OPTIMIZATION MODELING FOR A STEEL STRUCTURE PROJECT WITH MULTIPLE SOURCING POINTS AND STOCHASTIC DELIVERY TIMES

by

O. Serkan İleri

B.S., Civil Engineering, Boğaziçi University, 2000

M.S., Civil Engineering, Boğaziçi University, 2003

Submitted to the Institute for Graduate Studies in
Science and Engineering in partial fulfillment of
the requirements for the degree of
Doctor of Philosophy

Graduate Program in Civil Engineering

Boğaziçi University

2010

…dedicated to my Father, who has always encouraged me
all my life and through my long Ph.D. studies.

ACKNOWLEDGEMENT

I would like to express my special thanks to my master thesis supervisors Prof. Gülay Altay and Assoc. Prof. Necati Aras for providing me a basic foundation for this work, for their support and guidance during this study.

I would like to thank to Prof. Gökmen Ergün, Assoc. Prof. Aslı Sencer Erdem, Prof. Erol Güler and Asst. Prof. Aslı Pelin Gürgün for serving on my progress and thesis committees.

I would like to express my special thanks to Mr. Hüseyin Topaloğlu, who has been a guest instructor at Bogazici University for the 2010 spring semester. He has numerous studies and articles in the area of stochastic dynamic programming, some of which have been referenced in my previous progress reports. He has been highly influential in the construction of the mathematical modelling for the solution of the optimization problem at hand.

I would like to express my special thanks to my friend Okan Erkan for sharing all his knowledge and technical skills with me. It would not be possible to complete this study without his contribution. I am indebted to several people for helping me complete this work. I am thankful to my friend Uygun Alparslan for his support and friendship. I am sincerely grateful to my family for their never-ending trust and support.

ABSTRACT

OPTIMIZATION MODELING FOR A STEEL STRUCTURE PROJECT WITH MULTIPLE SOURCING POINTS AND STOCHASTIC DELIVERY TIMES

Due to the consolidation trends in steel industry and advancements in logistics, steelmakers now face the challenge of choosing the most cost-effective sourcing network for a steel structure project. This research deals with the problem of determining the optimal trade-off between the cost of material supplies and the tardiness cost of a project. The aim of this research study is to provide the mathematical background of the sourcing optimization models for steel structure projects. Another objective is to come up with an efficient solution method for this mathematical model.

Due to the stochastic nature of the delivery times of steel components, the optimization problem is solved by stochastic dynamic programming, which is a well-known technique in operations research. The model is applied to an on-going real project carried out in Siberia, and 9.54% and 19.99% savings are obtained in terms of total project cost. A C++ code is developed for the solution of the optimization problem. The required computation time is quite short, which implies that the proposed method can also be utilized on larger project networks.

The main research contribution is the formulation of stochastic dynamic programming model that optimizes the cost of sourcing. Furthermore, a fit-for-purpose computer code has been developed for the solution process. In addition, standing on the intersection of civil and industrial engineering disciplines, this study stimulates further studies on project scheduling and cost optimization, and use of linear, dynamic and stochastic programming methodologies on different civil engineering subject matters. Further research can be conducted on the optimization model that is formulated in this study – i.e. resource or capacity constraints can be introduced, optimization can be re-conducted at each stage, partial sourcing for activities can be allowed, continuous probability distribution can be used for activity durations.

ÖZET

ÇELİK KONSTRÜKSİYON PROJELERİNDE TEDARİK OPTİMİZASYONU: ÇOKLU TEDARİK NOKTASI VE STOKASTİK SEVKİYAT SÜRELERİ

Çelik endüstrisinde yaşanmakta olan konsolidasyon ve lojistik alanında yaşanan gelişmelere istinaden, çelik konstrüksiyon üreticileri artık çelik konstrüksiyon tedariğinde minimum maliyeti yakalayacak tedarik ağı seçimini yapmak zorundadırlar. Bu çalışma bir çelik konstrüksiyon projesinde malzeme maliyeti ve projenin gecikmesinden dolayi ödenmesi gerekecek cezai maliyetin optimizasyonu konusuna eğilmektedir. Bu çalışmanın amacı çelik konstrüksiyon projelerinde tedarik optimizasyonunu sağlayacak matematiksel bir modelleme geliştirmek ve bu modellemeyi çözebilecek bir algoritma oluşturmaktır.

Çelik elemanların sevkiyat sürelerinin stokastik biçim göstermelerinden dolayı, yöneylem araştırması alanında çokça kullanılan bir yöntem olan stokastik dinamik programlama vasıtası ile problem çözülebilmiştir. Oluşturulan modelleme Rusya'nın Sibirya Bölgesi'nde devam etmekte olan gerçek bir çelik konstrüksiyon projesine uygulanmış ve değişik cezi şartlar için yüzde 9.54 ve 19.99 maliyet tasarrufu sağlanmıştır. Modelleme C++ dilinde yazılan bir algoritma ile çözülmüştür. Çözüm için gerekli süre çok kısadır; dolayısıyla geliştirilmiş olan modelleme daha büyük inşaat projeleri üzerinde de kullanılabilir.

Bu çalışma söz konusu optimizasyonu ilk kez ele alması, bu ikilemi ilk kez stokastik dinamik programlama metodolojisi ile modellemesi ve bu modellemeyi bir programlama dili ile çözmesi açılarından akademik anlam kazanmaktadır. Ayrıca inşaat ve endüstri mühendisliği disiplinlerinin kesişiminde yer almakta ve bu bağlamda proje yönetimi ve zaman planlamasi, maliyet optimizasyonu ve inşaat mühendisliği problemlerinde doğrusal, dinamik ve stokastik programlama metodolojilerinin kullanıldığı ileriki çalışmaların önünü açmaktadır. Oluşturulmuş modelde yapılacak çeşitli revizyonlar yeni araştırma konularının ortaya çıkmasına sebep olacaktır – örneğin kaynak veya tedarik kısıtlarının getirilimesi, optimizasyonun her yeni proje safhasında tekrarlanması, bir aktivite için birden fazla tedarik noktası kullanılabilmesi ve aktivite sürelerinde sürekli olasılık dağılımı kullanılması gibi.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

AOA	Activity on Arc
CCPP	Combined Cycle Power Plant
CIS	Commonwealth of Independent States
CPM	Critical Path Method
DP	Dynamic Programming
DEM	Deterministic Equivalent Model
DEP	Deterministic Equivalent Program
EF	Earliest Finish Date
ES	Earliest Start Date
GAMS	General Algebraic Modeling System
GERT	Graphical Evaluation and Review Technique
JIT	Just In Time
LF	Latest Finish Date
LS	Latest Start Date
MILP	Mixed Integer Linear Programming
PERT	Program Evaluation and Review Technique
RCPS	Resource Constrained Project Scheduling
S-MILP	Stochastic Mixed Integer Linear Program
TCO	Total Cost of Ownership
VERT	Venture Evaluation and Review Technique

1. INTRODUCTION

1.1. Background

1.1.1. Steel Industry

Recently, steel industry has been experiencing significant consolidations and the industry has become less fragmented globally. As per World Steel Association statistics (World Steel Member Companies 2009 Crude Steel Production, 2009), in 2000, the largest steelmaker of the world had a total crude steel production of 28.4 million metric tons. Whereas due to establishment of Arcelor Group in 2001 via the merger of Arbed, Aceralia, and Usinor, followed by the merger of Arcelor and Mittal – the top two global steel producers – in 2006, the industry leader had a total crude steel production of 117, 116.4, 103.3, 77.5 million metric tons in years 2006, 2007, 2008, and 2009 consecutively. These figures represent market shares of 3,35%, 9,35%, 8,61, 7,77%, and 6,4% in years 2000, 2006, 2007, 2008, and 2009 respectively. This merger of Arcelor and Mittal fuelled mergers and acquisitions in the entire industry and the industry has been experiencing significant consolidations. It is now possible to talk about global players in the industry rather than regional or local firms. This consolidation in the industry will undoubtedly have serious effects on the industry itself and some other sectors, which utilize steel as a major raw material in its products.

Three main channels, to which steelmakers sell their products, are construction industry, automotive industry, and domestic appliances industry. The focus of this thesis study will be on the effect of consolidation in steel industry on construction sector.

Use of steel in various types of constructions has been continuously ramping up for the last decades. Grades of steel as well as types of available steel elements have also shown a rapid increase. World Steel Association statistics (Steel Statistical Yearbook, 2009) show that production of hot rolled long products have risen up from 338 million metric tons in year 2000

to 526 million metric tons in year 2008. Similarly production of hot rolled flat products have risen up from a total of 427 million metric tons in year 2000 to 665 million metric tons in year 2008. Thus, in this era, in a steel construction project, numerous combinations of steel elements exist to build a steel structure.

In the old days steel was merely supplied by local firms, and rarely by the regional players, for construction projects. There were many determinants that drove this result; steelmakers had production facilities only in their home countries, freight cost was extensively high especially for heavy loads, there were quite many trade barriers for international flow of goods, management of logistics and communication were very complicated and tough tasks to accomplish and so on. Globalisation movements, in addition to the advances in technology and communications, led to the fact that freight of steel elements internationally became much easier than it was before. Therefore, it is now a common practise among the steelmakers to acquire or merge with firms in other countries and buy new plants overseas.

Consequently, in the new world of steel, steel supply does not necessarily need to be sourced from the very same country of a construction project. Steelmakers now have the freedom of supplying the steel components from a network of plants spred across a wide geographical region. This makes optimization of steel supply a major issue in steel structure projects.

1.1.2. Steel Structure Projects

Steel structures have a wide range of applications within the construction sector. Industrial buildings, high-rise buildings, bridges, big off-shore platforms, shopping malls, residential buildings, office buildings and many more to list are examples of projects where steel structures are assembled.

One of the major differences between a steel structure project and a reinforced concrete structure project is the fact that for a steel structure project much work needs to be performed

upfront; during the design and planning phase, than for a concrete structure. Because the components composing a steel structure are usually prefabricated ones, which are processed in steel mills and are ready-delivered to the construction site. Whereas, in the concrete structure projects, much more work and processing can be performed on-site. This fact shows the importance of a thorough planning phase for steel structure projects; the right components including all necessary accessories, fit-for-purpose to meet the design requirements, with the right colors, right dimensions, right raw materials, should be delivered to site 'on time'.

Steel structures are composed of numerous components. Drawings of an indicative steel structure are provided in Appendix A[1]. The presented structure is a primary sign tower used for commercial advertising in the front yard of a big shopping mall, being constructed in Russia. In addition, the associated Bill of Quantities[2] is also provided in Appendix A. As can be seen from this Appendix, the list of components is quite exhaustive, even for a small sign tower in the yard. Bills of quantities belonging to the buildings themselves are naturally much more exhaustive and complicated in nature.

When it comes to delivery of steel components to the construction site, contractors prefer to receive them in a "Just In Time" manner. There are various reasons for this JIT delivery concept:

- Too many materials brought on-site long before they will be used, will cause a chaotic environment and disturb the smooth flow of works,
- The materials might get damaged, if they stay too long on the site,
- There might not be enough storage space on site in most circumstances,
- The handling costs might be substantial due to movement of materials on site,
- In case of potential design revisions, that will be more favorable to order the materials as late as possible to avoid wrong components being brought on site,
- It is in contractor's interest to tie the finances related to the cost of materials as late as possible.

[1] Necessary permits are obtained from the construction company to present these drawings and the concerning Bill of Quantities.
[2] Bill of Quantities is the inclusive list of components that build up the structure.

On the other hand, JIT brings along some difficulties; the possibility of late deliveries increases significantly. As can be inferred, a delay in the delivery of components to site, might delay the completion of the entire structure.

A brief summary of what has been said is:

- Steelmakers tend to source their supply from diverse locations due to cost savings on materials, even though these locations might be far away from the location of the project;
- Contractors aim to receive the materials on site in a JIT manner.

Combination of these two facts arises the dilemma; How can the materials be brought from distant locations on time, given the fluctuations in delivery times due to many factors?

1.2. Selection of the Thesis Subject

1.2.1. Statement of the Problem

In a narrower context, this thesis study will focus on optimizing the cost of supply of steel components in a steel structure project. The dilemma raised in the last section constitutes the basis for the problem at hand.

The unit cost of components ordered from distant located plants might be cheaper. However, since the delivery times of the products are not deterministic, there might be significant fluctuations in the makespan of a project, which, in turn, have an associated tardiness cost. The optimal balance should be set between benefiting from the lower product cost and having to pay penalties due to late completion of the project.

Thus, the hypothesis to be tested is;

H_0 – null hypothesis: Only unit costs of components are adequate, when trying to decide on the optimal sourcing strategy for components from a supply network;

H_1 – alternative hypothesis: Unit costs of components are not adequate to find the optimal sourcing strategy, but their delivery times and associated tardiness costs that could be incurred due to late completion of a project should also be taken into account.

The parties involved in the fabrication and erection of a steel structure are the Contractor of the project and the Steel Structure Subcontractor ("Supplier"). The output of this study will optimize the costs of the Contractor, which are incurred due to construction of the steel structure. That said, it should be noted that the Contractor ends up with cost savings as far as the Supplier finds ways to lower the costs; meaning any cost savings achieved by the Supplier are considered to be fully passed on to the Contractor.

A steel structure project consists of much more than the assembly of the steel components. The main steps of a steel structure project are:

- Phase 1: Planning of the project – timeline, resources, budget etc.;
- Phase 2: Preparation of the preliminary architectural design of the structure;
- Phase 3: Tender and contracting process for appointment of the Supplier;
- Phase 4: Preparation of the structural design of the structure;
- Phase 5: Preparation of the execution drawings of the structure;
- Phase 6: Preparation of the shop drawings – to be used in fabrication of the components;
- Phase 7: Fabrication of the steel components;
- Phase 8: Packing and loading of the components;
- Phase 9: Transportation of goods from the fabrication plant to the construction site – this step includes all sorts of logistical processes such as issuance of transportation documents, clearance of the goods from the customs, loading-unloading in transshipment locations etc.;
- Phase 10: Receipt and storage of goods at the construction site;
- Phase 11: Erection of the components of the steel structure according to the execution drawings;
- Phase 12: Inspections and hand-over of the project.

The optimization model to be derived as the output of this study is related to the 1st, 7th, and 9th phases of the above process flow. The relationship between the cost of the project and these processes are as follows:

- Cost – Planning relationship: The construction contracts include provisions for the makespan of the projects. In some cases, these contracts even have predefined due dates for certain milestones. Most contracts include penalty payments due by the Contractor, if the project completion due date is not met. Most significant penalty clauses are encountered in the contracts of construction of revenue-generating buildings, in which steel structures are usually used. Thus, as long as a steel construction project is delayed, the Contractor will be obliged to pay a certain amount of money per unit amount of time – this unit being a day or a week in most cases. Thus the time planning of the project should be carefully determined and a Contractor should know in advance whether it will be able to keep up with the project due date. In this study, due dates for intermediary milestones will not be considered and the focus will be on the final completion date of the project.
- Cost – Fabrication relationship: In the first section, it was stated that steelmakers recently have had the freedom of choosing their sourcing points from steel mills in diverse locations. Each of these mills will have quite different cost structures, especially if they are in different countries, due to various reasons such as cost of capital, cost of labor, efficiency level of the mill, location of the mill, benefits such as low-cost electricity obtained from local authorities and so on. Thus, pricing of the same steel component will be different in each mill. Undoubtedly, the Supplier will try to order the materials from these mills in such a way that it will incur the least cost for the entire project. If only the ex-works3 prices of materials were considered, this would have been an easy exercise. However, "Total Cost of Ownership" (TCO) should be considered in this respect and TCO includes transportation of goods, duties in the customs and more importantly tardiness cost that could be incurred, if the project completion was delayed due to late delivery of products to the site.

[3] Ex-works is a clause in Incoterms international trade standards set forth by the International Chamber of Commerce and represents the price of a product when it is ready for shipment at the plant where it is produced. Incoterms regulates the rules in global and domestic sales transactions.

- Cost – Transportation relationship: Transportation affects the cost of the project, since it is itself a cost driver. However, in this research study another affect of transportation to the project cost will also be investigated; tardiness costs incurred due to late delivery of materials, which is a result of prolonged transportation durations.

1.2.2. Goals and Objectives

Selection of the aforementioned problem as the subject of this research study has various reasons behind. Firstly, construction projects would benefit to a large extent from industrial engineering applications on cost savings, efficiency increases and more effective planning. Construction projects – as well as all other projects – have three dimensions: Budget (or cost), Time, and Scope. This study specifically delves deeper into the "cost" and "timing" dimensions of the management of construction projects.

Techniques such as CPM and PERT have been used by both the construction industry and civil engineering academic world for many years. However, not many studies focusing on the trade-off between material cost and cost of timing exist in literature; thus, this study will be a first-in-the-literature within its scope; combining cost minimization and project scheduling concepts, which are usually approached separately. Accordingly, this study will open a way for further studies on this interdisciplinary subject.

The aim of this research study is to provide the mathematical background of the sourcing optimization models for steel structure projects. The model to be constructed will be applied on a real construction project and the applicability of the theoretical framework will be tested.

Another objective is to come up with an efficient solution method for this mathematical model. The project activity networks, which will set the main scene for the optimization model, represent highly complicated networks and mathematical relations among activities.

The solutions of such formulations are sometimes not possible in practice due to very large size of the possible scenarios. In this research, an efficient algorithm will be sought after that will solve the optimization problem within a reasonable amount of time.

The model to be formulated will open ways for further research studies. Because the activity networks and their real life applications represent countless combinations of problem settings. Even small modifications on the way the network is optimized within this research will require a new mathematical model that would be subject to another research study.

2. LITERATURE REVIEW

Much research has been conducted on the optimization of supply cost and scheduling fields separately. Although this study will deal with the combination of these two fields – a relatively untouched area, very useful insight could be gained from past work.

> *"The theory of sequencing and scheduling, more than any other area in operations research, is characterized by a virtually unlimited number of problem types." Shmoys (1993, p.446)*

Uetz [1] defines the generic formulation of a scheduling problem as the design of a 'plan of procedure' for a given set of so–called jobs, generally in such a way that several side constraints are respected and some given objective is minimized. On the other hand, the twist from deterministic to stochastic processing times changes the nature of the scheduling problem considerably. The stochastic scheduling problem itself is no longer a combinatorial optimization problem, but can rather be embedded into the framework of stochastic dynamic optimization. Particularly the solution of a stochastic scheduling problem is no longer a simple schedule, but a scheduling policy [1]. Uetz also proposes linear programming relaxations for the solution of Resource Constrained Project Scheduling problems (RCPS).

Stork [2] defines the RCPS as a problem where a set of jobs with random processing times has to be executed subject to both precedence and resource constraints. There is, however, an important new aspect that comes into play: What is a solution to a stochastic RCPS problem? A solution should define for each possible 'event' that appears within the execution of the project an appropriate 'action'. This 'action' typically is a decision on which jobs should be executed next and an 'event' may be the completion of some jobs. To make such a decision one may want to exploit the information given by the current state of the project. Such a solution is called a 'policy'.

Yang et al. [3] provide a summary of past work on RCPS problems and presents a new RCPS model with a minimization of cost objective. Their model is applicable to a contractor

in the construction industry facing project deadlines with limited resources and penalties for late completion. However, their modeling is based on deterministic activity durations. They also provide a classification of RCPS problems, identifying 6 types, but do not elaborate on the stochastic RCPS problem.

Stork [2] has determined the deviation of the makespan in case of stochastic processing times from that of the one with deterministic processing times. The below graphic represents the deviation of the average and maximum makespans of the schedules with stochastic processing times from the deterministic makespan. The overruns have been respectively provided for Preselective, Linear Preselective, Earliest Start, and Job-Based Priority Policies. As can be seen, makespans of stochastic schedulings are approximately 4% and 10.5% longer than those of the deterministic schedules on average and maximum cases respectively. These percentages mean an approximate difference of 15 and 38 days for a 1-year project. The implication is; a project scheduled according to deterministic job processing times will most likely overrun 15 days, with a possibility to overrun up to 36 days in the worst case. These overruns will end up in substantial amount of penalty paid by the Contractor.

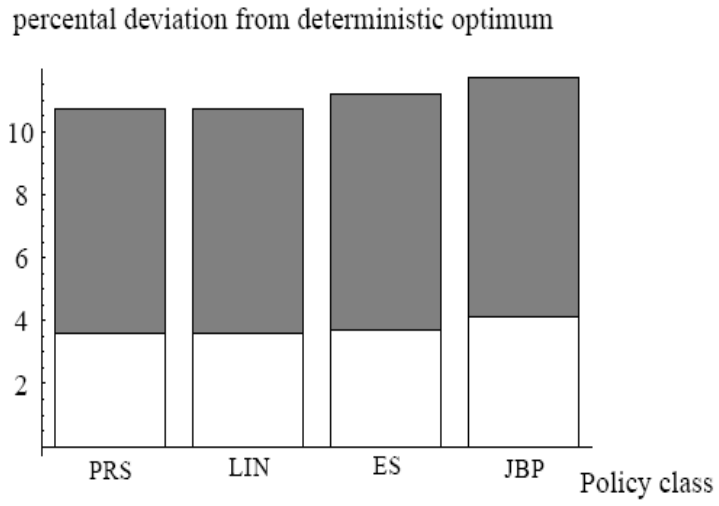

Figure 2.1. Average and maximum percental deviation from the deterministic optimum[4] [2]

[4] The abbreviations on the chart represent Preselective, Linear Preselective, Earliest Start, and Job-Based Priority Policies respectively, which are the main subclasses of policies.

Several Branch and Bound Methods have been extensively used in the solution of the RCPS problems. They are either used to reduce the computational effort of exact algorithms, or they provide ‘performance guarantees’ for heuristic solutions [1].

Bidot [4] classifies the scheduling problems according to precedence constraints among the activities. For open-shop scheduling problems there is no precedence constraint between the operations of a job. For job-shop scheduling problems each job is a different sequence of operations while for flow-shop scheduling problems each job defines the same sequence of operations. Accordingly, construction projects are mainly of the job-shop type. According to Bidot, typical optimization criteria of scheduling problems are makespan, tardiness, number of tardy operations, and allocation costs.

Bidot [4], furthermore, lists some of the factors that cause uncertainty in scheduling:

- some operation processing times/durations or some earliest operation start times are not precisely known a priori;
- some resource capacities or states are imprecise a priori;
- some resource quantities required by some operations are imprecise and/or uncertain a priori;
- uncertain and/or imprecise production orders;
- new operations to be scheduled and executed may only be known during execution.

Choi et al. [5] have developed a way to combine heuristic solutions for the RCPS problem through dynamic programming in the state space that the heuristics generate. The main idea of the algorithmic framework is to first find an important set of states via a large number of simulations with various heuristic policies and then solving the DP over the set of states visited by the heuristics to obtain an optimal solution within the confined state space. This way they overcome the burden of computational load and impracticality caused by the stochastic RCPS problem. The heuristics they utilize are:

- Heuristic 1: High success probability task first;
- Heuristic 2: Short duration task first;

- Heuristic 3: High reward task first.

Subsequently, the mathematical programming approaches can account for the uncertainties of stochastic problems via scenario generation. However, their common problem is that they are limited to small number of scenarios due to exponential increase of computational load.

Lambrechts et al. [6] introduced a new variant of RCPS, where resource availabilities, which are subject to unforeseen breakdowns, are subject to uncertainty. They introduce the concept of weighted instability, which is the expected weighted absolute deviation between the planned and the actually realized activity starting times. W_i denotes the marginal cost of deviating from the planned starting time of activity i during project execution. This marginal cost can be seen as an extra cost for having subcontractors start later than originally agreed or as an inventory cost for storing raw materials between the delivery time and the time they are needed [6]. The activities are sequenced in increasing order of their instability weights and a heuristic method, specifically the largest-instability-weighted-activity-first principle, is used to produce a schedule that is both time and resource feasible.

Project Evaluation and Review Technique (PERT) is a stochastic project network that consists of a set V of jobs with precedence constraints among pairs of jobs. However, in contrast to the RCPS problems, no resource constraints are imposed in the network. Each job can be started as soon as all its predecessors in the partial order defined by the precedence constraints are completed. The difficulty arises from the additional assumption that the processing time of each job is uncertain and is indicated by a random variable p_j [2]. The aim in a PERT problem is to determine the *statistical distribution of the project makespan*.

Copertari and Archer [7] claim that a normally distributed project completion time as per the PERT assumption leads into optimistic planning with less-than-actual estimated project makespans. They also assume beta distributed activity duration times but their assumption differs from conventional PERT that they determine the probability density function of the

project makespan as a beta distribution, of which they also provide the mathematical background.

Cottrell [8] defines the critical path as a sequence of activities that cannot be delayed without jeopardy to the entire project. In PERT, project network is given as a diagram, where each arc represents an activity and each node symbolizes an event. The expected duration of each sum is equal to the sum of the expected durations. Similarly, the variance of each sum is the sum of the variances. Cottrell's simplified PERT produces shorter expected project durations, but greater project duration variances, than does conventional PERT. The simplification is to reduce the number of estimates required for activity durations from three, as in conventional PERT, to two. These two estimates are the most-likely and the pessimistic estimates, and normal distribution is applied to the activity durations.

Ahuja and Thiruvengadam [9] have conducted an extensive investigation of research on project scheduling and monitoring, including the Critical Path Method and PERT. They have touched upon the research dealing with limitations of PERT / CPM, linear scheduling techniques, simulation techniques, scheduling of fast-track construction projects, time-cost optimization, resource allocation, and project monitoring and controlling.

Pontrandolfo [10] has developed an algorithm for determining the duration of stochastic networks by the so-called PERT-path technique. Via this algorithm, he derives the duration of every possible PERT path. In his numerical example, he used exponentially distributed activity durations. When it comes to networks with dependencies between activity durations, his approach provides an approximate estimate of the project duration.

Fetz et al. [11] have studied the use of fuzzy numbers and methods for processing uncertainty in project scheduling and cost planning. Nevertheless, his work does not help much for the optimization problem to be tackled within this study.

Hardie [12] questions the assumption of PERT that the activities of a project are independent from each other and occur in a linear sequence. He defends that projects occur as recursive networks, with reversions to earlier stages for rectification or to incorporate changes.

Abbasi and Mukattash [13] promote the concept of crashing in PERT networks to shorten the length of the pessimistic time estimate via inserting additional money in the critical activities, thus increasing the resource level for these activities. In doing so, the activities with the lowest cost slope, which will shorten the critical path, have the priority over others. The minimization of the pessimistic time estimate both decreases the project duration and reduces its variance, thus enables a more representative planning process. They make use of an algorithm for the application of their model to numerical examples.

Mummolo [14] states that criticality and uncertainty measures determined by PERT Monte Carlo simulation are more accurate than PERT estimates, because PERT estimates underestimate the project duration. He proposes new uncertainty and criticality measures based on PERT Path Network Technique. The approach considers the evolving nature of a project network and takes into account the non-critical activities, thus all possible completion sequences of a project as well.

Dawson and Dawson [15] present extensions to existing software tools that are used to plan projects with uncertainty involved – i.e. Graphical Evaluation and Review Technique (GERT) and Venture Evaluation and Review Technique (VERT). He approaches the issue from a practical point of view as well, and highlights an average project manager's inability to define probability distributions for the activity durations. He advises to ask only three estimates – minimum possible time, most-likely time, and maximum possible time – from the project managers and use an automatically derived simple distribution shape such as triangular distribution.

3. METHODOLOGY

3.1. Supply Network

The supply network for a steel structure project consists of the construction site and several steel mills, which could potentially be used as sourcing points. Figure 3.1 provides a graphical illustration of such a supply network.

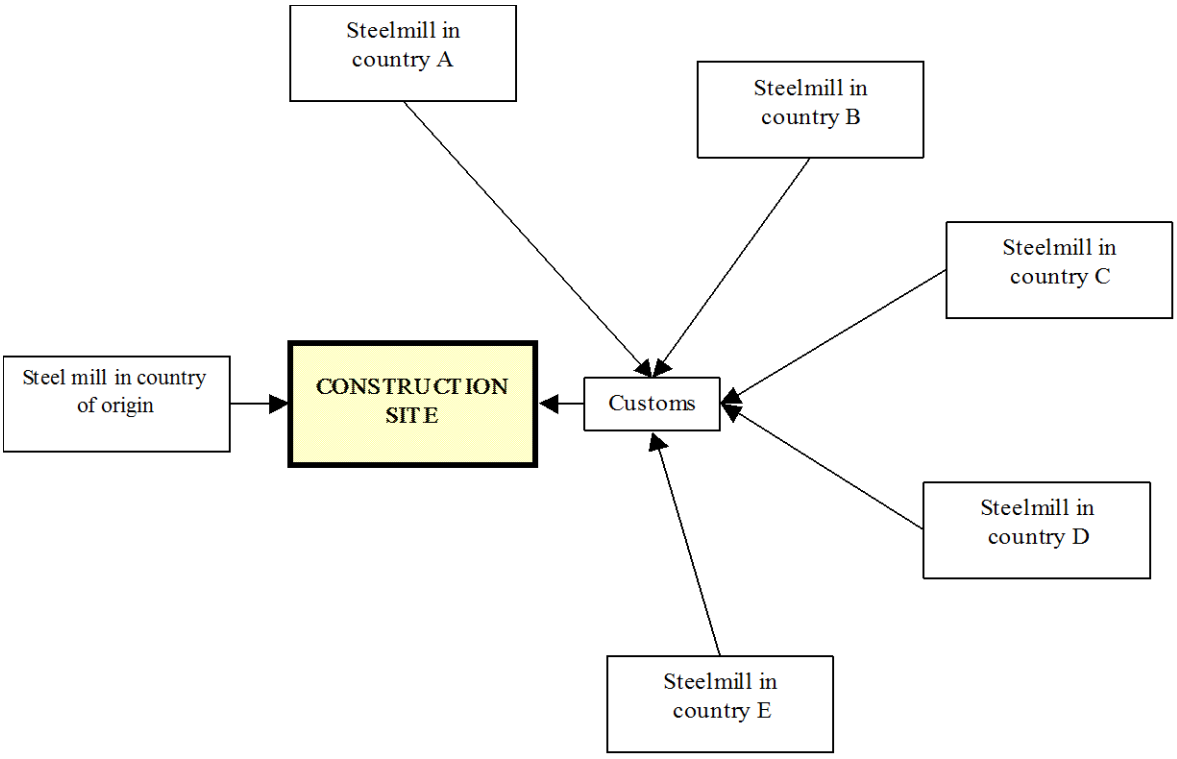

Figure 3.1. Supply network for a steel structure project

All the components of a steel project are not necessarily produced in each of the steelmills. Rather, each mill produces some variety of components. Thus, for each component there is certain number of potential sourcing points and that many different unit prices applicable in each of these locations.

Furthermore, each steel mill is at a certain distance from the construction site and the ordering processes, shipment processes, freight, customs clearance, transfer to the site, and receipt and handling of goods at the site take different time periods for each of these mills. In the model to be structured in this study, total time to receive the goods from the date of order will be considered to behave in a stochastic nature, when the concerning components are ordered from abroad. The construction site will be at the location of a real project in Russia and the sourcing mills will be considered to be in the region of Russia, CIS countries, Balkans, Central Eastern Europe, and Turkey. The amount of time spent in the customs clearance is also an important factor and thus will be included in the calculations. That said, transportation time from a steel mill in the country of origin will be considered deterministic.

The final output of the model to be constructed will be providing answers to the trade-off issue; (for each of the components) whether to source from the country of origin with high unit prices or to source from abroad with lower unit prices but expect to pay late-completion penalties due to uncertainty in the delivery time of components. Another extension to this decision will then be which components should be ordered from which steel mill.

3.2. Project Network

Let N denote a probabilistic PERT/CPM network composed of nodes (vertices) $N = \{1,2, ..., n\}$ and directed arcs $A = \{(i,j) \mid i=1 ... n-1, j=2 ... n\}$ where n is the total number of nodes and N is the adjacency matrix[5]. Let m be the total number of activities so that the set of directed arcs, A, can also be denoted as $A = \{k \mid k=1 . . .m\}$. The duration of arc (i,j) is a random variable t_{ij} with a known probability density function $f_{ij}(t)$ over the closed interval $[a_{ij},b_{ij}]$ where μ_{ij} denotes the mean (expected) duration of activity k in arc (i,j) and σ_{ij}^2 its variance. (Activity on Arc notation or AOA is implicit, where i indicates node of origin and j node of destination.) The completion time at sink node j, T_j, is the time at which all activities coming into j have been completed. The completion time at source node i, T_i, is the earliest time at which any activity k in arc (i,j) located between nodes i and j is allowed to start. An

[5] The content of this section is extracted from [8]

example AOA network graph belonging to a Gas Pipeline Operation is provided in Figure 3.2. The three figures on each arc represent the optimistic, most-likely, and pessimistic activity durations respectively (*a*, *m*, and *b*).

