UNIVERSITY OF PECS
FACULTY OF BUSINESS AND ECONOMICS
DOCTORAL SCHOOL OF BUSINESS ADMINISTRATION

A new Bi - Objective Hybrid Metaheuristic Algorithm for the Resource-Constrained Hammock Cost Problem (RCHCP)

Oren Eliezer

DOCTORAL DISSERTATION

Supervisors: Prof. György Csébfalvi
Prof. Roni Levi

Pécs, 2011

Acknowledgements

It is a pleasure to acknowledge the tutoring, advice, support and encouragement of Prof. György Csébfalvi and Prof. Roni Levi, under whose direction the following study has been written. Their input proved to be invaluable throughout the writing period. They are both mentors and friends – as well.

I would also like to thank the Head of the PhD. program at the University of Pécs, Professor Ivan Belyacz, as well as the other staff members of the department of Economic and Business who provided me with the opportunity to study and learn.

I want to express my gratitude to the opponents: Prof. Peter Dobay and Dr. Ference Kruzslicz, whom contribute a lot of smart and useful comments. I absolutely learnt a lot out of their remarks, and upgraded this disserataion.

In addition, I would also like to express my gratitude to Mrs. Stephanie Oren for the technical support they afforded me throughout the writing of the dissertation.

Last but not least, I owe a special debt of gratitude to my family for their time, patience and encouragement: my wife Shlomit , my parents – Ester and Sammy and my two daughters : Liel and Lihi.

A new Bi - Objective Hybrid Metaheuristic Algorithm for the Resource-Constrained Hammock Cost Problem (RCHCP)

Contents

Abstract

The concept of hammock activities plays a central role in project management. Hammock activities are used to fill the time span between other "normal" activities since their duration cannot be calculated or estimated at the initial stage of project planning. Typically they have been used to denote the usage of equipment needed for a particular subset of activities without predetermining the estimated time the equipment must be present on site. However, the recent literature does not offer a general and useful method to estimate the hammock's duration in the resource-constrained case. This study presents a new ***Resource Constrained Hammock Cost*** measure **(RCHC)** to solve this problem. The presented unconstrained project total hammock cost was dealt by Harhalakis, G. (1990), whom proposed the first rigorous algorithm to calculate the hammock's duration. Following it, hammock cost equals to the time difference between its hangers. Csébfalvi, G. and Csébfalvi, A. (2005) developed an implicit enumeration algorithm with effective pruning rules for the resource-constrained hammock cost problem (RCHCP). According to the NP-hard nature of the problem, the proposed implicit enumeration algorithm provides exact solutions for small-to-medium sized problems within a reasonable time, therefore in practice heuristic algorithms are needed to generate near-optimal schedules for large and highly constrained projects. In the proposed approach, a resource-constrained project is characterized by its "best" schedule, where best means a schedule in which the RCHC measure is minimal. Theoretically the resource-constrained case can be formulated as a MILP problem which can be solved for small-scale projects within a reasonable time. The presented MILP model, which is a reformulation of the traditional time oriented resource constrained project modelling MILP model, can used as a new resource constrained project scheduling model in its own right. The presented approach, in according to the NP-hard nature of the problem, includes both explicit enumeration algorithm which can be solved directly for small-scale projects and implicit enumeration algorithm which provides exact solutions for small to medium size problems in reasonable time. Large-scale projects can be managed by introducing an efficient hybrid metaheuristic for the resource-constrained hammock cost problem (RCHCP), based on Sound of Silence (SoS) approach, developed by Csébfalvi, G. (2007). In the proposed primary-secondary criteria approach, a project is characterized by its "best" schedule, where best means a

makespan minimal resource-constrained schedule for which the total hammock cost (the distance of the hammock hangers) is minimal. In order to illustrate the essence and viability of the proposed new approach, the J30 subset from a project scheduling problem library (PSPLIB) had been chosen. To solve the problems a "state-of the-art" MILP solver (CPLEX 8.1) was used. The presented benchmark problems can be used for testing the quality of exact and heuristic solution procedures to be developed in the future for this problem type.

1. Introduction

Project management has become a mainstream discipline in every institute in the world, and a challenge that every manager has to deal with during his/her career.

Project management is sometimes considered an "art of management", since it demands a variety of skills, abilities, and the implementation of many other disciplines. It covers wide areas of management like: budget, human resources management, quality control, and risk and time management.

People recognize that most of the simplest activities, like traveling from one city to another, require prior planning. This planning entails resource management. A project is defined as: any process whereby its activities and sub-activities need resources and so a preset goal is reached. However since ancient times people are in fact dealing with projects on a daily basis; only in the 1950's the formal tools and definitions of project management were developed. Firstly, techniques were developed to deal with time allocations. These techniques ensured that every activity would get its time window (its beginning and ending times), and that the whole project would be concluded by the dictated deadline. Next, another time technique was developed which dealt with resource allocation under time limits. All of these techniques were collectively termed: ***scheduling.*** Scheduling is a theory which deals with ***time constraints*** and in particular ***resource constraints.***

Thus scheduling became one of the most important issues in project management ─ both in research and practice. Resource scheduling, if done correctly, can make the difference between successful and unsuccessful projects. Many projects had a lot of activities and many kinds of resources, so scheduling becomes a NP-hard problem in project management. Many kinds of heuristics have been developed as a result of projects becoming more and more complex. This problem comes into the category of project time management as defined by the Project Management Institute (PMI). PMI is the leading organization that deals with the main principles and issues of project management. This institute defines the rules, methods and techniques of project management, and has many branches all over the world (the center is in USA). Its main objective is to train and qualify project managers.

The PMI defines the PMBOK – Project Management Body of Knowledge. The PMBOK contains rules and processes that are the main basis of project management; these are sorted into several categories. The categories include: planning and control procedures, human resource management, knowledge management, budget and finance control, monitoring the activities' progress and ***project time management.*** The latter is one of the main and most challenging issues in project management, and is one of the main issues in this dissertation. It includes five distinct processes: Activity definition, Activity development, Activity duration, Schedule development, and Schedule control. All of which will be addressed in detail later.

Project time management deals with the definition of the activities. It refers to documentation and recognition of activities, determining the order of those activities, duration estimation and scheduling.

All of the five processes are described below:

- ***Activity definition*** – identifies all activities that have to be performed and their documentation.
- ***Activity sequencing*** – identifies the precedence relations between all activities.
- ***Activity duration estimation*** – determines the time durations of every activity in the project in order to complete them within a set deadline.
- ***Schedule development*** – is a time allocation for all activities. This is done by allocating starting and ending times for all activities, and so the project's deadline is also determined. At the same time resource allocation is also determined, thus a feasible schedule is prepared.
- ***Schedule control*** – this procedure is carried out with the purpose of schedule improvement. This process ensures that after changes, the new improved schedule will remain feasible. It improves the schedule's flexibility and declines unwanted gaps between activities.

This dissertation addresses time scheduling, which includes the problem of resource allocation, and deals mainly with the scheduling of regular and hammock activities. The main objectives of this research are:

1. Minimization of a project's makespan as a primary criterion.

2. Minimization of hammock cost as a secondary criterion. (Hammock cost will be explained later.)

This research is part of a larger project, conducted under supervision of Prof. Csébfalvi G., and is geared toward developing a new model for resource scheduling ─ a model that will implement a MILP approach to solve the RCHCP. This new approach is a metaheuristic algorithm. It combines two innovative algorithms: the SoS algorithm (Csébfalvi G., 2007) and conflict repairing harmony search metaheuristic (Csébfalvi G. et. al, 2008b), and the hybrid algorithm (Eliezer and Csébfalvi G., 2009). This new approach aims towards getting a minimal makespan as the primary objective and a minimal Hammock Cost (HC) as the secondary one.

Hammock activities play a significant role in project management. Most of the research conducted in this area concentrated on unconstrained cases. Although pioneering attempts to address the constrained case were made by Vanhoucke and Van Osselaer (2004), and Csébfalvi G. and Csébfalvi A. (2005), the literature does not offer a general and useful method to compute the constrained hammock activities' durations. This dissertation presents a new metaheuristic algorithm to address this kind of issue; this new approach will calculate the hammock's duration by minimizing the hammock cost and makespan of the whole project. In keeping with this new approach, minimization of the hammock cost should reduce the total project's cost; this should be done by reorganizing the inner activities in the hammock. The hammocks' cost can be significantly reduced when more and more activities are allocated to each hammock.

The main objectives of the research are:

- To present a new approach towards dealing with hammock activities under resource constraints.
- To introduce a new algorithm, one that is based on: solving the MILP problem, and the forbidden set principle. This algorithm cannot provide an exact and optimal solution owing to the NP-hard of the problem.
- To implement a heuristic of Sound of Silence (SoS) in order to get a near-optimal solution for the MILP model above.

- To emphasize new results, and prove that those results are better than most other known scheduling approaches.

Research's stages:

- We present the traditional approach to solving RCPSP problems.
- We present a scheduling of hammock activities without constraints – according to the Harhalakis paper (1990).
- We describe research that was done on the RCHCP (scheduling hammock activities under resource constraints) – mainly on small-to-medium sized problems.
- We present SoS heuristic, in order to treat large-scale problems.
- We present new results and conclusions.

The structure of this dissertation is:

- ***First part*** (chapters 2, 3, 4) includes: theoretical background and a review of approaches, attitudes and research methodologies.

- ***Second part*** (chapter 5) presents: the new approach to solving the RCHCP by using a combination of the SoS and hybrid algorithms.

- ***Third part*** (chapters 6,7) deals with: the results of this new approach, the comparison to the traditional models, conclusions and other issues for future investigation.

Chapter 2: Theoretical Background

2.1 Overview

Throughout history organizations have been considered complicated systems that interact with the environment to achieve clear and well defined goals (Ackoff, 1970; Simon, 1977). Based on the premise that "every journey starts with a single step", organizations have adopted a philosophy that in order to reach a main target, they first must achieve intermediate goals. Therefore many tasks were organized as projects, whereby its throughput was considered as an intermediate goal. In conclusion, organizations recognized the importance of project management within its operational management.

Project management is a complex decision-making process that involves the unrelenting pressures of time and cost. It can be described by the following two theoretical definitions:

– "A temporary endeavor undertaken to create a unique product or service"
(PMI 2000).

– "A unique set of co-ordinate activities, with definite starting and finishing points, undertaken by an individual or organization to meet specific performance objectives within defined schedule, cost and performance parameters"
(BS 6079; 2000).

Tavares (2002) defined project management as a process whereby the goal is to transform a system from its initial status to a final one:

–"Any purposeful transformation leading a system Ω from an initial state s, to a specific state s' and so s' should represent the target to be achieved."

Demeulemeester and Herroelen (2002) defined it more operatively:

– "A unique process, consisting of a set of coordinated and controlled activities with start and finish dates, undertaken to achieve an objective conforming to specific requirements including constraints of time, cost and resources."

The latter definition is quite different from the one above since it emphasizes project management as an important tool for managers. It describes project management as a tool that can be used for coordinating and controlling projects more coherently, it can therefore be combined with Harhalakis's definition (1990), so a revised definition of a project with hammock activities is:

"A unique process, consisting of a set of coordinated and controlled activities. Those activities have mutual starting and finishing dates, and are undertaken to achieve an objective while conforming to specific requirements including: constraints of time, cost and resources. Some of the activities join two specific events that may be regarded as spanning two or more activities; however its duration cannot be estimated at the beginning of the project". The following elements are common to all projects:

- Each project has goals, objectives or targets and has to produce an end product, a result or any kind of output.
- Each project is unique. Each project differs from one another with regard to its targets, resources, goals and results. Even a pair of projects, which have the same target, cannot be identical in all elements.
- Constraints are mutual in all projects. Constraints are a limitation of time, and resources like: money, manpower, machines etc. Most projects deal with time and resource constraints as well as with other aspects of project management and their outcomes.
- Projects are temporary in character and all projects have starting and ending dates:
    - No project "lives" forever.
    - All resources are allocated at the beginning of the project and released after the project had been completed.
- Projects are complicated processes. Managing them entails controlling the execution of all activities and expertise in many professional issues like: budget planning, engineering, tool management and decision-making in such a way that all activities should be performed within the right time frame and should get the right output. In conclusion, project management requires the mobilization of many skills in order to get the correct and expected results.

- All projects need early planning before they can be activated, in so doing they carry a non-neglecting amount of risk and uncertainty.
- All projects have a life cycle. It begins with the conceptual design stage, and usually ends with the performance evaluation stage.

In light of the above, Tavares (2002) defined project management as:

> "The process of conceiving, preparing, organizing, driving and controlling such transformations in order that s' (the end result) will be achieved under the most convenient conditions"

From the beginning of civilization, people used to apply principles of project management, for example: hunting food or building the pyramids in ancient Egypt. However, for many decades and prior to World War II, the discipline of project management was not developed. Thereafter new technologies were developed, the world economy experienced an enormous growth, and computer technology became more and more popular; therefore new tools of project management were developed and new challenges were met. As a result of this technological progress, a new field of mathematics was developed – a field which had a major effect on the area of project management ─ Operational Research (OR). This discipline was discovered and proved powerful in solving project management problems.

The field of OR was developed during the 1950s. It concentrates on developing new heuristics and exact algorithms to deal with large and complex projects. OR is mainly applied to solve resource and time constraints. The growing use of more powerful computers greatly helped in obtaining new algorithms which in turn were developed to effectively plan the time and resource allocation of projects.

During the period of time that followed the development of industry all over the world, projects became more complicated and had many activities and constraints, consequently many of the tools that had been developed previously became less and less relevant. Projects became more dynamic. Many changes were enforced in quite a short time so that mathematicians had to develop new OR tools. At the same time many organizations recognized new opportunities to grow by their involvement in new projects. Dealing simultaneously with many projects significantly increased the amount of information and manpower that organizations had to deal with; at the same time the

factor of uncertainty also increased. In order to overcome the latter, and to deal more efficiently with all of the information, by using OR tools new heuristics and methods were developed.

During 1990's, project managers had to combine all kinds of knowledge, human and non-human resources, budget etc in order to bring a project to its target. Thus a project manager becomes an integrator of skills, resources and talents. The project manager's responsibility includes scoping the project accurately and carefully in order to be aware of changes, or unpredictable events like changes in cash flow, a reduction of an entire project's budget, or a reduction in one or more of the resources which might have significant consequences on the project's completion time. Under these kinds of circumstances, the manager has to change his/her prediction about the project's completion.

This responsible policy of management becomes an important strategy in as far as it must combine all resources to get the best result in the shortest time, and with the lowest budget. Even nowadays this approach is used supported by computer technologies (Maylor, 2003). In fact project management has become an increasingly important part of the organizations' strategy.

In conclusion, OR concentrates on the optimal ways to get the best results in the shortest time, therefore it is one of the main approaches used to solve project management issues.

There are several parameters by which a successful project can be judged:

1. Completion within the shortest time and within a pre-set budget.
2. Ending in a minimal time excluding budget limitation.
3. Getting the exact results within a time limitation and with minimal idle-time.

In our case, a project should be considered successful if it achieves the exact result, within the shortest time and with resource constraints.

When dealing with project management, three main categories must be addressed:

- Project modeling
- Project evaluation
- Project scheduling and monitoring

Project modeling:

Since a project is a collection of activities with precedence relations between them, there is a necessity to develop a new tool to describe a project: a project's model. Prior to describing the project's model, 'precedence relations between two activities' must be defined formally. Activity no. i is defined as a predecessor of activity no. j if activity no. j cannot be started before activity no. i has ended then in this case activity no. j is called a successor of activity no. i.

Battersby (1967) had developed a directed and acyclic network to represent projects is quite a basic concept in OR. Each project can be modeled by:

a) A discrete and finite set of activities $A = \{A_i : i = 1,2, \dots \dots N\}$. This set is called "jobs or activities".

b) A set of precedence conditions, $\{j_i : i = 1,2, \dots \dots N\}$ where j_i is the set of activities which are immediate predecessors of an activity A_i.

