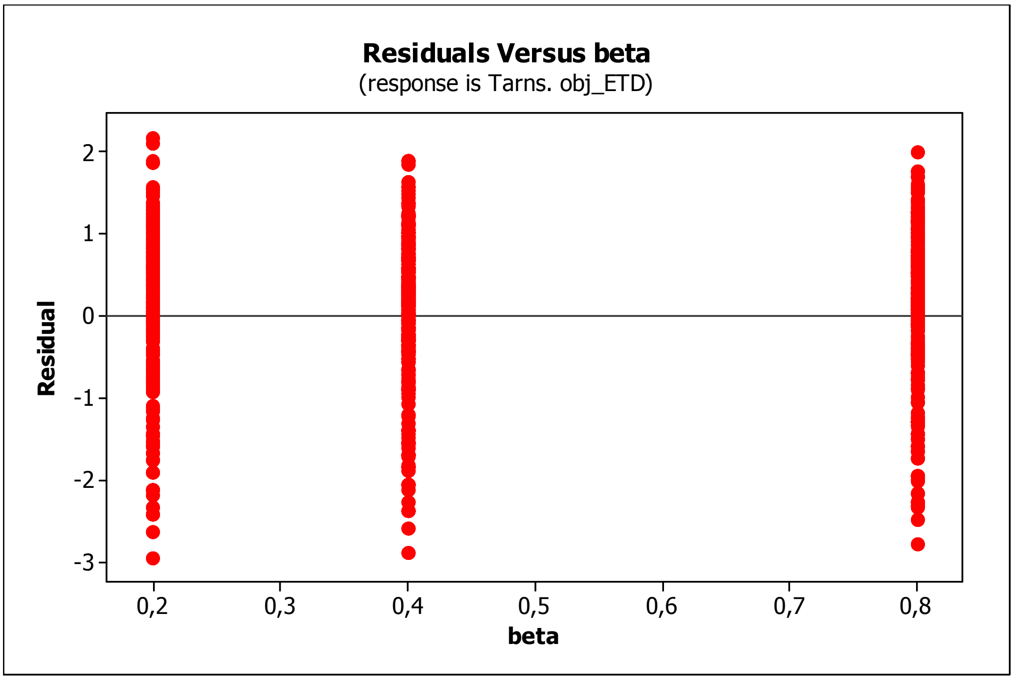

Figure D.28. Residuals vs. Beta for Transformed obj_ETD response

APPENDIX E

MAIN EFFECT AND INTERACTION PLOTS