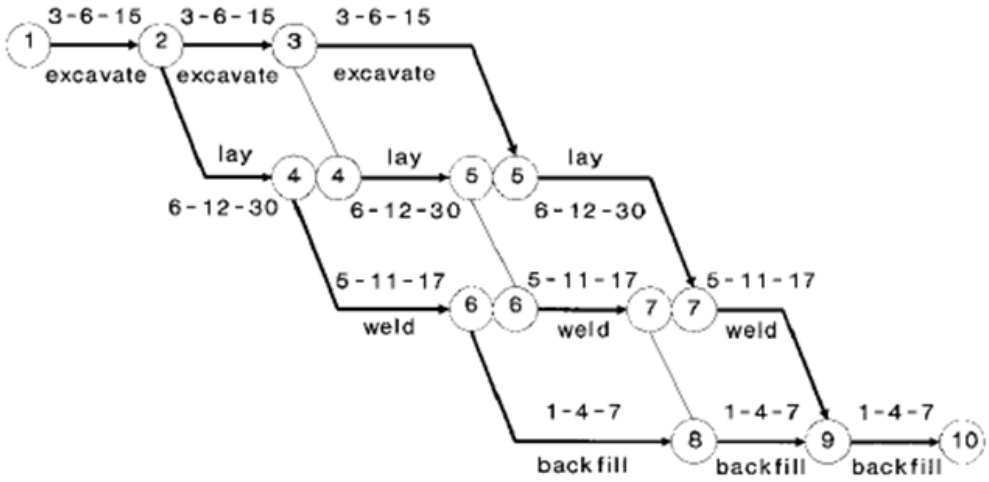

Figure 3.2. AOA graph of a gas pipeline operation [8]

N, the adjacency matrix, contains all precedence relationships. Let B_j (j=2,.., n) denote the set of predecessor nodes connecting to node j. Let i be one such node ($i \in B_j$). Figure 3.3 provides illustrations of both the notation for B_j and the adjacency matrix.

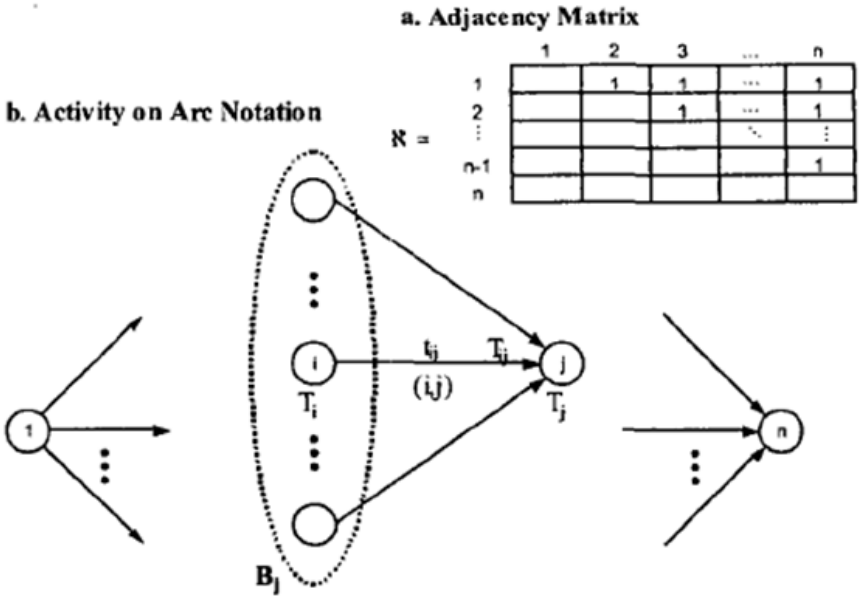

Figure 3.3. Adjacency matrix and illustration of B_j [7]

Denoting activities using their nodes of origin and destination facilitates writing equations for forward pass computations, as is the case in below equation:

$$T_j = \underset{i \in Bj}{\text{Max}} \left\{ T_i \oplus t_{ij} \right\} ..... \forall j = 2...n \quad (3.1)$$

Above equation is a stochastic sum across the network. The symbol $\oplus$ is used to denote the addition of two stochastic variables, T_i and t_{ij} which yields another stochastic variable, T_{ij} where $T_{ij} = T_i \oplus t_{ij}$, so that $T_i<T_{ij}<T_j$. T_{ij} indicates the completion time that would occur at node j if activity (i,j) happens to be critical across the network (longest duration time in a particular combination of random duration times), whereas t_{ij} is the duration time of activity k in arc (i,j). The completion time at node j is by definition the set of all maximum duration time combinations of the set B_j of all nodes i preceding node j.

Random duration times are described using probability density functions. In particular, PERT assumes that each activity's duration is given by a beta density function. Range and shape parameters are required to specify beta density functions. The range parameters are a and b (minimum and maximum), and the shape parameters are α and β.

PERT assumes the mean completion time at node j is the maximum of the mean completion time of all the arcs preceding node j. Let λ_j denote the mean completion time at node j for all j=1, ..., n, where $\lambda_1=0$. Then, the mean completion time at node j in PERT is given according as follows.

$$\lambda_j = \underset{i \in Bj}{\text{Max}} \left\{ \lambda_i \oplus \mu_{ij} \right\} ..... \forall j = 2...n \quad (3.2)$$

The minimum completion time must be the maximum of the minimum completion time of each and every path. The same reasoning applies to the analysis of maximum path duration times. Therefore, although equation for T_j cannot be applied to include the mean and the variance of path duration times, it can be applied to the range (minimum and maximum) in

which the randomly distributed completion time is allowed to vary. Let A_j and B_j be the minimum and maximum completion times at node j for all j=2, ..., n where A_i=0 and B_i=0. Then, the minimum and maximum completion times at each node j are given according to below equations. These are absolute bounds to the completion time at node j because no T_j can be less than A, nor greater than B, at node j, as shown below.

$$A_j = \underset{i \in B_j}{\text{Max}} \left\{ A_i \oplus a_{ij} \right\} ..... \forall j = 2...n \tag{3.3}$$

$$B_j = \underset{i \in B_j}{\text{Max}} \left\{ B_i \oplus b_{ij} \right\} ..... \forall j = 2...n \tag{3.4}$$

$$A_j < T_j < B_j$$

Above equations provide the range parameters of the project completion time (a and b) when j=n (a=A_n, and b=B_n). The shape parameters can be obtained as a function of the mean. But calculating the mean (and by extension the variance) requires knowing the probability density function of the completion time at each node. Denote the probability density function of the completion time at node j by $f_j(T)$, where $F_j(T)$ is the corresponding cumulative distribution. Assuming that all the stochastic completion times between each node i and each activity (i,j) given by the stochastic sum $f_{ij}(T)$=$f_i(T) \oplus f_{ij}(t)$ for all nodes i preceding node j ($i \in B_j$) are independently distributed, the joint cumulative distribution at node j, $F_j(T)$, is given by the product of all $F_{ij}(T)$ as shown in the equation below. (Notice that $F_{ij}(T)$ is the cumulative distribution of $f_{ij}(T)$)

$$F_j(T) = \prod_{i \in B_j} F_{ij}(T) \tag{3.5}$$

Then, the probability density function of the completion time at node j, $f_j(T)$, is given by the derivative of the cumulative distribution at node j as indicated below. The mean and the variance of the completion time at each node j, μ_j and σ_j^2, are given according to below equations for all j=2,...,n.

$$f_j(T) = \frac{\partial F_j(T)}{\partial T} \tag{3.6}$$

$$\mu_j = \int_{A_j}^{B_j} T(f_j(T))T \quad dT \tag{3.7}$$

$$\sigma_j^2 = \int_{A_j}^{B_j} (T - \mu_j)^2 f_j(T)\, dT \tag{3.8}$$

$$\alpha_k' = \left(\frac{\mu_k - a_k}{b_k - a_k}\right)\left((\mu_k - a_k)(b_k - \mu_k) - \sigma_k^2\right) \tag{3.9}$$

$$\beta_k' = \left(\frac{b_k - \mu_k}{b_k - a_k}\right)\left((\mu_k - a_k)(b_k - \mu_k) - \sigma_k^2\right) \tag{3.10}$$

Using above equations, the shape parameters of the beta distributed completion time at node j, α_j and β_j, are calculated. The probability density function of the project completion time is a beta distribution with range parameters $a=A_n$ and $b=B_n$ and shape parameters α_n and β_n.

3.3. Proposed Work

First, it is important to define what an activity duration will mean in the model to be constructed. Figure 3.4 illustrates an entire activity duration, including the timing for both the transportation of the components to the site and installation time required. In the case of deliveries from the country of origin, total time required from the shipment of goods to the end of installation will be a deterministic figure. On the other hand, if the goods are delivered from mills abroad, this total time will show a stochastic pattern and will have a probability distribution.

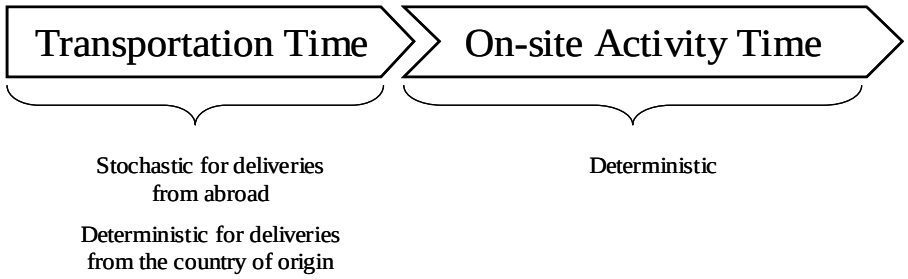

Figure 3.4. Transportation and activity durations

Let us assume that there are *m* components to be supplied for a steel structure project. For the sake of simplicity, it will be considered that these *m* components will be used in *c* different activities. Let us further assume that each of these components could be sourced from the steel mill at the country of origin, as well as two more steel mills in different locations[6]. This leaves us with a possibility to source each component from three different locations. In this case there exist 3^m different project networks, all with different probability distributions for the completion time. In each case, the supply network will also be different, leaving us with 3^m different total material cost figures.

Consequently for the optimization problem, there will be 3^m scenarios with different tardiness costs due to late completion of the project and different material costs. Naturally, the number of scenarios will be subject to change depending on the availability of components in the mills of the supply network, hence the quantity of 3^m is just given as an example. The following formula represents the cost incurred at each scenario:

$$TC_s = P(C_s - DD) + \left(\sum_{i=1}^{m} MC_{is} Q_i\right) \tag{3.11}$$

where,

TC_s is the total cost of the s^{th} scenario,

P is the penalty cost incurred for each unit time of delay,

C_s is the makespan of the network for s^{th} scenario,

[6] These two mills will be chosen among the supply network at hand.

DD is the due date of the project, after which penalty costs are incurred,
MC_{is} is the unit cost of component for the i^{th} activity in s^{th} scenario network,
Q_i is the quantity of the components needed for the i^{th} activity.

The optimization problem outlined above will be solved via utilization of a mathematical programming. C_s in the above equation is to be derived from the algorithm to be structured. In addition, for each scenario s, there will be a certain unit material cost for the components needed for each activity. Depending on from which mill in the supply network a component is sourced in the corresponding scenario, the algorithm will insert this cost figure into the calculations.

Undoubtedly, a Contractor should choose the optimal supply network in the scenario with the lowest total cost.

4. MATHEMATICAL PROGRAMMING TYPES

Mathematical programming types could be used for solving optimization problems. Within this chapter properties of Deterministic Linear Programming, Dynamic Programming, and Stochastic Programming will be provided for the evaluation of most viable programming type for the concerning optimization problem.

4.1. Deterministic Linear Programming

A deterministic linear program deals with finding a solution to

$$\min z = c^T x \tag{4.1}$$

subject to

$$Ax = b$$
$$x \geq 0$$

where x is an *(nx1)* vector of decisions and c, A, and b are known data of sizes *(nx1)*, *(mxn)*, and *(mx1)* respectively. The value $z=c^T x$ corresponds to the objective function, while

$$\{x \mid Ax = b, x \geq 0\} \tag{4.2}$$

defines the set of feasible solutions. An optimum x^* is a feasible solution such that

$$c^T x \geq c^T x^* \tag{4.3}$$

for any feasible x. Linear programs typically search for a minimal-cost solution under some requirements (e.g. demand) to be met or for a maximum profit solution under limited resource [16].

This programming type is not suitable for the solution of the problem in hand, since the problem set includes stochastic variables, whereas deterministic linear programming solves only problems with deterministic data.

4.2. Dynamic Programming

Dynamic programming solves the problems in stages, with each stage involving exactly one optimizing variable. The computations at different stages are linked through recursive computations in a manner that yields a feasible optimal solution to the entire problem. The main unifying theory in dynamic programming is the principle of optimality. It basically dictates how a properly decomposed problem may be solved in stages – rather than as one entity – through the use of recursive computations [17].

There are two solution methods in DP; forward procedure and backward procedure. If the computations start at the last stage and then proceed backwards to stage 1, this method is then named backward procedure. The opposite method, where computing starts at stage 1 and proceeds forward to last stage, is named as forward procedure. Main difference between these two methods is the way states of the system are defined.

Dynamic programming is interesting as a means of solving problems that evolve over time. Typical examples are production planning under varying demand, capacity expansion to meet an increasing demand and investment planning. Dynamic programming, in fact, can also be used to solve problems that are not sequential in nature [18].

Important concepts in dynamic programming are the time horizon, state variables, decision variables, return functions, accumulated return functions, optimal accumulated returns and transition functions. The time horizon refers to the number of stages (time periods) in the problem. State variables describe the state of the system, for example the present production capacity, the present age and species distribution in a forest or the amount of money one has in different accounts in a bank. Decision variables are the variables under one’s control. They can represent decisions to build new plants, to cut a certain amount of

timber, or to move money from one bank account to another. The transition function shows how the state variables change as a function of decisions. That is, the transition function dictates the state that will result from the combination of the present state and the present decisions. For example, the transition function may show how the forest changes over the next period as a result of its present state and of cutting decisions, how the amount of money in the bank increases, or how the production capacity will change as a result of its present size. A return function shows the immediate returns (costs or profits) as a result of making a specific decision in a specific state. Accumulated return functions show the accumulated effect, from now until the end of the time horizon, associated with a specific decision in a specific state. Finally, optimal accumulated returns show the value of making the optimal decision based on an accumulated return function, or in other words, the best return that can be achieved from the present state until the end of the time horizon [18].

The main idea behind dynamic programming is to take one stage at a time, starting with the last stage. For each stage, find the optimal decision for all possible states, thereby calculating the optimal accumulated return from then until the end of the time horizon for all possible states. Then move one step towards the present, and calculate the returns from that stage until the end of the time horizon by adding together the immediate returns, and the returns for all later periods based on the calculations made at the previous stage. In each case the problem must be solved for all possible values of the state variable z_t, which might be multi-dimensional [18].

4.2.1. Stochastic Dynamic Programming

There are two major properties of the solution procedure of dynamic programming. First, the procedure (i.e. dynamic programming) produces one solution per possible state in each stage. These solutions are not stored, since they are not needed in the procedure, but the extra cost incurred by doing so would be minimal. Secondly, if there is only one given value for the initial state z_0, we can use these decisions to produce a series of optimal solutions—one

for each stage. In other words, given an initial state, we can make plans for all later periods [18].

In the area of stochastic dynamic programming, we shall keep one property of the dynamic programming algorithm, namely that there will be one decision for each state in each stage, but it will no longer be possible to plan for the whole period ahead of time. Decisions for all but the first period will depend on what happens in the mean time. This is the same as is observed for stochastic decision trees [18].

A stochastic decision tree, as a method for solving multistage stochastic problems, requires a tree that branches off for each possible decision x_t and each possible realization of $\tilde{\xi}_t$. Therefore these must both have many possible finite values. The state z_t is not part of the tree, and can therefore safely be continuous. A stochastic decision tree easily grows out of hand. The second approach is stochastic dynamic programming. There a decision for each possible state z_t in each stage t must be made. Therefore, it is clearly an advantage if there are many possible finite states. However, the theory is also developed for a continuous state space. Furthermore, a continuous set of decisions x_t is acceptable, and so is a continuous distribution of $\tilde{\xi}_t$, provided the expectation with respect to $\tilde{\xi}_t$ could be performed [18].

It is also important to determine the importance of randomness. The most straightforward way to check if randomness is unimportant is to compare the optimal objective value of the stochastic model with the corresponding optimal value of the deterministic model (probably produced by replacing all random variables by their means) [18].

4.3. Stochastic Programming

Stochastic programming deals with a class of optimization models and algorithms in which some of the data may be subject to significant uncertainty. Such models are appropriate when data evolve over time and decisions need to be made prior to observing the entire data stream. Such models yield plans that are better able to hedge against losses. Because of these

properties, stochastic programming models have been developed for a variety of applications, including electric power generation, financial planning, telecommunications network planning, and supply chain management, to mention a few. Nevertheless, stochastic programming models remain one of the more challenging optimization problems [17].

Recourse programs are those in which some decisions or recourse actions can be taken after uncertainty is disclosed. To be more precise, data uncertainty means that some of the problem data can be represented as random variables. An accurate probabilistic distribution of random variables is assumed available, under the form of probability distributions or more generally probability measures. As usual, the particular values the various random variables will take are only known after the random event, i.e., the vector $\xi = \xi(w)$ is only known after the event [16].

The set of decisions are then divided into two groups:

- A number of decisions have to be taken before the event. All these decisions are called first-stage decisions and the period when these decisions are taken is called the first-stage.
- A number of decisions can be taken after the event. They are called second-stage decisions. The corresponding stage is called the second stage [16].

First stage decisions are represented by the vector *x*, while the second-stage decisions are represented by the vector *y* or *y(w)* or even *y(w,x)* if one wishes to stress that second-stage decisions differ as functions of the outcome of the random event and of the first-stage decision – here the outcome of the random event is represented by *w* and the first stage decision is represented by *x* [16]. The sequence of events and decisions is thus summarized as

$$x \rightarrow \xi(w) \rightarrow y(w,x) \tag{4.4}$$

4.3.1. The Basics

χ represents the space of decision variables. It is assumed that there is a given set $X \subset \chi$ of feasible (or permissible) decisions and an objective function $F(x,w)$ of decision vector $x \subset X$ and random element w. In an abstract setting, w is considered an element of a sample space Ω. In typical applications, the involved random data is formed by a finite number of parameters. Consequently, the objective function is given in the form $F(x,w) := V(x,\xi(w))$, where $\xi(w)$ is a finite dimensional random vector and $V(x,\xi)$ is a function of two vector variables x and ξ [19].

The mathematical programming problem of minimization (or maximization) of $F(x,w)$ subject to $x \in X$ depends on w and does not make much sense. For different realizations of the random parameters one would obtain different optimal solutions without any insight on which one is better than the others. A way of dealing with that is to optimize the objective function on *average*. This leads to the following mathematical programming problem:

$$\min_{x \in X} \{ f(x) := \mathrm{E}[F(x,w)] \} \tag{4.5}$$

The function $f(x)$ is referred to as the expectation or expected value function. The above formulation of stochastic programming problems assumes implicitly that the expected value is taken with respect to a known probability distribution P on (Ω, F) and that the expected value operator

$$\mathrm{E}[F(x,w)] = \int_{\Omega} F(x,w)\, dP(w) \tag{4.6}$$

is well defined [19].

4.3.2. Stochastic Programming Types

4.3.2.1. Two-Stage Stochastic Linear Programs with Fixed Recourse The basic two-stage stochastic linear programming with fixed recourse is formulated as follows:

$$\min z = c^T x + \mathrm{E}_{\xi}\left[\min q(w)^T y\right] \tag{4.7}$$

subject to

$$Ax = b$$
$$T(w)x + Wy = h(w)$$
$$x \geq 0, y \geq 0$$

where c and b are known vectors, A and W are known matrices of size m_1xn_1 and m_2xn_2 respectively, and W is called the *recourse matrix*, which we assume here is fixed. For each w, $T(w)$ is m_2xn_1 [16].

The so-called deterministic equivalent program (DEP) of the above problem is:

$$\min z = c^T x + Q(x) \tag{4.8}$$

subject to

$$Ax = b$$
$$x \geq 0$$

where,

$$Q(x) = \mathrm{E}_{\xi}(Q(x,\xi))$$

and

$$Q(x,\xi(w)) = \min\{q(w)^T y \mid Wy = h(w) - T(w)x, y \geq 0\}$$

This DEP apply for both discrete and continuous random variables.

This representation clearly illustrates the sequence of events in the recourse problem. First-stage decisions *x* are taken in the presence of uncertainty about future realizations of ξ. In the second stage, the actual value of ξ becomes known and some corrective actions or recourse decisions *y* can be taken. First-stage decisions are, however, chosen by taking their future effects into account. These future effects are measured by the value function or recourse function, $Q(x)$, which computes the expected value of taking decision *x*. The difficulty inherent in stochastic programming clearly lies in the computational burden of computing $Q(x)$ for all *x* [16].

Two-stage stochastic linear programming clearly cannot be used for the solution of the cost optimization problem outlined in this study, because there are more than two stages in the activity network to be optimized.

4.3.2.2. Probabilistic or Chance Constraints Constraints in a stochastic programming model do not have to hold deterministically. They could instead hold with some probability or reliability level. This type of probabilistic or chance constraints are of the form:

$$P\left\{A^i(w)x \geq h^i(w)\right\} \geq \alpha^i \tag{4.9}$$

where $0 < \alpha^i < 1$ and $i=1,\ldots,I$ is an index of the constraints that must hold jointly. One could model these constraints in a general expectational form $E_w(f^i(w, x(w)) \geq \alpha^i$ where *fi* is an indicator of $\left\{w \middle| A^i(w)x \geq h^i(w)\right\}$ but then there will be a need to deal with a discontinuous function [17].

$$\min_{x \in X} E_\xi g_0(x, \tilde{\xi}) \tag{4.10}$$

subject to

$$P\left\{\xi \middle| g_i(x, \xi) \leq 0, i = 1,\ldots m\right\} \geq \alpha$$

The above problem is called a probabilistically constrained or chance constrained program (or a problem with joint probabilistic constraints) [18].

If we define $\alpha^i \in [0,1], i = 1,...m$ and analogous "payoffs" for every single constraint, resulting in

$$f_0(x,\xi) = (g_0)x,\xi \tag{4.11}$$

$$f_i(x,\xi) = \alpha_i \text{ , if } g_i(x,\xi) \le 0 \quad i=1,\ldots,m$$

$$f_i(x,\xi) = \alpha_i, \text{ otherwise } \quad i=1,\ldots,m$$

then we get the problem with single (or separate) probabilistic constraint:

$$\min_{x\in X} E_\xi g_0(x,\tilde{\xi}) \tag{4.12}$$

subject to

$$P\{\xi | g_i(x,\xi) \le 0\} \ge \alpha_i, i = 1,...m$$

If, in particular, we have that the functions $g_i(x,\xi)$ are linear in x, and if furthermore the set X is convex polyhedral, i.e. we have the stochastic linear program.

$$\min z = c^T(\tilde{\xi})x \tag{4.13}$$

subject to

$$Ax = b$$

$$T(\tilde{\xi})x \ge h-\tilde{\xi}$$

$$x \ge 0$$

then the problem becomes

$$\min_{x\in X} E_\xi c^T(\tilde{\xi})x \tag{4.14}$$

subject to

$$P\{\xi \mid T(\xi)x \geq h(\xi)\} \geq \alpha$$

and, with $T_i(.)$ and $h_i(.)$ denoting the ith row and ith component of $T(.)$ and $h(.)$ respectively,

$$\min_{x \in X} E_{\xi} c^T(\xi) x \tag{4.15}$$

subject to

$$P\{\xi \mid T_i(\xi)x \geq h_i(\xi)\} \geq \alpha_i, i = 1,....,m$$

the stochastic linear programs with joint and with single chance constraints respectively.

Obviously there are many other possibilities to generate types of deterministic equivalents for a stochastic programming model [18].

In chance constraint programming the objective function is often an expectational functional (the *E-model*), or it may be the variance of some result (the *V-model*), or the probability of some occurrence such as satisfying the constraints (the *P-model*). Another variation includes an objective that is a quantile of a random function. This modeling cannot be utilized for the solution of the optimization problem on the activity network, because the data of the optimization problem do not fit into any of the above categories.

4.3.2.3. Stochastic Integer Programs In this class of problems probabilistic constraints do not exist. In a stochastic mixed integer linear program (S-MILP), if only the first stage decisions include integer restrictions, then the remaining problem inherits the properties of a Stochastic Linear Program. In general though (i.e when integer variables appear in future stages) the S-MILP is much more challenging.

Formulation for a two-stage integer program is similar to the two-stage linear case. The only variation is that some variables, either in the first stage or the second stage, are integer. In

many practical situations the variables must be binary – taking only values 0 or 1. The formulation is as follows:

$$\min_{x \in X} z = c^T x + \mathrm{E}_{\xi} \min \left\{ q(w)^T y \mid Wy = h(w) - T(w)x, y \in Y \right\} \qquad (4.16)$$

subject to

$$Ax = b$$

where the definitions of c, b, ξ, A, W, T, and h are as before. However, X and/or Y contains some integrality or binary restrictions on x and/or y. Thus, we could define the DEP as follows:

$$\min_{x \in X} z = c^T x + Q(x) \qquad (4.17)$$

subject to

$$Ax = b$$

where $Q(x)$ the expected value of the second stage was defined earlier [16].

Finding $Q(x)$ for a given x becomes an extremely difficult task for a general integer second stage. Cases where $Q(x)$ can be computed in a reasonable amount of time should be considered exceptions. One such exception is the case of Simple Integer Recourse.

Let ξ be a random vector with expectation μ, and cumulative distribution F with $F(t) = \mathrm{P}\{\xi \le t\}, t \in R^m$. A two-stage stochastic program with simple integer recourse is as follows [13]:

$$\min z = c^T x + \mathrm{E}_{\xi} \left\{ \min(q^+)^T y^+ + q^{-T} y^- \right\} \qquad (4.18)$$

$$y^+ \ge \xi - Tx, y^- \ge Tx - \xi$$

$$y^+ \in Z_+^m, y^- \in Z_+^m$$

subject to

$$Ax = b, x \in X$$

Apparently, the S-MILP is a closer modeling for the solution of the optimization problem on an activity network. Because the decision variables of the optimization problem to be solved are binary – either 0 for a mill, meaning the concerning component will not be supplied from that mill, or 1 for the same mill, meaning this component will be supplied from that mill. Nevertheless, S-MILP models are not appropriate for the optimization of such activity networks, since these networks are composed of many stages and all these stages include these binary (in other words integer) variables. Computing time would become burdensome in such solutions.

4.3.2.4. Two-Stage Stochastic Nonlinear Programs with Recourse The definition of the two-stage stochastic nonlinear program with (additive) recourse is given below. The additive form of the recourse is used to obtain separation of the first- and second- period problems [16].

$$\inf z = f^1(x) + Q(x) \tag{4.19}$$

subject to

$$g_i^1(x) \le 0, i = 1,\ldots,\bar{m}_1$$

$$g_i^1(x) = 0, i = \bar{m}_1,\ldots, m_1$$

where

$$Q(x) = E_w[Q(x,w)]$$

$$Q(x,w) = \inf\{f^2(y) \mid t_i^2(x,w) \le 0, i = 1,\ldots,\bar{m}_2\}$$

subject to

$$t_i^2(x,w) = g_i^2(y) - w, w \le 0, i = 1,\ldots,\bar{m}_2$$

$$t_i^2(x,w) = g_i^2(y) - w, w = 0, i = \bar{m}_2 + 1,\ldots, m_2$$

where all functions $f^2(.,w)$, $t_i^2(.,w)$, and $g_i^2(.,w)$ are continuous for any fixed *w* and measurable in *w* for any first argument. Given this assumption, $Q(x,w)$ is measurable and hence $Q(x)$ is well defined [16].

Since the optimization problem we are trying to solve is not non-linear, this modeling type is also not appropriate for the solution of the problem subject to this study.

4.3.2.5. Multistage Stochastic Programs with Recourse Up to this section, two-stage stochastic programs were investigated. In reality, most decision problems involve a sequence of decisions that react to outcomes that evolve over time. Thus, there exists a need to investigate multistage problems.

The multistage stochastic linear program with fixed recourse has the following form:

$$\min z = c^1x^1 + E_{\xi^2}\left[\min c^2(w)x^2(w) + \ldots + E\left[\min_{\xi^H}\left[(c^H(w)x^H(w)\right]\right]\right] \quad (4.20)$$

subject to

$$W^1x^1 = h^1$$

$$T^1(w)x^1 + W^2x^2(w) = h^2(w)$$

$$\ldots\ldots$$

$$T^{H-1}(w)x^{H-1}(w) + W^Hx^H(w) = h^H(w)$$

$$x^1 \geq 0, x^t(w^t) \geq 0, t = 2,\ldots,H$$

where c^1 and h^1 are known vectors, $\xi^t(w)^T = (c^t(w)^T + h^t(w)^T, T_1^{t-1}(w),\ldots,T_{m_t}^{t-1})$ is a random N_t-vector defined on $(\Omega, \sum^t, P)$ (where $\sum^t \in \sum^{t+1}$) for all *t*=2,…,H, and each W^t is a known $m_t x n_t$ matrix. The decisions *x* depend on the history up to time *t*, which we indicate by w^t [16].

The DEP of this problem should be described in terms of a dynamic program. If the stages are 1 to *H*, we could define states as $x^t(w^t)$. Noting that the only interaction between

periods is through this realization, we could define a dynamic programming type of recursion. For terminal conditions we have:

$$Q^H(x^{H-1}, \xi^H) = \min \{ (c^H)^T x^H \} \tag{4.21}$$

subject to

$$W^H x^H = h^H - T^H x^{H-1}$$
$$W^H x^H = h^H - T^H x^{H-1}$$

Letting $Q^{t+1}(x^t) = E_{\xi^{t+1}} \left[Q^{t+1}(x^t, \xi^{t+1}) \right]$ for all t, we obtain the recursion for t=2,……,H-1.

$$Q(x^{t-1}, \xi^t) = \min \{ (c^t)^T x(x^t) + Q^{t+1}(x^t) \} \tag{4.22}$$

subject to

$$W^t x^t = h^t - T^{t-1} x^{t-1}$$
$$x^t \geq 0$$

where we use x^t to indicate the state of the system. Other state information in terms of the realizations of the random variables up to time t should be included if the distribution of ξ^t is not dependent on the past outcomes [16].

The value seeked is:

$$\min z = c^1 x^1 + Q(x^1) \tag{4.23}$$

subject to

$$W^1 x^1 = h^1$$
$$x^1 \geq 0$$

which has the same form as the two-stage DEP.

A difficulty with multi-stage stochastic programs is that these problems become extremely large as the number of the stages increases, even if only a few realizations are allowed at each stage [16].

Having multiple stages, this modeling type is also desirable for the solution of the optimization problem on the activity network. However, since the decision variables in the stages will be binary – thus integer – this modeling type is also not applicable to the optimization problem subject to this study. Only if some variables were to be identified as ineteger – or more explicitly binary -, this model would be utilized for the desired solution. However, the solution would have so many scenarios that the computation would not be possible within tolerable timing.

4.3.3. Computational Challenges

The computational challenges associated with SP problems vary a great deal with the class of problems being addressed. As with any large-scale optimization problem, exploiting properties and the structure of problems provides the key to effective algorithms. The main computational challenges can be attributed to presence of multi-dimensional integration (to calculate either expectation or probability) within an optimization algorithm. Even in cases where the random variables are discrete, the total number of outcomes of a multi-dimensional random vector can be so large that calculations associated with the summations may be far too demanding. Hence even in the case of discrete random variables, one may have to resort to approximations. Discretizations and/or aggregations in multistage problems result in alternative data scenarios or sample paths. These scenarios may be organized in the form of a scenario tree which is a structure representing the evolution of information over the stages. In such a tree, two scenarios that share a common history until stage *t* are indistinguishable until that stage, and thereafter they are represented by distinct paths. Thus every distinct scenario represents a path from the root node to a leaf node of the scenario tree. In the absence of appropriate approximations, these trees can become extremely large, and the model difficult to manage and solve.

One of the more demanding problems in stochastic programming involves the solution of S-MILP problems. In cases where the first stage has binary variables, Laporte and Louveaux [1993] have proposed an extension of the so-called L-shaped method for two stage S-MILP problems. Unfortunately, it requires that the second stage problem (possibly a mixed-integer linear program (MILP)) be solved to optimality. Given the computational difficulties associated with MILPs, this is cumbersome. One possible way to alleviate this difficulty may involve an extension that incorporates the stochastic branch and bound algorithm proposed by Norkin, Ermoliev and Ruszczynski [1998]. Since the latter allows the use of sampled bounds, it may provide a more computationally feasible approach to practical S-MILP problems [17].