   An alternative definition of J_i is: $J_i = \{k : (k \in J_i') \cap (k \in J_m')\}$ for any m belongs to j_i' where j_i' or j_m' is the whole set of activities which have to be completed before starting i or before starting m. Similarly, the set of activities which are the immediate successors of i, K_i, can be defined by k_i $\{k : i \epsilon J_k\}$

c) A finite set of attributes $\{B_1(i), \dots \dots B_p(i)\}$ with $p \geq 1$ is defined for each activity and describes the properties that are relevant for project management such as: duration, cost, and consumption required of each resource, etc.

d) A finite set of criteria $\{V_1, \dots \dots . . V_q\}$ which emphasizes the values and the preferences of the project manager and enables him/her to compare alternative decisions concerning the management of the project. The most common criteria are: the total duration, the total cost, a cost-benefit function, and the net present value (NPV) of the project.

The best way to represent the network of activities is by using a directed graph. A directed graph contains nodes with arcs connecting them.

One improvisation is to treat the graph's nodes as activities that have to be done, and the graph's edges as the path between a precedence activity that has to be finished before the given activity starts – which means that an arc (edge), (u, v) tells us that task v must

not start before task u ends. This kind of representation is called activity on node (AON).

Another possibility is to represent a directed graph as activity on arc (AOA). Every arc represents an activity; each node symbolizes that the activity that was written in it has already been completed; and every node of this graph describes an event, for example: an edge (u,v) in an AOA graph describes an activity (u,v) which begins in event u and finishes in event v.

Hereunder is an example of a network which is described by those two approaches:

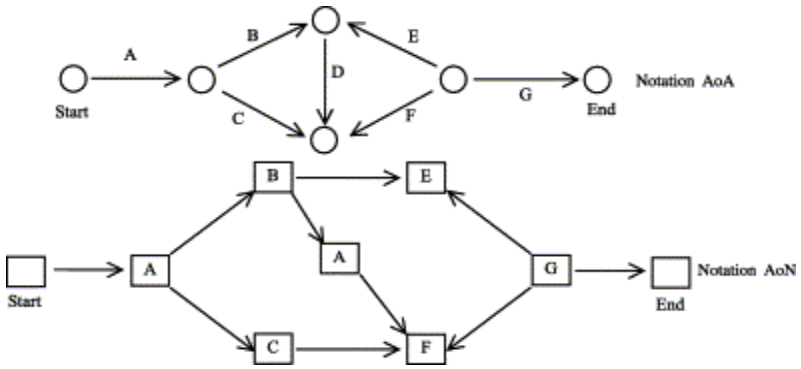

Figure 1: The network model of a project using the AOA (I) and the AON (II) notation (Tavares, 1998)

Regarding the AOA model, when two or more activities have to be finished by the same event, there is a dummy arc (a dummy activity) that must be added to the project's model. In order to understand the function of the dummy activity see figure 2 hereunder:

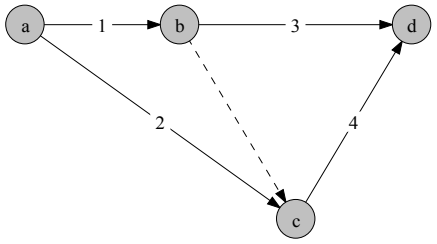

Figure 2: An example of an AOA with a dummy activity

This dotted arc is a ***dummy activity***. Dummy activities often have a zero completion time, and are used to represent precedence relationships that cannot be easily (if at all) represented using the actual activities involved in the project. Conventionally dummies are always shown as dotted arcs in network diagrams.

Regarding figure 2, both activities no.1 and no.2 have to be finished before starting activity no. 4, however, activity no.1 and no.2 are not connected to each other directly. As a result, a dummy activity between the events of completing no.1 and no.2 must be added. This dummy describes the precedence relationship between no.1 and no.2 to no.4

Often in drawing large networks we find that the easiest way to represent some precedence relationships is by dummy activities. It is important to note however that in AOA networks there is often more than one way to correctly represent the network logic by using dummies. Noteworthy too is the key to drawing dummies:

- ensure that they correctly represent the logic that is required; but also that
- they do not introduce unintended consequences (introduce precedence restrictions that are not required)

However in large projects which contain many activities, the more dummies we have to add the more cumbersome the project's model will become. The latter is a justification for discussing the whole theory by using an AON model.

After the 1960's, network modeling was developed. Some of the network models which have been developed are:

a) Construction of "generalized networks." According to Kauffmann and Desbazeille (1964) some activities can be performed under probabilities assumptions, or in terms of outcomes of precedence activities that finished before them.

b) Construction of "logical networks". These occur when there is a logical connection between a given activity and its predecessors (Battersby, 1967).

c) Modeling of overlapping activities in terms of time domain. In this kind of model the network is described as an AON network. Every activity is analyzed according to a given time unit. The sum of the resources is calculated according to the estimation that in a given time unit there are some activities that have to be performed within the same active time

window in such a way that every activity consumes its resource demand. Frequently, under those circumstances, a problem of an exaggeration of resource quantity can be expected (Leashman, Dinceler and Kim, 1990).

A work breakdown structure (WBS) has to be developed at this stage to watch over all activities which are operated within the same time frame. A WBS is a flowchart which describes the exact activities and sub-activities in a project. It creates the framework for a project's control, and contributes to the monitoring of the project's execution. An example will be given and explained in detail later.

d) The introduction of hammock activities. These are activities that can be operated during the time span between events, for example: the use of equipment between the start (or end) of an activity, and the start (or end) of another activity. The duration of a hammock activity is equal to the difference of times between the two activities that the hammock is hanging off. Those times are determined as two specific events. The construction of networks using these activities was studied by Harhalakis (1990).

e) Morphologic models of project network. The study of morphology networks was studied by Tavares (Tavares, Farreira and Cuelho, 1997; Tavares 1998).Tavares developed a theory that classifies project networks by two important concepts: progressive and regressive levels (Elmaghraby, 1977, Tavares et al., 1997). In the past managers experienced some difficulties in monitoring projects: they miscalculated the time executing estimation. Those miscalculations were a result of their lacking the right estimating tools, thus this produced the motivation to create the tool of a morphology network.

- The progressive level of the activity i is m(i) is defined:

$$m(i) = \begin{cases} max(m(j) + 1) \;\; if \; J_i \neq \emptyset \; and \; j \in J_i \\ else \;\; m(i) = 1 \end{cases}$$

Where $m(j)$ is the progressive level of j and J_i is the set of activities which are directly precedence of j. The maximal $m(i)$ is defined as M.

The precedence link connecting any pair of activities, j and i, is called a direct precedence link. If J_i is empty, then $m(j) = 1$ by definition.

- The regressive level of the activity i is $n(i)$. $n(i)$ is defined as :

$$n(i) = \begin{cases} min(n(k) - 1) \ \ if \ K_i \neq \emptyset \ and \ k \in K_i \\ else \ \ n(i) = M \end{cases}$$

Where $n(i)$ is the regressive level of i and K_i is the set of activities including i as a direct precedent activity. If K_i is empty, then $n(i) = M$, means that all activities were recently scheduled.

The level float $\Delta(i)$ is defined for every activity as $\Delta(i) = n(i) - m(i)$

The level length of any precedence link between i and j is given by:

$$L(i,j) = m(j) - m(i) \ .$$

The morphology of the network depends on several indicators which can be built in terms of the number of activities. Two of them are the number of activities per level and of the level length of the precedence links.

As an example, a network is shown below (figure no. 3), using AON and presenting $\big(n(i), m(i)\big)$ for each activity i and the level length of each precedence link.

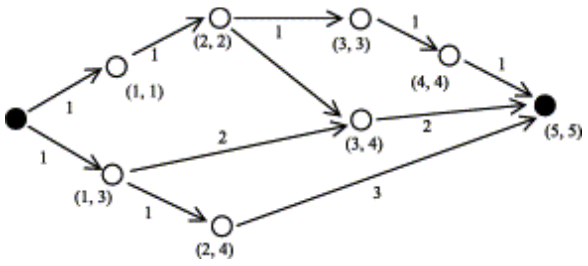

Figure 3: The progressive and regressive order of the activities of a network (Tavares, 1998)

In this example, the absolute length of this network is 5, which is equal to the maximal value of m(i). Each arc has its length, for instance: the length between (1,3) and (3,4) equals 3-1=2.

The morphological model looks the same as forward and backward scheduling, and by using it project managers can estimate the earliest (or latest) start (or finish) of any activity of the project. An activities' float Δ(i) can be used as a slack indicator since it is a difference between the latest start and the earliest start of any activity, however concerning m any interpretation can be made.

f) Constructing hierarchical networks. Each project can be viewed as a set of interconnected sub-projects ("macro-activities"). Each of these macro-activities can be modeled by another network constructed by using more detailed activities. This process of modeling can be studied using: multiple hierarchical levels as was presented by Speranza and Vercellis (1993), or partitioning methods which were proposed by Rafael (1990).

g) Aggregation of projects' networks to be transformed into simpler and more synthetic networks. Two major approaches were proposed: the decomposition and the reduction methods.

   The method of modular decomposition is based on the identification of "modules" which can be synthesized by equivalent macro-activities (Muller and Spinrad, 1989).

   A module (α) is defined as a subset of activities of the project network, which satisfies the following properties:

   - The set of precedent activities j of any activity $i \in \alpha$ with $j \notin \alpha$ is the same for any i.
   - The set of succedent activities k of any activity $i \in \alpha$ with $k \notin \alpha$ is the same for any i.

In figure 4, there is an example which uses successive levels of the two modules α and α':

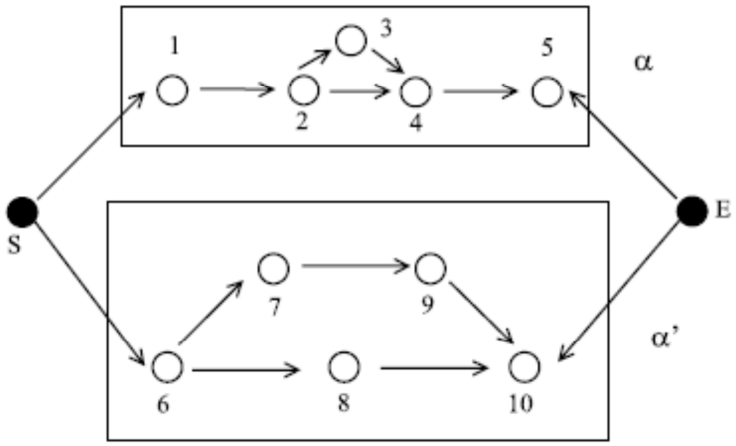

Figure 4: Modular decomposition (Tavares, 1998)

Following figure 4, activities 2, 3, 4 were collected as a sub-project, and in the next step those activities were added to activities no. 1 and no. 5. As a result, a module of α was created in the same way that the model α' had been created. This principle is called the *aggregation concept*. It unites some activities under the same roof so that all those activities have the same starting and ending points. The concept of aggregation contributes a lot to the use of hammocks. A hammock activity, which was discussed briefly in section d), is a collection of activities which have the same starting and ending times.

Over time there has been a lot of progress in the development of: activity modeling, description of activities, modeling of the activities' duration (Clark, 1962; Dean, Mertel, Roepke, 1969; Tavares, 1986), and use of resources (Blazewicz, Ecker, Scehmidt, Weglarz., 1985; Tavares, 1994).

Furthermore they were developed before the rise of the Multi-criteria Decision Theory, and so OR can now enrich this field/domain with new contributions. A multi-criteria model in terms of NPV, duration, and risk of delay, was developed and applied by Tavares (1984), to a real case-study. Another model (MACMODEL) is now available as a decision-aid to support the process of a multi-criteria evaluation of a project (Tavares, 1998). This model helps the decision-maker to estimate: the risks in the most accurate way, the predicted profits against the profits in reality, and to make the right decision under given circumstances.

Project evaluation

The main object is to enable the decision maker to get a realistic estimation when evaluating the real value of a project, it can be: profit, cost, or time, and can reflect on the risk level and the resources' efficiency. Indicators can be the NPV of the project and /or number of activities which were delayed. According to the above, project evaluation reflects the project's risk by using the principles of the scheduling theory. Therefore scheduling becomes one of the major issues in project management.

Project scheduling and monitoring

Following Demeulemeester and Herroelen (2002) a correct and practical scheduling of activities has to be done. Practical scheduling includes: the correct and most reasonable order between activities, supply of the right amount of resources, and exact time duration of each activity. All of those issues are the responsibilities of a project manager. Project scheduling can be formally defined as:

An ordered list of activities whereby every activity has its required amount of resources is followed by the appropriate successor activity, and has feasible start and completion dates.

Operatively, scheduling deals with developing time schedules and determining the time frame for operated activities. Some of the resources are: manpower, staff, equipment etc. The necessary processes of scheduling are:

- Exact definition of activities
- Exact definition of the relationship between activities
- Exact estimation of activities' duration
- Time control
- Development of schedules

A schedule combines information about operating times and the order of operations with regard to time and resources constraints at each time period. The termination of a project within its deadline has an important significance for the project's success therefore it is one of the main objectives.

In order to plan the project's scheduling properly, a manager should define precisely which activities that should be applied.

Before the complex job of planning and executing a project can be started, it is necessary to plan the strategy, objectives and scope of the project as clearly as possible. This plan has to be done in accordance with the proposed project's result, i.e. the set of products and/or services to be delivered. The further subdivision of the project result into its various components creates a WBS. This defines the project activities in relation to the project result; it creates a framework for project control; and it provides the basis for obtaining relevant insight into the time and cost status of a project through the various management echelons.

The main and basic requirements that a manager should demand from a WBS are:

- Hierarchic and systematic structure of all components of the project including: processes, activities, and sub-activities. A project contains a lot of activities that have to be performed in order to reach a well-defined goal. Some of those activities are quite complicated therefore the project management has to be aware of all components of all activities, including all sub-activities, in order to ensure a fruitful and exact execution.

- Availability of all objectives, services and tools which are needed to: create, develop, build, control, and supply the final product/service. All activities need some resources in order to be carried out, for example: manpower, money, appropriate machinery and equipment. However a manager needs the correct tools to: monitor the entire project and also each process during its execution. Initially a careful plan of the project's main targets is an essential tool. When all targets are clear to the project's team it is much easier to apply the resources and stages of the entire project.

- Use of a verbal or graphical model to detail the relationship between all of the products' components and the relationship between the components and the final product. Thus a manager should get a clear picture of the integration between all work ingredients, for example: Figure 5 below adequately describes all the ingredients and authorities of a software project. The model has to be simple enough for a manager can understand the relationship between all departments that are involved in the project.

- Suitability for: the structure of the work, work process, budget, final product and its overall structure. The WBS must connect the departments in keeping with the organizational structure of the organization that produces the project and work processes. An example of this is a factory that used to work according to a vertical structure, but which had to be changed to a matrix structure. This change must be applied right at the beginning and should be acknowledged on the WBS.

- Division of the activities into sub-activities which are considered as work units in order that:
    - every work unit can be managed easily
    - the dependency between project's ingredients can be reduced
    - a quite simple logical order between ingredients enables easy integration
    - there is easy control of all packages
- Clearance to all of the project's staff.
- Every work package must contain: targets, resources, timing, budget (or other financial predictions), operative indicators and responsibilities.