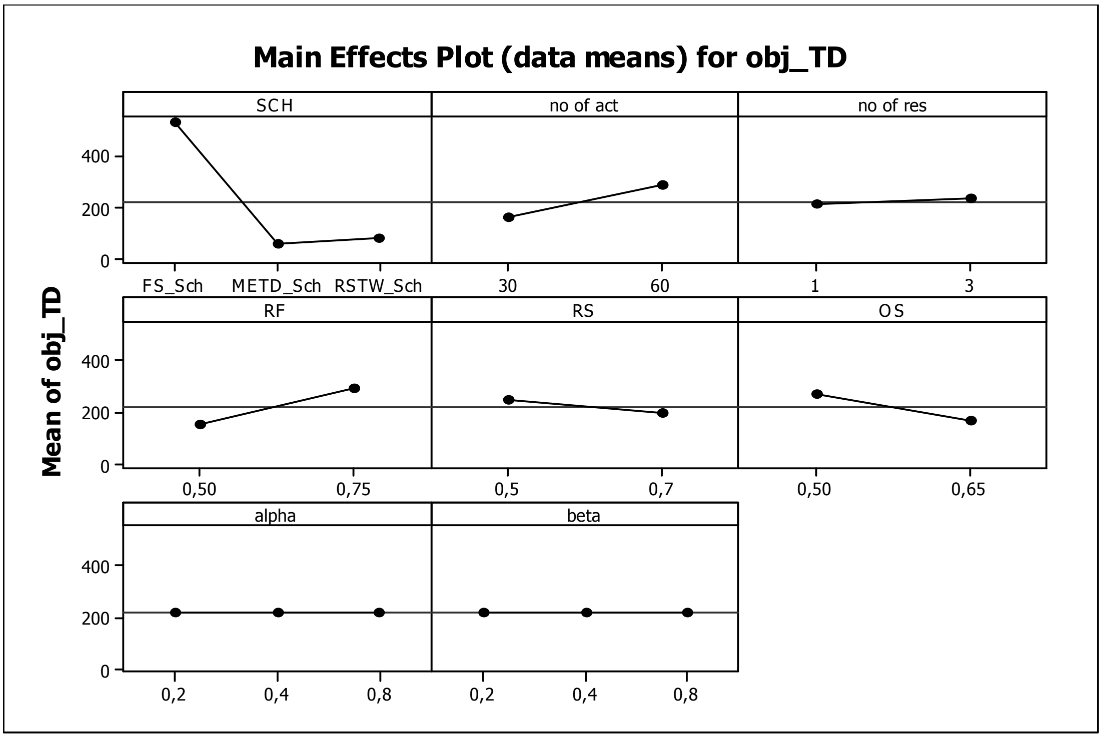

Figure E.1. Main Effects Plot – Data Means for obj_TD response