Finally, one area that has not attracted as much attention as it should is the development of software systems, which integrate stochastic modeling with stochastic programming algorithms. Such a system would provide software tools to build models, validate them, and experiment with alternative algorithms. With few exceptions (e.g. Kall and Mayer [1996] and Gassmann and Ireland [1996]), there has been relatively little activity in this important area. Stochastic programming would only be able to attain its potential with the development of systems that allow easy interactions between stochastic models and stochastic programming algorithms. The development of a high level language or system that allows manipulation and representation of models and data, together with the ability to experiment with alternative solvers, is long overdue [17].

5. APPLICATION OF DETERMINISTIC LINEAR PROGRAMMING ON THE PROBLEM

It would be worthwhile to try to tackle the decision problem in hand with a deterministic model first and only afterwards switch to a stochastic model. Accordingly, in this chapter, a deterministic model will be developed for a fictitious network setting. Presentation of a fictitious case will enable the readers to visualize the model to be structured better. Subsequently, in chapter 6, a stochastic model, which would represent the case in real life much more rigorously, will be constructed.

Let us now assume a steel construction project composed of seven activities. In reality, a steel construction project involves many more activities. However, for the sake of simplicity, the number of activities is kept at seven for this sample case. The below AOA network provides the sequence of these activities.

The nodes in the below project schedule denote the events (stages as named in stochastic programming), whereas the arcs between the nodes represent the activities. In the rectangles on the arcs; the first number denotes the number of the activity and the figure in parentheses denotes the duration of that specific activity in days. For instance the first activity lasts three days.

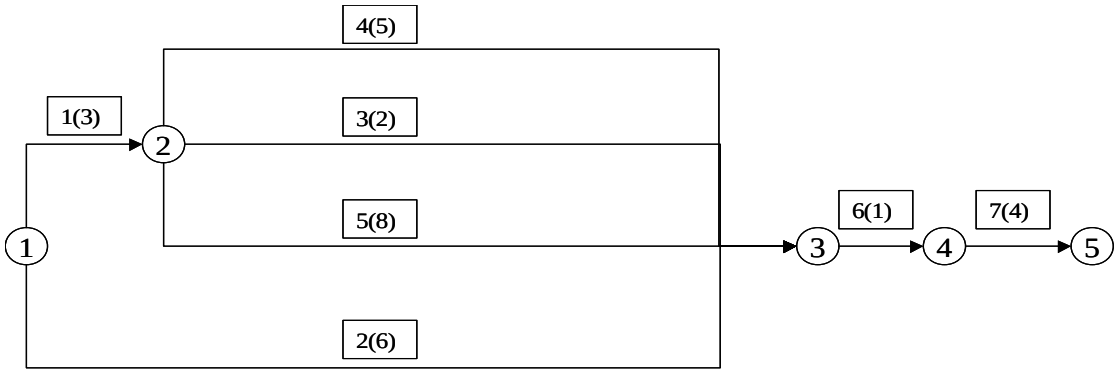

Figure 5.1. Schedule of the project

To make the above sample case more realistic, some activities could be associated to each arc:

- Arc 1: Erection of columns
- Arc 2: Welding of roof trusses
- Arc 3: Installation of bracings
- Arc 4: Installation of beams
- Arc 5: Installation of façade cladding
- Arc 6: Installation of trusses
- Arc 7: Installation of roofing

In accordance with the above illustrated base case, the project is supposed to be finalized in 16 days, as also computed in the below Earliest Start Date table. Since the nodes represent events, the earliest start date of a node represents the end of all the preceding activities.

Table 5.1. Earliest start dates of nodes

Stage:	Node:	Earliest Start (days):
1	1	0
2	2	3
3	3	11
4	4	12
5	5	16

In addition, the Earliest Start, Earliest Finish, Latest Start, and Latest Finish Dates of activities are provided in Table 5.2 below. The cells highlighted with shading show the activities on the Critical Path.

Table 5.2. ES, EF, LS, LF dates of activities

Activity:	Earliest Start (days)	Earliest Finish (days)	Latest Start (days)	Latest Finish (days)
1	0	3	0	3
2	0	6	5	11
3	3	5	9	11
4	3	8	6	11
5	3	11	3	11
6	11	12	11	12
7	12	16	12	16

It will be assumed that there are six available manufacturing plants for the supply of steel components – P = {1 (the local plant), 2, …, 6}. Table 5.3 below lists the effect of supplying components from a specific plant on the number of days an activity takes due to potential delays in delivery of components[7].

Table 5.3. Effect of supplying components from a specific plant on activity durations

Plant:	Extra Days Needed for an Activity to be Finalized:
1	0 (this is the local plant)
2	1
3	2
4	1
5	3
6	2

Table 5.4 below provides the set of plants, which could supply components needed for an associated activity. This selection is done on a random basis; it is assumed that different sets of plants, each having three elements, would supply for each activity. In the same table, effects of plant selection on the activity durations are also provided. As a consequence, total activity durations for different selections are determined.

In accordance with this availability table, there would be $3^6.2=1{,}458$ [(8)] different project time-plans depending on the selection of the plant for each activity. In this network of activities, all tasks are considered to require material supply. In effect, a construction project has much more than seven activities, thus, this fictitious activity network is considered the network of material supply required activities only. It should be noted that the local plant (plant # 1) is considered an option for all the activities and deliveries from this plant do not delay the finalization of any activity.

[7] The delays are given as deterministic in this table. In real life, no such case would exist; because if the exact delay was known upfront, the goods would be ordered earlier to offset the effect of delay. This contradiction will be resolved in the stochastic model to be formulated in the forthcoming chapter.

[8] For activity 2, in case of supply from both plants 3 and 4, the activity would take 7 days. Thus, number of different scenarios for the project schedule should be $3^6.2=1{,}458$, rather than $3^7=2{,}187$.

Table 5.4. Supply options and associated delays on activity durations

Activity:	Plant options: (Delays on Activity Durations - Days)			Plant options: Total Activity Durations - Days		
1	1	3	4	1	3	4
	(0)	(2)	(1)	3	5	4
2	1	2	4	1	2	4
	(0)	(1)	(1)	6	7	7
3	1	4	5	1	4	5
	(0)	(1)	(3)	2	3	5
4	1	2	6	1	2	6
	(0)	(1)	(2)	5	6	7
5	1	3	5	1	3	5
	(0)	(2)	(3)	8	10	11
6	1	5	6	1	5	6
	(0)	(3)	(2)	1	4	3
7	1	2	3	1	2	3
	(0)	(1)	(2)	4	5	6

These total costs are determined via the multiplication of the quantity of components for each activity by the unit price of the concerning component when supplied from the associated plant ($MC_{ij} = UC_{ij}$ x Q_i).

Given all the above data, the decision to be made is from which plant to supply components for each activity in the most cost effective way. The following deterministic mixed integer linear program would provide the answer to our dilemma under these circumstances.

Table 5.5. Total cost of supply from optional plants[9]

Activity:	Plant options: Total Cost of Supply - MCij (USD)		
1	1	3	4
	USD 10.000	USD 6.000	USD 7.000
2	1	2	4
	USD 42.000	USD 40.000	USD 45.000
3	1	4	5
	USD 20.000	USD 15.000	USD 10.000
4	1	2	6
	USD 35.000	USD 30.000	USD 25.000
5	1	3	5
	USD 64.000	USD 55.000	USD 50.000
6	1	5	6
	USD 20.000	USD 8.000	USD 9.000
7	1	2	3
	USD 12.000	USD 10.000	USD 9.000

[9] A heuristic simplification is hidden in the values provided in this table. For instance, why would one choose to get supply from plant 4 for activity 2 for 45,000 USD, when there is a chance to get cheaper supply from the local plant 1 without any delays for only 42,000 USD? This is named “domination” in decision problems.

Let:

P_{ij} = plant selection for the i^{th} activity from set j;

where

i = 1, 2, 3, 4, 5, 6, 7 - activities;

j = 1, 2, 3, 4, 5, 6 - plants;

ES_i = earliest start date of activity i

EF_i = earliest finish date of activity i;

$S7^+$ = if activity 7 finished earlier, number of days from EF_7 to the due date of the project;

$S7^-$ = if activity 7 finished later, number of days from due date of the project to EF_7;

D_{ij} = duration of activity i, if supplied from plant j;

MC_{ij} = material cost of activity i, if supplied from plant j;

TC = tardiness cost per day (fixed penalty per a day of late completion);

The Linear Program to be solved is provided below:

Objective Function:

$$\min z = S_7^- TC + \sum_{i=1}^{7}\sum_{j=1}^{6} P_{ij} MC_{ij} \qquad (5.1)$$

The extensive form of the objective function is:

$$\min z = S_7^-.TC +$$
$$10{,}000P_{11}+0P_{12}+6{,}000P_{13}+7{,}000P_{14}+0P_{15}+0P_{16}+$$
$$42{,}000P_{21}+40{,}000P_{22}+0P_{23}+45{,}000P_{24}+0P_{25}+0P_{26}+$$
$$20{,}000P_{31}+0P_{32}+0P_{33}+15{,}000P_{34}+10{,}000P_{35}+0P_{36}+$$
$$35{,}000P_{41}+30{,}000P_{42}+0P_{43}+0P_{44}+0P_{45}+25{,}000P_{46}+$$
$$64{,}000P_{51}+0P_{52}+55{,}000P_{53}+0P_{54}+50{,}000P_{55}+0P_{56}+$$
$$20{,}000P_{61}+0P_{62}+0P_{63}+0P_{64}+8{,}000P_{65}+9{,}000P_{66}+$$
$$12{,}000P_{71}+10{,}000P_{72}+9{,}000P_{73}+0P_{74}+0P_{75}+0P_{76}$$

Constraints:

Supply Constraints:

$P_{ij} \in \{0,1\}$ - meaning P_{ij} are binary variables – allowed to take values either 0 or 1 only

$$\sum_{j=1}^{6} P_{ij} = 1 \quad i=1, 2,\ldots,7 \tag{5.2}$$

meaning components for an activity could only be supplied from one plant[10].

The extensive form of the above constrain is:

$P_{11}+P_{12}+P_{13}+P_{14}+P_{15}+P_{16} = 1$ – for activity 1

$P_{21}+P_{22}+P_{23}+P_{24}+P_{25}+P_{26} = 1$ – for activity 2

$P_{31}+P_{32}+P_{33}+P_{34}+P_{35}+P_{36} = 1$ – for activity 3

$P_{41}+P_{42}+P_{43}+P_{44}+P_{15}+P_{16} = 1$ – for activity 4

$P_{51}+P_{52}+P_{53}+P_{54}+P_{55}+P_{56} = 1$ – for activity 5

$P_{61}+P_{62}+P_{63}+P_{64}+P_{65}+P_{66} = 1$ – for activity 6

$P_{71}+P_{72}+P_{73}+P_{74}+P_{75}+P_{76} = 1$ – for activity 7

We could not supply components for some activities from some plants – as per Table 5.4:

$P_{12} = P_{15} = P_{16} = 0$ – we could not supply components for activity 1 from plants 2, 5, 6 (5.3)

$P_{23} = P_{25} = P_{26} = 0$ – we could not supply components for activity 2 from plants 3, 5, 6 (5.4)

$P_{32} = P_{33} = P_{36} = 0$ – we could not supply components for activity 3 from plants 2, 3, 6 (5.5)

$P_{43} = P_{44} = P_{45} = 0$ – we could not supply components for activity 4 from plants 3, 4, 5 (5.6)

$P_{52} = P_{54} = P_{56} = 0$ – we could not supply components for activity 5 from plants 2, 4, 6 (5.7)

$P_{62} = P_{63} = P_{64} = 0$ – we could not supply components for activity 6 from plants 2, 3, 4 (5.8)

$P_{74} = P_{75} = P_{76} = 0$ – we could not supply components for activity 7 from plants 4, 5, 6 (5.9)

There are no additional supply constraints considered. The optional supply plants provided in Table 5.4 are assumed to have adequate capacity for the demand needed for that

[10] Components for an activity could actually be supplied from more than a single plant in portions. This would provide the flexibility to order urgent deliveries from the local plant with a higher unit price, and the less urgent deliveries from abroad with better unit prices. However this is not a common practice in real life and would unnecessarily complicate the model to a great extent.

specific activity. For further studies, some supply constraints on the capacities of plants can easily be introduced to the model.

Precedence Constraints:

$$ES_1 = 0 \quad \text{– earliest start date for activity 1} \tag{5.10}$$

$$ES_2 = 0 \quad \text{– earliest start date for activity 2} \tag{5.11}$$

$$ES_3 \geq ES_1 + \sum_{j=1}^{6} D_{1j} P_{1j} \quad \text{– earliest start date for activity 3} \tag{5.12}$$

The extensive form of this constraint is:

$$ES_3 = ES_1 + 3P_{11} + 5P_{13} + 4P_{14}$$

$$ES_4 \geq ES_1 + \sum_{j=1}^{6} D_{1j} P_{1j} \quad \text{– earliest start date for activity 4} \tag{5.13}$$

The extensive form of this constraint is:

$$ES_4 = ES_1 + 3P_{11} + 5P_{13} + 4P_{14}$$

$$ES_5 \geq ES_1 + \sum_{j=1}^{6} D_{1j} P_{1j} \quad \text{– earliest start date for activity 5} \tag{5.14}$$

The extensive form of this constraint is:

$$ES_5 = ES_1 + 3P_{11} + 5P_{13} + 4P_{14}$$

$$ES_6 \geq ES_2 + \sum_{j=1}^{6} D_{2j} P_{2j} \quad \text{– earliest start date for activity 6} \tag{5.15}$$

The extensive form of this constraint is:

$$ES_6 \geq ES_2 + 6P_{21} + 7P_{22} + 7P_{24}$$

$$ES_6 \geq ES_3 + \sum_{j=1}^{6} D_{3j} P_{3j} \quad \text{– earliest start date for activity 6} \qquad (5.16)$$

The extensive form of this constraint is:

$$ES_6 \geq ES_3 + 2P_{31} + 3P_{34} + 5P_{35}$$

$$ES_6 \geq ES_4 + \sum_{j=1}^{6} D_{4j} P_{4j} \quad \text{– earliest start date for activity 6} \qquad (5.17)$$

The extensive form of this constraint is:

$$ES_6 \geq ES_4 + 5P_{41} + 6P_{42} + 7P_{46}$$

$$ES_6 \geq ES_5 + \sum_{j=1}^{6} D_{5j} P_{5j} \quad \text{– earliest start date for activity 6} \qquad (5.18)$$

The extensive form of this constraint is:

$$ES_6 \geq ES_5 + 8P_{51} + 10P_{53} + 11P_{55}$$

$$ES_7 \geq ES_6 + \sum_{j=1}^{6} D_{6j} P_{6j} \quad \text{– earliest start date for activity 7} \qquad (5.19)$$

The extensive form of this constraint is:

$$ES_7 = ES_6 + 1P_{61} + 4P_{65} + 3P_{66}$$

$$EF_7 \geq ES_7 + \sum_{j=1}^{6} D_{7j} P_{7j} \quad \text{– earliest finish date for activity 7} \qquad (5.20)$$

The extensive form of this constraint is:

$$EF_7 = ES_7 + 4P_{71} + 5P_{72} + 6P_{73}$$

$$EF_7 + S7^+ - S7^- = 18 \quad \text{– for the compilation of } S_7^- \qquad (5.21)$$

Non-negativity Constraints

$$ES_i, EF_7, S7^+, S7^- \geq 0 \qquad (5.22)$$

The Mixed Integer Linear Programming Model[11] formulation for the above problem is provided in Appendix B. The concerning model is applied and the optimal results are computed in GAMS, which is a software used for solving linear, non-linear, and mixed integer optimization problems.

Utilizing the model in Appendix B, leaving all the variables unchanged, some optimization results are derived for different values of *TC* – tardiness cost. Plant selections for each activity, associated material costs, project finalization dates, associated tardiness costs, and total costs are provided in Table 5.6 below.

Table 5.6. Plant selections, project makespans, and total costs for different *TC*

Activity:	1	2	3	4	5	6	7
				Plant Selction for Each Activity			
Tardiness Cost (USD/Day):							
$ 20.000	1	2	5	6	1	6	1
$ 10.000	1	2	5	6	1	6	1
$ 5.000	1	2	5	6	1	6	1
$ 2.500	4	2	5	6	5	6	1
$ 1.000	4	2	5	6	5	6	2

				Material Cost for Each Activity			
Tardiness Cost (USD/Day):							
$ 20.000	$ 10.000	$ 40.000	$ 10.000	$ 25.000	$ 64.000	$ 9.000	$ 12.000
$ 10.000	$ 10.000	$ 40.000	$ 10.000	$ 25.000	$ 64.000	$ 9.000	$ 12.000
$ 5.000	$ 10.000	$ 40.000	$ 10.000	$ 25.000	$ 64.000	$ 9.000	$ 12.000
$ 2.500	$ 7.000	$ 40.000	$ 10.000	$ 25.000	$ 50.000	$ 9.000	$ 12.000
$ 1.000	$ 7.000	$ 40.000	$ 10.000	$ 25.000	$ 50.000	$ 9.000	$ 10.000

	Total Material Cost	Project Makespan	Tardiness Cost	Total Cost
Tardiness Cost (USD/Day):				
$ 20.000	$ 170.000	18	$ -	$ 170.000
$ 10.000	$ 170.000	18	$ -	$ 170.000
$ 5.000	$ 170.000	18	$ -	$ 170.000
$ 2.500	$ 153.000	22	$ 10.000	$ 163.000
$ 1.000	$ 151.000	23	$ 5.000	$ 156.000

[11] Mixed Integer Linear Programming Model is used for the solution of this problem, because P_{ij} are binary variables and can only assume integer values of 0 or 1.

As illustrated above, the solution matrix does not change for the cases of 20,000 USD, 10,000 USD, and 5,000 USD per day tardiness costs. For all these tardiness costs, it does not make sense to source cheaper materials when there is a risk to prolong the finalization of the project and having to pay penalty charges, which are very significant compared to material cost. The project is finalized just on time and no penalties are charged to the contractor. However, as the tardiness cost decreases to bearable levels – e.g. 2,500 USD or 1,000 USD per day – the contractor might actually decide to source the components from cheaper cost plants and pay some penalty cost. For instance for both the latter two tardiness cost cases, the optimum decision would be to source the components of Activity 1 from Plant 4 rather than Plant 1, which was the optimum case for higher tardiness costs. It takes longer to source from Plant 4, but the decrease in material cost more than offsets the penalty cost due. Same situation is valid for Activity 5; even though there are 3 days difference between sourcing from Plant 1 and 5, the optimum decision would be to source from Plant 5. Accordingly the optimum project set-up gets finalized with a tardiness of 4 days (22 days in total) for the tardiness cost of 2,500 USD per day, and with a tardiness of 5 (23 days in total) days for the tardiness cost of 1,000 USD per day.

It should here be noted that the dominance rule is obeyed in the optimum results. For none of the tardiness cost options, the components are sourced from Plant 4 for Activity 2, because both Plants 1 and 2 are dominant options over Plant 4 – Plant 1 is dominant on timing and Plant 2 is dominant on supply cost.

Manipulating the due date data, while leaving all the other variables unchanged and fixing the tardiness cost to 5,000 USD/day, we obtain the results in Table 5.7.

The interpretation of the below results is four-fold:

- The total cost of the project decreases as the tight due date constraint on the project makespan is released;

- Even though there might be some tardiness cost associated with the decisions when there is a tight deadline, the contractor might choose to supply from cheaper plants, as is the case in the first and second scenarios;
- When there is a loose deadline (due date = 24 days), the contractor chooses to source from the cheapest option for all activities, with the exception of activity 1;
- If we use a tardiness cost of 10,000 USD/day instead of 5,000 USD/day and set the due date to 16 days, we end up with interesting results. The components are supplied from plants 1, 2, 5, 6, 1, 1, and 1 respectively with a total material cost of 181,000 USD. Project makespan in this case is kept at 16 days due to severe tardiness costs. If we only switch the tardiness cost to 1,000 USD/day at this scenario, we get the solution where the materials are sourced from plants 4, 2, 5, 6, 5, 5, and 2 with a total material cost of 150,000 USD, and a total tardiness cost of 8,000 USD due to a late finish of 24 days.

Table 5.7. Plant selections, project makespans, and total costs for different due dates

Activity:	1	2	3	4	5	6	7
			Plant Selction for Each Activity				
Due Date (Day):							
16	1	2	5	6	1	6	1
17	1	2	5	6	1	6	1
18	1	2	5	6	1	6	1
19	4	2	5	6	1	6	1
20	3	2	5	6	1	6	1
24	4	2	5	6	5	6	3

			Material Cost for Each Activity				
Due Date (Day):							
16	$ 10.000	$ 40.000	$ 10.000	$ 25.000	$ 64.000	$ 9.000	$ 12.000
17	$ 10.000	$ 40.000	$ 10.000	$ 25.000	$ 64.000	$ 9.000	$ 12.000
18	$ 10.000	$ 40.000	$ 10.000	$ 25.000	$ 64.000	$ 9.000	$ 12.000
19	$ 7.000	$ 40.000	$ 10.000	$ 25.000	$ 64.000	$ 9.000	$ 12.000
20	$ 6.000	$ 40.000	$ 10.000	$ 25.000	$ 64.000	$ 9.000	$ 12.000
24	$ 7.000	$ 40.000	$ 10.000	$ 25.000	$ 50.000	$ 9.000	$ 9.000

	Total Material Cost	Project Makespan	Tardiness Cost	Total Cost
Due Date (Day):				
16	$ 170.000	18	$ 10.000	$ 180.000
17	$ 170.000	18	$ 5.000	$ 175.000
18	$ 170.000	18	$ -	$ 170.000
19	$ 167.000	19	$ -	$ 167.000
20	$ 166.000	20	$ -	$ 166.000
24	$ 150.000	24	$ -	$ 150.000

There have been some research studies present in the literature on time/cost trade-off problems. The trade-off in those problems is on whether to devote additional more costly resources for activities to make them last shorter or to simply let the project makespan overrun and cause some penalty payments. The devotion of resources and resource allocations become an important factor in these problems. However, a problem incorporating the supply cost in project scheduling problems has not been considered in research studies [9].

6. APPLICATION OF STOCHASTIC PROGRAMMING ON THE PROBLEM

The deterministic model constructed in the last chapter is a beneficial method to visualize the problem. However, that model does not truly reflect the reality. The starting point of the problem in hand was that the transportation and logistics duration of the deliveries to site behaves in a stochastic nature. This stochastic nature, in turn, effects the finalization date of activities associated with these deliveries. Related to the delay in finalization date of some activities, the project might get completed with significant delays, causing the Contractor to pay substantial amounts of tardiness cost. Accordingly, one cannot consider the delays caused by deliveries from abroad to have a deterministic effect on the completion time of activities. Thus, utilization of a Stochastic Program is essential for an optimal, or at least an approximate solution of the problem.

Accordingly, it would be viable to use Stochastic Integer Programming for the solution of the problem at hand, since there are binary variables included in the model (i.e. P_{ij}). On the other hand, there obviously is a need to use the Multistage Stochastic Programming with Recourse because there are certain *stages* inherent in the schedule.

A policy where the whole vector of decision variables is anticipated at stage 1 is called a simple recourse.

Scenario immunization models anticipate decisions in x that for multistage applications may not be needed at stage $t = 1$. Frequently, the decisions for stage $t = 1$ are the only decisions to be made, since at stage $t = 2$ one may realize that some of the data has been changed, some scenarios vanish, etc. In this case, the models will usually be re-optimized in a rolling planning horizon mode. When only spot decisions (i.e., decisions for stage $t = 1$) are to be made, the information about future uncertainty is only taken into account for a better spot

decision making. This type of approach is termed full recourse. [20]. Accordingly, in the stochastic programming model to be constructed, full recourse solution will be utilized.

The deterministic model developed in the previous chapter needs to be enhanced to reflect the stochastic nature of the delivery times of materials from different plant locations. The most appropriate stochastic behaviour that fits real-life examples of material deliveries is discrete probability distribution. Accordingly, the deterministic sourcing matrix has been modified as follows in order to reflect stochastic behaviour.

Table 6.1. Supply options and lead time probabilities for the activities of the network

Activity:	1			2			3			4			5			6			7		
Plant Option:	1			1			1			1			1			1			1		
Duration:	3			6			2			5			8			1			4		
Probability:	100%			100%			100%			100%			100%			100%			100%		
Plant Option:	3			2			4			2			3			5			2		
Duration:	4	5	7	5	7	8	2	3	4	5	6	7	8	10	12	3	4	5	4	5	6
Probability:	30%	40%	30%	20%	50%	30%	20%	45%	35%	30%	40%	30%	20%	45%	35%	20%	40%	40%	20%	30%	50%
Plant Option:	4			4			5			6			5			6			3		
Duration:	3	4	5	6	7	8	4	5	6	5	7	8	10	11	13	2	3	4	4	6	8
Probability:	30%	40%	30%	20%	30%	50%	30%	40%	30%	20%	50%	30%	30%	40%	30%	35%	40%	25%	30%	40%	30%

The project network is split into stages at epochs as illustrated in Figure 1. The stages correspond to the decision points in time, where the sourcing options for different activities will be evaluated and certain plants will be assigned to certain activities for sourcing the components needed. Accordingly, the model will consist of five stages; the last one representing the finalization of the project, hence having no decision involved.

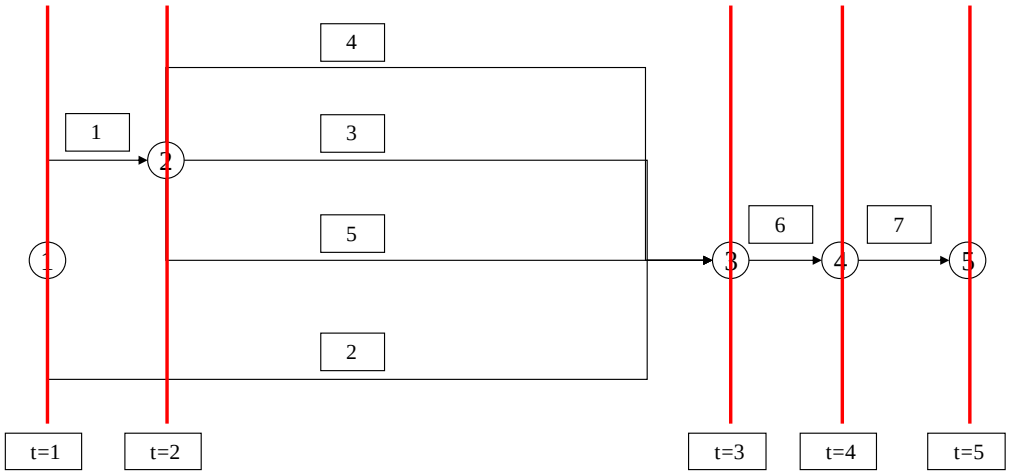

Figure 6.1. Stages on the project network

In order to utilize a stochastic programming solution for the problem at hand, the uncertainty within the supply chain optimization could be treated via a scenario analysis approach. An illustration of the scenario tree concept is provided in Figure 6.2. Each node in this figure corresponds to a point in time where a decision can be made – these nodes also correspond to the stages shown in Figure 6.1. Once a decision is made, possible outcomes might occur. For instance there are 9 possible contingencies for t=2 in this tree. Information concerning these contingencies is available at the beginning of the next stage – t=3. Each root-to-leaf path represents a scenario and is actually a realization of all the uncertain parameters. Given the supply options for the 7 activities of our network, which is provided in Table 6.1, there exist 4,782,969 possible scenarios. As a matter of fact, even a small number of activities have a huge number of possible realizations in terms of activity durations. As the project network gets bigger, the number of scenarios would easily grow out of hand.

A model that would solve this scenario analysis would require splitting variable representation – rather than the compact representation – of the DEM for the multi-stage stochastic problem at hand. Thus, the strategic 0-1 variables (P variables) would be provided as Pij^w rather than Pij. Nevertheless, this mathematical formulation brings a huge difficulty along due to the number of scenarios. We would obtain $|\Omega|$ independent scenario related 0-1 models.

For the stochastic optimization problem, we need to identify the following notation:

T = set of time periods along the time horizon;

Ω = set of scenarios;

w = scenario so that $w \in \Omega$;

w = weight (likelihood) of scenario such that $\sum_{w \in \Omega} w_w = 1$

Accordingly the deterministic version of the mathematical model should be reformulated in order to reflect the stochastic nature of the problem. Thus, the variables of the deterministic model will be transformed as follows:

$Pij \rightarrow Pij^w$ = plant selection for the i^{th} activity from set j under scenario w $(w \in \Omega)$;

where

i = 1, 2, 3, 4, 5, 6, 7 - activities;

j = 1, 2, 3, 4, 5, 6 - plants;

$ESi \rightarrow ESi^w$ = earliest start date of activity i under scenario w $(w \in \Omega)$;

$EFi \rightarrow EFi^w$ = earliest finish date of activity i under scenario w $(w \in \Omega)$;

$S_7^+ \rightarrow S_7^{+w}$ = if activity 7 finished earlier, number of days from EF_7^w to the due date of the project under scenario w $(w \in \Omega)$;

$S_7^- \rightarrow S_7^{-w}$ = if activity 7 finished later, number of days from due date of the project to EF_7^w under scenario w $(w \in \Omega)$;

$Dij \rightarrow Dij^w$ = duration of activity i, if supplied from plant j under scenario w $(w \in \Omega)$;

$MCij$ = material cost of activity i, if supplied from plant j;

TC = tardiness cost per day (fixed penalty per a day of late completion);

The Linear Program to be solved is provided below:

Objective Function:

$$min\ z = \min \sum_{w \in \Omega} w_w S_7^- TC + \sum_{i=1}^{7} \sum_{j=1}^{6} P_{ij}^w MC_{ij} \qquad (6.1)$$

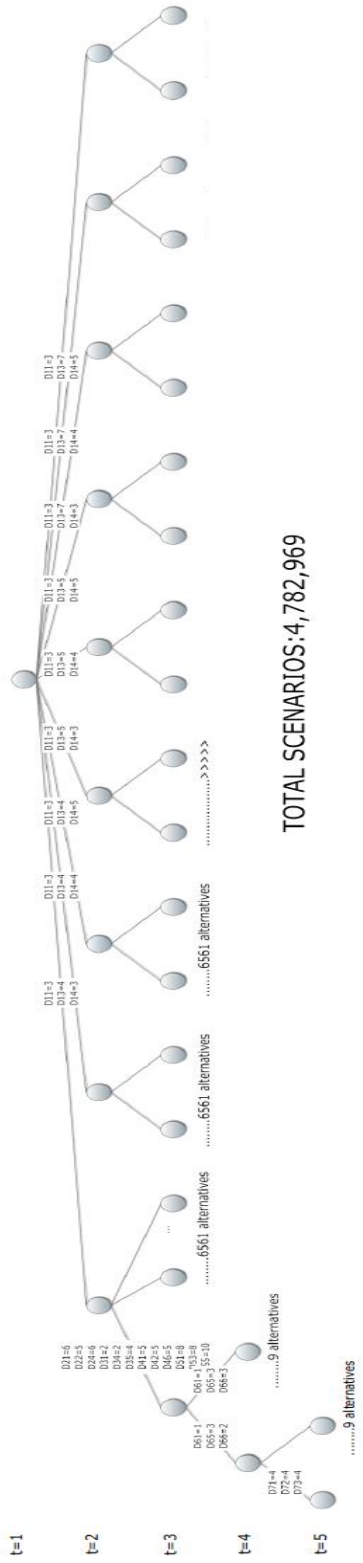

Figure 6.2. Scenario tree of the problem

Constraints:

1st Stage Constraints:

Precedence Constraints:

$ES_1^w = 0$ – earliest start date for activity 1 for scenario *w* ($w \in \Omega$) (6.2)

$ES_2^w = 0$ – earliest start date for activity 2 for scenario *w* ($w \in \Omega$) (6.3)

$$ES_3^w \geq ES_1^w + \sum_{j=1}^{6} D_{1j}^w P_{1j}^w \quad (6.4)$$

Earliest start date for activity 3 for scenario *w* ($w \in \Omega$)

$$ES_4^w \geq ES_1^w + \sum_{j=1}^{6} D_{1j}^w P_{1j}^w \quad (6.5)$$

Earliest start date for activity 4 for scenario *w* ($w \in \Omega$)

$$ES_5^w \geq ES_1^w + \sum_{j=1}^{6} D_{1j}^w P_{1j}^w \quad (6.6)$$

Earliest start date for activity 5 for scenario *w* ($w \in \Omega$)

Supply Constraints:

$P_{ij}^w \in \{0,1\}$ - meaning *Pijw* are binary variables – allowed to take values either 0 or 1 only for scenario *w* ($w \in \Omega$)

$$\sum_{j=1}^{6} P_{1j}^w = 1 \quad (6.7)$$

Meaning components for activity 1 could only be supplied from one plant for scenario *w* ($w \in \Omega$)

$P_{12}{}^{w} = P_{15}{}^{w} = P_{16}{}^{w} = 0$ – we could not supply components for activity 1 from plants 2, 5, 6 for scenario w ($w \in \Omega$) (6.8)