Hereunder is a flowchart of the WBS of a software product creation project:

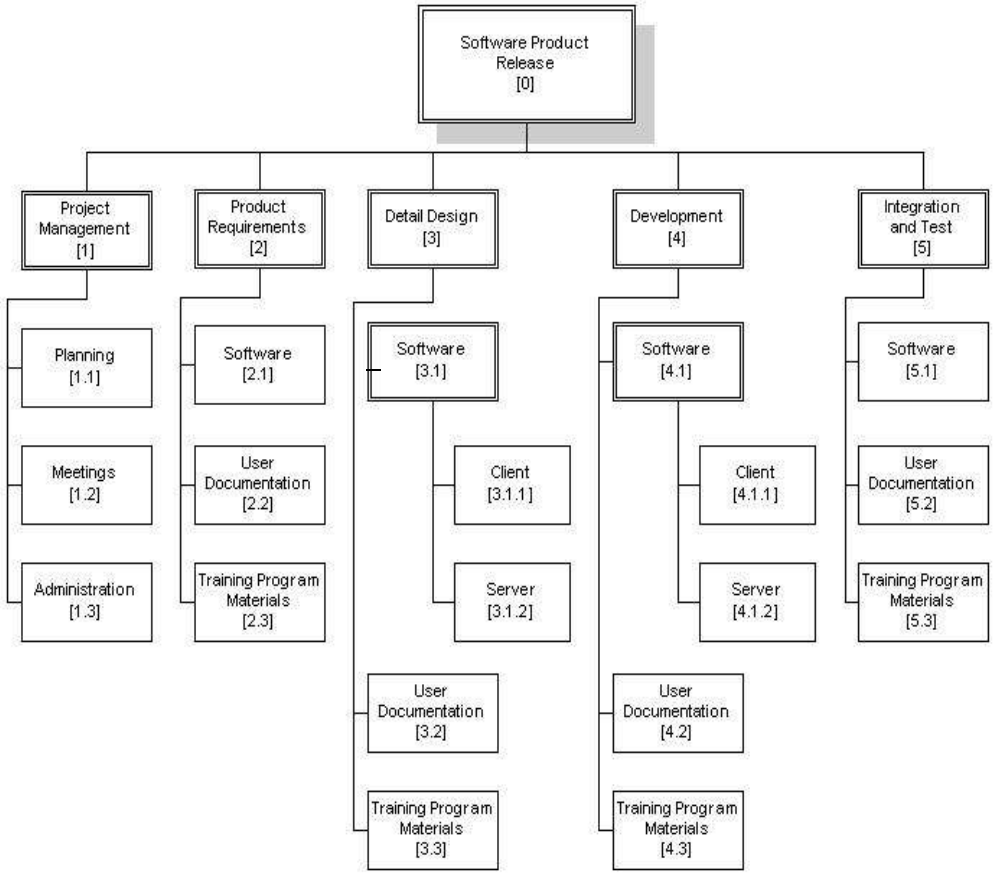

Figure 5: A work breakdown structure (WBS) of a software project

(http://www2.cit.cornell.edu/computer/robohelp/cpmm/Phase3_Process_Descriptions.htm)

Each item in the WBS above (figure 5), has a catalog number in each box, which is used as an identifier of the work component. Every component that belongs to a specific section has three numbers separated by dots. The first one is the department's number, the second one is a sub-department and the third one is an entity: i.e., detail design is department no. 3, the software ingredient of it is 3.1, which is divided into client (3.1.1) and server (3.1.2). This division process continues as far as needed.

One of the main factors in deciding how deeply the tree above should be divided is the organization's structure. OBS shows the various organizational units that are going to work for the project, and allows these responsibilities to be linked to the WBS. The lowest stages of OBS and WBS are the work packages. Work packages are divided into sets of activities and subtasks.

The development of the WBS begins at the highest level of the program with a definition of the project's end items (hardware, services, equipment and facilities). The major end items are then divided into their component parts (sub-systems, components); the components are divided again into more detailed units. After any division, we get a smaller unit, with a smaller authority, value and most important: complexity. When the process of division has ended, the responsibility for one organizational unit is assigned to the smallest work unit.

The most important conclusion here is that after a WBS has been established all activities are well-defined, and only at that moment planning the scheduling process can begin.

A project can be defined as a successful one if: it is completed on time, is set within a reasonable time frame and budget, and achieves satisfactory results. In conclusion: the manager's responsibility is to allocate all resources according to the need, while at the same time considering all constraints so as to finish the project on time, and achieve the actual results within financial limits, therefore the most important issue in project management is resource management.

The activities' duration has an important effect on the project's completion time. There are two types of resources which effect the duration of a project:

1) ***Precedence resources*** – there are some technological and reasonable limitations which dictate that an activity should not be executed before its predecessor is complete.

2) ***Resource constraints*** – in the most part, resource availability at any given time is limited. It means that when we are able to carry out some activities in parallel, we have to pay attention to the quantity of resources; we have to be precise, and check that the correct amount of resources are available during every time unit of the activity's execution, and without any violation of the predetermined resource allocation.

Both claims above are motivated to define several of a project's scheduling objectives:

- ***Timed-based objectives*** – one of the most common objectives is the minimization of makespan. The objective is practical, in many situations it: leads to a timely release of resource capacities for future projects, reduces the

risk of violating the deadline, and generates timely incoming cash flows etc. Another option concerning the time parameter is the ***lateness*** of the task i.e. it is possible to calculate the difference between the completion time of a task and the due date of every activity in the project.

1. ***Resource-based objectives*** – these objectives are established to control the resources' availability during the project's execution. Therefore a manager can ensure that all activities will not be delayed due to a lack of an adequate supply. One example of an objective is: ***resource availability cost problem*** (Demeulemeester, 1995) where the capacities of the renewable resources are determined so that a given deadline T is met, and the resource availability costs are minimized. Another example is: the ***resource leveling problem*** (Konstantinidis, 2002) which involves the generation of a time-feasible schedule in order to get resource profiles as balanced as possible, but without violating the project's deadline. The degree to which the resource use is leveled can be expressed in various ways. Burgess and Killebrew (1962), Demeulemeester and Herroelen, (2002) suggested the following procedure:

    1. Calculating the average of the resource requirements.

    2. Calculating the squared deviation of each resource requirement out of the average.

    3. Sum all squared deviation out of step no. 2.

    4. Minimize the sum of step no.3

Financial objectives – an important objective is related to the incoming and outgoing cash flows which are generated during the execution of a project. From the management's point of view, cash flows are influenced by the execution of project activities and the use of resources. Cash inflows then typically result from payments made upon the completion of certain parts of the project. An example of this is: maximizing the NPV of the project (Csébfalvi et al., 2008a, 2008b).

- ***Quality oriented objectives*** – maximizing quality is gaining importance in project scheduling. An example of quality is: the minimization of repeated

work and/or ensuring the continuity of the project's progress (Icmeli-Tukel and Rom, 1997).

- ***Regular and non-regular objectives*** – a regular measure of performance is a non-decreasing function of the activity's completion time, i.e. when the activity completion time increases, the objective function value does not decrease. Regular measures of performance are also referred to as: ***early completion measures.*** For instance, the minimization of makespan is a regular measure of performance. As an example of a ***non-regular measure performance***, we can cite the case when delaying activities should improve the project's quality, even if such a delay is not feasible due to resource constraints.
- ***Multiple objectives*** – entails using different kinds of objectives at the same time. The first one belongs to the first priority criteria (this objective should be achieved before any other), while others should be achieved later, dependent upon the first major one.

Scheduling has significant meaning and effect, and is one of the main tools that enables a project to be successful as opposed to unsuccessful. In the beginning, most scientists concentrated on solving precedent constraints, and did not pay enough attention to the availability of resources. Nowadays, tools like "critical path" helped researchers develop new algorithms to solve the non-resourced unconstrained project. However, when resource constraints became more critical, new algorithms, attitudes and heuristics were developed to deal with them. Resource constraints in projects' schedules motivated researchers to redefine some new measures and techniques to evaluate projects. Many new scheduling algorithms have been developed, so scheduling has become an increasingly crucial issue in project management.

2.2 Float and resource constrained float

A detailed explanation of some important concepts is necessary before discussing the main issue of scheduling, and before presenting the new measures and algorithms. Important definitions worthy of note here are:

Earliest starting time (EST) – the earliest possible starting time of an activity

Latest starting time (LST) – the latest possible starting time of an activity

Earliest finishing time (EFT) – the earliest possible finishing time of an activity

Latest finishing time (LFT) – the latest possible finishing time of an activity

Actual Start (S) – the actual starting point of an activity. This starting point is between *EST* and *LST*.

Duration (D) – the time needed in order to compete in an activity.

Precedence relation – the order in which activities are performed during a project. This order emanates from technological constraints.

Finish to start – An activity can only start when all its preceding activities have finished.

Data Date (DD) – refers specifically to a date in the life of an ongoing project. It is an arbitrary date which is considered important by the project's management. During the execution of the project the manager has to check out some intermediate stages of the project to see if the desired target has been reached, this can achieved by using data date.

Demeulemeester and Herroelen (2002) have defined three typical types of float associated with each activity within a project network:

- ***Total float (total slack)*** – this is the maximum time period that an activity can be delayed without violating the whole project's deadline. Total float of a known activity j is represented as TS_j (Total Slack) and defined as:

  $$TSj = LST_j - EST_j = LFT_j - EFT_j \quad (2.1)$$

- ***Free float (free slack)*** – the spare time of a given activity, which means the time that a given activity can be delayed without violating the starting time of its immediate successors.

  Under the assumption that i is a direct successor of j, the mathematical definition of it is:

  $$FSji = min\{EST_i\} - EFT_j \quad (2.2)$$

- ***Safety float*** – the number of time periods by which the duration of an activity may be stretched whereby all its predecessors start as late as possible but do not delay the possible completion time at all.

  Under the above assumption (i is a direct successor of j), the mathematical definition of it is:

$$SS_{ji} = LST_j - max\{LFT_i\} \quad (2.3)$$

All three of these measures help to improve the flexibility of the project scheduling. A manager can get an estimate of the time delay of every activity and its effect on the project as a whole, and/or its effect on some specific activities within it. Therefore, the changing of a starting time of a single activity (or activities), is likely to change the project's makespan. Definitely, resource constraints have a major effect on the timing of executing activities within the project. So when resource constraints become more and more important in project scheduling, all of these three measures have an important effect on the project's makespan. However, in order to treat scheduling under resource constraints we need a new definition: a resource constraints float.

A resource constrained float is defined as: 'a period of time that a given activity can be delayed, and/or its duration can be extended, without negative effects to the whole project's makespan, (if the resource availability stays as it is)'. A resource constrained float is a kind of measure of robustness; it describes a time-axis shifting ability of an activity, considers its resource constraints, or extends its duration without delaying the project's makespan. This kind of shifting of the activity's time range, or extending its duration, is necessary when a kind of essential resource (a resource that is necessary for an activity's operation) is unavailable, or for any other reason which forces a manager to delay.

The latter definition applies only when the resource availability stays constant during the project's entire duration. The process of determining the resource constrained float requires the conversion of resource dependability to form a permanent linkage between activities. By that we get resource dependability. This assumption is usually warranted and realistic in projects with a definitive design and a specific plan for resource allocation. These conditions usually exist when there are specialized resources, such as: a team of experts, or special equipment that cannot be easily transferred between

activities at short notice, or whenever several projects are managed at the same time and compete for the same resources.

However, in projects with less specialized resources that can be transferred easily between activities, the concept of resource constrained float is less useful. In conclusion, known measures of resource constrained float deal successfully with certain types of projects, but since their flexibility is not tested their applicability is restricted, regardless of the type of resources being used.

The resource constrained float principal deals very well with certain types of projects. However its use is very limited in as far as it cannot cope satisfactorily with flexible time and /or resource constraints changes. The fairly large number of resources and variables and the criticality of the considered number of activities, are the main obstacles for the robustness of scheduling ─ a problem could arise from shifting some activities and extending the duration of the project.

2.3 Critical path and critical chain

Critical path is the longest chain of activities that have to be performed following their precedence relations in order to finish a project on time. A critical activity is typed by zero total float. Any delay of a critical activity along this path would delay the whole makespan. In a project it is possible to find more than one critical path. Therefore, the critical path is one tool that will ensure that a project will be executed on time.

There are two approaches for determining the critical path:

- Critical Path Analysis (CPA), which deals with the activity's supplied time (A Monte Carlo simulation, (Willis, 1985)).
- Program Evaluation and Review Technique (PERT) or Critical Path Method (CPM), which estimates a triple duration per activity (Ragsdale, 1989; Schonberger, 1981).

These methods deal only with the parameter of time.

After defining the correct order of activities, the time duration is allocated for every activity regardless of the resource constraints, later the longest path is calculated (the critical path). In conclusion, this tool is insufficient since a project deals with motives of time and resources, so a new tool had to be developed.

Wiest (1964) treated activities' scheduling as a matter of time and resource allocation. He tried to find activities which share the same time period, and need the same resources for that known operation period. This was the first time that the critical path dealt with time and resource constraints. Wiest then proposed a new term: ***critical sequence***. Now managers have to focus not only on a time parameter, but also on resource allocation which is another important parameter that has to be taken into consideration.

When ***critical sequence*** was defined, a competition was created between the activities for resources. All of the activities that had to be scheduled over a known period had to compete with each other for the resources which they needed in order to operate properly.

This new philosophy forced managers to monitor activities and pay more attention to the scheduling process while at the same time to redefine definitions like critical path and float. The definition of critical sequence represents a new (and complex) problem without a solution; therefore researchers had to develop a new float definition, new algorithms and new enhanced research directions in scheduling.

2.4 Solving resource allocation problem

Wiest's work motivated others to tackle the issue of criticality in resource constraints projects. Bowers (1995, 2000), Raz and Marshall (1996) produced useful algorithms which did very well in scheduling under resource constraints, however those solutions were quite limited since they challenged problems with constant resources.

Two categories of resource scheduling are known:

- ***Resource leveling problem*** (RLP) – the project's makespan is known and cannot be exceeded. No resource limitation exists at all. The task is to allocate all resources as often and as uniformly as possible (meaning: scheduling with a minimized variation between two nearby time intervals), so that the makespan should not be exceeded.
- ***Resource allocation problem*** (RAP) – no resource violation is allowed i.e. resource limitation is dictated for any given time period. All activities have to be scheduled according to their time-window, as close as possible to the

critical path, and in such a way that the project's makespan should be kept to the minimum.

Konstantinidis (2002) developed a new model of resource scheduling – *resource leveling oriented*, by using the MILP technique. This research is a kind of RAP problem, so only related issues will be discussed below.

The main target of scheduling is to minimize the project's makespan. In light of this goal, a traditional approach ─ the RCPSP is used to solve scheduling under resource constraints. It is a time-oriented approach – but a NP-hard problem.

There are two major procedures used for dealing with RAP:

Exact procedures **–** these procedures provide an exact and optimal solution for the RCPSP, however in reality most projects have many resources and variables, so in terms of mathematics, achieving an exact solution requires a lot of computational effort. Therefore it is used mostly in small-medium scale projects. Small-scale problems can be solved by using a linear programming procedure where an exact and optimal solution is achieved; however with regard to medium-scale problems, a search-tree should be developed – like the Branch and Bound algorithm.

Usually small-medium projects contain between 30-50 activities, however there are small-medium projects which contain more than 50 activities, but because those projects contain quite a low number of constraints, there is an exact (polynomial) algorithm that solves it. Large projects usually contain more than 50 activities so heuristic algorithms (defined below) are needed.

Heuristic procedures **–** heuristics is defined as: a procedure that dictates some stages in order to get a near-optimal solution at the end. Mostly, the optimal solution is not known, however an approximation is reached. Since a large-scale problem contains many resource constraints and variables, the optimal solution is hard to reach, so heuristics are usually used in order to solve it. Most heuristics have been tested on large-scale randomly generated problems.

The first exact MILP model was developed by Pritsker, Watters and Wolfe (1969). It was the first time this kind of problem was addressed. They assumed that a project

contains n-1 activities, when activity no. n is a finish dummy activity. Out of necessity there are some definitions that are written below prior to writing the model itself:

(1) A binary variable $X_{it} = 1$ if activity i finishes at time instant t, otherwise, $X_{it} = 0$. X_{it} can only be defined over a time interval between the earliest and latest finishing times of the activity.

(2) An activity is defined as a letter i=1, 2,……n, when n is a dummy finishing activity.

(3) The set A below presents in the following model a precedence relationship between activities, namely: $(i,j) \in A$ emphasizes that activity j cannot be started before activity i ends.

It is important to note that the definition of set A is different from set A defined on page 14.

The model is described below:

$$\min \sum_{t=EFT_n}^{LFT_n} tX_{nt} \qquad (2.1)$$

Subject to:

$$\sum_{t=EFT_i}^{LFT_i} X_{it} = 1 \qquad \text{for i} = 1,2,\dots\dots \text{N} \qquad (2.2)$$

$$\sum_{t=EFT_i}^{LFT_i} tX_{it} \leq \sum_{t=EFT_j}^{LFT_j} tX_{jt} - d_j \qquad (2.3)$$

$$\text{for all } (i,j) \in A$$

$$\sum_{i=1}^{n} \sum_{q=\max\{t,EFT_i\}}^{\min\{t+d_j-1.LFT_i\}} r_{ik}X_{iq} \leq a_k \qquad (2.4)$$

$$X_{it} \in \{0,1\} \qquad (2.5)$$

Objectives:

- (2.1) minimizes the completion time of the dummy end activity and thus the completion time of the project.
- (2.2) clearly defines the fact that every activity should be completed only at one time point, which lies only in the interval between the earliest and latest finishing times.
- (2.3) describes the precedence constraints – i.e. the completion time of every activity cannot exceed the starting time of its successor.
- (2.4) the resource constraints for every resource type k are specified by: considering for every time instant t and every resource type k all possible completion times for all activities, I is such that the activity is in progress during period t. The weighted sum of the corresponding decision variables should not exceed the resource availability.
- (2.5) is a binary variable which dictates if an activity i is finished during time t or not.