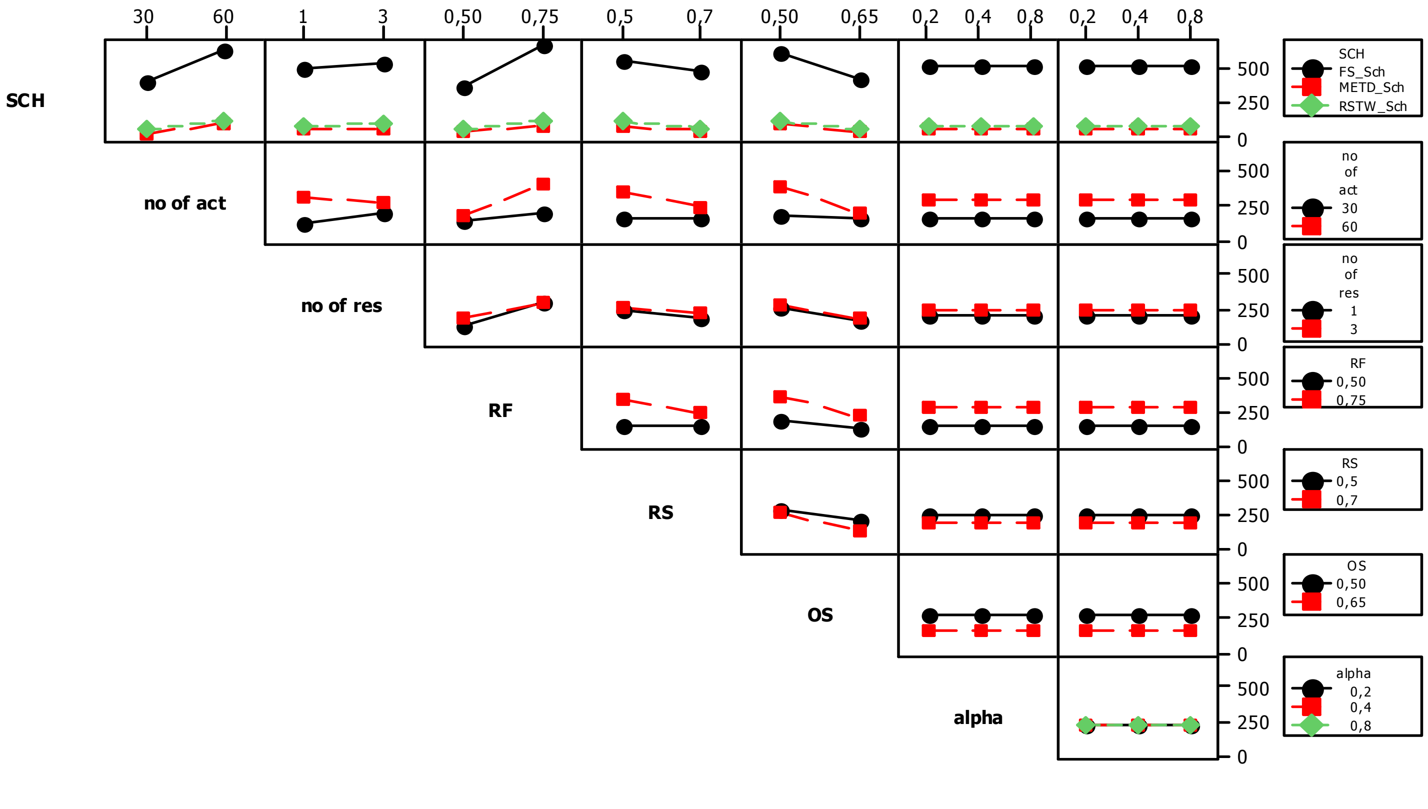

Figure E.2. Interaction Plot (data means) for obj_TD

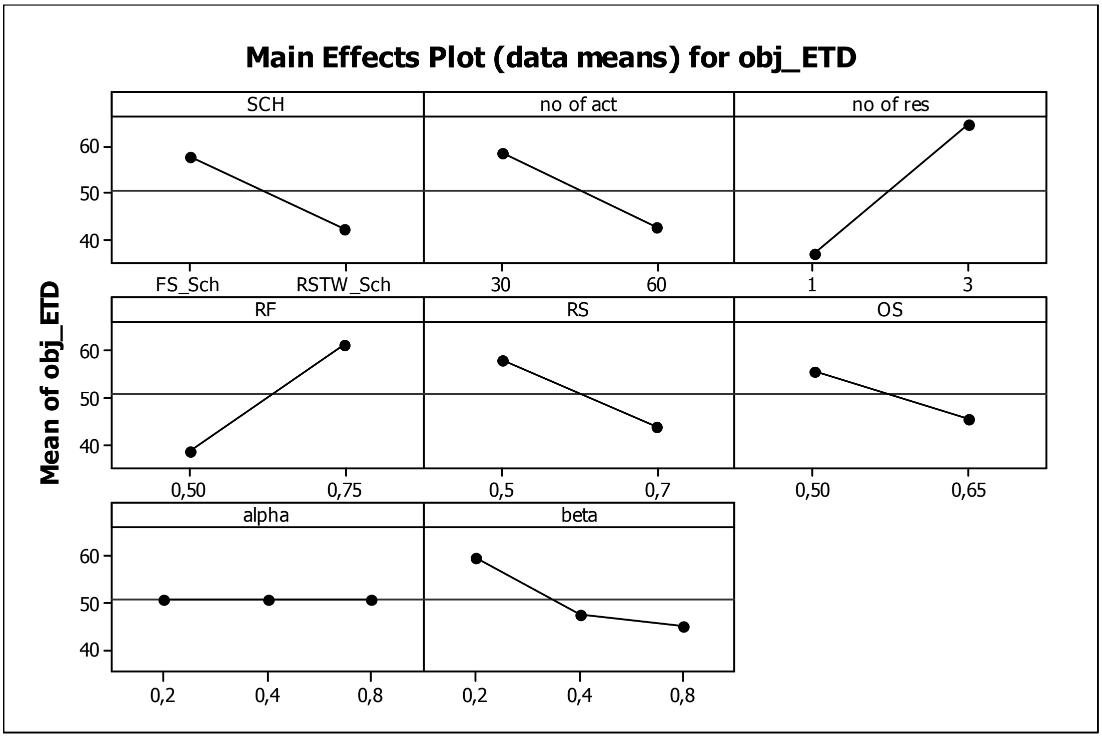