2nd Stage Constraints:

Precedence Constraints:

$$ES_6{}^{w} \geq ES_2{}^{w} + \sum_{j=1}^{6} D_{2j}{}^{w} P_{2j}{}^{w} \quad (6.9)$$

Earliest start date for activity 6 for scenario w ($w \in \Omega$)

$$ES_6{}^{w} \geq ES_3{}^{w} + \sum_{j=1}^{6} D_{3j}{}^{w} P_{3j}{}^{w} \quad (6.10)$$

Earliest start date for activity 6 for scenario w ($w \in \Omega$)

$$ES_6{}^{w} \geq ES_4{}^{w} + \sum_{j=1}^{6} D_{4j}{}^{w} P_{4j}{}^{w} \quad (6.11)$$

Earliest start date for activity 6 for scenario w ($w \in \Omega$)

$$ES_6{}^{w} \geq ES_5{}^{w} + \sum_{j=1}^{6} D_{5j}{}^{w} P_{5j}{}^{w} \quad (6.12)$$

Earliest start date for activity 6 for scenario w *(*$w \in \Omega$*)*

Supply Constraints:

$$\sum_{j=1}^{6} P_{2j}{}^{w} = 1 \quad (6.13)$$

Meaning components for activity 2 could only be supplied from one plant for scenario w ($w \in \Omega$)

$$\sum_{j=1}^{6} P_{3j}{}^{w} = 1 \tag{6.14}$$

Meaning components for activity 3 could only be supplied from one plant for scenario w ($w \in \Omega$)

$$\sum_{j=1}^{6} P_{4j}{}^{w} = 1 \tag{6.15}$$

Meaning components for activity 4 could only be supplied from one plant for scenario w ($w \in \Omega$)

$$\sum_{j=1}^{6} P_{5j}{}^{w} = 1 \tag{6.16}$$

*M*eaning components for activity 5 could only be supplied from one plant for scenario w ($w \in \Omega$)

$P_{23}{}^{w} = P_{25}{}^{w} = P_{26}{}^{w} = 0$ – we could not supply components for activity 2 from plants 3, 5, 6 for scenario w ($w \in \Omega$) (6.17)

$P_{32}{}^{w} = P_{33}{}^{w} = P_{36}{}^{w} = 0$ – we could not supply components for activity 3 from plants 2, 3, 6 for scenario w ($w \in \Omega$) (6.18)

$P_{43}{}^{w} = P_{44}{}^{w} = P_{45}{}^{w} = 0$ – we could not supply components for activity 4 from plants 3, 4, 5 for scenario w ($w \in \Omega$) (6.19)

$P_{52}{}^{w} = P_{54}{}^{w} = P_{56}{}^{w} = 0$ – we could not supply components for activity 5 from plants 2, 4, 6 for scenario w ($w \in \Omega$) (6.20)

3rd Stage Constraints:

Precedence Constraints:

$$ES_7^w \geq ES_6^w + \sum_{j=1}^{6} D_{6j}{}^w P_{6j}{}^w \tag{6.21}$$

Earliest start date for activity 7 for scenario w ($w \in \Omega$)

Supply Constraints:

$$\sum_{j=1}^{6} P_{6j}{}^w = 1 \tag{6.22}$$

Meaning components for activity 6 could only be supplied from one plant for scenario w ($w \in \Omega$)

$P_{62}{}^w = P_{63}{}^w = P_{64}{}^w = 0$ – we could not supply components for activity 6 from plants 2, 3, 4 for scenario w ($w \in \Omega$) (6.23)

4th Stage Constraints:

Precedence Constraints:

$$EF_7^w \geq ES_7^w + \sum_{j=1}^{6} D_{7j}{}^w P_{7j}{}^w \tag{6.24}$$

Earliest finish date for activity 7 for scenario w ($w \in \Omega$)

$$EF_7^w + S7^{+w} - S7^{-w} = 18 \tag{6.25}$$

For the compilation of S_7^{-w} for scenario w ($w \in \Omega$)

Supply Constraints:

$$\sum_{j=1}^{6} P_{7j}^{w} = 1 \tag{6.26}$$

Meaning components for activity 7 could only be supplied from one plant for scenario w ($w \in \Omega$)

$P_{74}^{w} = P_{75}^{w} = P_{76}^{w} = 0$ – we could not supply components for activity 7 from plants 4, 5, 6 for scenario w ($w \in \Omega$) (6.27)

Non-negativity Constraints:

$$ES_i^{w}, EF_7^{w}, S7^{+w}, S7^{-w} \geq 0 \text{ for scenario } w \ (w \in \Omega) \tag{6.28}$$

Binary Variables Constraint:

$$P_{ij}^{w} \in \{0,1\} \tag{6.29}$$

Meaning P_{ij}^{w} are binary variables – allowed to take values either 0 or 1 only – for scenario w ($w \in \Omega$)

As per the number of scenarios, we would need to solve 4,782,969 independent scenario related 0-1 models for this optimization model. Since there are also binary – thus integer – variables in the second and proceeding stages, we could not use classical L-shaped decomposition method for the solution. This problem would fall under Stochastic Mixed Integer Linear Programming; which is quite an un-touched field. There are very few general purpose solution methods that would work under large number of scenarios, especially in the

presence of integer variables in the second and proceeding stages. And especially since the problem has more than two stages, computation of these methods would not really be efficient.

Accordingly, a rough-cut approximation would be designed in order to reach a solution for the problem. The number of scenarios will be reduced to a digestible level and the deterministic equivalent model will be solved using existing software. Dynamic programming will be the method for the solution of this model, as will be explained in the proceeding chapter.

7. APPLICATION OF DYNAMIC PROGRAMMING ON THE PROBLEM

Dynamic Programming is a mathematical technique designed primarily to improve the computational efficiency of certain optimization problems. The basic idea of the technique is to decompose the problem into smaller sub-problems that are computationally more manageable [17]. In this respect, dynamic programming is the utmost suitable mathematical formulation for the supply optimization problem that is the subject of this research study.

Dynamic programming decomposes the problem into sub-problems and each sub-problem is optimized separately over its *alternatives*, so that it is not necessary to enumerate all combinations in advance. These sub-problems correspond to the *stages*. Optimization is performed in each sub-problem, thus the non-optimal combinations are systematically discarded. The stages are linked to each other in a special way so that optimization does not occur over infeasible combinations. These links are provided via *states* of the system at each stage [17]. Accordingly, three determinants of the dynamic programming are:

- stages
- decision variables (alternatives) at each stage and their return functions
- state of the system at each stage.

Most important of these three determinants is the state of the system. That links all the stages to each other, so that separate optimization of each stage is feasible over the entire problem.

The use of stages and states to decompose a dynamic programming problem is computationally implemented by means of *recursive equation* that allows one to optimize each stage separately. It also keeps track of the *cumulative optimal return* for all previously considered stages so that when the last stage is reached the total optimal return for the entire problem is available. The nature of the approach is recursive since it computes the optimal

return for the first *k* stages from the optimal return for the first (*k*-1) stages and the return for stage *k*.

To set the initial scene for the solution of our example, some definitions would first be provided:

- Stages: Stages of the optimization problem have already been identified as the epochs on the project network;
- Decision variables: Decision variables have also been identified as the binomial variables (*Pij*) that would provide us the sourcing selection matrix for all activities on the network;
- States: States of the model (x_k) we have will be the time period left to the pre-identified due date of the project. This state representation will link the stages, the decisions at all stages and their associated optimal returns to each other.

The solution methodology proposed within this study will be as follows:

- First the DEM of the entire problem will be solved via Dynamic Programming. In order to do that, lead times having probabilistic distributions will be converted into their Expected Values.
- For the first activities belonging to the first stage, decisions will be made according to the DEM solution and components for these activities will be sourced from the associated plants.
- The algorithm to be constructed in the latter phases of this study will enable the users to update the states of the stages to reflect the realization of the lead times up to the concerning stage. The problem will be resolved taking the lead time realizations to that stage into account. This re-solution procedure will carry on until the last stage is reached. This way a better approximation will be achieved for the minimum cost optimization problem.

Subsequently, the expected values of the lead times for sourcing of components from different plants are determined as follows:

Table 7.1. Expected values of lead times for sourcing of components

Activity:	1	2	3	4	5	6	7
Plant Option:	1	1	1	1	1	1	1
Duration:	3	6	2	5	8	1	4
Plant Option:	3	2	4	2	3	5	2
Duration:	5,3	6,9	3,15	6	10,3	4,2	5,3
Plant Option:	4	4	5	6	5	6	3
Duration:	4	7,3	5	6,9	11,3	2,9	6

The associated material costs from different plants are as follows:

Table 7.2. Material cost for components from different plants

Activity:	Plant options: Total Cost of Supply - MCij (USD)		
1	1	3	4
	USD 10.000	USD 6.000	USD 7.000
2	1	2	4
	USD 42.000	USD 40.000	USD 45.000
3	1	4	5
	USD 20.000	USD 15.000	USD 10.000
4	1	2	6
	USD 35.000	USD 30.000	USD 25.000
5	1	3	5
	USD 64.000	USD 55.000	USD 50.000
6	1	5	6
	USD 20.000	USD 8.000	USD 9.000
7	1	2	3
	USD 12.000	USD 10.000	USD 9.000

Let

$f_k(x_k)$ = cumulative optimal cost for stages 1, 2, ……, and k given the state of the system is x_k, where x_k is the time period remaining to the due date of the project

The value of x_1 is 18, since at the start of the project time remaining to the due date is pre-defined to be 18 days. The values of x_2, x_3, and x_4 are not exactly known, but they comply with

$$0 \le x_2 \le 18 \quad (7.1)$$

$$0 \le x_3 \le 18 \quad (7.2)$$

$$0 \le x_4 \le 18 \quad (7.3)$$

Definitions of x_1, x_2, x_3, and x_4 implies that all feasible combinations of sourcings will be considered. We should start with the computation of the minimum cost for stage 4, $f_4(x_4)$. Next, the cumulative minimum cost for stages 3 and 4, $f_3(x_3)$, will be determined from $f_4(x_4)$. Thereafter comes the cumulative minimum cost for stages 2, 3, and 4, $f_2(x_2)$ – this one is computed from $f_3(x_3)$. Finally, using $f_2(x_2)$, cumulative minimum cost for all stages, $f_1(x_1)$, will be computed. This method is named as backward computation, since we start the computation with the last stage and proceed towards the preceding stages.

It should here be noted that state of the system at a stage could only assume integer values, since start of an activity would happen only the next day, even if the preceding activity is finalized half way through the day. Accordingly, state of a stage should be rounded up to the closest smaller integer.

The associated computations are provided in Appendix C, D, and E. The optimization problem is solved by Dynamic Programming methodology for three cases – tardiness costs of 10,000 USD/day, 4,000 USD/day, and 1,000 USD/day. Excel application is used for the solution of this dynamic programming model in order to test the feasibility of the problem. Hence, considerable computation time is spent for this preliminary solution. In the proceeding chapters of this research study, an algorithm constructed in C++ will be utilized and the computation time will be reduced drastically.

The computation of the cost for stages is conducted as follows:

Stage 4: (states $x_4 = 0, 1, 2, 3, \ldots, 18$)

For all states, there are three alternative sourcing options – plants 1, 2, and 3 for activity 7. Material costs and tardiness costs for all states and sourcing options are computed. Then, the minimum cost and determining plant option (decision variable) is computed for each state.

Stage 3: (states $x_3 = 0, 1, 2, 3, \ldots, 18$)

For all states, there are three alternative sourcing options – plants 1, 5, and 6 for activity 6. Material costs and tardiness costs for all states and sourcing options are computed for stage 3. In addition, associated x_4's for all state-plant combinations are computed. The optimum cost of stage 4 is inserted in the computation for all these associated x_4's. Afterwards the total cost of stages 3 and 4 is calculated for all state-plant combinations. The minimum cost and determining plant options (decision variables) are computed for each state.

Stage 2: (states $x_2 = 0, 1, 2, 3, \ldots, 18$)

For all states and activities (there are 3 activities associated with Stage 2 – activities 3, 4, and 5), there exist three alternative sourcing options. This makes $3^3=27$ activity-plant combinations for each state. Material costs and tardiness costs for all states and activity-plant combinations are computed for stage 2. In addition, associated x_3's (remaining time for the rest of the activities) for all state-plant combinations are computed. The optimum cost of stage 3 is inserted in the computation for all these associated x_3's. Afterwards the total cost of stages 2, 3 and 4 is calculated for all state-plant combinations. The minimum cost and determining plant options (decision variables) are computed for each state.

Stage 1: (state $x_1 = 18$)

There exists only one state for stage 1 and that's 18 days, since stage 1 represents the start of the project. For all activities (there are 2 activities associated with Stage 1 – activities 1 and 2), there exist three alternative sourcing options. This makes $3^2=9$ activity-plant combinations for state $x_1 = 18$. Material costs and tardiness costs for all activity-plant combinations are computed for stage 1. In addition, associated x_2's (remaining time for the rest of the activities) for all state-plant combinations are computed. The optimum cost of stage 2 is inserted in the computation for all these associated x_2's. Afterwards the total cost of stages 1, 2, 3 and 4 is calculated for all state-plant combinations. The minimum cost and determining plant options (decision variables) are then computed.

This optimum representation provides the solution to the supply optimization problem. Following table lists the optimum decision variables (plant options) for all activities for three different tardiness cost cases (10,000 – 4,000 – 1,000 USD/day).

As can be interpreted from the table, as the unit tardiness cost decreases, there occurs a tendency to choose the less costly alternatives, even though this decision might mean longer activity durations and eventually a project schedule that is overrun.

Table 7.3. Optimum solutions for different tardiness costs

OPTIMUM SUPPLY SOURCES:

Tardiness Cost: 10,000 USD/day					Tardiness Cost: 4,000 USD/day					Tardiness Cost: 1,000 USD/day				
Activity:	Plant:	Time:	MC:	Critical Path:	Activity:	Plant:	Time:	MC:	Critical Path:	Activity:	Plant:	Time:	MC:	Critical Path:
1	1	3	10.000	3	1	1	3	10.000	3	1	4	4	7.000	4
2	2	6,9	40.000		2	2	6,9	40.000		2	2	6,9	40.000	
3	5	5	10.000		3	5	5	10.000		3	5	5	10.000	
4	6	6,9	25.000		4	6	6,9	25.000		4	6	6,9	25.000	
5	1	8	64.000	8	5	1	8	64.000	8	5	5	11,3	50.000	11,3
6	6	2,9	9.000	2,9	6	6	2,9	9.000	2,9	6	6	2,9	9.000	2,9
7	1	4	12.000	4	7	1	4	12.000	4	7	3	6	9.000	6
Total MC:				170.000	Total MC:				170.000	Total MC:				150.000
Total Project Duration:				17,9	Total Project Duration:				17,9	Total Project Duration:				24,2
Total Tardiness Cost:				-	Total Tardiness Cost:				-	Total Tardiness Cost:				6.900
Total Cost:				170.000	Total Cost:				170.000	Total Cost:				156.900

The conclusions that would be reached from this DP analysis are manifold:

- Project networks and scheduling problems represent high numbers of scenarios, especially if some parameters building up the networks behave in a stochastic nature. Therefore, their mathematical solutions would require some rough-cut approximations – these be either heuristic approximations or utilization of deterministic equivalents of these parameters.
- Dynamic programming represents a highly applicable methodology for the solution of scheduling problems under uncertainty. The DEM of stochastic models does not provide the optimum solution; however these are quite close approximations to the optima and provide the users with adequately satisfactory results.
- While deciding on sourcing options, not only material cost but also tardiness cost should be taken into consideration. Depending on the magnitude of tardiness cost, optimum solutions that are very different from the case, where only material costs are considered,

can be computed. As an example optimum sourcing networks for tardiness costs of 4,000 and 1,000 are quite different from each other.

- Depending on the circumstances, the decision makers might choose not to complete the project on time, but to choose the minimum cost sourcing options even if they bear the risk of paying tardiness penalty – see the optimum supply network for tardiness cost of 1,000 USD/day. While concluding deals for construction projects, this dimension should also be taken into consideration. Thus, the end result of this research study will constitute a useful device for not only decision makers for sourcing decisions but also project owners.

8. CASE STUDY – A REAL WORLD STEEL FRAMED CONSTRUCTION PROJECT

One of the main focuses of this research study is the application of the optimization model on a real world steel framed construction project. For this aim, the 2x400 MW Combined Cycle Power Plant Project in Surgut, Russia is chosen. This project, which will provide electricity to the Tumenskaya Region of Siberia, is an extension to an already existing and active facility and is undertaken by the Joint Venture of General Electric and Gama Power Systems, a well-reputable international Turkish contractor. Essential approvals are obtained from the project management for the share of certain information concerning the project.

Within this CCPP, there exist three main buildings, Energoblock 7, Energoblock 8, and Gas Compressor Station. All these three buildings are constructed fully with steel frames over the +0.00 level. Components of the steel frame are supplied by different parties, and there exists a steel structures division within the site management that controls the supply and erection works of the steel frames.

Some of the drawings of the steel frame of Energoblock 7 are provided in Appendix F. As illustrated, the frame has quite a complex structure and is composed of numerous different elements.

Since the site is within an existing power station, there exist highly limited storage facilities within the site. Accordingly, the materials are brought to site with a just-in-time manner. Consequently, logistics planning and timely delivery of steel elements become an utmost important issue. All the machinery and equipment within the buildings require weather-tight conditions and a delay in the construction of the buildings directly affects the opening date of the entire facility.

In addition, Surgut is a city within the Siberian region of Russia and especially in winter time the weather conditions in the region is quite severe – i.e. the temperature goes down to -50 degrees Celsius. Due to this geographical disadvantage, material deliveries to site is quite problematic and delays on the deliveries to site are pretty common; these delays sometimes being up to two months. Timely delivery is also more important in such projects, because the weather conditions allow working in exterior spaces only in certain months.

In accordance, the decisions on material supplies depend as much on the location of the manufacturers as the cost of materials themselves.

The work-schedule at site for the steel frame is provided in Appendix G. Since we will concentrate on the steel frame of Energoblock 7, only the schedules of Energoblocks 7 and 8 are provided; the schedule of Gas Compressor Station is not provided. It should be noted that this schedule is a simplified version of the master steel structures schedule being used at site. Naturally, there have been many revisions on this version of the schedule, which was the initial timeplan. However, since the subject of this study is on the optimization on the supply even before the project starts, the initial schedule, which was the only data available prior to project start date, is taken into account for analysis. It should also here be noted that many delays both on material supply and erection works have happened and this schedule has never been realized. To illustrate the dimension of the problem; the steel works progress is approximately 10 months behind this initial time-plan – this delay meaning a huge financial cost for the investor, contractors and subcontractors working at site.

The initial plan, as outlined in Appendix G, was to start the erection works at site in December 2008 for Energoblock 7 and in May 2009 for Energoblock 8. The consecutive makespans of these two buildings were 249 and 273 days.

8.1. Optimization Model for the Steel Supply

Another main focus of this research study is to modify the mathematical modelling constructed in the previous chapters in order to optimize the steel supply of the real life project

and come up with a computer code that would ease the solution method, which was first manually formulated and solved in Excel environment.

In order to have an efficient optimization model for the supply of steel elements for Energoblock 7, the full schedule provided for this structure should be condensed. Because there exists 99 activities, even in the simplified version of the schedule of works as enclosed in Appendix G. Please note that the original work schedule contains more than 2,000 activities – the one in Appendix G is a simplified version as was mentioned. If we presume 3 sourcing options for 99 activities, the model to be built will immediately grow out of hand, especially due to stochastic nature of the data.

For this condensation process, activities under the main work items – that are provided as headings – are shrunk and the totality of the main work items are taken into consideration. This condensed work schedule is provided in Appendix H. In this condensed version, there exist 13 activities. There are certain precedence relations between these activities, which are listed in Table 8.1 below.

Table 8.1. Precedence relations among the activities of the project

PRECEDENCE RELATIONS AMONG ACTIVITIES

Activity ID	Activity Name	Start Date	Finish Date	Prdecessor	Predecessor ID	Relation	Time Lag (days)
1	Column Anchorages	01.12.2008	25.01.2009				
2	Columns	05.12.2008	11.04.2009	Column Anchorages	1	Start-to-Start	4
3	Beam Supports	14.12.2008	13.04.2009	Columns	2	Start-to-Start	9
4	Beams - Level +5.48	17.12.2008	01.04.2009	Beam Supports	3	Start-to-Start	3
5	Beams - Level +10.48	02.04.2009	10.06.2009	Beams - Level +5.48	4	Finish-to-Start	
6	Windpost Supports	11.06.2009	26.06.2009	Beams - Level +10.48	5	Finish-to-Start	
7	Windposts	14.06.2009	01.07.2009	Windpost Supports	6	Start-to-Start	3
8	Bracings	02.07.2009	11.07.2009	Windposts	7	Finish-to-Start	
9	Window Supports	02.07.2009	25.07.2009	Windposts	7	Finish-to-Start	
10	Trusses	05.04.2009	09.05.2009	Columns	2	Finish-to-Start	-7
11	Roof Purlins	07.05.2009	24.05.2009	Trusses	10	Finish-to-Start	-3
12	Staircase Structures	02.07.2009	25.07.2009	Windposts	7	Finish-to-Start	
13	Fire Escape Stairs	26.07.2009	06.08.2009	Staircase Structures	12	Finish-to-Start	

Accordingly, activity network of the fictitious project provided in the previous progress reports could be reconstructed as follows:

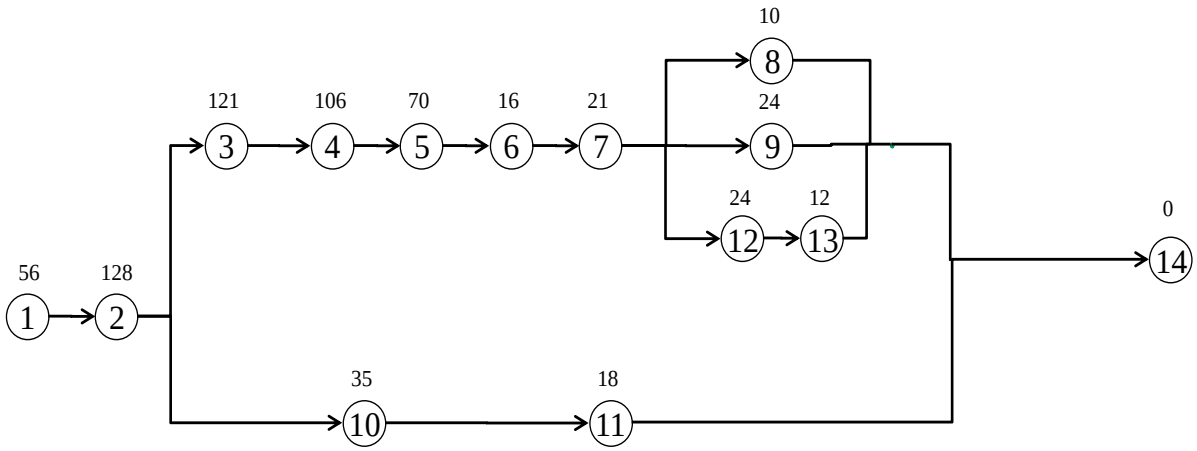

Figure 8.1. Activity-on-node network

The difference of this activity network from the previous ones is that this one is an activity-on-node network, rather than an activity-on-arc network. The figures in the circles indicate the ID of the activities and the figures over the circles indicate the duration of each activity. However, this activity network needs to be converted to an activity-on-arc network once again, so that the stages of the optimization model could be identified. Figure 8.2 below illustrates the activity-on-arc version of the concerning network.

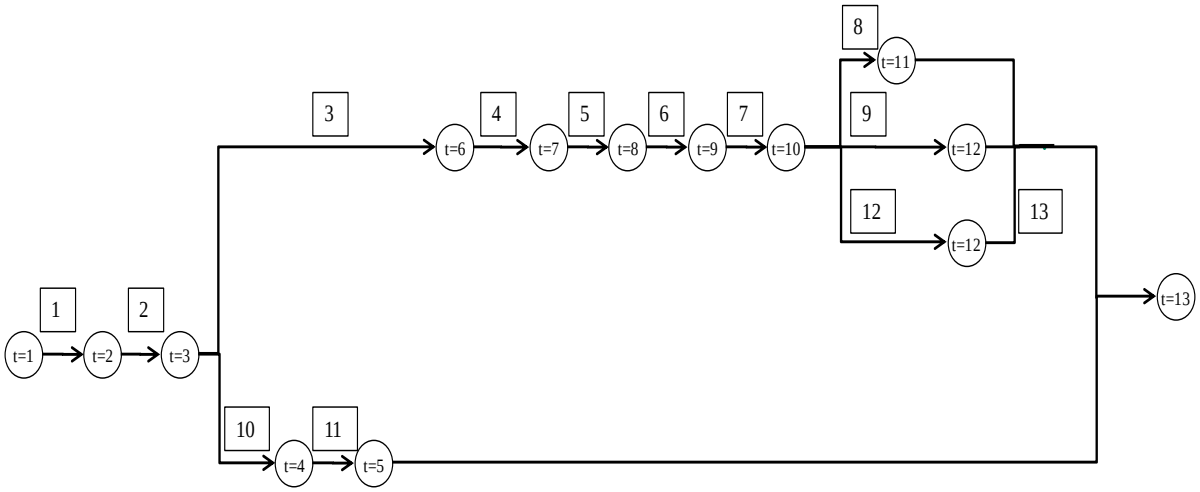

Figure 8.2. Activity-on-arc network and stages of the model

As illustrated, the project and thus the model has 13 stages. The figures inside the circles represent the stages and the figures inside the rectangles represent the activities, since this is an activity-on-arc network. This obviously is a bigger network than the fictitious one and thus will lead into a more complex and hard-to-solve model.

The deterministic models developed in the previous chapters need to be enhanced to reflect the stochastic nature of the delivery times of materials from different plant locations. The most appropriate stochastic behaviour that fits real-life examples of material deliveries is discrete probability distribution. Accordingly, the deterministic sourcing matrix has been modified as in Table 8.2 in order to reflect stochastic behaviour. The material cost for sourcing options also exist within this table.

As was explained in the previous chapters, the number of scenarios will be way too many to solve this problem, should there be a desire to utilize stochastic programming. Since there are also binary – thus integer – variables in the second and proceeding stages, this problem would fall under Stochastic Mixed Integer Linear Programming; which constitutes quite a complex methodology. And especially since the problem has more than two stages, computation of these methods would not really be efficient.

Accordingly, stochastic dynamic programming will be the method for the solution of this model. The problem will be divided into stages and at each stage there will exist a separate sub-optimization problem to be solved. These stages will then be inter-connected to each other by the means of states.

In order to provide a solvable model within dynamic programming principles, the equations of the model should be re-formulated as follows. The below formulation reflects the recursive nature of the dynamic programming; it commences the calculations from the last step and proceeds backwards to the first stage. The states of the model inter-connect the sub-optimization problems at each stage.

Table 8.2. Supply options, lead time probabilities, and material costs

Activity:	1			2				3				4				5				6				7				8				9				10				11				12				13			
Plant Option:	1			1				1				1				1				1				1				1				1				1				1				1				1			
Delivery Duration:	12			24				8				20				32				9				16				16				25				11				14				15				18			
Delivery Duration Probability:	100%			100%				100%				100%				100%				100%				100%				100%				100%				100%				100%				100%				100%			
Material Cost (MCij) - USD	602.000			4.448.000				304.500				3.196.000				2.988.200				265.000				3.225.600				582.100				277.600				9.564.300				1.543.700				2.065.080				922.100			
Plant Option:	3			2				4				2				3				5				2				3				2				4				5				2				3			
Delivery Duration:	14	19	25	25	28	32	8	12	16	20	24	28	32	40	42	12	16	20	16	20	24	16	19	25	25	28	32	11	12	16	14	16	20	16	20	24	18	19	25									
Delivery Duration Probability:	30%	40%	30%	20%	50%	30%	20%	45%	35%	30%	40%	30%	20%	45%	35%	20%	40%	40%	20%	30%	50%	30%	40%	30%	20%	50%	30%	20%	45%	35%	20%	40%	40%	20%	30%	50%	30%	40%	30%									
Material Cost (MCij) - USD	522.100			3.950.000				265.000				2.876.500				2.643.100				182.000				2.430.500				392.100				198.300				9.230.400				1.346.000				1.876.000				723.900			
Plant Option:	4			4				5				6				5				6				3				4				5				6				6				3				4			
Delivery Duration:	12	16	20	24	28	32	16	20	24	20	28	32	40	44	46	9	12	16	16	24	32	24	28	32	25	27	32	20	28	32	14	17	20	16	24	32	24	28	32									
Delivery Duration Probability:	30%	40%	30%	20%	30%	50%	30%	40%	30%	20%	50%	30%	30%	40%	30%	35%	40%	25%	30%	40%	30%	20%	30%	50%	30%	40%	30%	20%	50%	30%	35%	40%	25%	30%	40%	30%	20%	30%	50%									
Material Cost (MCij) - USD	544.300			3.822.000				232.500				2.542.100				2.221.000				211.000				2.133.400				285.400				212.500				7.283.900				1.302.500				1.795.300				654.700			
Activity Duration (at Site):	56			128				121				106				70				16				21				10				24				35				18				24				12			
Effect of Delivery Duration on Activity Duration:																																																	
Plant Option:	1			1				1				1				1				1				1				1				1				1				1				1				1			
Starting Delay (Sij):	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0									
Gross Activity Duration (Dij):	56	56	56	128	128	128	121	121	121	106	106	106	70	70	70	16	16	16	21	21	21	10	10	10	24	24	24	35	35	35	18	18	18	24	24	24	12	12	12									
Plant Option:	3			2				4				2				3				5				2				3				2				4				5				2				3			
Starting Delay (Sij):	2	7	13	1	4	8	0	4	8	0	4	8	0	8	10	3	7	11	0	4	8	0	3	9	0	3	7	0	1	5	0	2	6	1	5	9	0	1	7									
Gross Activity Duration (Dij):	58	63	69	129	132	136	121	125	129	106	110	114	70	78	80	19	23	27	21	25	29	10	13	19	24	27	31	35	36	40	18	20	24	25	29	33	12	13	19									
Plant Option:	4			4				5				6				5				6				3				4				5				6				6				3				4			
Starting Delay (Sij):	0	4	8	0	4	8	8	12	16	0	8	12	8	12	14	0	3	7	0	8	16	8	12	16	0	2	7	9	17	21	0	3	6	1	9	17	6	10	14									
Gross Activity Duration (Dij):	56	60	64	128	132	136	129	133	137	106	114	118	78	82	84	16	19	23	21	29	37	18	22	26	24	26	31	44	52	56	18	21	24	25	33	41	18	22	26									

The following notation holds for the entire model:

Y_{ij} = plant selection for the i^{th} activity from set j;

where

i = 1, 2, 3….., 13 - activities;

j = 1, 2, 3, 4, 5, 6 - plants;

D_{ij} = Duration of activity i, when supplied from plant j;

S_{ij} = Delay in the starting date of activity i, when supplied from plant j;

MC_{ij} = material cost of activity i, if supplied from plant j;

TC = tardiness cost per day (fixed penalty per a day of late completion);

MS = Target makespan of the project

$$V_{13} = \min_{j \in A_{13}} \left\{ MC_{13j} Y_{13j} \quad Y_{13j} TC \left[\sum_{k=0}^{\infty} \left[P \; B_{13j} \right) \; (k) W_{13}^{-} \; k \; \right] \right] \right\} \tag{8.1}$$

Supply Constraints:

$$\sum_{j=1}^{6} Y_{13j} = 1 \tag{8.2}$$

Meaning components for activity 13 could only be supplied from one plant

$Y_{13\text{-}2} = Y_{13\text{-}5} = Y_{13\text{-}6} = 0$ – we could not supply components for activity 13 from plants 2, 5, 6 (8.3)

$$Y_{ij} \in \{0,1\} \tag{8.4}$$

Meaning Y_{ij} are binary variables – allowed to take values either 0 or 1 only

Scheduling Constraint:

$$t + k + W_{13}^{+}(k) - W_{13}^{-}(k) = MS \text{ – for the compilation of } W_{13}^{-}(k) \tag{8.5}$$

$$MS = 250 \tag{8.6}$$

Non-negativity Constraints:

$$W_{13}^{+}(k), W_{13}^{-}(k) \geq 0 \tag{8.7}$$

Here,

$V_{12}(t)$ = state variable

$W_{13}^{+}(k)$ = if activity 13 finished earlier, number of days from EF_{13} to the due date of the project;

$W_{13}^{-}(k)$ = if activity 13 finished later, number of days from due date of the project to EF_{13};