Klein (2000) has rephrased Pritsker's definition by defining the binary variable X_{it}=1 if activity i is in progress in time t. Other Branch and Bound algorithms were developed by Patterson, Slowinski, Talbot and Weglarz (1989), Stinton, Davis and Khumawala (1978), and Csébfalvi G. and Csébfalvi A.(2005).

All of the exact algorithms achieved an optimal solution for small-medium scale RCPSP. Heuristics were used to get a near-optimal solution for large-scale problems. However, all of the above discuss scheduling standard activities under resource constraints, but there is no answer to a very special and dominant kind of activity: hammock activity (it will be clearly defined in the next section). The main motivation is to treat more than a single objective. This research solves an LP problem with a variety of objectives. The main objective is to first minimize a project's makespan,

and then any other criteria. By so doing, a model achieves more than one optimal value. This issue motivates the author to think about a hammock cost as a secondary objective that should be achieved as a result of an LP model.

2.5 Hammock activity

Hammock activity is an activity that we schedule between “regular” activities, since its duration cannot be estimated or calculated at the initial time of project planning. A hammock activity contains a collection of "regular" activities which have the same starting and ending points and/or need special equipment and/or special resource(s). Since hammock activity has the same starting and ending points similar to a standard project, hammock activity could be considered as a project itself, except that there is no significance in the order between its inner activities.

It should be noted that part of the hammock's members can be operated in parallel or serially, so the mutual starting and ending points are mutual as a whole. Importantly, all of the hammock's members need special equipment which is not necessarily a resource constraint of the whole project.

Hammock activities can play a useful role in project management. Typically, they have been used to denote usage of equipment needed for a particular chain of activities (e.g. a load lifting device), without predetermining the estimated time the equipment must be present on site. Similarly, it may be required to pick up the cost of a complete section of the project, or more usually some background cost related to a section. Such background costs could arise from storage, supervision etc. and can be allocated to hammock activity. Also for upper management reporting hammocks are used to collectively represent a sequence of consecutive normal activities, all of which form the task of one department, or relate to the same cost center.

The use of hammock activity has become more and more popular; computer programs are developed to handle project management dilemmas. Software helps treat hammocks as a part of a whole project. However there is some confusion about estimating the hammock's duration. Consequently every hammock activity is connected on both sides to regular activities; it is connected in such a way that all its activities have the same starting and ending points.

Hammock and the scheduling process – hammock activity constitutes a group of activities which are not concerned with regular precedence and resource constraint; they therefore do not apparently affect the scheduling process and the project makespan. However, its members can share the same equipment (or another resource) and same time allocation. The WBS, on the other hand, which had been dealt before, constitutes a well-connected package of activities which have a well-defined and logical order between them. This is one important difference between hammock activity and WBS. Another important difference is that WBS is planned before conducting the scheduling process itself, however hammocks are planned afterwards.

2.5.1 Characteristics of a hammock activity

In many projects there are many cases of activities' collections which need some special resources and/or special funds. This can cause many difficulties in forecasting their exact beginning and/or finishing times; or where there is an order problem a manager has difficulty in deciding which activity should precede the other one, however all these activities have a common starting and ending point. Another case that motivates managers to use hammock activities is a dependency between activities which are not related by precedence relations at all, however all of them do share a mutual resource (like a single special machine), which is needed at certain stages of the project to carry out the necessary activities. This dependency creates the linkage between non-related activities in such a way that a mutual resource becomes essential. In such a case, a definition of hammock activities in which its members comprise all of these activities is needed.

Firstly, in order to understand the main issue of hammock activities and the differences between regular activities and hammock activities, we have to characterize them.

A hammock activity has the following characteristics:

1. A hammock should have at least one predecessor and one successor. A hammock without a predecessor starts on the data date (which is an arbitrary time point defined by the management of the project), and a hammock without a successor ends on the project completion date.

2. A hammock activity bar starts on the actual date of a hammock activity in progress, and the computer program calculates: the original and remaining durations, and percent completion, based on the following formulas:

   a. Remaining Duration (RD) = (Early Finish Date – Data Date), or in other words, $RD_H = EF_H - DD_H$

   b. Original Duration (OD) = (Data Date – Actual Start) + Remaining Duration, or $OD_H = DD_H - S_H + RD_H$

   c. Percent Complete (PC) = (Original Duration – Remaining Duration) / Original Duration, meaning $PC_H = (OD_H - RD_H)/OD_H$

   Hereunder is an example of hammock activity:

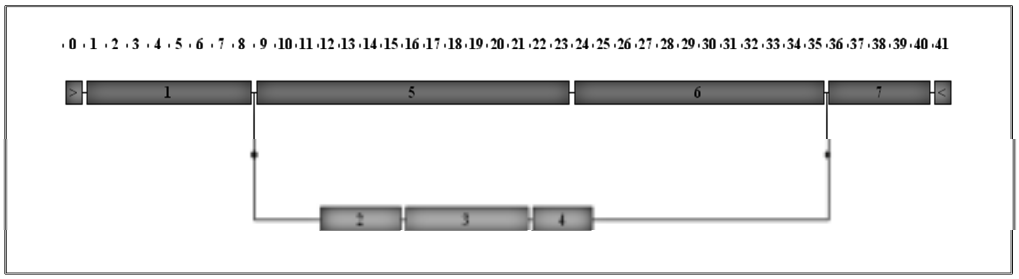

Figure 6: An example of hammock activity

The above example illustrates a hammock which contains three activities: 2, 3 and 4, which are considered as hammock members. Those hammock members are carried out parallel to activities no. 5 and 6, since they have the same time range: [9, 36]. The latter means that this hammock activity begins at t=9 and ends at t=36, so its duration equals 27 time units. Following the figure above, activity no. 1 is a hammock's predecessor, and activity no. 7 is its successor.

3. A hammock activity cannot drive out (push out) any task activities. It behaves like a “rubber band” as it is stretched or compressed according to its successor and predecessor ─ not the other way around.

4. All hammock members should have the same time-range – meaning that all have some overlap execution time.

5. All hammock members should need the same sized resource(s) and constraint(s).

6. A hammock member should not be a critical activity (the leg of it must not equal to zero).This helps a manager unite the activities under the same roof: a hammock.

.

2.5.2. A real-life example

The following example emphasizes a practical and useful use of hammocks. Since there are only a few papers which deal with hammock activities (or summary activities), there are hardly any practical examples in the literature. The following example illustrates a practical and useful use of hammock activities in a project; it describes a project of building an underground tunnel; the activities of the project were divided into groups. All of the groups are hammock activities. The main aim of citing this example is to illustrate a wise use of hammocks; as a result of grouping activities as hammock members, a lot of money was saved. The question that was raised was how to group the activities wisely. The main idea in this case was to recognize a mutual resource(s), which would be essential for many of the authorities at every stage of the project, and to unite regular activities into hammocks productively (that is grouping together all activities which needed this special resource(s), however which had no precedence relation between them). As a result, a lot of time was saved. The main time saving in this example, was the idle-time ─ the waiting time for the freezing machine (the essential and mutual resource).

The following example will emphasize the importance of hammock activities in solving scheduling problems wisely. In this example four hammock activities were used. This project contained 26 sub-projects, which apparently were unconnected. However these sub-projects had the same needs, i.e. the same resource constraints of freezing machine and manpower. This example shows how to ensure work continuity while the scheduling process is being carried out. While dealing with this dilemma, a trade-off between the freezing machine's time and manpower work's time was obvious. While the necessity of a freezing machine at every one of the 26 parts was certain, there was a manpower cost that had to be taken into consideration. The cost of one freezing machine was higher than the cost of the manpower, however the idle time of manpower, while waiting for a free freezing machine, was not free. Hammock activities were described here as activities which needed the same resources: freezing machine and

manpower. When this principle was clear to a project manager, he/she could trade-off between manpower's idle-time and/or the cost of buying more and more freezing machines. Both were considered to be hammocks, and in this example both were used to reduce the idle-times between activities and to ensure work continuity. However if for example there would be one freezing machine for every tunnel, the whole project would be finished very quickly with hardly any idle time, but the project would be expensive. On the other hand, a manpower trade-off could be achieved if all activities were scheduled in such a way that whenever a freezing machine would be free, there would be an available crew to work on it. By applying this approach, minimal idle time of manpower and equipment would be achieved. Both resources – machinery and manpower were examples of two important resources that if managed wisely, would ensure continuity of the project.

Vanhoucke and Van Osselaer (2004) focused on a huge project with a groundbreaking boring technique in the Netherlands: the Westerscheldetunnel. This tunnel provides a fixed link between Zeeuwsch-Vlaanderen and Zuid-Beveland in the Netherlands. The tunnel is very deep underground, so the planning and building process was very complicated.

Description of the tunnel's project

Every 250 meters the tunnel tubes are connected by transverse links (sometimes referred to as cross passages), as displayed in figure 7. Normally the doors to the transverse links are locked, however in the case of an emergency they are unlocked automatically, and one can walk to the other tunnel tube, and emergency services can use this road to reach the site in case of an accident.

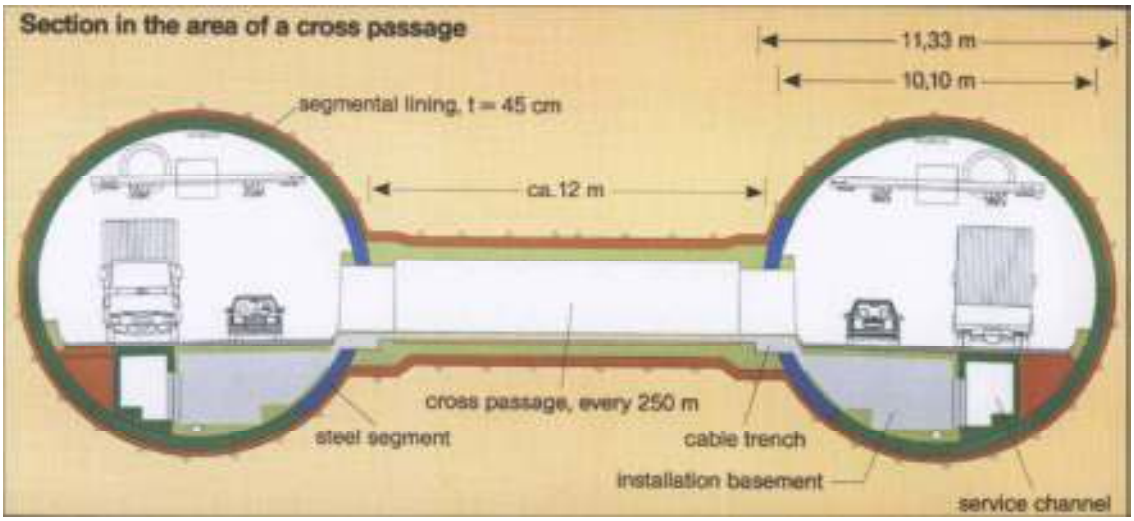

Figure 7: Drawing of traverse link (Vanhouke and Van Osselaer, 2004)

The transverse links account for ten percent of the construction budget. A freezing technique was used for the construction. This guarantees watertight transverse links and does not harm the environment.

The authors decided to treat the building of the traverse links as a sub-project (11 activities). These 11 activities were drawn on figure 8 as follows:

First, 26 pipes fitted with drill heads were bored from the eastern tube to the western tube (activity no.1 on fig. 8), and a refrigeration unit was built in the eastern tube (activity no. 2). After that a brine solution that had been cooled to -35°C by the refrigeration unit was pumped into 22 of the steel pipes (activity nos. 3 or 4, depending on whether it was sand or clay that had to be frozen). Two of the pipes were used to monitor the progress of the freezing activity and third pipe was used for drainage. When the ice around the future transverse link was sufficiently thick, the installation of the link could begin.

In the western tube the future entrance to the link was opened and the frozen ground was excavated step by step with a cutting machine (activity no. 5). To prevent the ice capsule from deforming under the great pressure, a layer of gunite concrete was immediately sprayed under high pressure against the exposed ground. Thus a concrete outer wall was created. Subsequently watertight foil, which prevents the mixing of thawing water and concrete, was applied (activity no. 6). After that, shuttering was laid for the spraying of concrete for: the inner casing that would form the floor (activity no. 7), and the wall and roof (activity no. 8) of the transverse link. Then the shuttering was

removed (activity no. 9) but the surrounding ground was still being frozen to discharge the concrete (activity no. 10). Only when the concrete had hardened sufficiently the freezing process could be discontinued. Finally, the refrigeration unit was drawn off (activity no. 11). The links were completed at a later stage.

In figure 8 we display the unit activity-on-node network of the sub-project 'build transverse links' with 11 activities. Although this unit network is a simplified and adapted version of the real network, it is useful to illustrate our overall results. All technological precedence relations are of the finish-start type with a time lag of zero (FS = 0), except where indicated.

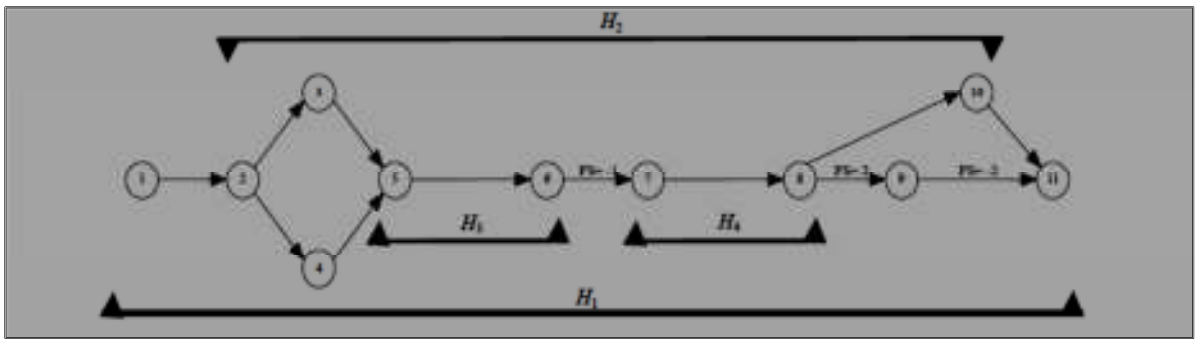

Figure 8: The unit project network of the sub-project "build traverse link"(Vanhouke and Van Osselaer ,2004)

Activities denoted by H_i are the hammock activities. Hammock 1 denotes the total sub-project (transverse links) per unit, and has a duration which equals the total makespan of the sub-project. Hammock 2 refers to the time span that is necessary for the freezing machine during the execution of the sub-project. Figure 8 reveals that this freezing machine is needed between the start of activity 2 and the end of activity 10. The freezing activities are split into two sub-activities: hammock 3 refers to all freezing activities on the gunite concrete, while hammock 4 refers to freezing activities during the construction of concrete.

Although the freezing machine is an important resource that is necessary during the construction of a transverse link at each unit, the crews that pass along the units are also considered as an important resource type. In the network, we incorporate 5 different crews (referred to, for the sake of simplicity, as crew 1, crew 2, etc.) i.e. activity 1

needs crew 1, activity 2 needs crew 2, activity 5 needs crew 3, activity 7 needs crew 4 and activity 8 needs crew 5. The crews mainly consist of ordinary employees and specialists.

The network displayed in figure 8 is a unit network, which means that it is a graphical representation of the sub-project for only one unit. A transverse link has to be built every 250 meters, resulting in 26 cross passages in the tunnel. The unit network will therefore, be repeated 26 times. Each unit makes use of the freezing machine, while the different crews pass along the units.