Figure E.3. Main Effects Plot – Data Means for obj_ETD response

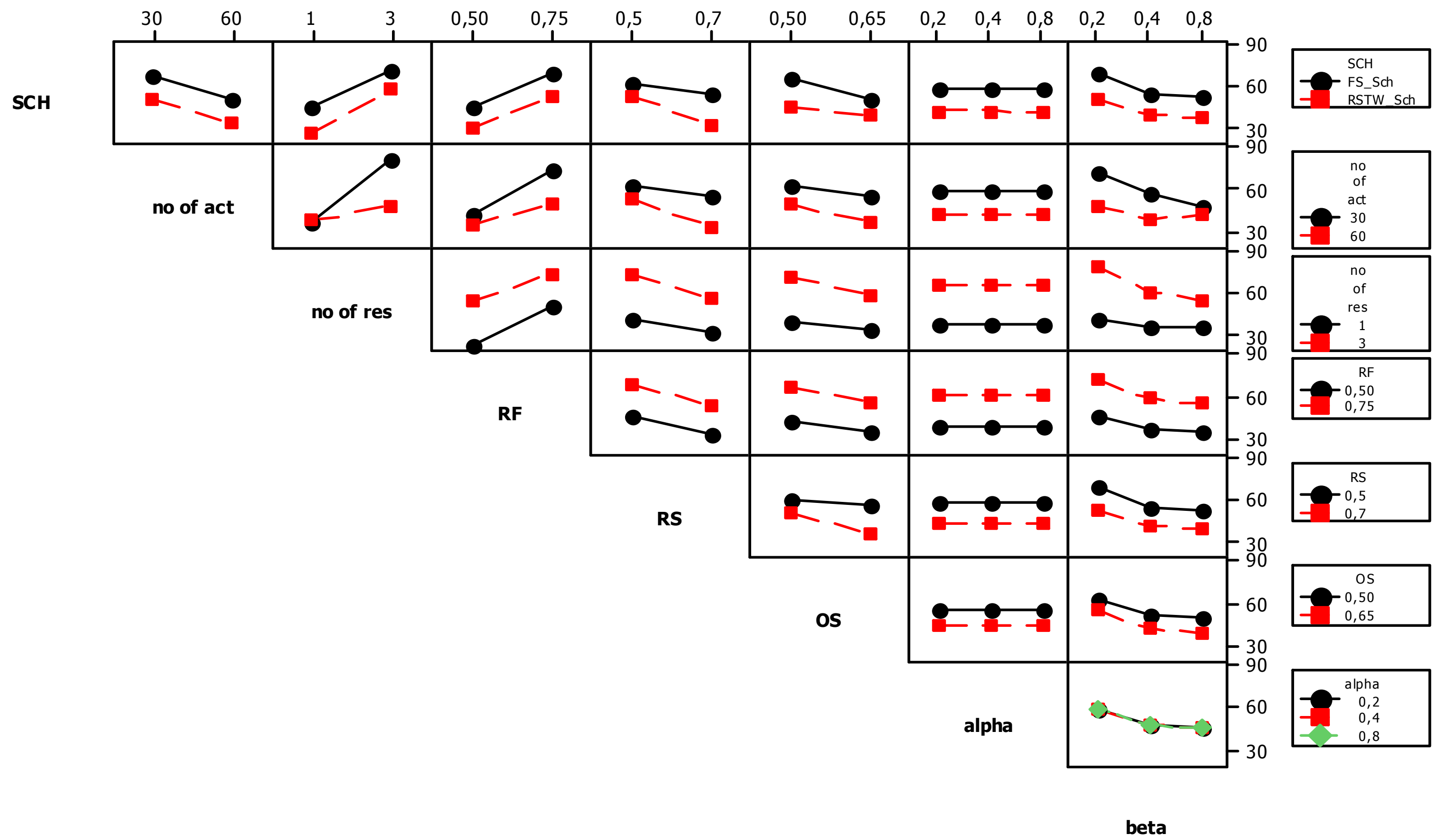

Figure E.4. Interaction Plot (data means) for obj_ETD