If,

$$\max\left\{\sum_{l=0}^{\infty}(PD_{8n})l(l,\sum_{m=0}^{\infty}P)D_{9o}\quad m\ m\right\} \geq \left[\sum_{k=0}^{\infty}PD_{12j}=k\ k\right] + \sum_{p=0}^{\infty}PD_{13i}=p\ p \tag{8.8}$$

$$j \in A_{12}, n \in A_8, o \in A_9, i \in A_{13}$$

Then,

$$V_{10}(t) = \min_{\substack{j\in A_{12}\\ n\in A_8\\ o\in A_9}}\left\{\begin{matrix} MC_{12j}Y_{12j} + Y_{12j}V_{12}(t+\left[\sum_{k=0}^{\infty}(PD_{12j})+k\ k\right] & MC_{8n}Y_{8n} \quad MC_{9o}Y_{9o} \\ Y_{8n}Y_{9o}TC\left[\sum_{m=0}^{\infty}\sum_{l=0}^{\infty}\left[R(D_{8n}=l)(P=D_{9o})\ n(W_{8\text{-}9}^{-}l,m\right]\right] & \end{matrix}\right\} \tag{8.9}$$

If,

$$\max\left\{\sum_{l=0}^{\infty}(P\ D_{8n}\ )\ l((\ ,\sum_{m=0}^{\infty}=P\ )D_{9o}\quad m(m\ \right\}=\left[\sum_{k=0}^{\infty}P(D_{12j}=\ k)k\quad\sum_{p=0}^{\infty}P\ D_{13i}\quad p\ p\ \right] \tag{8.10}$$

$$j\in A_{12}, n\in A_8, o\in A_9, i\in A_{13}$$

Then,

$$V_{10}(=\min_{\substack{j\in A_{12}\\ n\in A_8\\ o\in A_9}}\left\{MC_{12j}Y_{12j}\quad Y_{12j}(V_{12}\ t\ (\left(\left[\sum_{k=0}^{\infty}=P\ D_{12j}\quad k\ k\ \right|\ +MC_{8n}Y_{8n}\quad MC_{9o}Y_{9o}\right\} \tag{8.11}$$

Supply Constraints:

$$\sum_{j=1}^{6}Y_{12j}=1 \tag{8.12}$$

Meaning components for activity 12 could only be supplied from one plant

$$Y_{12\text{-}4}=Y_{12\text{-}5}=Y_{12\text{-}6}=0 \text{ – we could not supply components for activity 12 from plants 4, 5, 6} \tag{8.13}$$

$$\sum_{j=1}^{6}Y_{8n}=1 \tag{8.14}$$

Meaning components for activity 8 could only be supplied from one plant

$$Y_{82}=Y_{85}=Y_{86}=0 \text{ – we could not supply components for activity 8 from plants 2, 5, 6} \tag{8.15}$$

$$\sum_{j=1}^{6}Y_{9o}=1 \tag{8.16}$$

*M*eaning components for activity 9 could only be supplied from one plant

$Y_{93} = Y_{94} = Y_{96} = 0$ – we could not supply components for activity 9 from plants 3, 4, 6 (8.17)

$$Y_{ij} \in \{0,1\} \quad (8.18)$$

Meaning Y_{ij} are binary variables – allowed to take values either 0 or 1 only

Scheduling Constraint:

$t + \max\{l, m\} + W_{8\text{-}9}^{+}(l,m) - W_{8\text{-}9}^{-}(l,m) = 250$ – for the compilation of W^{-} (8.19)

$$MS = 250 \quad (8.20)$$

Non-negativity Constraints:

$$W_{8\text{-}9}^{+}(l,m), W_{8\text{-}9}^{-}(l,m) \geq 0 \quad (8.21)$$

Here,

$V_{10}(t)$ = state variable

$W_{8\text{-}9}^{+}(k)$ = if activity 8 & 9 (whichever is finished later) finished earlier, number of days from EF_8 or EF_9 (whichever is later) to the due date of the project;

$W_{8\text{-}9}^{-}(k)$ = if activity 8 & 9 (whichever is finished later) finished later, number of days from due date of the project to EF_8 or EF_9 (whichever is later);

$$V_9(t) = \min_{j \in A_9} \left\{ MC_{7j}.Y_{7j} + Y_{7j} V_{10}\left(t + \left\lceil \sum_{k=0}^{\infty} (P(D_{7j} = k)k) \right\rceil\right) \right\} \quad (8.22)$$

Supply Constraints:

$$\sum_{j=1}^{6} Y_{7j} = 1 \tag{8.23}$$

Meaning components for activity 7 could only be supplied from one plant

$Y_{74} = Y_{75} = Y_{76} = 0$ – we could not supply components for activity 7 from plants 4, 5, 6 (8.24)

$$Y_{ij} \in \{0,1\} \tag{8.25}$$

Meaning Y_{ij} are binary variables – allowed to take values either 0 or 1 only

Here,

$V_9(t)$ = state variable

$$V_8(t) = \min_{j \in A_6} \left\{ MC_{6j}.Y_{6j} + Y_{6j} V_9(t + \left\lceil \sum_{k=0}^{\infty} (P(S_{6j} = k)k) \right\rceil + 3) \right\} \tag{8.26}$$

Supply Constraints:

$$\sum_{j=1}^{6} Y_{6j} = 1 \tag{8.27}$$

Meaning components for activity 6 could only be supplied from one plant

$Y_{62}{}^{w} = Y_{63}{}^{w} = Y_{64}{}^{w} = 0$ – we could not supply components for activity 4 from plants 2, 3, 4 (8.28)

$$Y_{ij} \in \{0,1\} \tag{8.29}$$

Meaning Y_{ij} are binary variables – allowed to take values either 0 or 1 only

Here,

$V_8(t)$ = state variable

$$V_1(\quad) \quad \min_{j \in A_{\quad}} \left\{ MC_{5j} Y_{5j} \quad Y_{5j} V_8 \; t \left(\left\lceil \sum_{k=0}^{\infty} P \; D_{6j} \quad k \; k \right\rceil \right) \right\} \tag{8.30}$$

Supply Constraints:

$$\sum_{j=1}^{6} Y_{5j} = 1 \tag{8.31}$$

*M*eaning components for activity 5 could only be supplied from one plant

$$Y_{52} = Y_{54} = Y_{56} = 0 \text{ – we could not supply components for activity 4 from plants 2, 4, 6} \tag{8.32}$$

$$Y_{ij} \in \{0,1\} \tag{8.33}$$

Meaning P_{ij} are binary variables – allowed to take values either 0 or 1 only

Here,

$V_7(t)$ = state variable

$$V_6(\quad) \quad \min_{j \in A_{\quad}} \left\{ MC_{4j} Y_{4j} \quad Y_{4j} V_7 \; t \left(\left\lceil \sum_{k=0}^{\infty} P \; D_{4j} \quad k \; k \right\rceil \right) \right\} \tag{8.34}$$

Supply Constraints:

$$\sum_{j=1}^{6} Y_{4j} = 1 \tag{8.35}$$

Meaning components for activity 4 could only be supplied from one plant

$$Y_{43} = Y_{44} = Y_{45} = 0 \text{ – we could not supply components for activity 4 from plants 3, 4, 5} \quad (8.36)$$

$$Y_{ij} \in \{0,1\} \quad (8.37)$$

Meaning Y_{ij} are binary variables – allowed to take values either 0 or 1 only

Here,

$V_6(t)$ = state variable

$$V_4(t) = \min_{j \in A_i} \left\{ MC_{11j} Y_{11j} + Y_{11j} TC \left[\sum_{k=0}^{\infty} P(D_{11j} = k) W_{11}^{-}(k) \right] \right\} \quad (8.38)$$

Supply Constraints:

$$\sum_{j=1}^{6} Y_{11\text{-}j} = 1 \quad (8.39)$$

Meaning components for activity 1 could only be supplied from one plant

$$Y_{11\text{-}2} = Y_{11\text{-}3} = Y_{11\text{-}4} = 0 \text{ – we could not supply components for activity 1 from plants 2, 5, 6} \quad (8.40)$$

$$Y_{ij} \in \{0,1\} \quad (8.41)$$

Meaning Y_{ij} are binary variables – allowed to take values either 0 or 1 only

Scheduling Constraint:

$$t + k + W_{11}^{+}(k) - W_{11}^{-}(k) = MS \text{ – for the compilation of } W^{-} \tag{8.42}$$

$$\text{MS} = 250 \tag{8.43}$$

Non-negativity Constraints:

$$W_{11}^{+}(k), W_{11}^{-}(k) \geq 0 \tag{8.44}$$

Here,

$V_4(t)$ = state variable

$W_{11}^{+}(k)$ = if activity 11 finished earlier, number of days from EF_{11} to the due date of the project;

$W_{11}^{-}(k)$ = if activity 11 finished later, number of days from due date of the project to EF_{11};

$$V_{3\text{-}1}(t) = \min_{j \in A_3} \left\{ MC_{3j} Y_{3j} + Y_{3j} V_6 \left(t + \left\lceil \sum_{k=0}^{\infty} (P(S_{3j} = k)k) \right\rceil + 3 \right) \right\} \tag{8.45}$$

$$V_{3\text{-}2}(t) = \min_{n \in A_{10}} \left\{ MC_{10n} .. Y_{10n} + Y_{10n} V_4 \left(t + \left\lceil \sum_{l=0}^{\infty} (P(D_{10n} = l)l) \right\rceil - 3 \right) \right\} \tag{8.46}$$

Supply Constraints:

$$\sum_{j=1}^{6} Y_{3j} = 1 \tag{8.47}$$

*M*eaning components for activity 1 could only be supplied from one plant

$$Y_{32} = Y_{33} = Y_{36} = 0 \text{ – we could not supply components for activity 1 from plants 2, 5, 6} \tag{8.48}$$

$$\sum_{j=1}^{6} Y_{10n} = 1 \tag{8.49}$$

Meaning components for activity 10 could only be supplied from one plant

$$Y_{10\text{-}2} = Y_{10\text{-}3} = Y_{10\text{-}5} = 0 \text{ – we could not supply components for activity 10 from plants 2, 3, 5} \tag{8.50}$$

$$Y_{ij} \in \{0,1\} \tag{8.51}$$

Meaning Y_{ij} are binary variables – allowed to take values either 0 or 1 only

Here,

$V_3(t)$ = state variable

$$V_2(t) = \min_{j \in A_2} \left\{ MC_{2j}.Y_{2j} + Y_{2j}V_{3-1}(t + \left\lceil \sum_{k=0}^{\infty} (P(S_{2j} = k)k) \right\rceil + 9) + Y_{2j}V_{3-2}(t + \left\lceil \sum_{l=0}^{\infty} (P(D_{2j} = l)l) \right\rceil - 7) \right\} \tag{8.52}$$

Supply Constraints:

$$\sum_{j=1}^{6} Y_{2j} = 1 \tag{8.53}$$

Meaning components for activity 2 could only be supplied from one plant

$$Y_{23} = Y_{25} = Y_{26} = 0 \text{ – we could not supply components for activity 2 from plants 3, 5, 6} \tag{8.54}$$

$$Y_{ij} \in \{0,1\} \tag{8.55}$$

Meaning Y_{ij} are binary variables – allowed to take values either 0 or 1 only

Here,

$V_2(t)$ = state variable

$$V_2(0) = \min_{j \in A_1} \left\{ MC_{1j} Y_{1j} + V_2(1) + \left(\left\lceil \sum_{k=0}^{\infty} = P \right\rceil \right) \right\} \quad (8.56)$$

Supply Constraints:

$$\sum_{j=1}^{6} Y_{1j} = 1 \quad (8.57)$$

Meaning components for activity 1 could only be supplied from one plant

$Y_{12} = Y_{15} = Y_{16} = 0$ – we could not supply components for activity 1 from plants 2, 5, 6 (8.58)

$$Y_{ij} \in \{0,1\} \quad (8.59)$$

Meaning Y_{ij} are binary variables – allowed to take values either 0 or 1 only

Here,

$V_1(t)$ = state variable

A computer code is constructed in C++ for the solution of the above dynamic programming model. This algorithm is provided under Appendix I. This algorithm solves the model in a recursive manner, as is the case in dynamic programming. The makespan is identified as 250 days and the tardiness cost per day is taken as 200,000 USD/day and 10,000 USD/day respectively. The consecutive solutions determined by the code are as follows:

Table 8.3. Optimum solutions for the project network

Tardiness Cost: 200,000 USD/day	
Total Cost: 27,486,590	
Activity:	Plant Option:
1	1
2	1
3	1
4	1
5	1
6	1
7	1
8	4
9	2
10	6
11	6
12	1
13	1

Tardiness Cost: 10,000 USD/day	
Total Cost: 24,005,400	
Activity:	Plant Option:
1	3
2	4
3	1
4	6
5	5
6	5
7	3
8	4
9	2
10	6
11	6
12	3
13	4

This solution optimizes the model with a total cost of 27,486,590 USD and 24,005,400 USD consecutively for different tardiness costs, whereas if the components are purchased from plant option 1 all the time (without any optimization procedure), the total cost would be 30,384,190 USD and 30,004,190 USD consecutively for these tardiness costs. Thus the model helps us save 2,897,600 USD and 5,998,790 USD consecutively, which constitutes a consecutive 9,54% and 19,99% saving on the original cost determined without optimizing the sourcing process. The overrun in the case of 200,000 USD/day tardiness cost is 2 days, whereas the overrun in the case of 10,000 USD/day tardiness cost is 76 days.

The code constructed for the solution of the case application solves the optimization problem within less than a second. The consecutive steps of this algorithm solve each stage of the stochastic dynamic programming utilizing data provided within matrices. In effect, this step-by-step solution avoids the computational burden of a potential S-MILP algorithm. The largest matrices that this algorithm produces are the state variable matrices. As formulated in the beginning of the algorithm in Appendix I, the largest matrix among these state variables

belong to $V_{12}(t)$, since $V_{12}(t)$ represents the state variable of the last stage of the network. The number of rows for a state variable matrix is determined by the timespan from the start of the project to the latest finish date of the concerning stage. For instance for $V_{12}(t)$, the number of rows is 460, whereas for $V_{10}(t)$ that is 419 and for $V_1(t)$ that is 1. The number of columns for a state variable matrix is determined by the number of activities within the entire network plus a column for the tardiness cost.

Accordingly, even if the number of activities within the network was not condensed to 13 main activity flows, the computer code's matrix sizes would not grow out of hand. Because the number of rows for state variable matrices would still be at same maximum values, since the latest finish dates of the concerning stages would remain unchanged. The number of columns in state variable matrices would increase by the number of extra activities added to the network. However, this addition would not turn into a geometric increase in the matrix data, since the number of rows would not change. Another increase would be in the number of state variables, since more activities in series would mean more stages within the network. Nevertheless, since the stochastic dynamic programming methodology solves the stages of the network one at a time, this increase in the number of stages would not create an unsolvable problem. Thus, the computing time does not constitute an obstacle for the efficient work of the algorithm. The only disadvantage of having a larger network would be the length of repetitive formulations within the algorithm, since appropriate formulation for each stage would be added to the algorithm, once a stage is added to the network.

9. CONCLUSIONS AND RECOMMENDATIONS

The work done in this report proves that the optimization reached utilizing the stochastic dynamic programming principles provides significant savings for steel structures projects. 9.54% and 19.99% savings were obtained for a real project in Siberia, utilizing the optimization method formulated in this research.

This research study proves that the mathematical model determined for the solution of the optimization problem works efficiently and the solution can be provided by means of a computer code. The running time of the algorithm is negligible – is less than a second, meaning more complex networks with higher number of activities can still be solved with the same methodology. The addition of further activities to the network only increases the time needed to formulate the algorithm, however the computational time does not change significantly. Because the dimensions of the solution matrices are dependant on the makespan of the project and the number of activities within the project network. The makespan of the project does not change as the number of activities increases. The data within the matrices will increase by the ratio of increase in the number of activities only. Thus, there will be an arithmetic increase in the data to be processed rather than a geometric one. Subsequently, the method constructed is applicable on much larger project networks.

Another conclusion is that it is not always economically the most feasible decision to finish steel structures activities on time. Depending on the relative weight of tardiness cost against the material cost, that would be in the contractor's best interest to choose the cheapest sourcing option, even though this decision will mean a significant delay on the completion of works. In the real project optimized with the model built in this research, when the tardiness cost is low enough – like 10,000 USD/pay, the contractor chooses to source the materials from the cheapest supply options even though they delay the project to a large extent. As a consequence of these decisions, this 250-day project overruns 76 days, which is a highly

significant delay. This is a very important criterion that the investors should consider, while negotiating on contractual issues with the contractor they would appoint.

Given the methodology within this research, there exist much to add and modify. Including but not limited to, future research studies might focus on the following developments of the same subject matter:

- An algorithm that re-conducts the calculations at the beginning of each stage can be developed. This will enable the decision makers to take the realizations of the previous stages into account, while deciding for the further stages;
- The sourcing decisions for the parallel activities are made at once and this decision is not changeable within the model created. However, during the realization of these activities, at certain points of time, some of these parallel activities that were initially not on the critical path might turn out to be on the critical path due to some delays. In order to reach a closer proximity to absolute optimum, the sourcing decisions will need to be revised when such delays occur. A modified mathematical modeling will need to be constructed to include such changes in sourcing decisions;
- A different model can be built for networks with activity durations behaving in stochastic nature that fits continuous probability distributions;
- There might be projects where the sourcing of a single activity might be realized not from only one mill, but from a set of mills. This case will require significant re-formulation of the mathematical model;
- Some supply constraints can be introduced to the model; some mills might not be able to supply all the materials needed for an activity and thus sourcing decisions might differ due to these capacity constraints;
- Some resource constraints can be introduced to the model; some activities might not be performed in parallel due to scarcity of resources such as labor capacity. These constraints will require a brand new modeling with a different solution approach.

APPENDIX A. STEEL STRUCTURE PROJECT

A.1. Structural Drawings of a Steel Structure

- Drawing in Figure A.1 shows the 3D view of the primary sign tower,
- Drawing in Figure A.2 shows plan views of the primary sign tower,
- Drawing in Figure A.3 and A.4 show cross sections of the primary sign tower,
- Drawing in Figure A.5 provides the stairs of the primary sign tower,
- Drawing in Figure A.6 provides the joint details of the primary sign tower.