The incorporation of work continuity constraints in scheduling has revealed two important insights. Firstly, the minimization of resource idle time results in a dramatic decrease in the cost of the resource used. Indeed resource idle time results from the fact that the resource has to be paid for when it is not really necessary. In this example, crews had to wait to pass along units and the freezing machine that could have been in operation but when no real work was being done at the time, were referred to as: unnecessary resources. Secondly, the minimization of the resource's idle time involves a trade-off between the cost of idle time and a delayed project deadline. Consequently as the project's deadline is delayed, there is more freedom to schedule the activities, and as a result to lower the resource's idle time. Resources that move along the units can be subjected to idle time between these units. Work continuity constraints can also be of importance to minimize within-unit idle time, as was the case with the freezing machine.

We will now endeavor to show how the new approach deals with this problem more successfully. Initially it is important to define: ***hammock cost***. Hammock cost is the total duration of a known hammock's activity; it is sometimes multiplied by a cost per time-unit. In the example above, the crew's total idle-time was 165 days, and its cost was 1200 Euros per day, so the total hammock cost was 165*1200=198000 Euros; the idle-time of the freezing machine was 343 days multiplied by 3000 Euros (freezing machine's idle-time cost per day) which equaled 1029000 Euros. In the example above, Vanhoucke and Van Osselaer (2004) represented an algorithm which saves the idle-time of equipment and manpower by finding the right timing between manpower and equipment availability. We compared their approach to the example cited above.

The new approach reduced the cost of the crews' idle-time to 107*1200=128400 Euros; and the cost of the freezing machine's idle-time to 5*3000=15000 Euros. In conclusion: 1083600 Euros were saved.

However, a conflict arose. As the project's makespan became longer, the machine's idle-time became shorter, while the opposite was true for the manpower. This conflict was created as result of the limited amount of resources (the freezing machine in this case). This solution significantly reduced the total hammock cost of this project (it reduced the idle-time between activities), but it was planned to be an algorithm to solve sub-projects – meaning this algorithm was developed for a procedure that had to be repeated 26 times (the number of tunnels). The new algorithm however will reduce the total hammock cost of a large-scale project to a minimum – almost exactly like that of Vanhoucke and Van Osselaer however it will apply to large-scale projects which are not divided into sub-projects and which do not have repetition routines.

2.6 Summary

Project management utilizes many skills to get the desired and expected results. The discipline of project management became more and more popular during the 1950's as a result of the development of OR. Nowadays projects have become more complicated with large-scale and combined resource and time constraints. The area of OR has contributed heuristics and useful algorithms so as to address the problem. As a result, one of the main concerns is the definition of a successful project. A project can be defined as successful if it ends on time without an exaggeration of either time or budget, and achieves satisfactory results. The latter is the main object of scheduling. In order to ensure that the scheduling process is carried out correctly, that is: all activities should be covered by the parameters of executing times, resource constraints, money and budget requirements, responsibilities etc, a sharp and exact planning of WBS had to be done. When the latter process is finished, the activities and the authorities units in the project and who is responsible for executing them become clear to a project manager, only then the scheduling process can be started. The most challenging problems in project management are scheduling problems under resource constraints or RCPSP. This element of project management has been studied over the course many years. In the history of research, many categories of objectives were developed,

according to the specific needs of the project. Many algorithms have coped with RCPSP and its different versions quite successfully. However early on there were many algorithms based on the critical path principle; in the meantime the issue of resource allocation was quite neglected. Willis (1985) and Ragsdale (1989) were the pioneers of dealing with time under resource constraints, and using the critical sequence approach. Later on different definitions of 'float' were developed to calculate the float in resource constrained projects. (Bowers, 1995, 2000; Raz and Marshall, 1996).

As projects became more and more complex, hammock activities became more and more common. The rising number of activities that had to be applied contemporaneously and which needed the same equipment (or other resources), and the order of precedence between them, which was not known at the initial stages of the project, enforced managers to treat the collection of them differently than they would have done previously with regular activities. This different attitude motivated researchers of scheduling to develop a new discipline to treat them, and so a new definition was defined: hammock activity. Managers realized that hammocks alone can be treated as a project therefore often requiring separate management. However when those hammock activities are quite widespread in that they contain a large number of activities, they become much more difficult to manage, and as a result it is quite probable that this can affect the whole project's makespan. As a result a need for an algorithm to treat hammock activities more carefully was recognized.

A hammock activity is the milestone of a project. It can play a useful role in project management. Typically hammock activities have been used to denote the usage of equipment needed for a particular chain of activities (e.g. a load-lifting device), without predetermining the estimated time the equipment must be present on site. Similarly, they may be required to pick up the cost of a complete section of the project, or more usually of some background cost related to a section**.** As hammocks have a particular definition a useful tool was developed to deal with them. As discussed earlier, a good example that illustrates the implementation of hammocks was the Westerscheldetunnel project in the Netherlands: Vanhoucke and Van Osselaer (2004), used hammocks to treat this huge project with a literally ground breaking boring technique. In this project, they defined four different hammock activities.

The use of hammock activities in the example above helped the scheduling process. A project manager scheduled activities in such a way that a crew of workers would not have to wait for equipment to become available. One hammock activity referred to the time span during which the freezing machine was needed i.e. between the start of activity 2 and the end of activity 10. The second hammock dealt with freezing activities. Activities were timed in such a way that when the use of a freezing machine in one tunnel was finished, that would be the right moment for an available crew to begin work on another tunnel. In conclusion, the idle-time between activities was minimal. Hammocks here helped define the problem's resource constraints: manpower and equipment, and reduce the idle-time between activities. As mentioned above, a conflict arose while running the algorithm between manpower and equipment. The new approach algorithm will reduce the total hammock cost of a large-scale project to a minimum – very similar that of Vanhoucke and Van Osselaer, however contrary to their algorithm, it will apply to large-scale projects without repetition routines.

By analyzing this example, it can be concluded that hammock activities play useful roles in solving scheduling problems; by using them: manpower and mutual resource division dilemmas will be solved, idle time between activities will be minimized, and there will be a shorter deadline for the whole project.

This research proposes a new hybrid algorithm which defines a new measure of the float – RCHC.

This new measure is a secondary optimal criterion: firstly, the algorithm finds a schedule with a minimal makespan as a primary objective, and then it finds a schedule with a minimal hammock cost, i.e. a schedule which contains hammock activities, which have a short termination time sometimes multiplied by a cost per time unit.

Managers should use this algorithm to complete a project under resource constraints on time, without delays or resource violations, and with hammock activities with quite short durations i.e. hammocks that are quite easy to manage.

Chapter 3: definitions and notations

A project is a set of interrelated activities. The general model (Battersby, 1967) developed to represent projects is quite a basic concept in OR: a directed and acyclic network. Actually, each project can be modeled by:

a) A discrete and finite set of activities $A = \{A_i : i = 1,2, \dots \dots N\}$. *T*his set is called "jobs or activities".

b) **A *set of precedence conditions***, $\{j_i : i = 1,2, \dots \dots N\}$ where j_i is the set of activities which are immediate predecessors of an activityA_i. An alternative definition toJ_i is: $J_i = \{k : (k \epsilon J_i') \cap (k \epsilon j_m') = \emptyset\}$ for any m belongs to j_i' where j_i' or j_m' is the whole set of activities which have to be completed before starting i or before starting m. Similarly, the set of activities which are the immediate successors of *i,* K_i, can be defined by K_i $\{k : i \epsilon J_k\}$

c) A finite set of attributes $\{B_1(i), \dots \dots B_p(i)\}$ where $p \geq 1$ is defined for each activity and describes the properties that are relevant for project management such as: duration, cost, consumption required of each resource, etc.

d) A finite set of criteria $\{V_1, \dots \dots . V_q\}$ which emphasizes the values and the preferences of the project manager in order to compare alternative decisions concerning the management of the project. The most common criteria are: the total duration, the total cost, a cost-benefit function, and the NPV of the project.

In order to represent the network of activities, the easy way of doing it is by using a directed graph. A directed graph contains nodes with arcs connecting them. In this research the author uses AON representation, meaning that: activities are treated as the graph's nodes, and the graph's edges are treated as the paths between an activity that has to be finished before the given activity starts. This means that an edge $i \rightarrow j$ is a finish-start precedence relation between activity i and j.

We consider that a project consists of N real activities $i \in \{1,2,3, \dots \dots \dots N\}$ with a non-preemptable duration of D_i periods.

There are two dummy activities: *i=0* is a dummy source and *i=N+1* is a dummy sink. The activities are interrelated by precedence and resource constraints. Precedence

constraints prevent an activity from being started before all its predecessors are finished.

Let $PS = \{i \to j, i \neq j, i \in \{0,1,\ldots..N\}, j \in \{1,2,\ldots\ldots N+1\}, FS_{ij} = 0\}$ denote the set of immediate predecessor-successor relations.

Every activity demands some resources or even just a single one. Resource constraints arise as follows: in order to be processed, activity i requires R_{ir} units of resource type $r \in \{1,2,\ldots\ldots R\}$ during every period of its duration.

We assume that the periods are labeled $t \in \{0,1,2,\ldots\ldots T, T+1\}$ where T is defined as ***maximal project duration.*** In our notation, T+1 is defined as a milestone of N+1 (dummy sink), which emphasizes the end of project.

A schedule is a vector $(S_0, S_1, \ldots\ldots S_N)$ which assigns the starting time S_i for every activity$i \in \{1,2,\ldots\ldots N\}$. In this research, we deal with a single-project case, so$S_0 = 0$.

The maximal project duration is defined T+1 – as mentioned above, so $S_{N+1} = T + 1$ is not affected by the resource balancing procedure.

A schedule is considered as ***network-feasible*** if it satisfies the network relations. Let R denote the number of renewable resources required for carrying out the project.

Let $AS = \{AS_{it} \mid i = 1,2,\ldots,N: t = EST_i, \ldots, LST_i\}$ denote the set of ***zero-one*** starting time variables, where EST_i (LST_i) are the earliest (latest) starting times for activity i. usually, AS_{it} =1 if activity i starts in period j, and 0 otherwise. In a schedule, every activity has exactly one starting time, therefore:

$$SS_{ji} = LST_j - max\{LFT_i\} \tag{3.1}$$

$$\sum_{ES_i}^{LS_i} AS_{it} = 1 \ and \ S_i = \sum_{ES_i}^{LS_i} t\, AS_{it} \tag{3.2}$$

$$for\ i = 1,2,\ldots\ldots,N$$

3.1 General definitions

Project management **-** a unique process, consisting of a set of coordinated and controlled activities with start and finish dates, undertaken to achieve an objective conforming to specific requirements including: constraints of time, cost and resources (Demeulemeester and Herroelen, 2002).

Scheduling **–** an organizing procedure of activities, that complies with their beginning time, resource constraints and the order in which they have to be performed.

Project scheduling **–** a project based plan which specifies for each activity the precedence and resource feasible start and finishing dates.

Project makespan (T) **–** the latest permissible completion date for a project.

Earliest starting time (EST) **–** the earliest possible starting time of an activity.

Latest starting time (LST) **–** the latest possible starting time of an activity.

Earliest finishing time (EFT) **–** the earliest possible finishing time of an activity.

Latest finishing time (LFT) **–** the latest possible finishing time of an activity.

Duration (D) **–** the time needed in order to complete an activity.

Total float (or total slack) **–** determines the time period by which the beginning of an activity should be delayed without delaying the whole project. This delay should delay other activities which are not scheduled on the critical chain however the whole project should not be delayed.

Free float (or free slack) **–** determines the time period that the beginning of an activity should be delayed but without delaying its successors (and of course, not delaying the whole project). Every activity which is not scheduled on the critical chain, but its finishing event is – has a free slack. In conclusion, total slack can be bigger or equal to the free slack.

Precedence relation **–** the order in which activities are performed during a project. This order emanates from technological constraints.

Finish to start **–** an activity can only start as soon as all its predecessor activities have finished.

Resources **–** may be of different types including: financial, manpower, machinery, equipment, materials, energy, etc.

Renewable resources – are available at each period (available on a period by period basis).

Non-renewable resources – have a constrained total consumption over the life of the project (available on a total project basis).

Work Breakdown Structure (WBS) - a structure which: defines the project activities in relation to the project result, creates a framework for project control and provides the basis for obtaining relevant insight into the time and cost status of a project through the various management echelons.

Organizational Breakdown Structure (OBS) - shows the various organizational units that are going to work for the project, and enables these responsibilities to be linked to the WBS.

Polynomial algorithm – an algorithm which should be applied by a computer program on a reasonable time-running.

NP-hard algorithm –an algorithm which does not have a known exact solution, however it offers a heuristic method which gets an approximate solution, quite close to the optimal one.

Hammock duration – the difference between the hammock activity's finishing time and its beginning time.

Hammock cost – the hammock's time duration, sometimes multiplied by cost per time-unit.

Path – a series of activities in a network that lead from a starting point to its end.

Critical path – the longest path in the network.

Critical Chain – a critical path that takes into consideration both the order of precedence and the resource constraint.

Feasible solution – a solution which satisfies all constraints, with no violation.

AON - an activity representation whereby a known project is represented by a directed graph where every node represents an activity and an edge represents a precedence relationship between two activities.

AOA - an activity representation whereby a project is represented by a directed graph and each vertex represents an event and each arc represents an activity. When both

activities (arcs) have to end in the same event, they have to meet the same vertex. If an activity in the project should begin only after two or more of its predecessors finish, then a dummy arc is necessary to create a mutual event where all activities can meet.

Makespan **–** the project's time duration, and/or a proposed upper limit. One of the popular objectives is to minimize the total makespan of a project.

RCPSP **–** a classic problem which deals with scheduling activities on a project under resource constraints. There are many methods and approaches for solving it. In large-scale projects heuristics are needed.

RCHCP **–** a secondary objective of the RCPSP problem whereby its objective is to minimize the duration of hammock activities under resource constraints.

Robustness **–** the ability of a known schedule to cope with delays in one or more activities.

3.2 Hammock activity – a formal definition

The concept of hammock activities plays a central role in project management. Hammock activities are used to fill the time span between other "normal" activities since their duration cannot be calculated or estimated at the initial stage of project planning. Typically they have been used to denote usage of equipment needed for a particular subset of activities without predetermining the estimated time the equipment must be present on site. For example: a construction contractor must build a new building. The building process includes many stages and sub-processes like: producing a skeleton of the building, pouring concrete, plastering the walls, etc. Every sub-process is quite complicated. In the most part, the contractor cannot estimate the correct order of activities involved in each process. However, the contractor has a fairly precise schedule which means that he knows the starting and ending times of each sub-process. In such a case every sub-process has the same starting and ending point, however the exact order between activities is not clear. Therefore it is an activity which contains sub-activities with the same starting and ending time points. This is a hammock activity.

Hammock cost (HC) - is the hammock's duration (hammock's finishing time - hammock's starting time) multiplied by a cost per time unit. If the cost per time unit is

equal to one, the hammock cost should reach the hammock's duration. In this research we tried to get it to a minimum as a secondary objective. The formal objective function is: $\min HC = \sum_{h=1}^{H} C_h H_h$ (3.2), when $\overleftrightarrow{H_h}$ represents a known hammock activity's duration, and C_h is equal to a value which represents a price (cost) that has to be paid in order to activate the hammock activity for example: special equipment that has to be bought, sum expense, etc.

In light of all of the above, we can summarize the research's objective as:

The main objective is to get a minimal makespan of the project with a secondary objective of obtaining a minimal hammock cost.

Or in other words:

Under the assumption of the resource constrained and precedence relationship between activities, we have to find the shortest feasible scheduling (feasible – means no resource violation at all) with minimal hammocks' durations.

Chapter 4: The Resource constraint hammock problem

4.1 Introduction

In this chapter, the author discusses the following important issues in depth:

1. Research on hammock activities – with and without resource constraints.
2. A survey of different heuristic methods (including different approaches), and the advantages and disadvantages of every method.
3. Description of HS metaheuristics and SoS, (the core of the research)
4. A description of the author's research, including papers that had been published.
5. A new methodology to deal with RCPSP with hammock activities, and which that has been created as a result of this research.

4.2 Heuristic methods

With time, projects have included more and more activities, and projects have become more and more complicated. As a result exact algorithms could not reach a solution to scheduling problems in polynomial time. Therefore many different heuristics were developed.