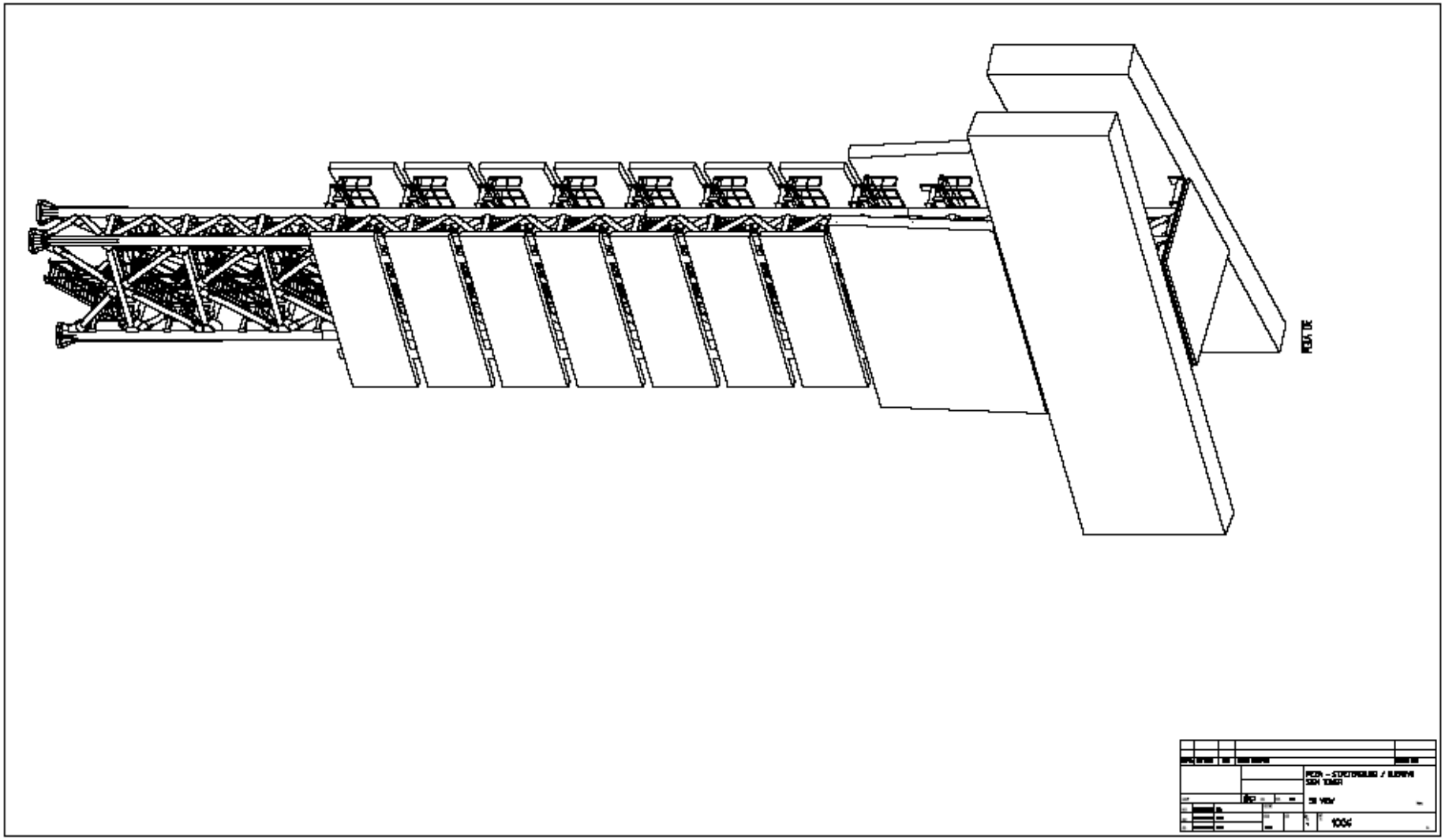

Figure A.1. 3D view of the primary sign tower

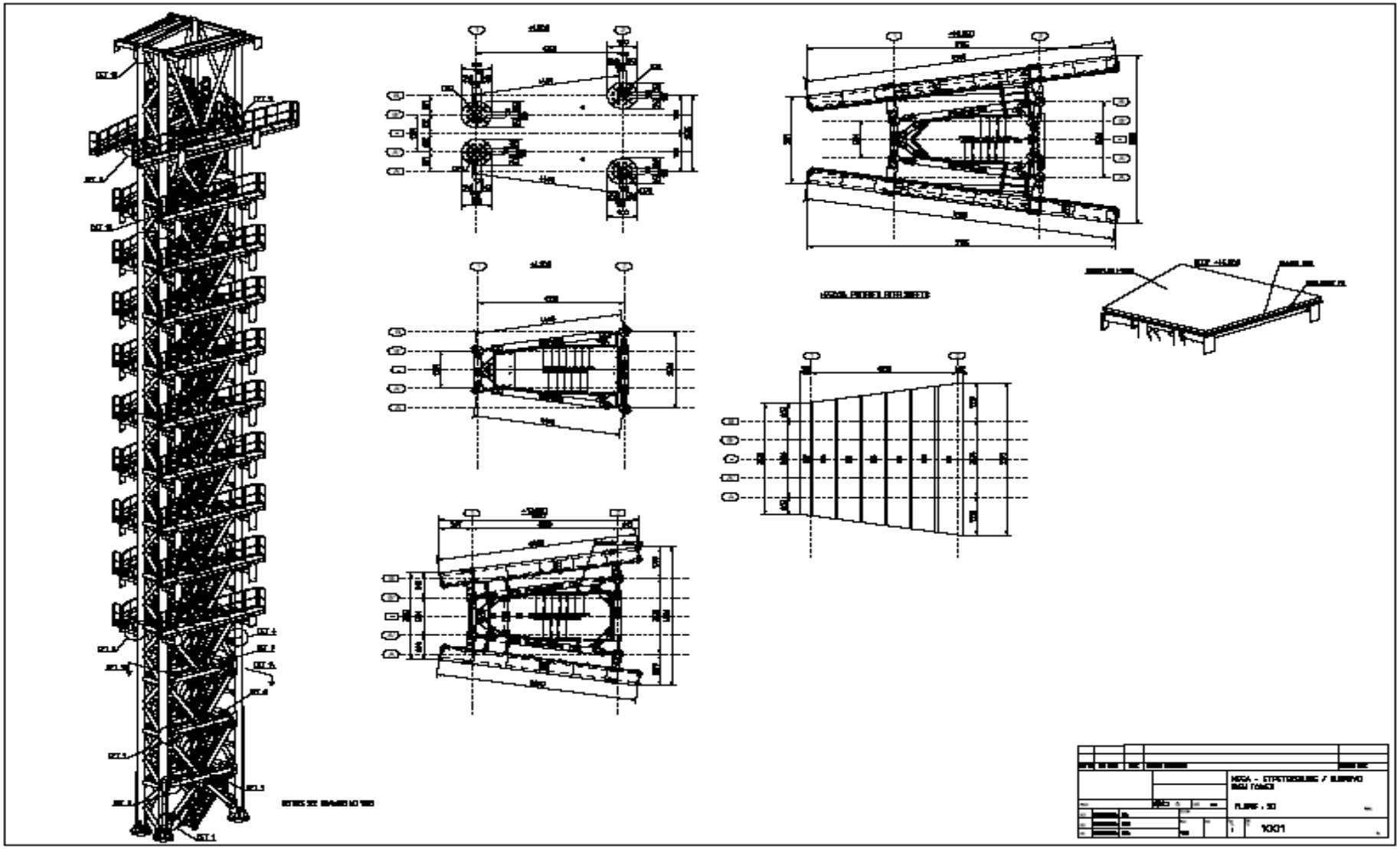

Figure A.2. Plans of the primary sign tower

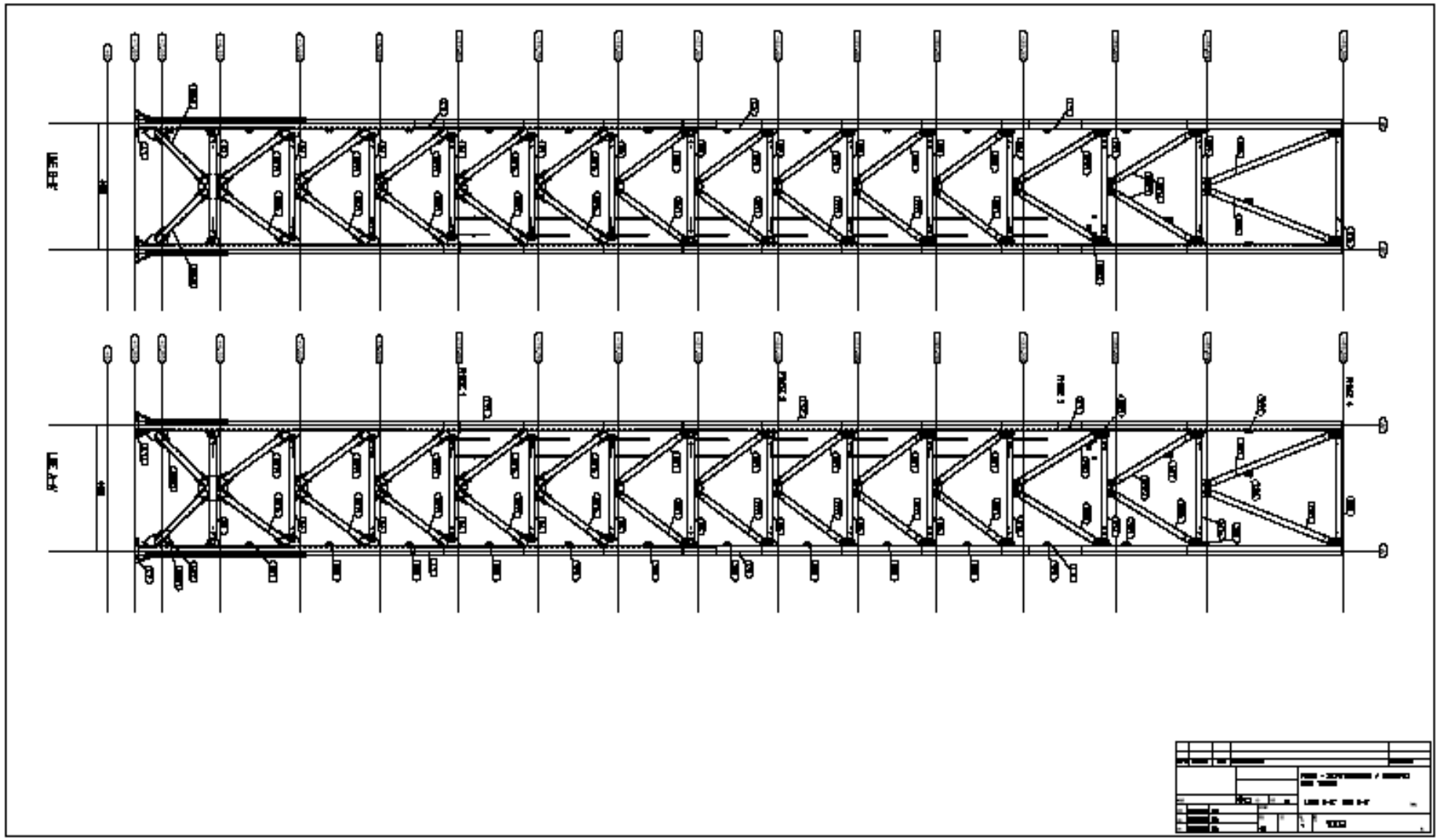

Figure A.3. Sections of the primary sign tower

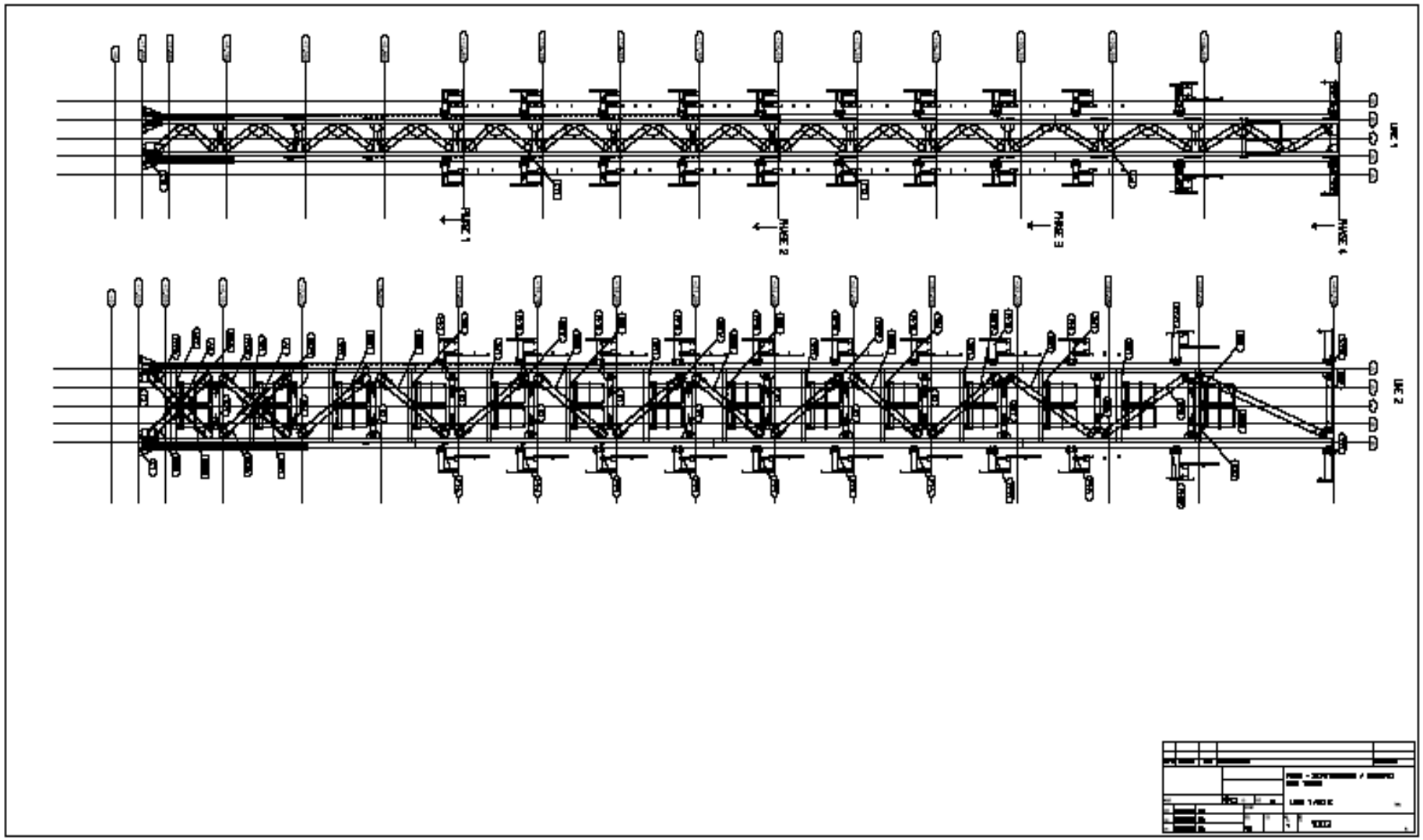

Figure A.4. Sections of the primary sign tower

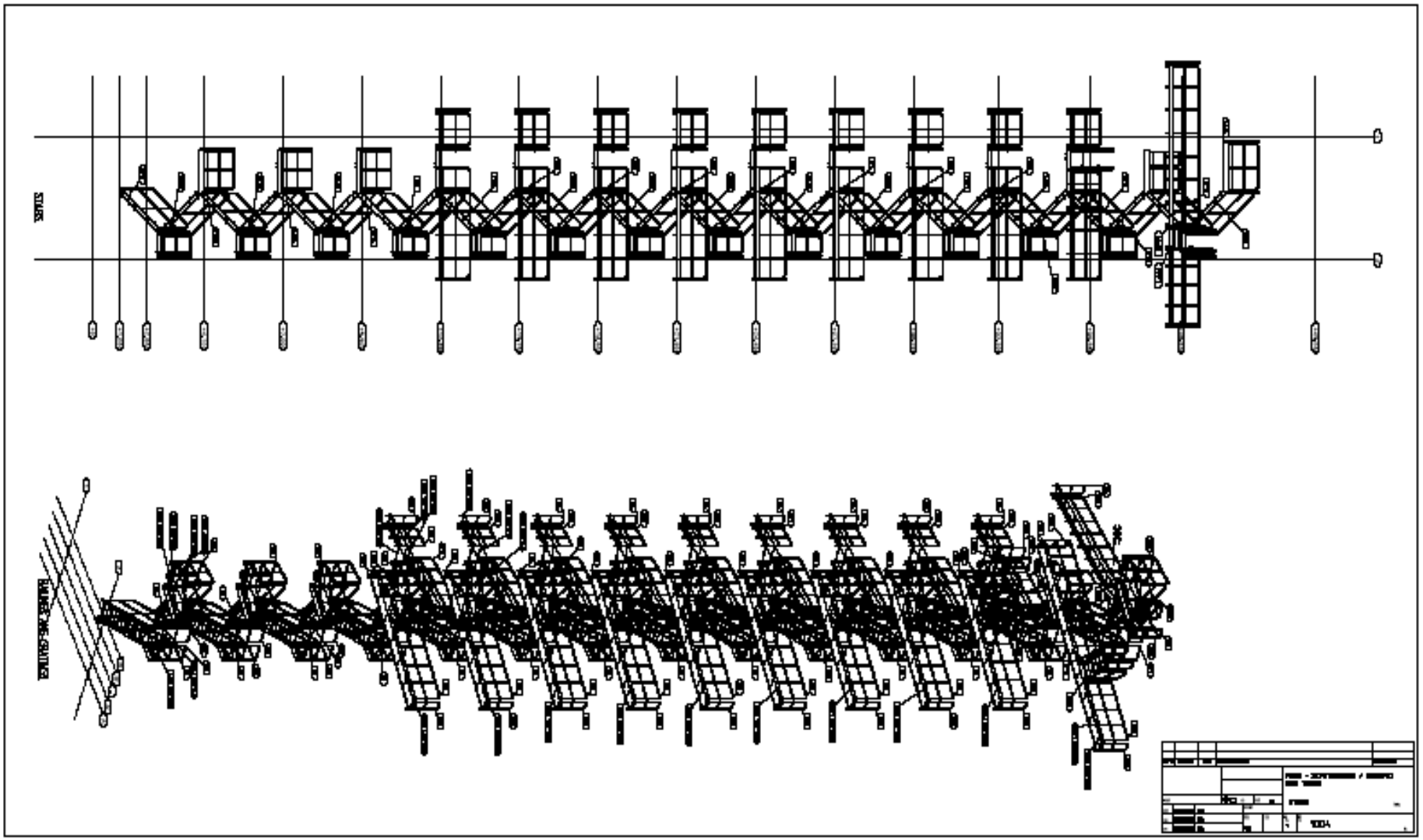

Figure A.5. Stairs of the primary sign tower

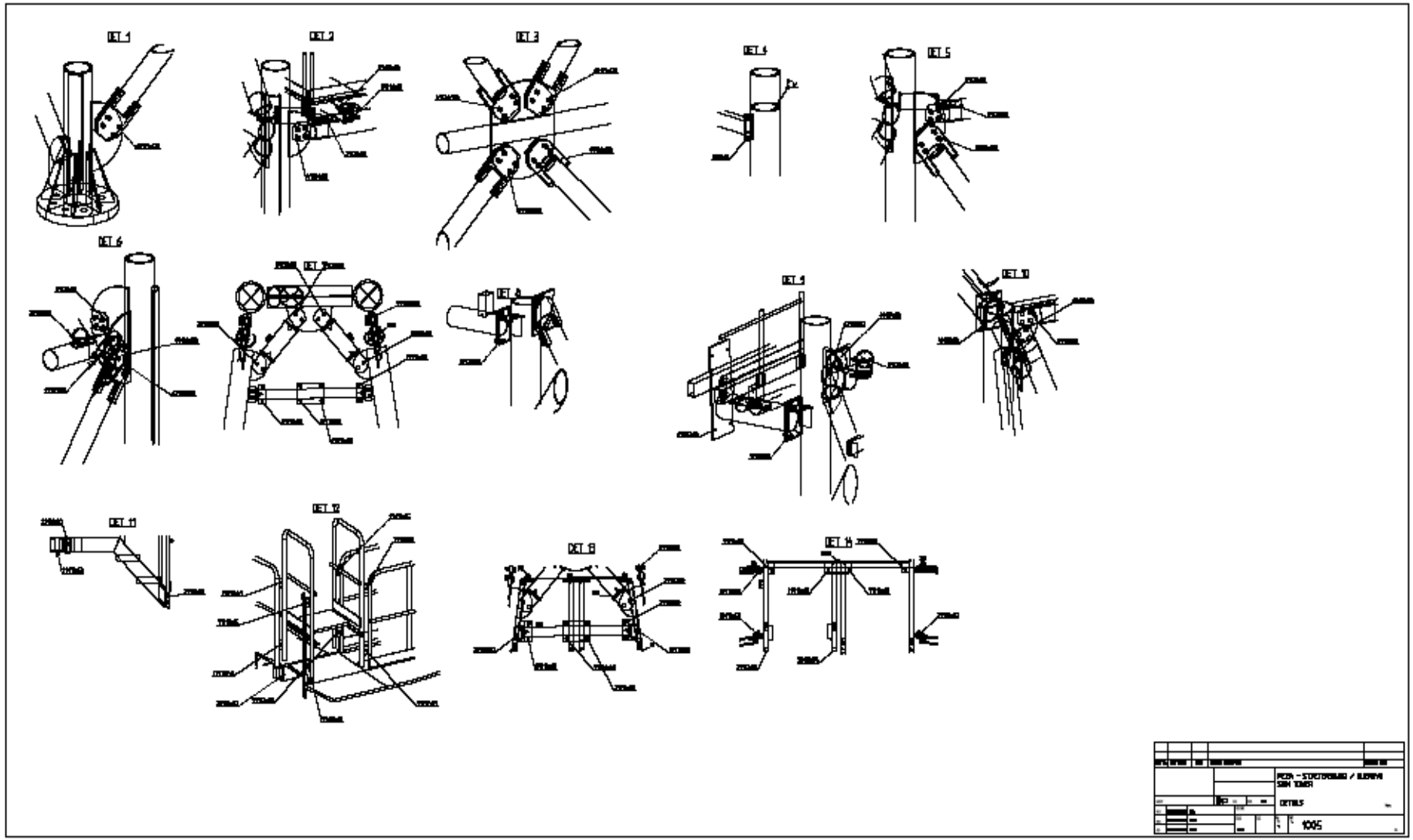

Figure A.6. Joint details of the primary sign tower

A.2. Bill of Quantities of a Steel Structure

Table A.1. Bill of quantities for the primary sign tower

MEGA - ST.PETERSBURG / KUDRO Date 05/05/2006
SIGN TOWER

Steel structures / Assemblylis Total

all parts are galvanized

No.	Ass.	Qty.		Size	Grade	Main	Fixing	Both	Total	Rev	Rev.date
1	A1	1	PLATE	PL25*900	S355J2G3	123.8	0.0	123.8	123.8		
2	A2	1	PLATE	PL25*900	S355J2G3	123.8	0.0	123.8	123.8		
3	B1	1	BEAM	CFCHS219.1*6	S355J2H	110.6	176.9	287.5	287.5		
4	B2	1	BEAM	CFCHS219.1*6	S355J2H	110.6	176.9	287.5	287.5		
5	B3	4	BEAM	CFCHS219.1*6	S355J2H	103.6	86.6	190.3	761.0		
6	B4	4	BEAM	CFCHS219.1*6	S355J2H	103.6	86.6	190.3	761.0		
7	B5	1	BEAM	CFCHS219.1*6	S355J2H	103.6	49.9	153.5	153.5		
8	B6	1	BEAM	CFCHS219.1*6	S355J2H	103.6	49.9	153.5	153.5		
9	B7	8	BEAM	CFCHS219.1*6	S355J2H	47.1	29.1	76.2	609.5		
10	B8	4	BEAM	CFCHS219.1*6	S355J2H	109.2	49.9	159.0	636.1		
11	B9	4	BEAM	CFCHS219.1*6	S355J2H	109.2	49.9	159.0	636.1		
12	B10	1	BEAM	CFCHS219.1*6	S355J2H	109.2	49.9	159.0	159.0		
13	B11	1	BEAM	CFCHS219.1*6	S355J2H	109.2	48.8	157.9	157.9		
14	B12	1	BEAM	CFCHS219.1*6	S355J2H	109.2	48.8	157.9	157.9		
15	B13	1	BEAM	CFCHS219.1*6	S355J2H	109.2	49.9	159.0	159.0		
16	B14	1	BEAM	CFCHS219.1*6	S355J2H	109.2	48.8	157.9	157.9		
17	B15	1	BEAM	CFCHS219.1*6	S355J2H	109.2	48.8	157.9	157.9		
18	B16	2	BEAM	CFCHS219.1*6	S355J2H	109.1	54.0	163.1	326.3		
19	B17	1	BEAM	CFCHS219.1*6	S355J2H	50.2	29.1	79.4	79.4		
20	B18	1	BEAM	CFCHS219.1*6	S355J2H	51.0	29.1	80.2	80.2		
21	B19	1	BEAM	CFCHS219.1*6	S355J2H	51.0	33.2	84.2	84.2		
22	B20	1	BEAM	CFCHS219.1*6	S355J2H	51.2	29.1	80.3	80.3		
23	B21	26	BEAM	CFCHS168.3*5	S355J2H	10.0	12.1	22.1	574.7		
24	B22	26	BEAM	CFCHS168.3*5	S355J2H	8.1	13.0	21.1	549.2		
25	B23	2	BEAM	CFCHS219.1*6	S355J2H	38.9	29.1	68.0	136.0		
26	BB1	20	BRACING	CFCHS219.1*6	S355J2H	86.2	20.1	106.3	2126.0		
27	BB2	1	BRACING	CFCHS219.1*6	S355J2H	100.4	23.1	123.5	123.5		

28	BB3	1	BRACING	CFCHS219.1*6	S355J2H	100.4	19.9	120.3	120.3		
29	BB4	1	BRACING	CFCHS219.1*6	S355J2H	146.5	20.3	166.8	166.8		
30	BB5	1	BRACING	CFCHS219.1*6	S355J2H	146.5	20.3	166.8	166.8		
31	BB6	1	BRACING	CFCHS219.1*6	S355J2H	146.5	24.5	171.0	171.0		
32	BB7	1	BRACING	CFCHS219.1*6	S355J2H	146.5	24.5	171.0	171.0		
33	BB8	1	BRACING	CFCHS219.1*6	S355J2H	105.1	20.3	125.5	125.5		
34	BB9	1	BRACING	CFCHS219.1*6	S355J2H	106.2	19.9	126.0	126.0		
35	BB10	1	BRACING	CFCHS219.1*6	S355J2H	149.5	19.9	169.4	169.4		
36	BB11	1	BRACING	CFCHS219.1*6	S355J2H	100.5	19.9	120.3	120.3		
37	BB12	1	BRACING	CFCHS219.1*6	S355J2H	100.5	22.9	123.3	123.3		
38	BB13	1	BRACING	CFCHS219.1*6	S355J2H	100.3	20.0	120.3	120.3		
39	BB14	1	BRACING	CFCHS219.1*6	S355J2H	100.3	20.0	120.3	120.3		
40	BB15	20	BRACING	CFCHS219.1*8	S355J2H	104.5	77.4	181.9	3638.9		
41	BB16	2	BRACING	CFCHS219.1*8	S355J2H	31.8	77.4	109.3	218.6		
42	BB17	4	BRACING	CFCHS219.1*8	S355J2H	111.7	77.4	189.1	756.4		
43	BB18	4	BRACING	CFCHS219.1*8	S355J2H	111.7	77.4	189.2	756.6		
44	BB19	1	BRACING	CFCHS219.1*8	S355J2H	112.7	77.4	190.1	190.1		
45	BB20	2	BRACING	CFCHS219.1*8	S355J2H	81.7	75.3	157.0	314.1		
46	BB21	2	BRACING	CFCHS219.1*8	S355J2H	83.0	75.3	158.3	316.6		
47	BB22	1	BRACING	CFCHS219.1*6	S355J2H	100.3	24.7	125.1	125.1		
48	BB23	1	BRACING	CFCHS219.1*6	S355J2H	100.3	24.7	125.1	125.1		
49	BB24	2	BRACING	CFCHS219.1*8	S355J2H	36.1	77.4	113.6	227.1		
50	BB25	2	BRACING	CFCHS219.1*8	S355J2H	36.5	77.4	113.9	227.8		
51	BB26	2	BRACING	CFCHS219.1*8	S355J2H	32.5	77.4	109.9	219.8		
52	C1	1	COLUMN	CFCHS323.9*25	S355J0	979.2	952.7	1931.8	1931.8	1	5.5.2006
53	C2	1	COLUMN	CFCHS323.9*25	S355J0	979.2	1016.1	1995.3	1995.3	1	5.5.2006
54	C3	1	COLUMN	CFCHS323.9*12.5	S355J2H	1001.3	2077.4	3078.6	3078.6		
55	C4	1	COLUMN	CFCHS323.9*12.5	S355J2H	1094.4	356.9	1451.2	1451.2		
56	C5	1	COLUMN	CFCHS323.9*12.5	S355J2H	1094.4	337.6	1432.0	1432.0		
57	C6	1	COLUMN	CFCHS323.9*25	S355J0	1056.0	437.8	1493.7	1493.7		
58	C7	1	COLUMN	CFCHS323.9*25	S355J0	1056.0	437.8	1493.7	1493.7		
59	C8	1	COLUMN	CFCHS323.9*12.5	S355J2H	1097.3	211.6	1308.9	1308.9		
60	C9	1	COLUMN	CFCHS323.9*12.5	S355J2H	1097.3	235.3	1332.5	1332.5		
61	C10	1	COLUMN	CFCHS323.9*25	S355J0	1139.5	3856.0	4995.5	4995.5	1	5.5.2006
62	C11	1	COLUMN	CFCHS323.9*25	S355J0	1113.6	2714.5	3828.0	3828.0		
63	C12	1	COLUMN	CFCHS323.9*12.5	S355J2H	972.5	2181.7	3154.2	3154.2		
64	EE5	8	STRINGER	CFRHS150*50*5	S355J2H	11.0	189.6	200.6	1604.9		
65	EE11	1	STRINGER	CFRHS150*50*5	S355J2H	12.4	128.7	141.1	141.1		
66	EE14	1	STRINGER	CFRHS150*50*5	S355J2H	12.4	183.9	196.3	196.3		
67	EE21	1	STRINGER	CFRHS150*50*5	S355J2H	12.6	27.9	40.4	40.4		
68	EE22	1	STRINGER	CFRHS150*50*5	S355J2H	12.6	27.9	40.4	40.4		
69	EE24	3	STRINGER	CFRHS150*50*5	S355J2H	12.4	192.1	204.5	613.6		
70	EE25	8	STRINGER	CFRHS150*50*5	S355J2H	12.4	184.8	197.2	1577.6		
71	EE27	1	STRINGER	CFRHS150*50*5	S355J2H	11.0	188.7	199.7	199.7		
72	EE28	3	STRINGER	CFRHS150*50*5	S355J2H	11.0	197.0	208.0	623.9		
73	EE29	1	STRINGER	CFRHS150*50*5	S355J2H	12.6	26.2	38.8	38.8		
74	EE30	1	STRINGER	CFRHS150*50*5	S355J2H	12.6	26.2	38.8	38.8		
75	EE31	1	STRINGER	CFRHS150*50*5	S355J2H	11.0	210.2	221.2	221.2		
76	EE32	1	STRINGER	CFRHS150*50*5	S355J2H	12.4	195.8	208.2	208.2		
77	EE33	1	STRINGER	CFRHS150*50*5	S355J2H	11.0	200.9	211.9	211.9		
78	EE34	1	STRINGER	CFRHS150*50*5	S355J2H	24.8	71.7	96.5	96.5		

79	PF3	5	PLATFORM	CFRHS150*100*5	S355J2H	110.0	390.3	500.3	2501.6	1	5.5.2006
80	PF4	1	PLATFORM	CFRHS150*100*5	S355J2H	110.0	390.3	500.3	500.3	1	5.5.2006
81	PF6	1	PLATFORM	CFRHS150*100*5	S355J2H	110.0	340.1	450.1	450.1	1	5.5.2006
82	PF7	1	PLATFORM	CFRHS150*100*5	S355J2H	110.0	340.1	450.1	450.1	1	5.5.2006
83	PF8	1	PLATFORM	CFRHS150*100*5	S355J2H	110.0	370.7	480.7	480.7	1	5.5.2006
84	PF13	1	PLATFORM	CFRHS150*100*5	S355J2H	110.0	430.0	540.0	540.0	1	5.5.2006
85	PF15	1	PLATFORM	CFRHS150*100*5	S355J2H	110.0	370.7	480.7	480.7	1	5.5.2006
86	PF18	1	PLATFORM	CFRHS150*100*5	S355J2H	110.0	430.0	540.0	540.0	1	5.5.2006
87	PF19	6	PLATFORM	CFRHS150*100*5	S355J2H	110.0	390.3	500.3	3002.0	1	5.5.2006
88	PF20	1	PLATFORM	CFRHS150*100*5	S355J2H	170.5	661.7	832.2	832.2	1	5.5.2006
89	PF21	1	PLATFORM	CFRHS150*100*5	S355J2H	170.5	662.0	832.6	832.6	1	5.5.2006
90	R1	12	RAILING	CFCHS42.4*2.6	S355J2H	2.9	9.8	12.7	152.7	1	5.5.2006
91	R2	14	RAILING	CFCHS42.4*2.6	S355J2H	2.9	9.0	11.9	167.0	1	5.5.2006
92	R3	14	RAILING	CFCHS42.4*2.6	S355J2H	2.9	9.0	11.9	166.6	1	5.5.2006
93	R4	13	RAILING	CFCHS42.4*2.6	S355J2H	2.9	8.4	11.3	146.9	1	5.5.2006
94	R5	10	RAILING	CFCHS42.4*2.6	S355J2H	2.9	6.4	9.2	92.4	1	5.5.2006
95	R6	10	RAILING	CFCHS42.4*2.6	S355J2H	2.9	6.4	9.2	92.4	1	5.5.2006
96	R7	9	RAILING	CFCHS42.4*2.6	S355J2H	2.9	6.4	9.2	83.2	1	5.5.2006
97	R8	11	RAILING	CFCHS42.4*2.6	S355J2H	2.9	15.4	18.3	201.5	1	5.5.2006
98	R9	1	RAILING	CFCHS42.4*2.6	S355J2H	2.9	6.7	9.6	9.6	1	5.5.2006
99	R10	1	RAILING	CFCHS42.4*2.6	S355J2H	2.9	6.7	9.6	9.6	1	5.5.2006
100	R11	16	RAILING	CFCHS42.4*2.6	S355J2H	2.8	48.1	50.9	814.9	1	5.5.2006
101	R13	1	RAILING	CFCHS42.4*2.6	S355J2H	2.8	68.4	71.2	71.2	1	5.5.2006
102	R15	8	RAILING	CFCHS42.4*2.6	S355J2H	2.8	16.1	18.9	151.5	1	5.5.2006
103	R16	8	RAILING	CFCHS42.4*2.6	S355J2H	2.8	16.1	18.9	151.5	1	5.52006
104	R19	1	RAILING	CFCHS42.4*2.6	S355J2H	2.8	34.8	37.7	37.7	1	5.5.2006
105	R21	1	RAILING	CFCHS42.4*2.6	S355J2H	2.9	15.4	18.3	18.3	1	5.5.2006
106	R24	1	RAILING	CFCHS42.4*2.6	S355J2H	2.8	48.0	50.8	50.8	1	5.5.2006
107	R27	1	RAILING	CFCHS42.4*2.6	S355J2H	3.5	11.5	15.1	15.1	1	5.5.2006
108	R31	1	RAILING	CFCHS42.4*2.6	S355J2H	2.8	68.4	71.2	71.2	1	5.5.2006
109	R32	1	RAILING	CFCHS42.4*2.6	S355J2H	2.8	34.8	37.7	37.7	1	5.5.2006
110	R34	1	RAILING	CFCHS42.4*2.6	S355J2H	3.5	11.5	15.1	15.1	1	5.5.2006
111	R35	1	RAILING	CFCHS42.4*2.6	S355J2H	3.5	11.5	15.0	15.0	1	5.5.2006
112	R36	1	RAILING	CFCHS42.4*2.6	S355J2H	3.5	11.5	15.0	15.0	1	5.5.2006
113	R37	1	RAILING	CFCHS42.4*2.6	S355J2H	2.8	48.0	50.8	50.8	1	5.5.2006
114	R38	1	RAILING	CFCHS42.4*2.6	S355J2H	2.8	16.6	19.4	19.4	1	5.5.2006
115	R39	1	RAILING	CFCHS42.4*2.6	S355J2H	2.8	16.6	19.4	19.4	1	5.5.2006
116	R42	1	RAILING	CFCHS42.4*2.6	S355J2H	2.9	11.7	14.5	14.5	1	5.5.2006
117	R46	1	RAILING	CFCHS42.4*2.6	S355J2H	2.9	9.8	12.7	12.7		
118	R47	1	RAILING	CFCHS42.4*2.6	S355J2H	2.9	9.0	11.9	11.9		
119	R48	9	RAILING	CFCHS42.4*2.6	S355J2H	2.9	6.4	9.2	83.2		
120	SB1	13	SUPPORT BEAM	CFRHS150*100*5	S355J2H	46.5	7.3	53.8	699.3		
121	SB2	1	SUPPORT BEAM	CFRHS150*100*5	S355J2H	46.5	7.1	53.6	53.6		
122	SB3	1	SUPPORT BEAM	CFRHS150*100*5	S355J2H	24.7	11.2	35.9	35.9		
123	SB5	11	SUPPORT BEAM	CFRHS150*100*5	S355J2H	21.5	13.5	35.0	384.9		
124	SB6	1	SUPPORT BEAM	CFRHS150*100*5	S355J2H	32.9	13.5	46.5	46.5		
125	SB7	1	SUPPORT BEAM	CFRHS150*100*5	S355J2H	30.3	13.5	43.8	43.8		

126	SB8	1	SUPPORT BEAM	CFRHS150*100*5	S355J2H	27.2	13.5	40.7	40.7
127	SS1	1	SUPPORT BEAM	CFCHS273*8	S355J2H	51.1	410.6	461.7	461.7
128	SS10	1	SUPPORT BEAM	CFCHS273*8	S355J2H	51.1	410.6	461.7	461.7

Total weight: 67446.1

Total number of assemblies: 420

APPENDIX B. MILP MODEL FOR THE FICTITIOUS PROJECT

Deterministic Model

SETS
i activities / 1,2,3,4,5,6,7 /
j plants / 1,2,3,4,5,6 / ;

TABLE mc(i,j) material cost - in USD - of activity i when supplied from plant j

	1	2	3	4	5	6
1	10000	0	6000	7000	0	0
2	42000	40000	0	45000	0	0
3	20000	0	0	15000	10000	0
4	35000	30000	0	0	0	25000
5	64000	0	55000	0	50000	0
6	20000	0	0	0	8000	9000
7	12000	10000	9000	0	0	0 ;

TABLE d(i,j) duration - in days - of activity i when supplied from plant j

	1	2	3	4	5	6
1	3	0	5	4	0	0
2	6	7	0	7	0	0
3	2	0	0	3	5	0
4	5	6	0	0	0	7
5	8	0	10	0	11	0
6	1	0	0	0	4	3
7	4	5	6	0	0	0 ;

SCALAR
tc tardiness cost per day / 5000 / ;

VARIABLE
z Total cost function
p(i,j) Plant selecion for activity i from set j
ef7 Earliest finish date of activity 7
es(i) Earliest start date of activity i
s7p Slack variable if project finishes before due date
s7m Slack variable if project finishes after due date ;

BINARY VARIABLES
p Plant selecion for activity i from set j ;
POSITIVE VARIABLES

```
ef7             Earliest finish date of activity 7
es              Earliest start date of activity i
s7p             Slack variable if project finishes before due date
s7m             Slack variable if project finishes after due date ;

EQUATIONS
TOTALCOST       define objective function
EarStr3         earliest start time of activity 3
EarStr4         earliest start time of activity 4
EarStr5         earliest start time of activity 5
EarStr6a        earliest start time of activity 6
EarStr6b        earliest start time of activity 6
EarStr6c        earliest start time of activity 6
EarStr6d        earliest start time of activity 6
EarStr7         earliest start time of activity 7
EarFin7         earliest finish time of activity 7
SlVar           slack variable for project due date
Plant(i)        assignment of only one plant for each activity i
NullPlant       avoidance of assignment of some plants to some activities ;

TOTALCOST ..    z =E= (s7m*tc)+SUM((i,j), p(i,j)*mc(i,j)) ;
EarStr3 ..      es('3') =G= SUM(j, d('1',j)*p('1',j)) ;
EarStr4 ..      es('4') =G= SUM(j, d('1',j)*p('1',j)) ;
EarStr5 ..      es('5') =G= SUM(j, d('1',j)*p('1',j)) ;
EarStr6a ..     es('6') =G= es('2')+SUM(j, d('2',j)*p('2',j)) ;
EarStr6b ..     es('6') =G= es('3')+SUM(j, d('3',j)*p('3',j)) ;
EarStr6c ..     es('6') =G= es('4')+SUM(j, d('4',j)*p('4',j)) ;
EarStr6d ..     es('6') =G= es('5')+SUM(j, d('5',j)*p('5',j)) ;
EarStr7 ..      es('7') =G= es('6')+SUM(j, d('6',j)*p('6',j)) ;
EarFin7 ..      ef7 =G= es('7')+SUM(j, d('7',j)*p('7',j)) ;
SlVar ..        ef7+s7p-s7m =E= 18 ;
Plant(i) ..     SUM(j, p(i,j)) =E= 1 ;
NullPlant ..
p('1','2')+p('1','5')+p('1','6')+p('2','3')+p('2','5')+p('2','6')+p('3','2')+p('3','3')+p('3','6')+p('4','3')+p(
'4','4')+p('4','5')+p('5','2')+p('5','4')+p('5','6')+p('6','2')+p('6','3')+p('6','4')+p('7','4')+p('7','5')+p('7'
,'6') =E= 0 ;


MODEL deterministic /ALL/ ;
SOLVE deterministic using mip minimizing z ;
DISPLAY z.l, ef7.l, p.l, s7p.l, s7m.l ;
```

APPENDIX C. DYNAMIC PROGRAMMING SOLUTION – TC=10,000 USD/DAY

The enclosed table provides the solution to the optimization problem, where the tardiness cost per day is 10,000 USD/day.

APPENDIX D. DYNAMIC PROGRAMMING SOLUTION – TC=4,000 USD/DAY

The enclosed table provides the solution to the optimization problem, where the tardiness cost per day is 4,000 USD/day.

APPENDIX E. DYNAMIC PROGRAMMING SOLUTION – TC=1,000 USD/DAY

The enclosed table provides the solution to the optimization problem, where the tardiness cost per day is 1,000 USD/day.

APPENDIX F. STEEL FRAME OF ENERGOBLOCK 7 – SURGUT 800MW CCPP

There exist the following drawings of Energoblock 7 in the enclosed figures:

- 3-D model of the frame,
- Sections of the building,
- Shop drawings of some steel components,
- Plan views of some portions of the building.

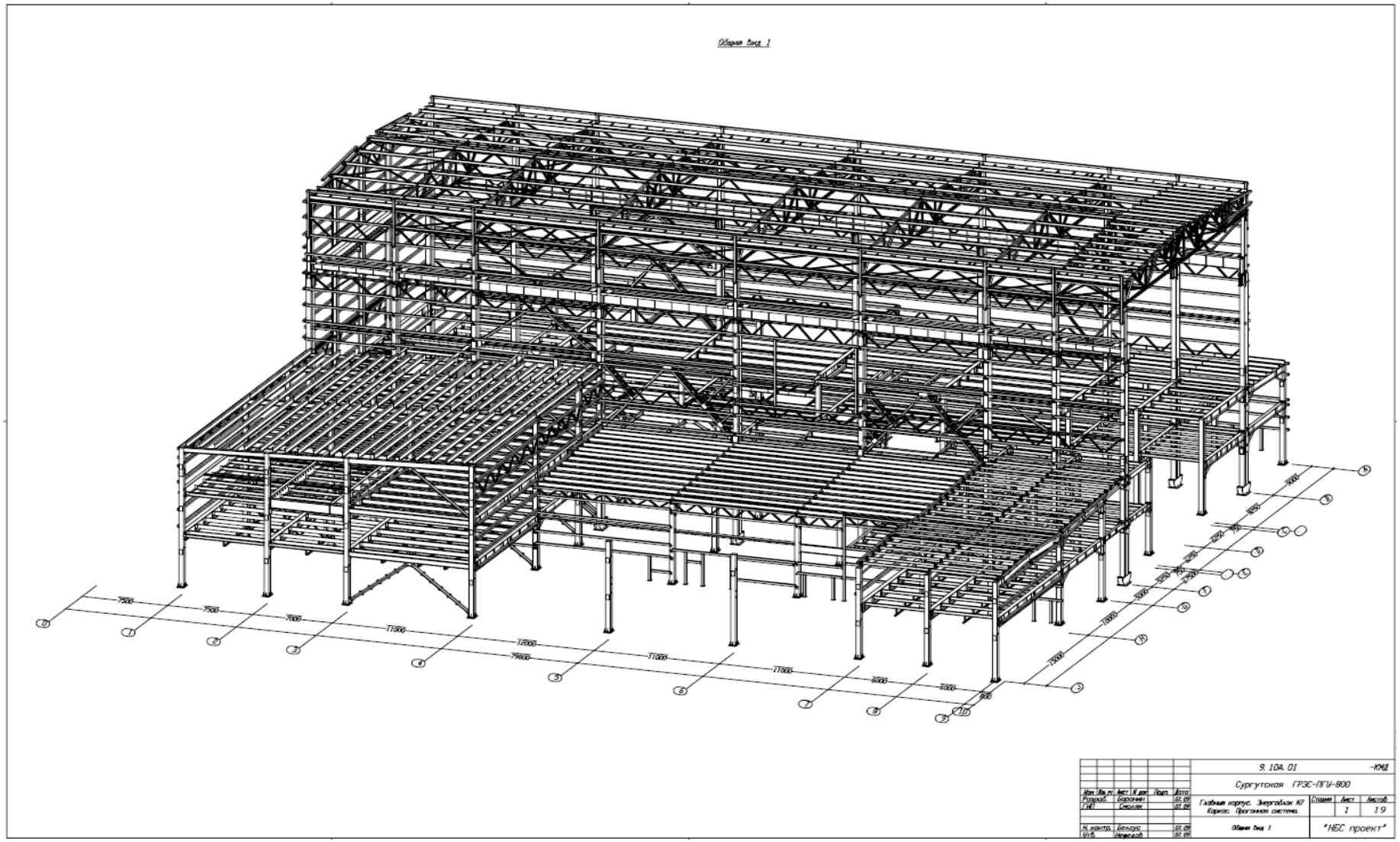

Figure F.1. 3-D model of the steel frame of Energonlock 7

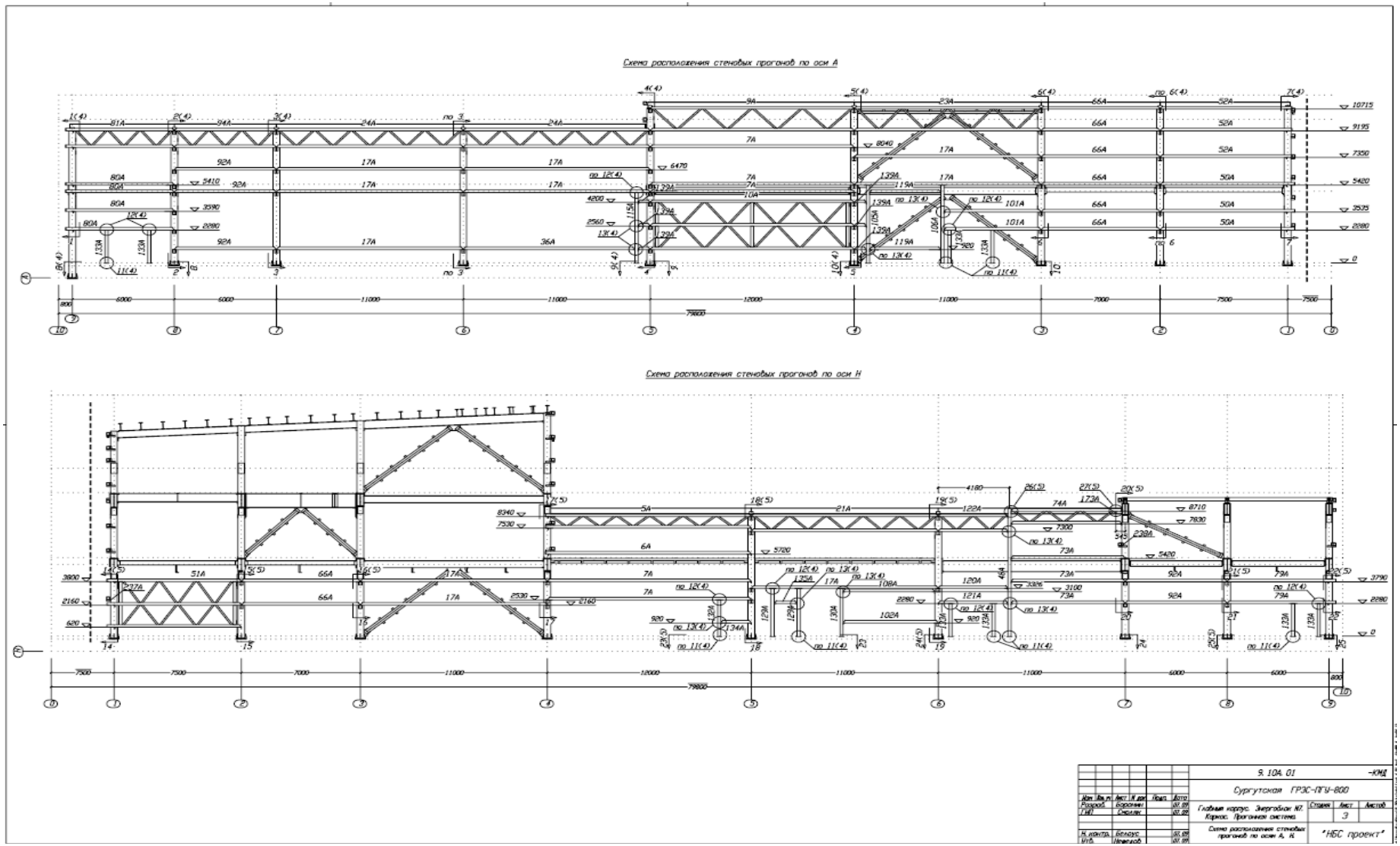

Figure F.2. Sectional view of the steel frame of Energonlock 7

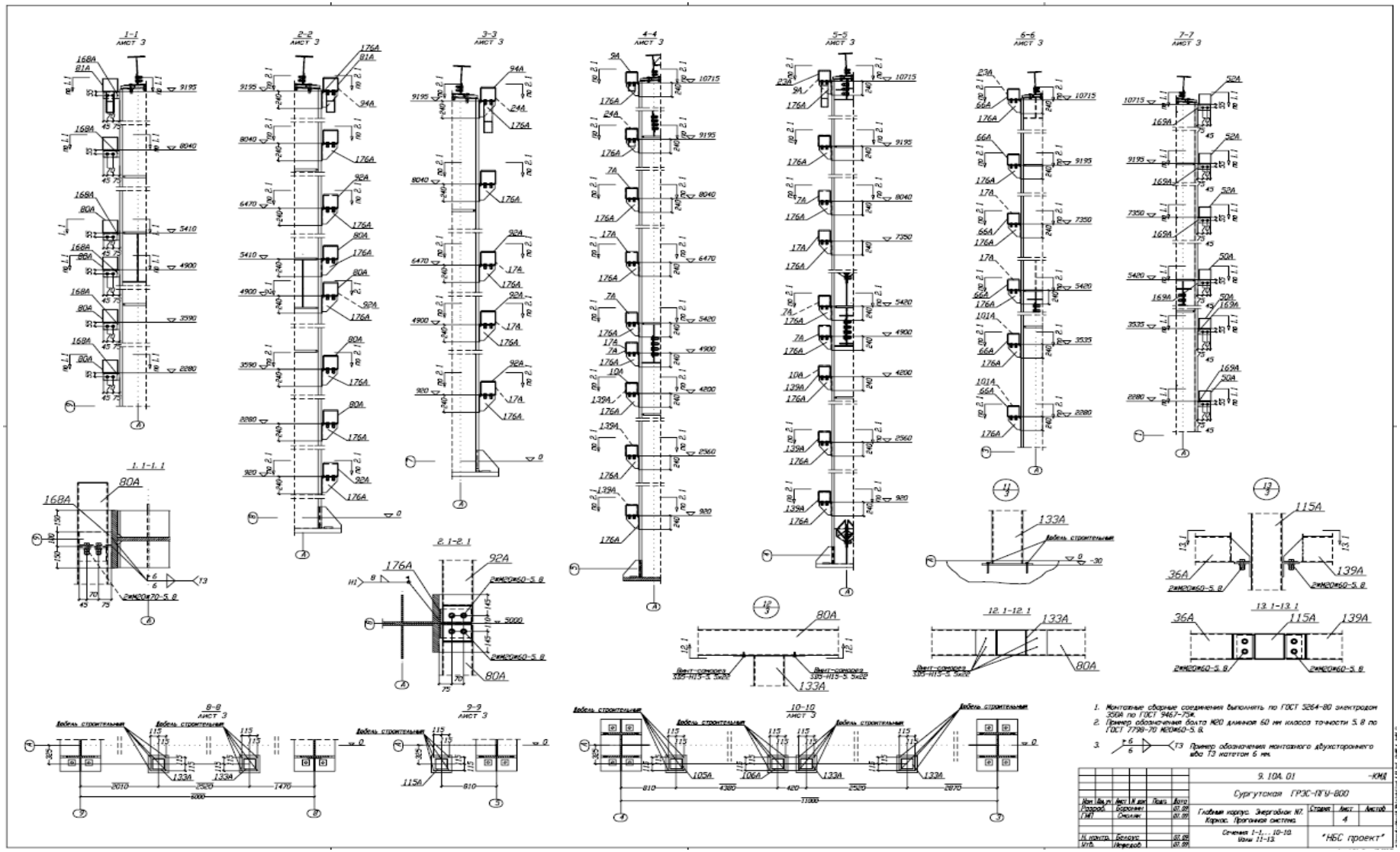

Figure F.3. Shop drawings of the components of the steel frame of Energonlock 7

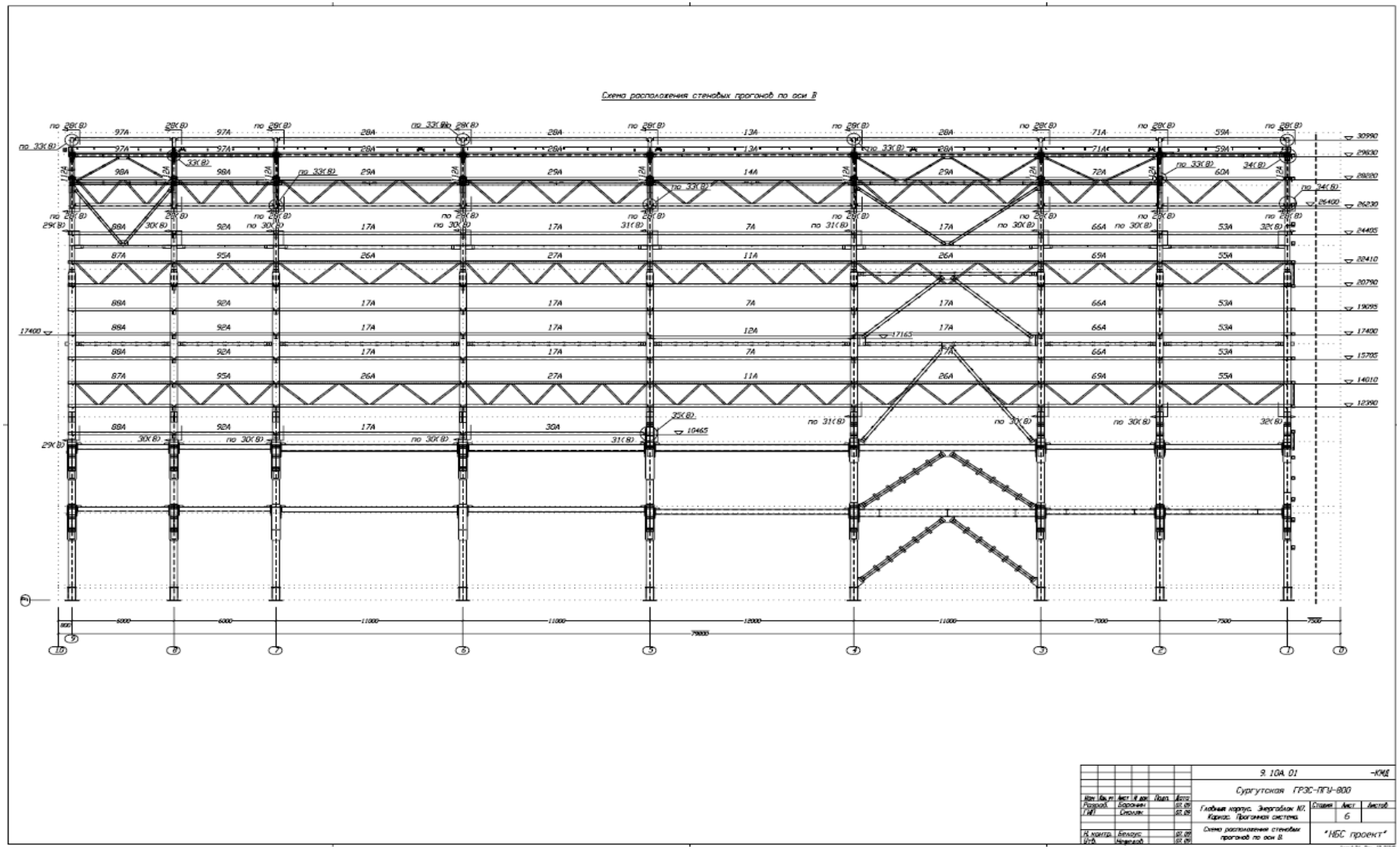

Figure F.4. Sectional view of the steel frame of Energonlock 7

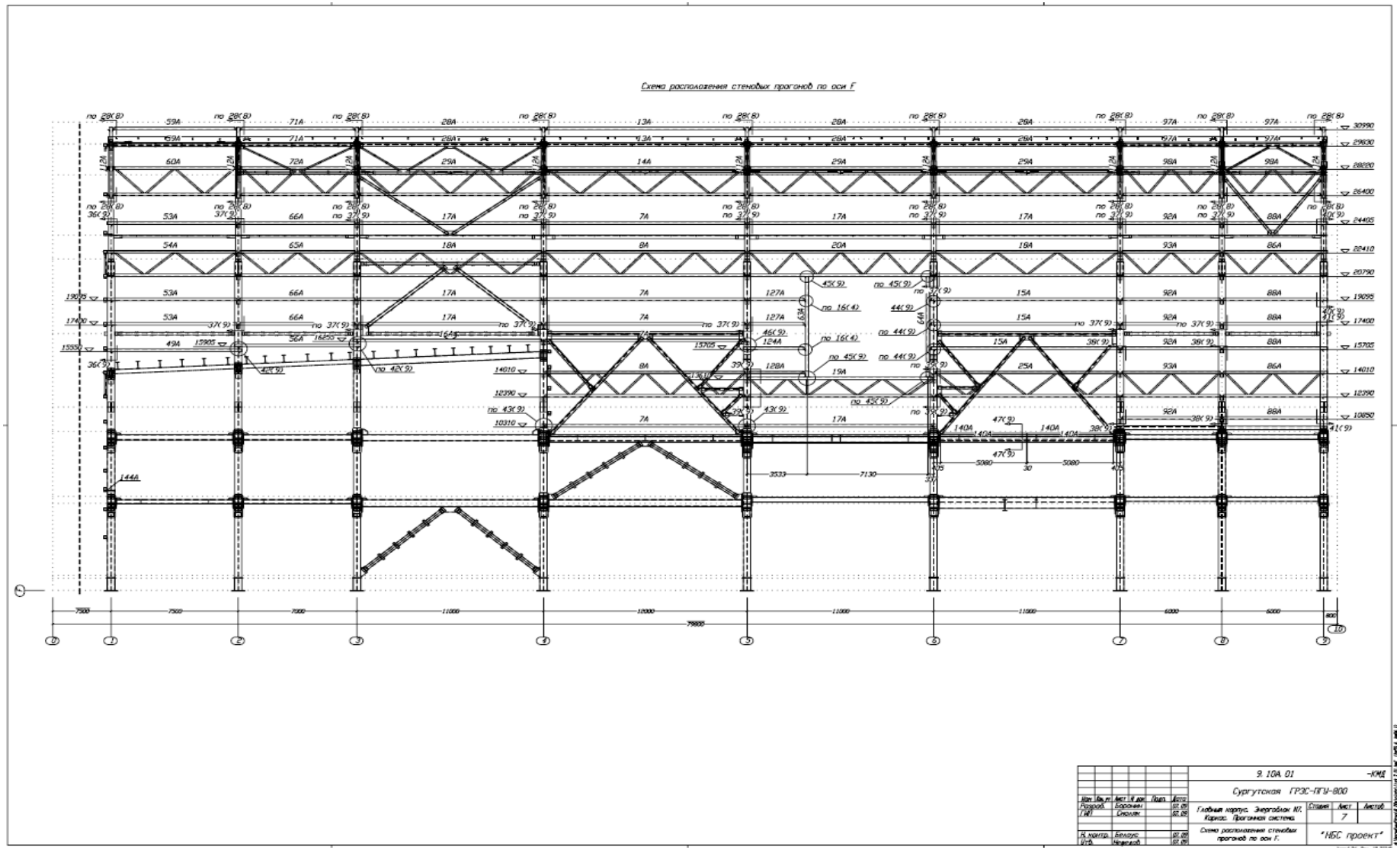

Figure F.5. Sectional view of the steel frame of Energonlock 7

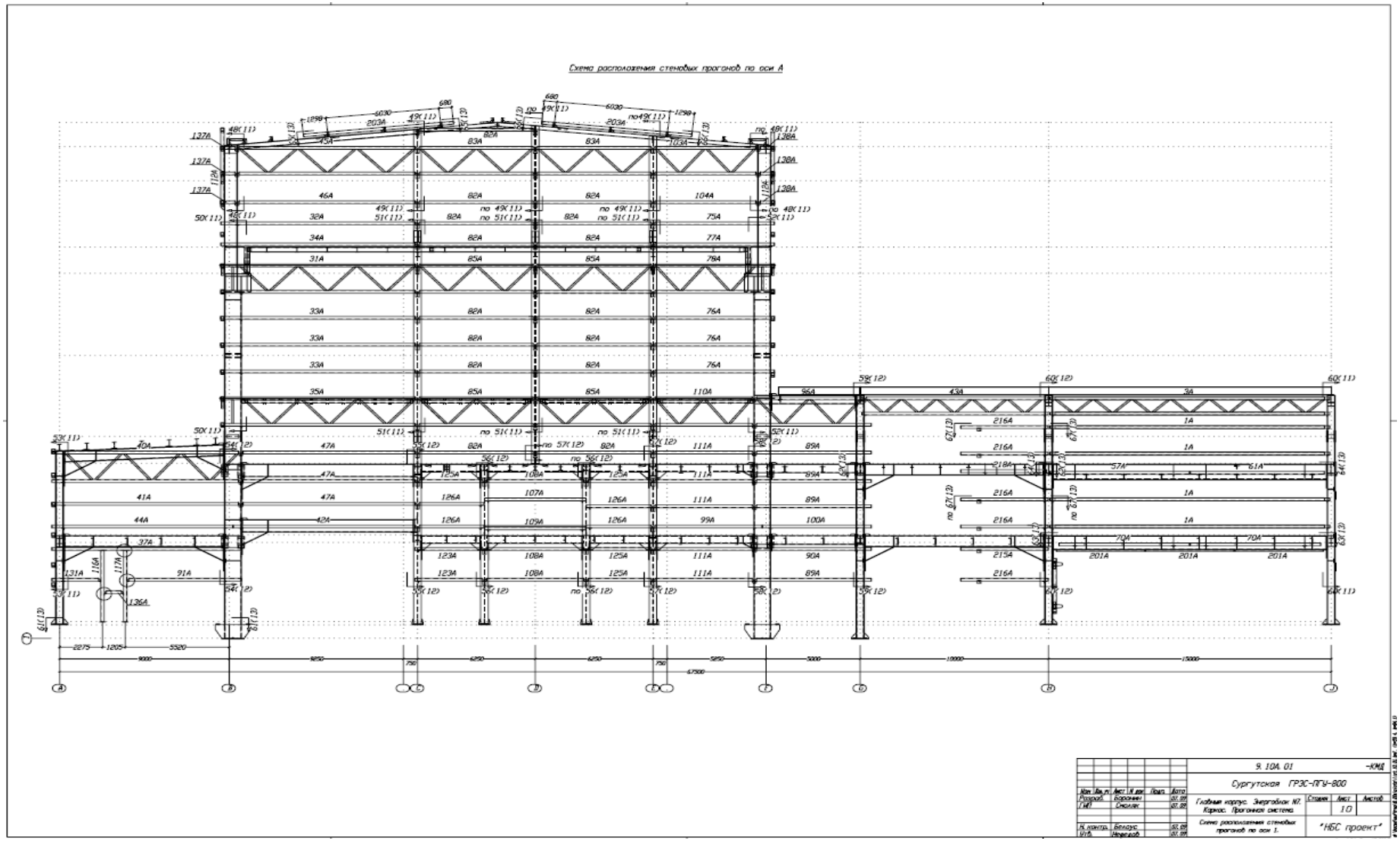

Figure F.6. Sectional view of the steel frame of Energonlock 7

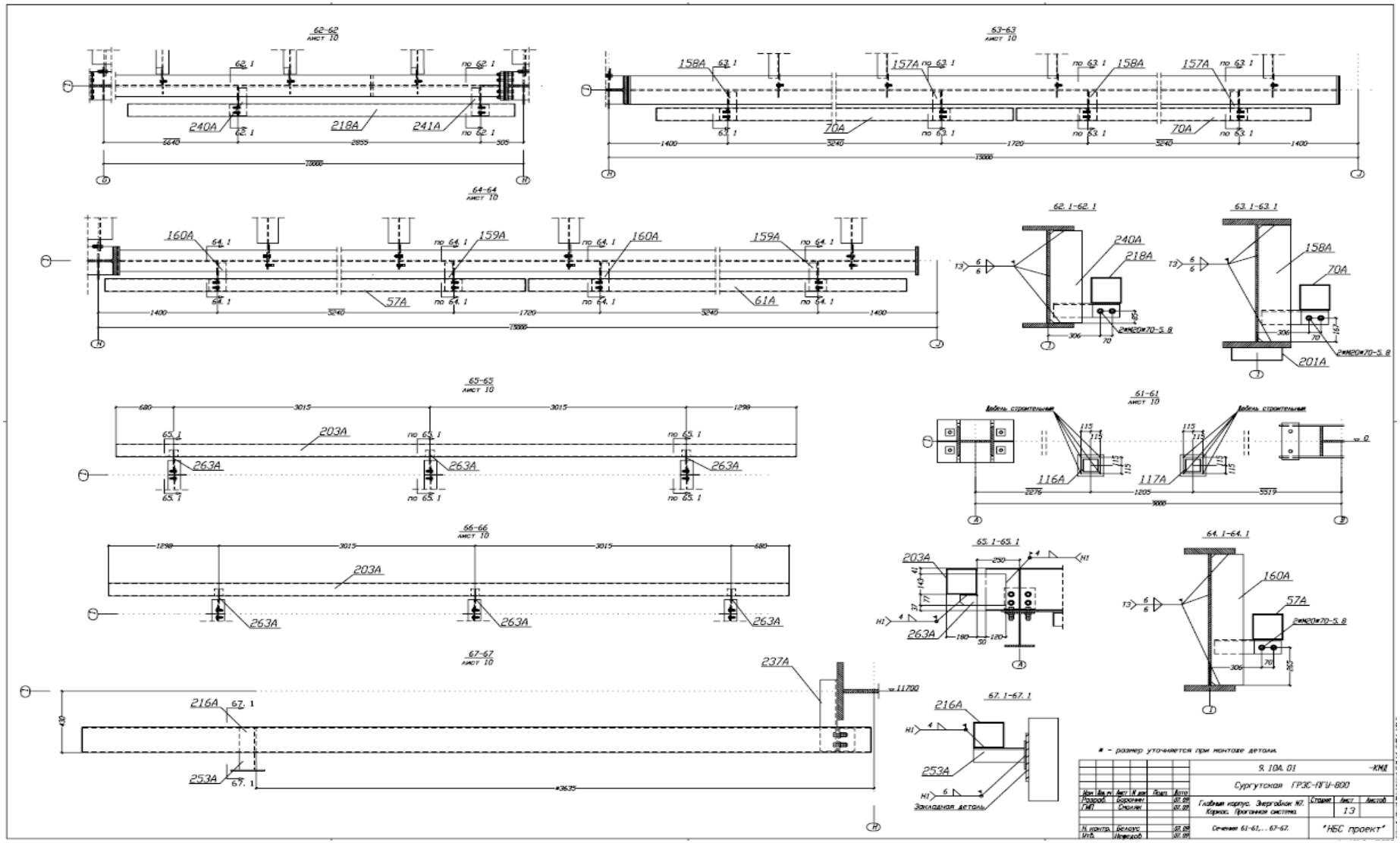

Figure F.7. Shop drawings of the components of the steel frame of Energonlock 7

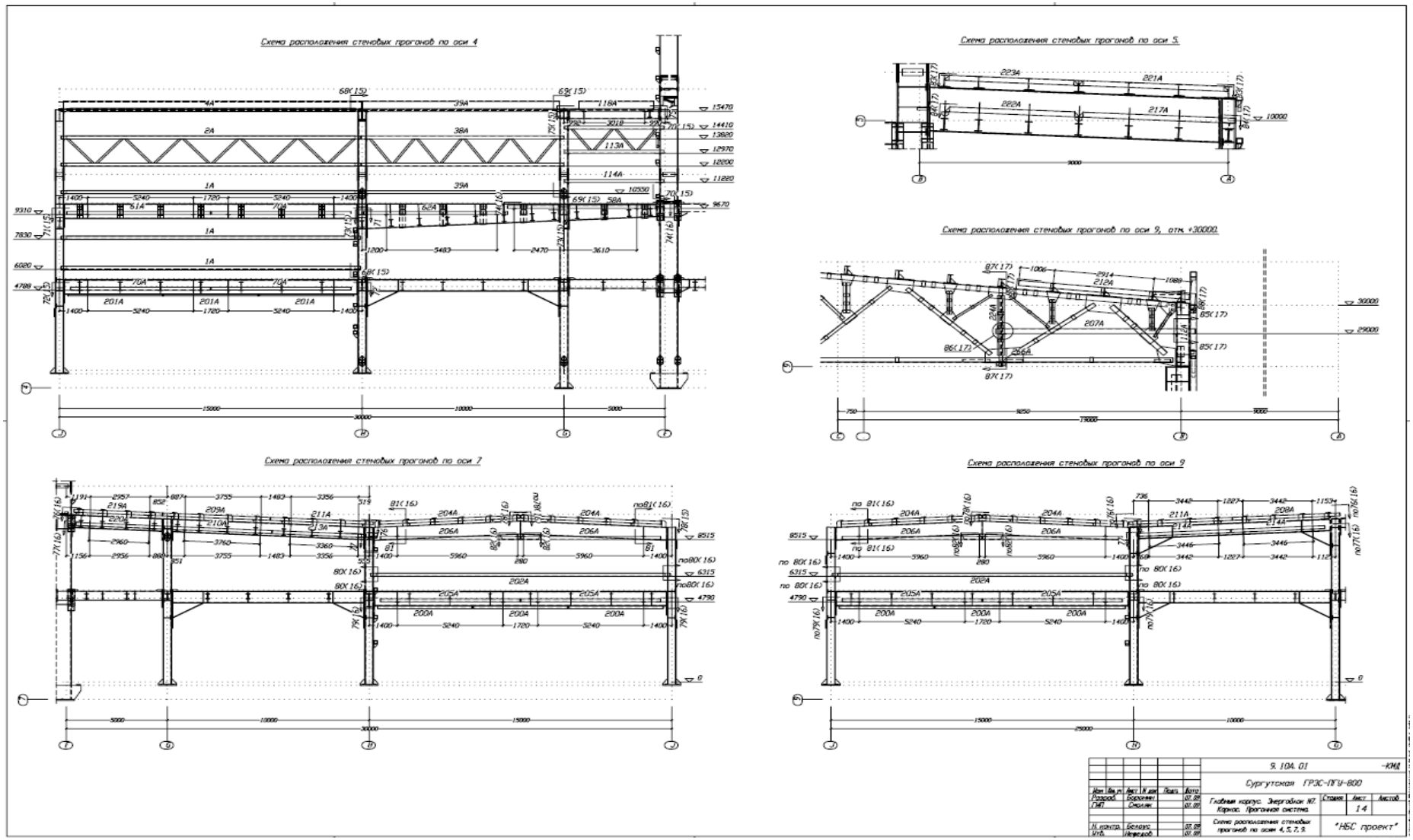

Figure F.8. Sectional view of the steel frame of Energonlock 7

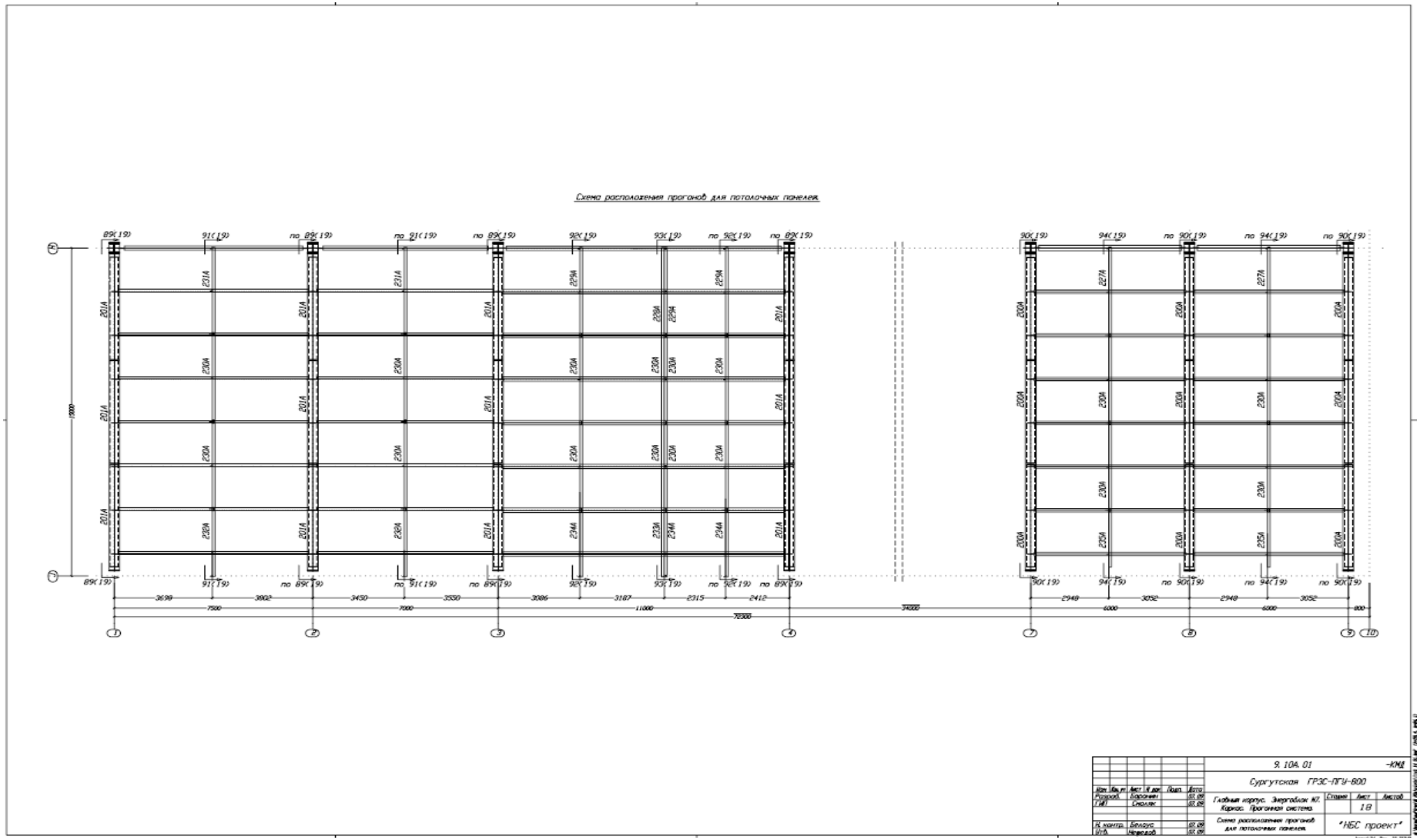

Figure F.9. Plan view of the steel frame of Energonlock 7

APPENDIX G. WORK SCHEDULE OF STEEL

Enclosed, there exist the details of the schedule of following work items for Energoblock 7 & 8:

- Column anchorages,
- Columns,
- Beam supports,
- Beams,
- Windpost supports,
- Windposts,
- Bracings,
- Window supports,
- Trusses,
- Roof purlins,
- Staircase structures,
- Fire escape stairs.

APPENDIX H. CONDENSED WORK SCHEDULE OF STEEL STRUCTURES

Enclosed, there exists the overall high-level schedule of following work items for Energoblock 7:

- Column anchorages,
- Columns,
- Beam supports,
- Beams,
- Windpost supports,
- Windposts,
- Bracings,
- Window supports,
- Trusses,
- Roof purlins,
- Staircase structures,
- Fire escape stairs.