Heuristics were based on a principle of getting an approximate solution that is quite close to the optimal one. This kind of approximation is a good tool for coping with NP-hard problems.

Some different heuristic approaches to solving RCPSP problems are:

1. Robust project management – according to Al-Fawzan and Haouari (2005), robust means: “the ability to cope with minor changes in the time duration of the project”, i.e. to extend the slack time of every activity in the project (which means delaying the starting and finishing time of the activities without delaying the project at all), and at the same time minimize the project’s makespan.
2. Proactive scheduling – is a kind of scheduling method which predicts uncertainties of the activities' operating times. When there is an exception the algorithm declines it and so as a result activities become more robust. Proactive

scheduling is based on methods of statistics or redundancy. Lambrechts, Demeulemeester and Herroelen (2007) developed a useful model that minimizes the sum of weighted absolute deviations between the expected real activity starting times and planned activity starting times. The algorithm was based on tabu search, (that will be discussed later in more detail) however the Lambrechts and colleagues' model is inexact and not suitable for all cases of uncertainty. A better algorithm was developed by Van de Vonder, Ballestin, Demeulemeester and Herreolen (2006); it deals with rescheduling activities according to their priorities. The most important activity should be scheduled earliest, and in every step of scheduling the best option of scheduling should be chosen. This is the most promising method of all. However there are many unpredictable events which can cause delays in executing activities, so in many cases a proactive attitude is not applicable.

3. One of the most popular algorithms is a tabu search algorithm (Nai-Hsin, Po-Wen H., and Kuei-Yen C, 2008). It suggests that an arbitrary (and not optimal) initial schedule should be chosen. This initial schedule is improved by generating new (and better) scheduling. All solutions are saved in a memory for later use. The number of iterations is constant and finite, so the algorithm must terminate after a limited time. This algorithm is iterative and is useful when dealing with a small-medium project; however this algorithm diverges when the number of activities in the project is quite high. Demeulemeester, Herroelen and Van De Vonder, (2008) developed a proactive algorithm which was based on a tabu search that in turn was based on a double neighborhood structure in order to generate more scheduling options that maintain time and resource allocations. This algorithm added some idle-time to some activities (mostly when a gap between a planned finished date, and an actual finite date was founded) in order to cope with delays and unpredictable events. This algorithm proved useful when compared to the traditional approach.

4. A simulated annealing (SA) method is a local search algorithm that repeats an iterative neighbor generation procedure, and follows search directions that improve the objective function. The technique replaces a known solution with a better one; then the algorithm compares the new generated solution to other

existing solutions. The algorithm stops when some candidates are optimally converged.

Bouleimen and Lecocq (2003) developed two algorithms which deal with RCPSP and resource-constrained project scheduling problem with multiple modes (MRCPSP). The objectives of both are to minimize the project's makespan. They claim that both procedures are proven efficient for benchmark problems.

5. A genetic algorithm (GA) (or SA, which is a version of it), is a useful metaheuristic method. It is based on two options :

    a. Taking some possible solutions, but not necessarily optimal, and mixing them by taking the first half of the first one and mixing it with the second half of the second one, and vice versa. This technique is called: crossover, and by applying it, two feasible solutions are generated out of one feasible candidate. Those two new solutions are candidates for an optimal solution.

    b. Taking a possible solution and making some minor changes in it. This technique is called: mutation.

    By applying mutation, the number of candidates for an optimal solution increase.

    The GA evolves the chromosomes which represent: the priorities of the activities and delay times. For each chromosome the following two phases are applied:

    (1) *Decoding of priorities' delay times.* This phase is responsible for transforming the chromosome supplied by the genetic algorithm into the activities' and delay times' priorities.

    (2) *Schedule generation.* This phase uses the priorities and the delay times defined in the first phase, and constructs parameterized active schedules.

    After a schedule is constructed, the corresponding measure of quality (*modified makespan*) is feedback to the genetic algorithm. As a result, the best makespan is achieved.

6. A metaheuristics method, which combines some of the above approaches. By using this method some algorithms are run together and so mutually improve each other. A good example of this method is the paper of Tseng and Chen (2005) which describes a combination of a GA, an ant colony optimization procedure and a local search process. The genetic algorithm and the ant colony optimization mutually improved each other so that all the good solutions were collected and saved in a "pool". Csébfalvi G. (2007) developed a metaheuristic algorithm, very similar to that of Tseng and Chen based on a HS approach. The algorithm produced a relatively fast solution in comparison to other algorithms.

   In conclusion, there are many useful heuristic algorithms which produce some very good solutions to NP-hard problems of RCPSP, thereby making the choice quite difficult. In this research we have adopted the HS approach, developed by Csébfalvi G. (2007). A more detailed explanation will be given later.

   Since calculating a hammock's duration is a type of RCPSP problem, the calculating process will be explained in more detail in the next sub-chapter.

4.3 The computing process of a hammock's duration

The main objective of this research is to find a new tool that can be useful in solving certain scheduling problems. By solving them, managers can schedule hammock and regular activities under resource constraints. Heuristic methods could be helpful here.

As already mentioned, the traditional approach of RCPSP does not have a satisfactory answer to a scheduling problem with hammock activities when it deals with medium-large scale projects. This is the main motivation for this research. The unconstrained hammock problem is introduced, from which we will develop the new model.

The unconstrained hammock problem

Harhalakis G. (1990) defined the hammock activity clearly, and developed an algorithm to calculate the hammock's duration.

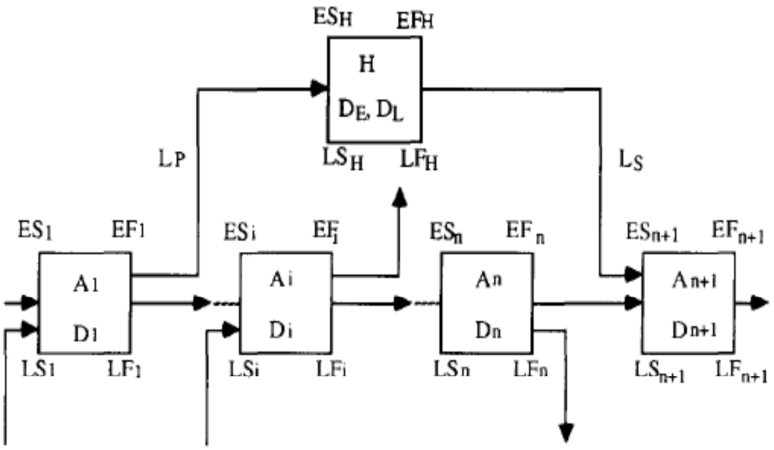

Figure 9: H is a hammock activity connected to two "regular" activities A_1 and A_{n+1} which are its hanging points. (Harhalakis, 1987)

In figure 9, H is a hammock activity which is connected to two regular activities: A_1 and A_{n+1}. Usually A_1 and A_{n+1} are considered to be the dummy activities of the project. Their main function is to mark the clear starting and finishing points of the hammock – following its formal definition (see chapter 3.2).

These are regular activities in the sense that it is easy to estimate their durations. The total float of A_1 and A_{n+1} is:

$$TF_1 = LF_1 - EF_1 \ \ and \ \ TF_{n+1} = LS_{n+1} - ES_{n+1} \qquad (4.1)$$

Where:

LF = Latest Finish of an activity LS = Latest Start of an activity

EF = Early Finish of an activity ES = Early Start of an activity

TF = Total Float of an activity

During the forward run, the earliest starting time of the hammock can be determined unambiguously as:

$$ES_H = EF_1 + L_p \qquad (4.2)$$

Where L_p = Lag time between activity A_1 and H (meaning that L_p is the free slack between A_1 and H).

The latest starting time of a hammock based on the latest finishing time of its predecessor and the lag between them, meaning:

$$LS_H = LF_1 + L_p \quad (4.3)$$

Similarly, during the backward run, the latest finishing time of the hammock can be determined as:

$$LF_H = LS_{n+1} - L_S \quad (4.4)$$

Where Ls = Lag time between hammock activity H and A_{n+1} (meaning that L_s is the free slack between H and A_{n+1}).

The hammock's early finish can be calculated by:

$$EF_H = ES_{n+1} - L_S \quad (4.5)$$

Next the hammock's duration (D_E) can be calculated by subtracting the hammock's early finishing Es from the hammock's early starting E_F:

$$D_E = EF_H - ES_H \quad (4.6)$$

(based on early start and finish), similarly, D_L = hammock's duration based on late starting and finishing, and can be calculated as:

$$D_L = LF_H - LS_H \quad (4.7)$$

Equation (4.6) leading to calculate the latest start of a hammock:

$$LS_H = LF_H - D_E \quad (4.8)$$

In order to get an unambiguous hammock's duration, Harhalakis (1990) concluded that$D_E = D_L$, if and only if $TF_1 = TF_{n+1}$, meaning that the hammock activity has no leg at all (so $D_H = D_E = D_L$), however, if $TF_1 \neq TF_{n+1}$ then $D_E \neq D_L$ and $D_H = min(D_E, D_L)$

Proof: following the above (4.6 and 4.7) $D_E - D_L = (EF_H - ES_H) - (LF_H - LS_H)$, or based on (4.5) and (4.3) $D_E - D_L = (ES_{n+1} - L_s - ES_H) - \left(LF_H - LF_1 - L_p\right)$, when using (4.2) and (4.4) into the latter, we get:

$$D_E - D_L = (LF_1 - EF_1) - (LS_{n+1} - ES_{n+1})$$

Finally, using (4.1), we get $D_E - D_L = TF_1 - TF_{n+1}$ so as a conclusion, D_E equals D_L iff $TF_1 - TF_{n+1} = 0$.■

As a conclusion, Harhalakis calculated the hammock cost in the unconstrained case.

In order to understand Harhalakis' research a description of simple instances are given below:

Harhalakis dealt with two problems:

1. Estimation of the hammock's duration
2. Dealing with the criticality of hammock activity. Both need a different analysis.

. Figure no.10 illustrates example #1 whereby a hammock is linked between two ordinary activities that have an equal total float.

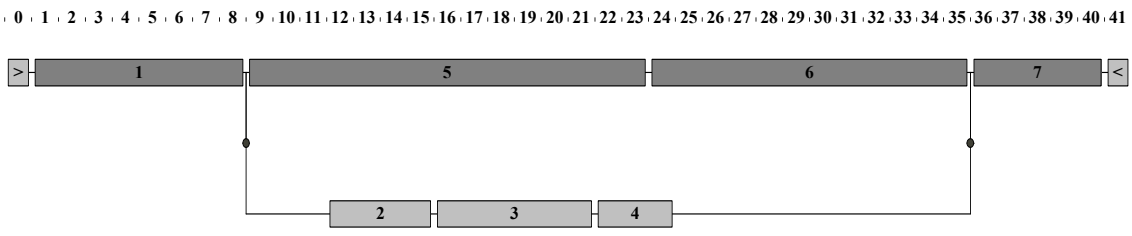

Figure 10: A hammock linked between two ordinary activities with an equal total float

As shown in figure 10, activities no.1 and no. 7 are regular, critical activities – their total slack is equal to zero. The hammock contains activities no. 2, 3, and 4, which all have the same total float, meaning $TF_2 = TF_3 = TF_4 = 14$.

Since there is no lag between activity no.1 (the hammock's predecessor) and the hammock activity, and there is no lag between the hammock activity and activity no. 7 (the hammock's successor), the hammock duration is equal to:

$$D_H = EF_H - ES_H = ES_7 - EF_1 = 36 - 9 = 27.$$

As a result, we have an estimation of the hammock duration in the unconstrained case.

As activities no. 2, 3, and 4 are not critical, it leads us to conclude that a hammock as a whole can be moved – according to the limitation of the time range [9, 36] in this example (as usual the range flows between ES_1 and EF_{n+1}).

Figure no. 11 illustrates example no. 2, whereby a hammock is scheduled between two regular activities with a different total float.

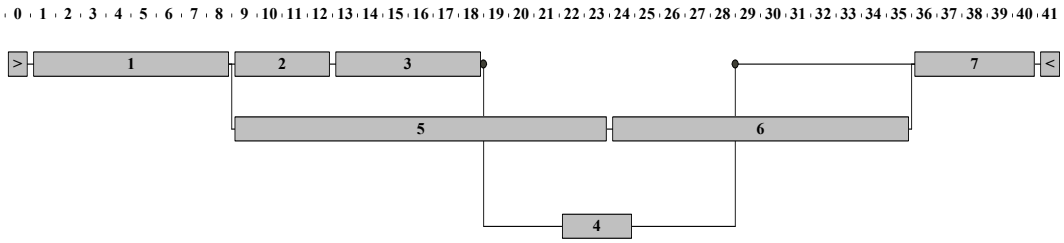

Figure 11: A hammock is connected between two activities with a different total float

As shown in figure 11, the hammock contains two activities: activity no. 4 – which begins at t=19 (as a successor of activity no.3), and is bounded by a milestone in t=29 (meaning that the latest finishing time is t=29), and activity no. 6, which is critical and its time range is [24, 36]. The hammock's hangers are activity no. 3 and activity no. 7. Activity no. 7 is critical ($TF_7 = 0$), however activity no. 3 is not– its total slack is not zero but rather $TF_3 = 14$. D_L (=the hammock duration based on latest times), should be calculated as $D_L = 36 - 33 = 3$, and conversely, based on the earliest times, the earliest finish of the hammock is equal to 35 (the earliest finish of activity 7 – the successor of the hammock) and the earliest start of the hammock equals to 19, so D_E (=the hammock duration based on earliest times), is 36-19=17.

In conclusion $D_H = min\,(D_E, D_L) = 3$, and the LS of the hammock equal 19.

Remark: the difference of $D_E - D_L = 17 - 3 = 14$, equals the maximal total float of the hammock members.

In conclusion, the 2nd example deals with the critically of the hammock: in this case, a non-critically inner activity can be moved, however other critically inner activities cannot. In conclusion, we can understand that the hammock itself behaves in the same way as a whole project, meaning that a critically activity cannot be moved, and the hammock duration is determined by the longest path (=critical path) between its hangers. The question that we have to ask here is: what is the best position of every non-critically hammock member? We can approach this question by introducing: 'best position' of an inner activity.

An example of best position:

Regarding the underground project in the Netherlands (Van-Houcke, Van Osselar, 2004), as explained above, the total cost of the project was reduced to a minimum; this optimization was achieved owing to two important paradigms by:

1. Reorganizing the whole project as a collection of hammock activities, Figure 8 shows that all 11 activities were organized as 4 hammock activities.

2. Reorganizing the order of inner activities in the hammock; In this example all activities were organized so that the freezing machine would be positioned correctly, and consequently the idle time would be minimized.

The routine was repeated 26 times, equal to the number of traverse links in the tunnel, and more than one million Euros were saved. The hammock cost in this example could be realized by reducing the idle time, which in turn reduced the total hammock cost, and enabled the authors to realize the best position of every hammock member.

The other important matter that we have to address is: the criticality of the hammock as a whole. As shown in figure no. 10, this hammock can be moved since there is a slack of 14.

By addressing this issue, we can estimate the flexibility of the hammock.

Regarding the hammock as a whole there are two possibilities:

1) The hammock is critical; in this case the hammock cannot be moved at all. Its time range is determined without any possibility to change it
2) The hammock is not critical, so its hangers can be fixed and its length can be minimized or maximized.

Although this initiative is very important, there is no estimation of hammock durations under resource constraints. Harhalakis (1990) was the first to deal with hammock activities. Accordingly the constraint case was not investigated. Indeed, very little research has been done to further Harhalakis' work; so this research endeavors to enrich Harhalakis' work and expand upon the knowledge of hammocks in project management.

4.4 The constraint hammock problem

Vanhoucke and Van Osselaer (2004) solved the tunnel problem in the Netherlands by applying the hammock theory**.** It was the first time that hammock activities were scheduled under resource constraints, however their model was quite restricted as was mentioned in chapter 2, because it was applied as a repetition procedure.

Csébfalvi G. and Csébfalvi A. (2005) proposed an algorithm which deals with the hammock's scheduling under resource constraints. Thus a new approach was developed: RCHCP that is a solution to the scheduling problem under resource constraints and hammock cost consideration. The algorithm is based on solving the MILP problem and IE (implicit enumeration), which are both based on: the *forbidden set principle*. Prior to discussing Csébfalvi's algorithm, we will explain the idea of the *forbidden set concept*.

Forbidden set: A forbidden activity set *F* is identified such that:

(1) All activities in the set may be executed concurrently.

(2) The usage of some resources by these activities exceeds the resource availability.

(3) The set does not contain another forbidden set as a recognized *subset.*

A resource conflict can be repaired explicitly by inserting a network feasible precedence relation between two forbidden set members, which will guarantee that not all members of a forbidden set can be executed concurrently. At the same time, an inserted explicit conflict repairing relation (as its side effect) might be able to repair one or more of the other conflicts implicitly. Let $i \rightarrow \cdots . . \rightarrow j$ denote that activity j is a direct (indirect) successor of activity i. An $i \rightarrow j$ explicit repairing relation might be replaced by an $p \rightarrow q$ implicit relation, where $i \rightarrow \cdots . \rightarrow p$ and $q \rightarrow \cdots . \rightarrow j , i \neq p \cup q \neq j$, if there is another forbidden set for which $p \rightarrow q$ is an explicit repairing relation. Let ER(F) (IR(F)) denote the set of implicit (explicit) repairing relations for forbidden set F. Hereunder figure 12 shows the early CPM schedule for a simple example.

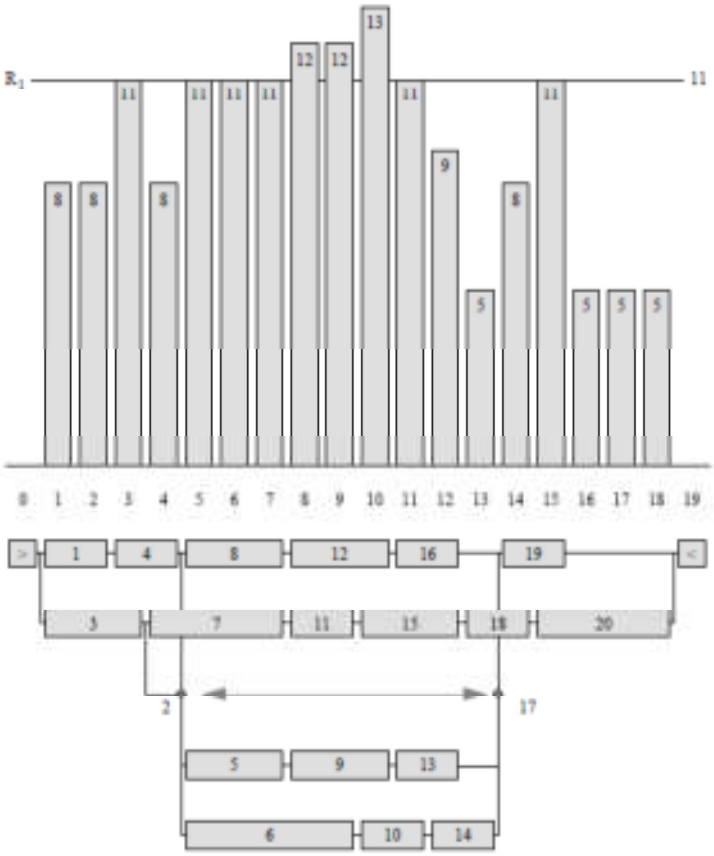

Figure 12: The early CPM of a simple example (Csébfalvi G.and Csébfalvi A., 2005)

The following tables show the forbidden sets and their explicit (implicit) repairing sets from the example above. In the presented earliest CPM schedule every conflict is feasible. Note that a feasible conflict may be "visible" or "hidden". A hidden conflict is "invisible" in the

earliest CPM schedule, but might be visible in a shifted schedule. In the example above the total number of forbidden sets is sixteen, but in the early schedule only three conflicts namely: $\{F_1, F_{13}, F_{14}\}$ are visible (active).

Table 1: Forbidden sets and explicit repairs

i	Visible	Interval	FS_i	$ER(FS_i)$
1	Yes	[8,9]	{6,9,11,12}	$\{6 \to 9{,}6 \to 12{,}11 \to 9{,}9 \to 12{,}12 \to 9, 11 \to 12\}$
2		[11,11]	{5,10,16}	$\{5 \to 10{,}5 \to 16{,}10 \to 16{,}16 \to 10\}$
3		[12,14]	{9,14,16}	$\{9 \to 14{,}9 \to 16{,}14 \to 16{,}16 \to 14\}$
4		[15,16]	{13,14,16,20}	$\{13 \to 14{,}14 \to 13{,}13 \to 16{,}16 \to 13{,}13 \to 20,$ $14 \to 16{,}16 \to 14{,}14 \to 20{,}16 \to 20\}$
5		[8,9]	{6,8,9,11}	$\{8 \to 6{,}6 \to 9{,}8 \to 9{,}8 \to 11{,}11 \to 9\}$
6		[13,14]	{10,13,16,18}	$\{10 \to 13{,}13 \to 10{,}10 \to 16{,}16 \to 10{,}10 \to 18,$ $13 \to 16{,}16 \to 13{,}13 \to 18{,}18 \to 13{,}16 \to 18,$ $18 \to 16\}$
7		[11,11]	{5,6,15,16}	$\{5 \to 6{,}5 \to 15{,}5 \to 16{,}6 \to 15{,}6 \to 16{,}15 \to 16\}$
8		[11,12]	{6,9,16}	$\{6 \to 9{,}6 \to 16{,}9 \to 16\}$
9		[13,14]	{9,10,18}	$\{9 \to 10{,}10 \to 9{,}9 \to 18{,}10 \to 18\}$
10		[11,14]	{9,10,16}	$\{9 \to 10{,}10 \to 9{,}9 \to 16{,}10 \to 16{,}16 \to 10\}$
11		[10,11]	{5,8,10,15}	$\{5 \to 8{,}8 \to 5{,}5 \to 10{,}5 \to 15{,}8 \to 10{,}8 \to 15,$ $15 \to 10\}$
12		[13,14]	{9,12,14,18}	$\{9 \to 12{,}12 \to 9{,}9 \to 14{,}9 \to 18{,}12 \to 14$ $, 12 \to 18{,}18 \to 14\}$
13	Yes	[10,12]	{9,10,15}	$\{9 \to 10{,}10 \to 9{,}15 \to 10\}$
14	Yes	[10,14]	{9,10,12}	$\{9 \to 10{,}10 \to 9{,}9 \to 12{,}12 \to 9{,}10 \to 12,$ $12 \to 10\}$
15		[10,11]	{5,10,12,15}	$\{5 \to 10\ ,5 \to 12{,}5 \to 15{,}10 \to 12{,}12 \to 10,$ $15 \to 10\}$

16		[10,11]	{8,9,10}	$\{8 \to 9,8 \to 10, 9 \to 10, 10 \to 9\}$

Table 2: Forbidden sets and implicit repairs

i	Visible	Interval	FS_i	$ER(FS_i)$
1	Yes	[8,9]	{6,9,11,12}	$\{10 \to 9, 10 \to 12\}$
2		[11,11]	{5,10,16}	$\{13 \to 10,5 \to 6,9 \to 10, 9 \to 12,9 \to 16,13 \to 16,5 \to 8,$ $5 \to 12,14 \to 16,10 \to 12\}$
3		[12,14]	{9,14,16}	$\{13 \to 14,13 \to 10,9 \to 10,9 \to 12,13 \to 16,16 \to 10\}$
4		[15,16]	{13,14,16,20}	$\{13 \to 10,13 \to 18,16 \to 10,16 \to 18\}$
5		[8,9]	{6,8,9,11}	$\{10 \to 9,12 \to 9,8 \to 5\}$
6		[13,14]	{10,13,16,18}	$\{14 \to 13,10 \to 9,14 \to 16,10 \to 12\}$
7		[11,11]	{5,6,15,16}	$\{9 \to 12,9 \to 16,13 \to 16,5 \to 8,5 \to 12,$ $6 \to 12,10 \to 16,14 \to 16,10 \to 12,18 \to 16\}$
8		[11,12]	{6,9,16}	$\{10 \to 9,6 \to 12,10 \to 16,14 \to 16,10 \to 12,9 \to 12,13 \to 16\}$
9		[13,14]	{9,10,18}	$\{13 \to 10,13 \to 18\}$
10		[11,14]	{9,10,16}	$\{13 \to 10,9 \to 12,13 \to 16,14 \to 16,10 \to 12\}$
11		[10,11]	{5,8,10,15}	$\{13 \to 10,5 \to 6,9 \to 10,16 \to 10,8 \to 6,12 \to 10,8 \to 11\}$
12		[13,14]	{9,12,14,18}	$\{13 \to 14,13 \to 10,9 \to 10,13 \to 18,16 \to 10,16 \to 14,$ $12 \to 10,16 \to 18\}$
13	Yes	[10,12]	{9,10,15}	$\{13 \to 10\}$
14	Yes	[10,14]	{9,10,12}	$\{13 \to 10,16 \to 10\}$
15		[10,11]	{5,10,12,15}	$\{13 \to 10 ,5 \to 6,9 \to 10,9 \to 12,5 \to 8,16 \to 10\}$
16		[10,11]	{8,9,10}	$\{12 \to 9,8 \to 5, 16 \to 10, 8 \to 6,12 \to 10,13 \to 10\}$

The hidden conflict, for example: {5, 10, 16} is not "visible" in the early CPM so at first sight it is unseen. However after entering some "visible" conflict repairs, some of the

activities are moved to the right and cause a new conflict later. The latter conflict is termed: "hidden".

Following the example and the tables above, this algorithm is innovative in solving scheduling problems. It recognizes and solves all conflicts before the scheduling process itself by ignoring their explicitly or implicitly and this is done by inserting relations termed: "conflicts repairs" between activities. The scheduling process of hammocks and regular activities begins only after repairing all conflicts. The objective is based on "hammock cost" – meaning minimizing the duration of all hammocks, and sometimes multiplying it by a cost per operating time unit. As a result of inserting all of the conflicts repairs, we get a robust schedule which is immune to all resource violations, that is: every activity should be shifted along its slack without violating the achieved solution.

Accordingly our objective function is the following:

$$min\left[HC = \sum_{K=1}^{H} C_k \cdot \vec{H}_k\right] \tag{4.9}$$

Assuming that a project contains H hammock activities, when each hammock activity demands C_k price units from each operation time-unit, the total hammock cost is defined as a sum of multiplying the hammock's duration (D_E, as defined before) with its cost. This cost should be minimized by the algorithm. The reason for such a definition is: hammock activities are defined as activities which need special funds, and/or extra equipment, and a budget per time-operation unit.

The existing algorithms are exact algorithms which provide a solution only for small to medium sized problems. These algorithms do not provide a solution in polynomial for a large scale problem, and so a heuristic algorithm is needed. As a result of this limitation, a new heuristic (and efficient) tool has to be developed. The heuristic algorithm in this research is SoS.

4.5 Harmony search (HS)

The HS process is an analogy between project management scheduling under resource constraints (RCPSP) and the music world. Lee and Geem (2005) developed a useful tool: a metaheuristic algorithm which efficiently solves a RCPSP problem. Every activity i ($i \in \{1,2,\ldots..N\}$) is represented by a player in the orchestra. Every player makes a single sound from his/her initial repertoire. This single sound has a known time-range and is chosen randomly by the player who determines its timing. All those sounds are collected into a melody which contains N sounds from N players. This melody is compared to the worst one in the known repertoire to date. When the new melody is better than the worst one in the repertoire it replaces it. Thus the repertoire's quality steadily improves with every step, as does the solution, and there is a better chance to get the near-optimal solution to the second criteria of the problem (RCHCP).

Every player is an activity that has to be scheduled in the RCPSP world. The timing of entering into the melody is represented by a time variable that is limited to the activities' time duration. In the end the N-players melody is a new schedule that was created by improvisation. This algorithm's purpose is to find the best schedule by improvisation, where ‘best’ means the shortest and most feasible makespan from the improvisation with a near-optimal, to the hammock cost as a second objective. During all this process, the aesthetic value of the sounds is important, meaning that in every scheduling period, no resource violation is allowed. In conclusion, ‘best scheduling’ means: the shortest and most feasible scheduling. The main advantage of HS over and above other methods is that there is no need for a complicated mathematical calculation to get initial values of time variables. Instead, HS uses stochastic variables to get random values. HS solves quite complicated problems, not only scheduling problems, but also problems in other areas of engineering.

It is necessary to describe the similarity between the music world and the RCPSP problem, therefore a mathematical description of the HS procedure is needed.

In HS, the optimization problem is specified as follows:

$$\max\{f(X)|X = \left\{X_i \left| \underline{X_i} \leq X_i \leq \bar{X_i}, i \in \{1,2, \dots \dots N\}\right.\right\}\right\} \quad (4.10)$$

In the language of music, X is a melody, that's aesthetic value is represented by f(X). The higher the value of $f(X)$ is, the higher the quality of the melody will be. In the band, the number of musicians is N, and musician i, $i\epsilon\{1,2.....n\}$, is responsible for sound X_i. The improvisation process is driven by two parameters: (1) According to the repertoire consideration rate (RCR), each musician chooses a sound from his/her repertoire with probability: RCR, or a totally random value with probability: (1-RCR); (2) According to the sound adjusting rate (SAR), the sound selected from his/her repertoire will be modified with probability: SAR. The algorithm starts with a totally random "repertoire upload" phase, after that the band begins to improvise. During the improvisations, when a new melody is better than the worst in the repertoire, the worst will be replaced by the better one. Naturally, the two most important parameters of the HS algorithm are the repertoire size and the number of improvisations. The HS algorithm is "explicit" because it operates directly on the sounds. In the case of RCPSP we can only define an "implicit" algorithm, and if we do not introduce a "conductor," we cannot manage the problem efficiently.

First, we show how the original problem can be transformed into the world of music. Here the resource profiles form a "polyphonic melody". Assuming that in every phrase only the "high sounds" are audible, the transformed problem will be to find the shortest "SoS" melody by improvisation. Naturally, the "high sound" in music is analogous to the overload in scheduling.

In the language of music, the RCPSP can be summarized as follows:

- the band consists of N musicians;
- the polyphonic melody consists of R phrases and N polyphonic sounds;
- each $i\epsilon\{1,2.....n\}$musician is responsible for exactly one polyphonic sound;
- each $i\epsilon\{1,2.....n\}$ polyphonic sound is characterized by the set of the following elements: $\left\{X_i, D_i, \{R_{ir}|r \in \{1,2, \dots \dots R\}\}\right\}$; the polyphonic sounds (musicians) form a partially ordered set according to the precedence (predecessor-successor) relations;

- each $r\epsilon\{1,2.....R\}$ phrase is additive for the simultaneous sounds;
- in each phrase only the high sounds are audible:

  $\{U_{tr}|U_{tr} > R_r, t = 1,2, \ldots..T\}$;
- in each repertoire's uploading (improvisation) step, each $i\epsilon\{1,2.....n\}$ musician has the right to present (modify) an idea $IP_i \in [-1,+1]$ about X_i where a large positive (negative) value means that the musician wants to enter the melody as early (late) as possible;
- in the repertoire's uploading phase the "musicians" improvise freely, $IP_i \leftarrow RandomGauss(0,1)$; the function $\eta \leftarrow RandomGauss(\mu,\sigma)$ generates random numbers from a truncated $(-1 \leq \eta \leq 1)$ normal distribution with mean μ and standard deviation σ;
- During the improvisation phase the "freedom of imagination" is monotonically decreasing step by step, $IP_i \leftarrow RandomGauss(0,1)$ where standard deviation σ is a decreasing function of the progress (see figures 13 and 14). Regarding figure 14, the conductor reduces the freedom of players by an asymmetric normal-distribution with a monotonic growth towards 1 (meaning that most of schedules should be timed as early as possible);
- each of the possible decisions of the HS process (melody selection and idea-driven melody construction) is the conductor's responsibility;
- the band tries to find the shortest "SoS" melody by improvisation.