APPENDIX I. COMPUTER CODE IN C++

```
#define _CRT_SECURE_NO_WARNINGS

#include <stdio.h>
#include <stdlib.h>
#include <math.h>

#define                 TC              10000
#define                 MS              250
#define                 SIZE12          460
#define                 SIZE10          419
#define                 SIZE9       382
#define                 SIZE8       368
#define                 SIZE7       284
#define                 SIZE6       166
#define                 SIZE4       200
#define                 SIZE3       147
#define                 SIZE2       18
#define                 SIZE1       1

int materialCost[13][6];
int d[13][18];
int s[13][18];
double p[13][18];

double V12[SIZE12][14];
double V10[SIZE10][14];
double V9[SIZE9][14];
double V8[SIZE8][14];
double V7[SIZE7][14];
double V6[SIZE6][14];
double V4[SIZE4][14];
double V3_2[SIZE3][14];
double V3_1[SIZE3][14];
double V2[SIZE2][14];
double V1[SIZE1][14];

void fillMatrices();
void funcV12();
void funcV10();
void funcV9();
void funcV8();
void funcV7();
void funcV6();
void funcV4();
void funcV3_1();
void funcV3_2();
void funcV2();
void funcV1();

int main()
{
      int i;
```

```
	FILE *fp = fopen("Last_Output.txt", "w");

	fillMatrices();

	funcV12();
	funcV10();
	funcV9();
	funcV8();
	funcV7();
	funcV6();
	funcV4();
	funcV3_1();
	funcV3_2();
	funcV2();
	funcV1();
	printf("Cost : %.1lf\n", V1[0][0]);
	fprintf(fp, "Cost : %.1lf\n", V1[0][0]);
	for (i = 1; i < 14; i++) {
		printf("%d. Factory : %.0lf\n", i, V1[0][i]);
		fprintf(fp, "%d. Factory : %.0lf\n", i, V1[0][i]);
	}
	fclose(fp);
	getchar();
	return 0;
}

void fillMatrices()
{
	int i, j, temp, tempD;

	FILE *materialF = fopen("material_cost.txt", "r");
	FILE *dF = fopen("d.txt", "r");
	FILE *sF = fopen("s.txt", "r");

	if (materialF == NULL || dF == NULL || sF == NULL) {
		printf("Can not read file!!!\n");
		getchar();
		exit(EXIT_FAILURE);
	}
	for (i = 0; i < 13; i++)
		for (j = 0; j < 6; j++)
			fscanf(materialF, "%d%d", &temp, &materialCost[i][j]);

	for (i = 0; i < 13; i++)
		for (j = 0; j < 18; j++) {
			fscanf(dF, "%d%d%d", &temp, &d[i][j], &tempD);
			p[i][j] = tempD / 100.0;
		}

	for (i = 0; i < 13; i++)
		for (j = 0; j < 18; j++)
			fscanf(sF, "%d%d%d", &temp, &s[i][j], &tempD);

	fclose(materialF);
	fclose(dF);
	fclose(sF);
```

```
}

int findMin(int *ptr, int size, int *index)
{
	int i, min;
	*index = 1;
	for (i = 0, min = ptr[0]; i < size; i++)
		if (ptr[i] > 0 && min > ptr[i]) {
			min = ptr[i];
			*index = i + 1;
		}
	return min;
}

int findMinV10(int *ptr, int size, int *indexJ, int *indexN, int *indexO)
{
	int i, min, temp;
	*indexJ = 1;
	*indexN = 1;
	*indexO = 1;
	for (i = 0, min = ptr[0]; i < size; i++)
		if (ptr[i] > 0 && min > ptr[i]) {
			min = ptr[i];
			temp = i % 6;
			*indexO = temp + 1;
			*indexN = ((i - temp) / 6) % 6 + 1;
			*indexJ = i / 36 + 1;
		}
	return min;
}

int findMax(int a, int b)
{
	int max = a;
	if (b > max)
		max = b;
	return max;
}

void funcV12()
{
	FILE *fp = fopen("V12.txt", "w");
	int i, j, t;
	int vTemp[6];
	int total;
	int w, index;

	if (fp == NULL) {
		printf("Cannot create file\n");
		getchar();
		exit(EXIT_FAILURE);
	}

	for (t = 0; t < SIZE12; t++) {
		for (j = 0; j < 6; j++) {
			if (d[12][j * 3] == 0) {
```

```
                        vTemp[j] = -1;
                        continue;
                  }
                  total = 0;
                  for (i = 0; i < 3; i++) {
                        w = MS - t - d[12][j * 3 + i];
                        if (w < 0)
                              total += (int)ceil(p[12][j * 3 + i] * abs(w));
                  }
                  vTemp[j] = materialCost[12][j] + TC * total;
            }
            V12[t][0] = findMin(vTemp, 6, &index);
            V12[t][13] = index;
            fprintf(fp, "t : %d Cost : %.1lf Factory of 13: %.0lf\n", t,
V12[t][0], V12[t][13]);
      }
      fclose(fp);
}

void funcV10()
{
      FILE *fp = fopen("V10.txt", "w");
      int t, j, i, n, o;
      int total;
      int vTempJ, vTempJON[216];
      int indexJ, indexN, indexO;

      if (fp == NULL) {
            printf("Cannot create file\n");
            getchar();
            exit(EXIT_FAILURE);
      }

      for (t = 0; t < SIZE10; t++) {
            for (j = 0; j < 6; j++) {
                  total = 0;
                  for (i = 0; i < 3; i++) {
                        total += (int)ceil(p[11][j * 3 + i] * d[11][j * 3 +
i]);
                  }
                  vTempJ = materialCost[11][j] + (int)V12[t + total][0];
                  for (n = 0; n < 6; n++) {
                        for (o = 0; o < 6; o++) {
                              if (d[11][j * 3] == 0 || d[7][n * 3] == 0 ||
d[8][o * 3] == 0) {
                                    vTempJON[j * 36 + n * 6 + o] = -1;
                                    continue;
                              }
                              vTempJON[j * 36 + n * 6 + o] = vTempJ +
materialCost[7][n] + materialCost[8][o];
                        }
                  }
            }
            V10[t][0] = findMinV10(vTempJON, 216, &indexJ, &indexN, &indexO);
            total = 0;
            for (i = 0; i < 3; i++)
```

```
                  total += (int)ceil(p[11][indexJ * 3 + i] * d[11][indexJ * 3
+ i]);
            V10[t][13] = V12[t + total][13];
            V10[t][12] = indexJ;
            V10[t][8] = indexN;
            V10[t][9] = indexO;
            fprintf(fp, "t : %d Cost : %.1lf Factory of 8: %.0lf Factory of
9: %.0lf Factory of 12: %.0lf\n", t, V10[t][0], V10[t][8], V10[t][9],
V10[t][12]);
      }
      fclose(fp);
}

void funcV9()
{
      FILE *fp = fopen("V9.txt", "w");
      int i, j, t;
      int vTemp[6];
      int total;
      int index;

      if (fp == NULL) {
            printf("Cannot create file\n");
            getchar();
            exit(EXIT_FAILURE);
      }

      for (t = 0; t < SIZE9; t++) {
            for (j = 0; j < 6; j++) {
                  if (d[6][j * 3] == 0) {
                        vTemp[j] = -1;
                        continue;
                  }
                  total = 0;
                  for (i = 0; i < 3; i++) {
                        total += (int)ceil(p[6][j * 3 + i] * d[6][j * 3 +
i]);
                  }
                  vTemp[j] = materialCost[6][j] + (int)V10[t + total][0];
            }
            V9[t][0] = findMin(vTemp, 6, &index);
            total = 0;
            for (i = 0; i < 3; i++)
                  total += (int)ceil(p[6][index * 3 + i] * d[6][index * 3 +
i]);
            for (i = 1; i < 14; i++)
                  V9[t][i] = V10[t + total][i];
            V9[t][7] = index;
            fprintf(fp, "t : %d Cost : %.1lf Factory of 7: %.0lf\n", t,
V9[t][0], V9[t][7]);
      }
      fclose(fp);
}

void funcV8()
{
```

```
	FILE *fp = fopen("V8.txt", "w");
	int i, j, t;
	int vTemp[6];
	int total;
	int index;

	if (fp == NULL) {
		printf("Cannot create file\n");
		getchar();
		exit(EXIT_FAILURE);
	}

	for (t = 0; t < SIZE8; t++) {
		for (j = 0; j < 6; j++) {
			if (d[5][j * 3] == 0) {
				vTemp[j] = -1;
				continue;
			}
			total = 0;
			for (i = 0; i < 3; i++) {
				total += (int)ceil(p[5][j * 3 + i] * s[5][j * 3 +
i]);
			}
			vTemp[j] = materialCost[5][j] + (int)V9[t + total + 3][0];
		}
		V8[t][0] = findMin(vTemp, 6, &index);
		total = 0;
		for (i = 0; i < 3; i++)
			total += (int)ceil(p[5][index * 3 + i] * s[5][index * 3 +
i]);
		for (i = 1; i < 14; i++)
			V8[t][i] = V9[t + total + 3][i];
		V8[t][6] = index;
		fprintf(fp, "t : %d Cost : %.1lf Factory of 6: %.0lf\n", t,
V8[t][0], V8[t][6]);
	}
	fclose(fp);
}

void funcV7()
{
	FILE *fp = fopen("V7.txt", "w");
	int i, j, t;
	int vTemp[6];
	int total;
	int index;

	if (fp == NULL) {
		printf("Cannot create file\n");
		getchar();
		exit(EXIT_FAILURE);
	}

	for (t = 0; t < SIZE7; t++) {
		for (j = 0; j < 6; j++) {
			if (d[4][j * 3] == 0) {
```

```
                        vTemp[j] = -1;
                        continue;
                  }
                  total = 0;
                  for (i = 0; i < 3; i++) {
                        total += (int)ceil(p[4][j * 3 + i] * d[4][j * 3 +
i]);
                  }
                  vTemp[j] = materialCost[4][j] + (int)V8[t + total][0];
            }
            V7[t][0] = findMin(vTemp, 6, &index);
            total = 0;
            for (i = 0; i < 3; i++)
                  total += (int)ceil(p[4][index * 3 + i] * d[4][index * 3 +
i]);
            for (i = 1; i < 14; i++)
                  V7[t][i] = V8[t + total][i];
            V7[t][5] = index;
            fprintf(fp, "t : %d Cost : %.1lf Factory of 5: %.0lf\n", t,
V7[t][0], V7[t][5]);
      }
      fclose(fp);
}

void funcV6()
{
      FILE *fp = fopen("V6.txt", "w");
      int i, j, t;
      int vTemp[6];
      int total;
      int index;

      if (fp == NULL) {
            printf("Cannot create file\n");
            getchar();
            exit(EXIT_FAILURE);
      }

      for (t = 0; t < SIZE6; t++) {
            for (j = 0; j < 6; j++) {
                  if (d[3][j * 3] == 0) {
                        vTemp[j] = -1;
                        continue;
                  }
                  total = 0;
                  for (i = 0; i < 3; i++) {
                        total += (int)ceil(p[3][j * 3 + i] * d[3][j * 3 +
i]);
                  }
                  vTemp[j] = materialCost[3][j] + (int)V7[t + total][0];
            }
            V6[t][0] = findMin(vTemp, 6, &index);
            total = 0;
            for (i = 0; i < 3; i++)
                  total += (int)ceil(p[3][index * 3 + i] * d[3][index * 3 +
i]);
```

```
			for (i = 1; i < 14; i++)
				V6[t][i] = V7[t + total][i];
			V6[t][4] = index;
			fprintf(fp, "t : %d Cost : %.1lf Factory of 4: %.0lf\n", t,
V6[t][0], V6[t][4]);
		}
		fclose(fp);
}

void funcV4()
{
		FILE *fp = fopen("V4.txt", "w");
		int i, j, t;
		int vTemp[6];
		int total;
		int w, index;

		if (fp == NULL) {
			printf("Cannot create file\n");
			getchar();
			exit(EXIT_FAILURE);
		}

		for (t = 0; t < SIZE4; t++) {
			for (j = 0; j < 6; j++) {
				if (d[10][j * 3] == 0) {
					vTemp[j] = -1;
					continue;
				}
				total = 0;
				for (i = 0; i < 3; i++) {
					w = MS - t - d[10][j * 3 + i];
					if (w < 0)
						total += (int)ceil(p[10][j * 3 + i] * abs(w));
				}
				vTemp[j] = materialCost[10][j] + TC * total;
			}
			for (i = 1; i < 14; i++)
				V4[t][i] = V6[t][i];
			V4[t][0] = findMin(vTemp, 6, &index);
			V4[t][11] = index;
			fprintf(fp, "t : %d Cost : %.1lf Factory of 11: %.0lf\n", t,
V4[t][0], V4[t][11]);
		}
		fclose(fp);
}

void funcV3_1()
{
		FILE *fp = fopen("V3_1.txt", "w");
		int i, j, t;
		int vTemp[6];
		int total;
		int index;

		if (fp == NULL) {
```

```
		printf("Cannot create file\n");
		getchar();
		exit(EXIT_FAILURE);
	}

	for (t = 0; t < SIZE3; t++) {
		for (j = 0; j < 6; j++) {
			if (d[2][j * 3] == 0) {
				vTemp[j] = -1;
				continue;
			}
			total = 0;
			for (i = 0; i < 3; i++) {
				total += (int)ceil(p[2][j * 3 + i] * s[2][j * 3 +
i]);
			}
			vTemp[j] = materialCost[2][j] + (int)V6[t + total + 3][0];
		}
		V3_1[t][0] = findMin(vTemp, 6, &index);
		total = 0;
		for (i = 0; i < 3; i++)
			total += (int)ceil(p[2][index * 3 + i] * s[2][index * 3 +
i]);
		for (i = 1; i < 14; i++)
			V3_1[t][i] = V6[t + total + 3][i];
		V3_1[t][3] = index;
		fprintf(fp, "t : %d Cost : %.1lf Factory of 4: %.0lf\n", t,
V3_1[t][0], V3_1[t][3]);
	}
	fclose(fp);
}

void funcV3_2()
{
	FILE *fp = fopen("V3_2.txt", "w");
	int i, j, t;
	int vTemp[6];
	int total;
	int index;

	if (fp == NULL) {
		printf("Cannot create file\n");
		getchar();
		exit(EXIT_FAILURE);
	}

	for (t = 0; t < SIZE3; t++) {
		for (j = 0; j < 6; j++) {
			if (d[9][j * 3] == 0) {
				vTemp[j] = -1;
				continue;
			}
			total = 0;
			for (i = 0; i < 3; i++) {
				total += (int)ceil(p[9][j * 3 + i] * d[9][j * 3 +
i]);
```

```
                  }
                  vTemp[j] = materialCost[9][j] + (int)V4[t + total - 3][0];
            }
            V3_2[t][0] = findMin(vTemp, 6, &index);
            total = 0;
            for (i = 0; i < 3; i++)
                  total += (int)ceil(p[9][index * 3 + i] * d[9][index * 3 +
i]);
            for (i = 1; i < 14; i++)
                  V3_2[t][i] = V4[t + total - 3][i];
            V3_2[t][10] = index;
            fprintf(fp, "t : %d Cost : %.1lf Factory of 4: %.0lf\n", t,
V3_2[t][0], V3_2[t][10]);
      }
      fclose(fp);
}

void funcV2()
{
      FILE *fp = fopen("V2.txt", "w");
      int i, j, t;
      int vTemp[6];
      int temp_1, temp_2;
      int index;

      if (fp == NULL) {
            printf("Cannot create file\n");
            getchar();
            exit(EXIT_FAILURE);
      }

      for (t = 0; t < SIZE2; t++) {
            for (j = 0; j < 6; j++) {
                  if (d[1][j * 3] == 0) {
                        vTemp[j] = -1;
                        continue;
                  }
                  temp_1 = 0;
                  temp_2 = 0;
                  for (i = 0; i < 3; i++) {
                        temp_1 += (int)ceil(p[1][j * 3 + i] * s[1][j * 3 +
i]);
                        temp_2 += (int)ceil(p[1][j * 3 + i] * d[1][j * 3 +
i]);
                  }
                  vTemp[j] = materialCost[1][j] + (int)(V3_1[t + temp_1 +
9][0] + V3_2[t + temp_2 - 7][0]);
            }
            V2[t][0] = findMin(vTemp, 6, &index);
            temp_1 = 0;
            temp_2 = 0;
            for (i = 0; i < 3; i++) {
                  temp_1 += (int)ceil(p[1][index * 3 + i] * s[1][index * 3 +
i]);
                  temp_2 += (int)ceil(p[1][index * 3 + i] * d[1][index * 3 +
i]);
```

```
			}
			for (i = 1; i < 14; i++)
				V2[t][i] = V3_2[t + temp_2 - 7][i];
			V2[t][3] = V3_1[t + temp_1 + 9][3];
			V2[t][2] = index;
			fprintf(fp, "t : %d Cost : %.1lf Factory of 2: %.0lf\n", t,
V2[t][0], V2[t][2]);
		}
		fclose(fp);
}

void funcV1()
{
		FILE *fp = fopen("V1.txt", "w");
		int i, j, t;
		int vTemp[6];
		int total;
		int index;

		if (fp == NULL) {
			printf("Cannot create file\n");
			getchar();
			exit(EXIT_FAILURE);
		}

		for (t = 0; t < SIZE1; t++) {
			for (j = 0; j < 6; j++) {
				if (d[0][j * 3] == 0) {
					vTemp[j] = -1;
					continue;
				}
				total = 0;
				for (i = 0; i < 3; i++) {
					total += (int)ceil(p[0][j * 3 + i] * s[0][j * 3 +
i]);
				}
				vTemp[j] = materialCost[0][j] + (int)V2[t + total + 4][0];
			}
			V1[t][0] = findMin(vTemp, 6, &index);
			total = 0;
			for (i = 0; i < 3; i++)
				total += (int)ceil(p[0][index * 3 + i] * s[0][index * 3 +
i]);
			for (i = 1; i < 14; i++)
				V1[t][i] = V2[t + total + 4][i];
			V1[t][1] = index;
			fprintf(fp, "t : %d Cost : %.1lf Factory of 1: %.0lf\n", t,
V1[t][0], V1[t][1]);
		}
		fclose(fp);
}
```

REFERENCES

1. Uetz, M., *Algorithms for Deterministic and Stochastic Scheduling,* Ph.D. Dissertation, Berlin Technical University, Germany, 2001.

2. Stork, F., *Stochastic Resource Constrained Project Scheduling.* Ph.D. Dissertation, Berlin Technical University, Germany, 2001.

3. Yang, B., J. Geunes and W. J. O'Brien, *Resource Constrained Project Scheduling: Past Work and New Directions,* University of Florida Press, Florida, 2001.

4. Bidot, J., *A General Framework Integrating Techniques for Scheduling Under Uncertainty,* Ph.D. Dissertation, Institute National Polytechnique de Toulouse, France, 2005.

5. Choi, J., M. J. Realff and J. H. Lee, *Dynamic Programming in a Heuristically Confined State Space: A Stochastic Resource-Constrained Project Scheduling Application,* Georgia Institute of Technology Press, Atlanta.

6. Lambrechts, O., E. Demeulemeester and W. Herroelen, *Proactive and Reactive Strategies for Resource-Constrained Project Scheduling with Uncertain Resource Availabilities,* Katholieke Universiteit Leuven Press, Belgium.

7. Copertari, L. F. and N. P. Archer, *Calculating the Theoretical Project Completion Time of Large Networks in Polynomial Processing Time,* McMaster School of Business Press, Ontario.

8. Cotrell, W. D., "Simplified Program Evaluation and Review Technique", *Journal of Construction Engineering and Management,* Vol. 125, No. 1, pp. 16-22, 1999.

9. Ahuja, V. and V. Thrivengadam, "Project Scheduling and Monitoring: Current Research Status", *Construction Innovation,* Vol. 4, No. 1, pp. 19-31, 2004.

10. Pontrandolfo, P., "Project Duration in Stochastic Networks by the Pert-Path Technique", *International Journal of Project Management,* Vol. 18, No. 3, pp. 215-222, 2000.

11. Fetz, T., M. Oberguggenberger, J. Jager, D. Köll, G. Krenn, H. Lessmann and R. F. Stark, "Fuzzy Models in Geotechnical Engineering and Construction Management", *Computer-Aided Civil and Infrastructure Engineering,* Vol. 14, No. 2, pp. 93-106, 1999.

12. Hardie, N., "The Prediction and Control of Project Duration: A Recursive Model", *International Journal of Project Management,* Vol. 19, No. 7, pp. 401-409, 2001.

13. Abbasi, G. Y. and A. M. Mukattash, "Crashing PERT Networks Using Mathematical Programming", *International Journal of Project Management,* Vol. 19, No. 3, pp. 181-188, 2001.

14. Mummolo, G., "Measuring Uncertainty and Criticality in Network Planning by PERT-Path Technique", *International Journal of Project Management,* Vol. 15, No. 6, pp. 377-387, 1997.

15. Dawson, R. J. and C. W. Dawson, "Practical Proposals for Managing Uncertainty and Risk in Project Planning", *International Journal of Project Management,* Vol. 16, No. 5, pp. 299-310, 1998.

16. Birge, J. R., and F. V. Louveaux, *Introduction to Stochastic Programming*, Springer, New York, 1997.

17. Taha, H. A., *Operations Research – An Introduction*, Macmillan Publishing Company, New Jersey, 1992.

18. Kall, P. and S. W. Wallace, *Stochastic Programming*, John Wiley & Sons, New York, 1994.

19. Ruszczynski, A. and A. Shapiro, *Stochastic Programming.* Elsevier, 2003.

20. Alonso-Ayuso, A., L. F. Escudero and M. T. Ortuno, “Modeling Production Planning and Scheduling Under Uncertainty”, in S. W. Wallace and W. T. Ziemba (eds.), *Applications of Stochastic Programming*, pp. 217-252, SIAM-MPS, Philadelphia, 2005.

REFERENCES NOT CITED

Alonso-Ayuso, A., L. F. Escudero, A. Garin, M. T. Ortuno and G. Perez, 2003, "An Approach for Strategic Chain Planning under Uncertainty based on Stochastic 0-1 Programming", *Journal of Global Optimization,* Vol. 26, No. 1, pp. 97-124.

Baptiste, P., F. D. Croce, A. Grosso and V. T'kindt, 2007, "Sequencing a Single Machine with Due Dates and Deadlines: an ILP-Based Approach to Solve Very Large Instances", *Journal of Scheduling,* Vol. 13, No. 1, pp. 39-47.

Bayraksan, G. and D. P. Morton, 2005, *Assessing Solution Quality in Stochastic Programs,* The University of Texas at Austin Press, Texas.

Bayraksan, G. and D. P. Morton, 2006, *A Sequential Sampling Procedure for Stochastic Programming,* The University of Texas at Austin Press, Texas.

Bayraksan, G. and D. P. Morton, , 2005, *Assessing Solution Quality in Stochastic Programs,* The University of Texas at Austin Press, Texas.

Bertsekas, D., 1987, *Dynamic Programming – Deterministic and Stochastic Models,* Prentice-Hall, New Jersey.

Bubshait, A. A. and M. J. Cunningham, 1998, "Comparison of Delay Analysis Methodologies", *Journal of Construction Engineering and Management,* Vol. 124, No. 4, pp. 315-322.

Buchanan, C. S., K. I. M. McKinnon and G. K. Skondras, 2001, "The Recursive Definition of Stochastic Linear Programming Problems within an Algebraic Modeling Language", *Annals of Operations Research,* Vol. 104, No. 1-4, pp. 15-32.

Caballero, R., E. Cerda, M. M. Munoz and L. Ray, 2002, "Analysis and Comparisons of Some Solution Concepts for Stochastic Programming Problems", *Sociedad de Estadistica e Investigacidn Operativa,* Vol. 10, No. 1, pp. 101-123.

Danso-Amoako, M. O., W. J. O'Brien and R. Issa, *A Case Study of IFC and CIS/2 Support for Steel Supply Chain Processes,* University of Florida Press, Florida.

De La Fuente, J. L., L. F. Escudero, C. Garcia and F. J. Prieto, 1999, "A Parallel Computation Approach for Solving Multistage Stochastic Network Problems", *Annals of Operations Research,* Vol. 90, No. 0, pp. 131-160.

Demeulemeester, E. L. and W. S. Herroelen, 2002, *Project Scheduling – A Research Handbook,* Kluwer Academic Publishers, The Netherlands.

Dempster, M. A. H., J. E. Scott and G. W. P. Thompson, 2005, "Stochastic Modeling and Optimization Using Stochastics", in S. W. Wallace and W. T. Ziemba (eds.), *Applications of Stochastic Programming,* pp. 137-158, SIAM-MPS, Philadelphia.

Dubois, D., H. Fargier and J. Fortin, 2005, "Computational Methods for Determining the Latest Starting Times and Floats of Tasks in Interval-Valued Activity Networks", *Journal of Intelligent Manufacturing,* Vol. 16, No. 4-5, pp. 407-421.

Dupacova, J., 1999, "Portfolio Optimization Via Stochastic Programming: Methods of Output Analysis", *Mathematical Methods of Operations Research,* Vol. 50, No. 2, pp. 245-270.

Dyer, M. and L. Stoguie, 2005, "Computational Complexity of Stochastic Programming Problems", *Mathematical Programming,* Vol. 106, No. 3, pp. 423-432.

Elazouni, A. M. and A. A. Gab-Allah, 2004, "Finance-Based Scheduling of Construction Projects Using Integer Programming", *Journal of Construction Engineering and Management,* Vol. 130, No. 1, pp. 15-24.

Entriken, R., 2001, "Language Constructs for Modeling Stochastic Linear Programs", *Annals of Operations Research*, Vol. 104, No. 1-4, pp. 49-66.

Escudero, L. F., J. L. De La Fuente, C. Garcia and F. J. Prieto, 1999, "A Parallel Computation Approach for Solving Multistage Stochastic Network Problems", *Annals of Operations Research*, Vol. 90, No. 0, pp. 131-160.

Escudero, L. F. and J. Salmeron, 2005, "On a Fix-and-Relax Framework for a Class of Project Scheduling Problems", *Annals of Operations Research*, Vol. 140, No. 1, pp. 163-188.

Escudero, L. F., 2009, "On a Mixture of the Fix-and-Relax Coordination and Lagrangian Substitution Schemes for Multistage Stochastic Mixed Integer Programming", *Sociedad de Estadística e Investigación Operativa*, Vol. 17, pp. 5-29.

Feng, C., L. Liu and S. A. Burns, 1997, "Using Genetic Algorithms to solve Construction Time-Cost Trade-Off Problems", *Journal of Computing in Civil Engineering*, Vol. 11, No. 3, pp. 184-189.

Fragniere, E. and J. Gondzio, 2005, "Stochastic Programming from Modeling Languages", in S. W. Wallace and W. T. Ziemba (eds.), *Applications of Stochastic Programming*, pp. 95-114, SIAM-MPS, Philadelphia.

Gaivoronski, A. A., 2005, "SQG: Software for Solving Stochastic Programming Problems with Stochastic Quasi-Gradient Methods", in S. W. Wallace and W. T. Ziemba (eds.), *Applications of Stochastic Programming*, pp. 37-60, SIAM-MPS, Philadelphia.

Gassman, H. I., 1998, "Modelling Support for Stochastic Programs", *Annals of Operations Research*, Vol. 82, No. 0, pp. 107-138.

Gassmann, H. I., S. W. Wallace, W. T. Ziemba, 2005, "Stochastic Programming Computer Implementations", in S. W. Wallace and W. T. Ziemba (eds.), *Applications of Stochastic Programming*, pp. 3-8, SIAM-MPS, Philadelphia.

Gassmann, H. I., 2005, "The SMPS Format for Stochastic Linear Programs", in S. W. Wallace and W. T. Ziemba (eds.), *Applications of Stochastic Programming*, pp. 9-20, SIAM-MPS, Philadelphia.

Gassmann, H. I. and D. M. Gay, 2005, "An Integrated Modeling Environment for Stochastic Programming", in S. W. Wallace and W. T. Ziemba (eds.), *Applications of Stochastic Programming*, pp. 159-176, SIAM-MPS, Philadelphia.

Gassmann, H. I., S. L. Schwartz, S. W. Wallace and W. T. Ziemba, 2005, "Introduction to Stochastic Programming Applications", in S. W. Wallace and W. T. Ziemba (eds.), *Applications of Stochastic Programming*, pp. 179-184, SIAM-MPS, Philadelphia.

Gassmann, H. I. and E. Schweitzer, 2001, "A Comprehensive Input Format for Stochastic Linear Programs", *Annals of Operations Research*, Vol. 104, No. 1, pp. 89-125.

Haneveld, W. K. K. and M. H. Van Der Vlerk, 1999, "Stochastic Integer Programming: General Models and Algorithms", *Annals of Operations Research*, Vol. 85, No. 0, pp. 39-57.

Hochreiter, R. and G. C. Pflug, 2007, "Financial Scenario Generation for Stochastic Multi-Stage Decision Processes as Facility Location Problems", *Annals of Operations Research*, Vol. 152, No. 1, pp. 257-272.

Jeng, T. and C. M. Eastman, 1999, "Design Process Management", *Computer-Aided Civil and Infrastructure Engineering*, Vol. 14, pp. pp. 55-67.

Kall, P. and J. Mayer, 2005, "Building and Solving Stochastic Linear Programming Models with SLP-IOR", in S. W. Wallace and W. T. Ziemba (eds.), *Applications of Stochastic Programming,* pp. 79-94, SIAM-MPS, Philadelphia.

Kall, P. and S. W. Wallace, 1995, *Stochastic Programming,* John Wiley and Sons, New York.

Kim, K. and J. M. De La Garza, 2005, "Evaluation of the Resource-Constrained Critical Path Method Algorithms", *Journal of Construction Engineering and Management,* Vol. 131, No. 5, pp. 522-532.

King, A. L., S. E. Wright, G. R. Parija and R. Entriken, 2005, "The IBM Stochastic Programming System", in S. W. Wallace and W. T. Ziemba (eds.), *Applications of Stochastic Programming,* pp. 21-36, SIAM-MPS, Philadelphia.

Lee, D. and D. Arditi, 2006, "Automated Statistical Analysis in Stochastic Project Scheduling Simulation", *Journal of Construction Engineering and Management,* Vol. 132, No. 3, pp. 268-277.

,Linderoth, J., A. Shapiro and S. Wright, 2006, "The Empirical Behavior of Sampling Methods for Stochastic Programming", *Annals of Operations Research,* Vol. 142, No. 1, pp. 215-241.

Linderoth, J. and S. J. Wrigh, 2005, "Computational Grids for Stochastic Programming", in S. W. Wallace and W. T. Ziemba (eds.), *Applications of Stochastic Programming,* pp. 61-78, SIAM-MPS, Philadelphia.

Lu, M. and S. M. AbouRizk, 2000, "Simplified CPM/PERT Simulation Model", *Journal of Construction Engineering and Management,* Vol. 126, No. 3, pp. 219-226.

Lu, M. and H. Li, 2003, "Resource-Activity Critical-Path Method for Construction Planning", *Journal of Construction Engineering and Management,* Vol. 129, No. 4, pp. 412.420.

Mattila, K. G. and D. M. Abraham, 1998, "Resource Leveling of Linear Schedules Using Integer Linear Programming", *Journal of Construction Engineering and Management*, Vol. 124, No. 3, pp. 232-244.

O'Brien, W. J., K. London, and R. Vrijhoef, 2002, "Construction Supply Chain Modeling: A Research Review and Interdisciplinary Research Agenda", *In Proc. IGLC-10*, Gramado, Brasil.

O'Brien, W. J., *Construction Supply-Chain Management: A Vision for Advanced Coordination, Costing, and Control*, University of Florida Press, Florida.

Powell, W. B. and H. Topaloglu, 2005, "Fleet management", in S. W. Wallace and W. T. Ziemba (eds.), *Applications of Stochastic Programming*, pp. 185-216, SIAM-MPS, Philadelphia.

Samuelson, P. A., 1997, "How Best to Flip-Flop If You Must: Integer Dynamic Stochastic Programming for Either-Or", *Journal of Risk and Uncertainty*, Vol. 15, No. 3, pp. 183-190.

Schultz, R., 2003, "Stochastic Programming with Integer Variables", *Mathematical Programming*, Vol. 97, No: 1-2, pp. 285-309.

Schultz, R., L. Stoguie and M. H. Van Der Vlerk, 1998, "Solving Stochastic Programs with Integer Recourse by Enumeration: A Framework Using Gröbner Basis Reductions", *Mathematical Programming*, Vol. 83, No: 1-3, pp. 229-252.

Sen, S., *Stochastic Programming: Computational Challenges and Issues*, University of Arizona Press, Tucson.

Sen, S., J. L. Higle and J. R. Birge, 2000, "Duality Gaps in Stochastic Integer Programming", *Journal of Global Optimization,* Vol. 18, No. 2, pp. 189-194.

Shaftel, T. L. and B. M. Wilson, 1999, "Time-Cost Evaluation of Alternative System/Process Designs", *International Transactions in Operational Research,* Vol. 6, No. 3, pp. 275-288.

Spengler, T., H. Puchert, T. Pekuhn and O. Rentz, 1997, "Environmental Integrated Production and Recycling Management", *European Journal of Operational Research,* Vol. 97, No. 2, pp. 308-326.

Thenie, J., C. Van Delft and J. P. Vial, 2006, "Automatic Formulation of Stochastic Programs via an Algebraic Modeling Language", *Computational Management Science,* Vol. 4, No. 1, pp. 17-40.

Tommelein, I. D., N. Akel and J. C. Boyers, *Capital Projects Supply Chain Management: SC Tactics of a Supplier Organization,* University of California Press, California.

Valente, P., G. Mitra and C. A. Poojari, 2005, "A Stochastic Programming Integrated Environment", in S. W. Wallace and W. T. Ziemba (eds.), *Applications of Stochastic Programming,* pp. 115-136, SIAM-MPS, Philadelphia.

Vanhoucke, M. and E. Demeulemeester, 2003, "The Application of Project Scheduling Techniques in a Real-Life Environment", *Project Management Journal,* Vol. 34, No. 1, pp. 30-42.

Wei, C., P. Liu and Y. Tsai, 2002, "Resource-Constrained Project Management Using Enhanced Theory of Constraint", *International Journal of Project Management,* Vol. 20, No. 7, pp. 561-567.

Yamin, R. A. and D. J. Harmelink, 2001, "Comparison of Linear Scheduling Model (LSM) and Critical Path Method (CPM)", *Journal of Construction Engineering and Management*, Vol. 127, No. 5, pp. 374-381.

Zhong, D. H. and J. S. Zhang, 2006, "New Method for Calculating Path Float in Program Evaluation and Review Technique", *Journal of Construction Engineering and Management,* Vol. 29, No. 5, pp. 501-506.