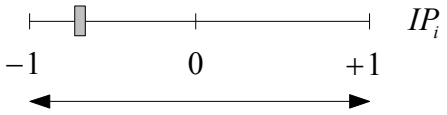

Figure 13: An idea IP_i about the "best" position (Csébfalvi et al., 2008b)

-

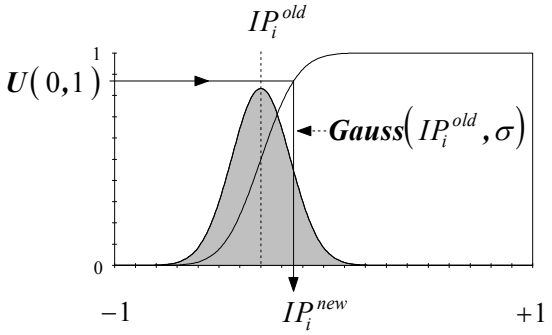

Figure 14: Perturbation of IP_i (Csébfalvi et al., 2008b)

The conductor solves a LP problem to balance the effect of the nearly diametrically opposed ideas about a shorter "SoS" melody. The LP problem which maximizes the satisfaction of the musicians with the sound positions is:

$$min\left[\sum_{i=1}^{N} IP_i \cdot X_i\right] \tag{4.11}$$

$$X_i + D_i \leq D_j, i \to j \in PS \tag{4.12}$$

$$\underline{X_i} \leq X_i \leq \overline{X_i}, \qquad i \in \{1,2,\ldots..N\} \tag{4.13}$$

The result of the optimization is a schedule (melody) which is used by the conductor to define the final starting (entering) order of the sounds (musicians). The conductor generates a soundless melody by taking the selected sounds one by one in the given order, and schedules them at the earliest (latest) feasible start time. After that by using the well-known forward-backward improvement (FBI) methods (Tormos and Lova, 2001) the conductor tries to improve the quality of the generated melody. Naturally, the conductor memorizes the shortest feasible melody found till that time.

A very similar approach was represented by Tseng and Chen (2005) which developed a combination of: a genetic algorithm, an ant colony optimization procedure and *local search.*

Local search – is a technique which tries to improve a known solution by finding new directions. In this algorithm the authors take a known schedule and apply the *local*

search procedure that first calculates the forward schedule; the backward schedule is calculated later when activities, which need less (more) resources, should be scheduled later (earlier). Thereafter the makespan of the new schedule is calculated and compared to the best known schedule so far. If it is found to be shorter, it should replace the worst known schedule up until that time.

This algorithm operates two algorithms – a genetic one and an ant colony optimization which mutually improve each other in such a way that all good solutions are collected and saved in a "pool". The *local search* improves both of the solutions in the two procedures. This algorithm presents good solutions to the RCPSP problem: the average deviation from the critical path, based on lower bounds of J60, is 11.94 out of 1000 schedules, and 34.49 out of 1000 schedules of J120. For this reason, and because the SoS leans on the same principles, ANGEL has been used as a good basis for the SoS algorithm.

4.6 The Sound of Silence (SoS) algorithm

According to the NP-hard nature of the problem, the proposed implicit enumeration algorithm provides exact solutions for small to medium sized problems within a reasonable period of time (Csébfalvi G. and Csébfalvi A., 2005). Large-scale problems can be managed by introducing an optimality tolerance. Csébfalvi G. (2007) developed a new algorithm based on a HS principle: the "SoS algorithm".

The central element of the “SoS algorithm” can be classified as a pure forward-backward list scheduling without improvement. In this element the only, but extremely important novelty, is the developed activity list generator, which is practically independent of the applied metaheuristic frame, and able to provide near-optimal solutions for large “academic” instances very quickly. The fast and effective activity list generator is based on an “unusual” integer linear programming problem which can be solved in polynomial time by relaxing the integrality assumption. The computational results show that the “SoS” is a fast and high quality algorithm. The SoS algorithm is an improvement version of HS approach.

At the beginning of the iterations the conductor collects the ideas. The ideas are random sounds which are produced by the players in the orchestra which is the initial basis of

the repertoire. All ideas are activities which have to be scheduled, and the schedule itself is an LP problem.

The conductor solves the inductive logic programming (ILP) problem as an LP using a fast interior point method. Usually the result of the optimization is a “very interesting” schedule (melody).

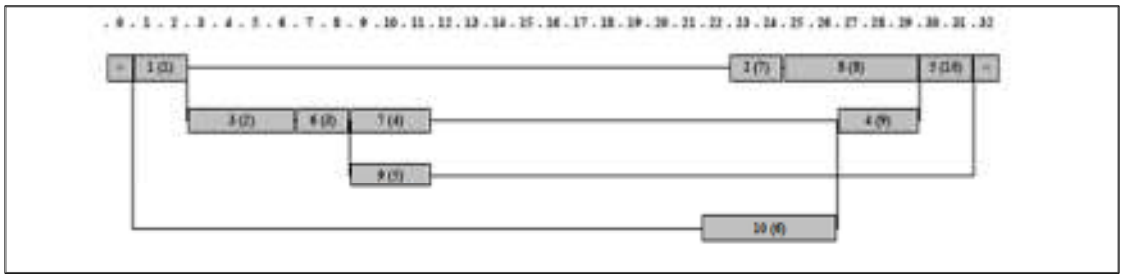

Figure 15: Activity list generation ─ two loud parts with a "long break" between them (Csébfalvi et al., 2008a)

In the case of a simple project the melody consists of two (loud) parts (themes) with a very long break between them (as in Figure 15 above). According to our objective function, in the schedule, activity 4 and activity 8 border one another $(-0.3S_4 - 0.6S_8)$ and push activity 5 forward as much as possible $(S_5 \leq 30)$, so activity 5 does not have enough force $(0.4S_5)$ to avoid the aggression.

This procedure is used by the conductor for determining an initial schedule which defines the final starting (entering) order of the sounds (musicians).

When two or more activities have the same starting time (e.g. activity 7 and 9), the conductor solves the problem by random permutation.

The essence of the activity list oriented (serial) forward-backward scheduling is well known.

The schedule generation schema transforms activity list L into a schedule S(L) by taking the activities one by one in the order of the activity list and scheduling them at the earliest (latest) precedence and resource-feasible starting time. In the example above, the order is a vector $I = \{1,3,6,7,9,10,2,8,4,5\}$. I is created as a result of tossing random numbers between [-1, 1]. Those numbers are tossed by the players in the orchestra. When a positive (negative) number is tossed the activity should be scheduled

earlier (later). After that, in the " SoS algorithm" the conductor selects the best (makespan minimal) schedule. When two or more schedules have the same makespan, the conductor chooses the forward one (earlier) between the two.

As previously mentioned, the vector I is created as a decision of the players, e.g. $I = \{0.3, -0.1, 0.5, -0.3, 0.4, -0.2, 0.3, -0.6, 0.7, -0.6\}$. This means that activities 1, 3,5,7,9 should be scheduled at their earliest starting times, however the rest at their latest. Therefore the LP problem that the conductor has to solve is:

$$\min(0.3S_1 - 0.1S_2 + 0.5S_3 - 0.3S_4 + 0.4S_5 - 0.2S_6 + 0.3S_7 - 0.6S_8 + 0.7S_9 - 0.6S_{10}) \quad (4.14)$$

s.t.

$$1 \le S_1 \le 16 \qquad S_1 + 2 \le S_2$$

$$3 \le S_2 \le 23 \qquad S_1 + 2 \le S_3$$

$$3 \le S_3 \le 18 \qquad S_2 + 2 \le S_8$$

$$12 \le S_4 \le 27 \qquad S_3 + 4 \le S_6$$

$$15 \le S_5 \le 30 \qquad S_4 + 3 \le S_5$$

$$7 \le S_6 \le 22 \qquad S_6 + 2 \le S_9$$

$$9 \le S_7 \le 24 \qquad S_6 + 2 \le S_9$$

$$5 \le S_8 \le 25 \qquad S_7 + 3 \le S_4$$

$$9 \le S_9 \le 29 \qquad S_8 + 5 \le S_5$$

$$1 \le S_{10} \le 22 \qquad S_{10} + 5 \le S_4$$

The constraints of this LP problem are the time durations of each activity and the operating time-ranges. Since this is a minimal objective, the conductor should schedule the activities which have positive (negative) coefficients as late (early) as possible. The solution to this LP problem is an initial schedule (non-optimal) that should be

improved. The improvement process moves the starting time of the earliest (latest) activities to the left (right) by controlling its freedom. Controlling its freedom is done by using of the model (4.11)-(4.13) - as was introduced above.

The conflict repairing version of the "SoS algorithm" is based on the forbidden set concept. In the conflict repairing version the primary variables are conflict repairing relations; a solution is a makespan minimal resource-feasible solution set in which every movable activity can be shifted without affecting the resource feasibility. In the traditional "time oriented" model the primary variables are starting times, therefore an activity shift may be able to destroy the resource feasibility. The makespan minimal solutions of the conflict repairing model are immune to the activity movements; therefore we can introduce a secondary performance measure to select the "best" makespan minimal resource feasible solution from the generated solution sets. In the "SoS algorithm", according to the applied replacement strategy (whenever the algorithm obtains a solution superior to the worst solution of the current population, the worst solution will be replaced by the better one), the quality of the population gradually improves. The larger the makespan minimal subset size becomes, the greater the chance to get a good solution for the secondary criterion. It is well-known that the crucial point of the conflict repairing model is the forbidden set computation. In the "SoS algorithm" the conductor uses a simple (but fast and effective) "thumb rule" to decrease the time requirement of the forbidden set computation. In the forward-backward list scheduling process the conductor (without explicit forbidden set computation) inserts a precedence relation $i \rightarrow j$ between an already scheduled activity i and the currently scheduled activity j whenever they are connected without lag ($S_i + D_i = S_j$). The result is a schedule without "visible" conflicts. Thereafter the conductor (in exactly one step) repairs all of the hidden (invisible) conflicts, and always inserts the "best" conflict repairing relation for each forbidden set. In this context "best" means a relation $i \rightarrow j$ between two forbidden set members for which the lag $(S_i - S_j - D_i)$ is maximal. In the language of music, the result of the conflict repairing process will be a robust (flexible) "SoS" melody, in which the musicians have some freedom to join the performance without affecting the aesthetic value of the composition.

Theoretically the resource-constrained case can be formulated as a MILP problem which can be solved for small-scale projects within a reasonable time. It is important to

note that in the HS algorithm the RCHC measure of a "repaired" schedule can be evaluated in polynomial time by using the presented unconstrained LP formulation.

A resource profile of the scheduling problem out of a forward and backward procedure presented hereunder, clarifies the abovementioned example:

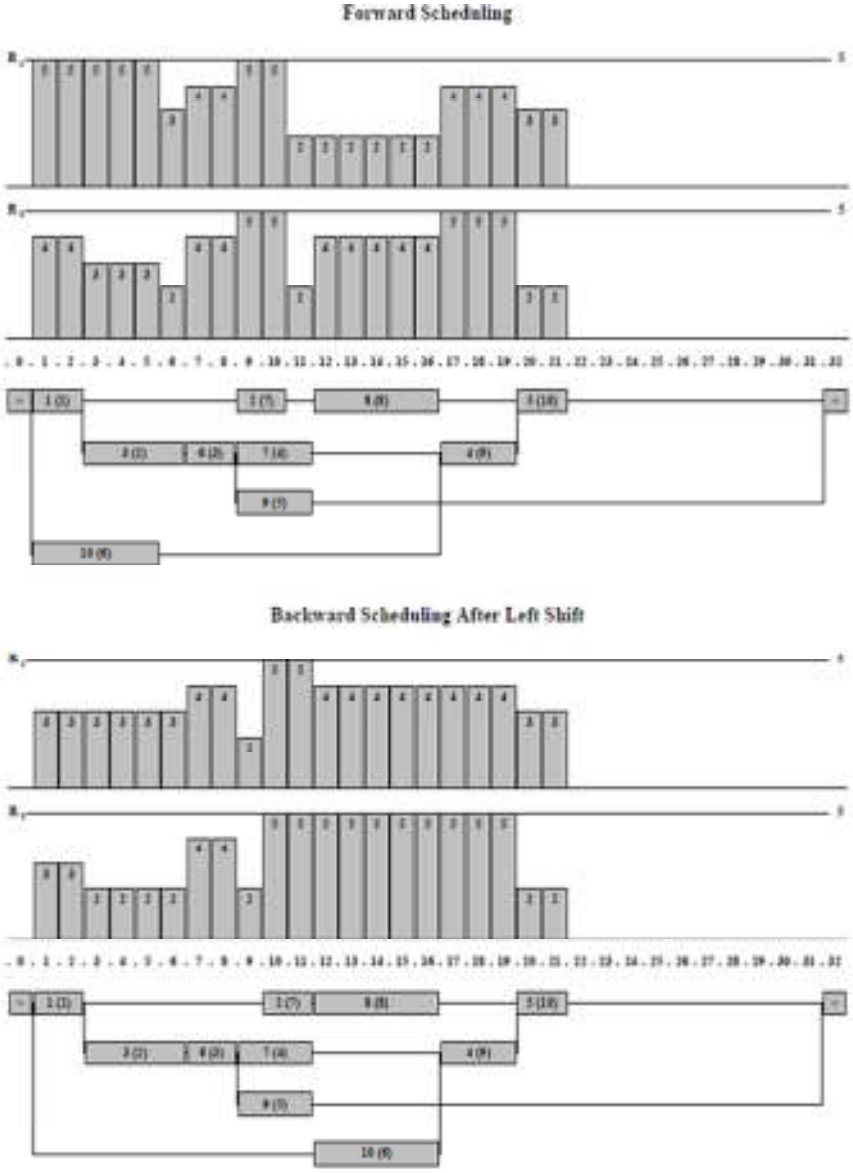

Figure 16: Activity list oriented serial forward-backward scheduling (Csébfalvi, et al., 2008a)

The solution is a feasible schedule with a makespan of 21. This value was achieved in the forward and backward scheduling, therefore the conductor should choose the forward one. (Figure 16 above). The lower bound of this problem's makespan is 20 time units, showing that there is a diversity of 5% (21-20)/20 *100).

Later, this idea was expanded to deal with hammock activities.

Csébfalvi et al. (2008a, 2008b), developed a multi-objective algorithm which, based on the "SoS" principle, reduces makespan significantly and at the same time minimizes the hammock's cost; this is a secondary objective when solving the problem.

The idea of SoS is to give all players in the orchestra the possibility to improvise freely. At the beginning of the process, several of the players should join the music by playing as early as possible – according to their "time window". However the rest should join as late as possible. (There is a simple objective which "dictates" this principle. By applying it, we get an initial playing order). As a result the time gap between the two groups of players is wide – which means that we get a wide makespan that should be reduced. The form of this melody is a continuous and noisy (unfeasible schedule t with a resource violation) rendition at the beginning with a long break (silence), and then another noisy unfeasible part at the end.

It is important to get an initial and arbitrary order of playing at the beginning of the process.

The players' freedom is then reduced from iteration to iteration therefore the time gap between the groups is reduced. With each step this gap is reduced until it hardly exists. When this is achieved, the algorithm terminates by getting a heuristic scheduling (solution). The players' freedom is reduced by controlling a stochastic variable that determines the timing of the playing, as will be demonstrated hereunder.

With regard to speed and efficiency the "SoS" is the best heuristics in that field Csébfalvi G. (2007) compared many heuristics and found that in the case of 600 'heavy' (including 120 activities) projects of the PSPLIB J120 (according to the result - with 30 replay (or 600 * 30 = 18000 run)) the average reliability of the model (the mean distance of the results from the lower bound) was 11.72 per cent which was at least three times better than the previously achieved result of 34.07 per cent (Hybrid Genetic

Algorithm (HGA)), (Valls, Ballestín and Quintanilla, 2008) which was considered as the best.

4.7 Extending the RCHCP

As with the above, there are some algorithms which solve RCHCP; Csébfalvi G. and Csébfalvi A.(2005) used an effective Branch-and-Bound tree, however it is restricted to small-medium problems. Csébfalvi G. (2007) developed a new model based on the SoS metaheuristic which produced good results for medium-large scale problems. Csébfalvi G., Eliezer, Láng and Levi (2008) are continuing this work. The paper presents an algorithm which deals with bi-objective resource-constrained project scheduling problems. It emphasizes dealing with many objectives like: minimal NPV, maximal robustness, and minimal hammock cost in addition to the minimal makespan. The algorithm was based on a conflict-repair SoS principle; by applying it, it was proven that many objectives can be dealt with at the same time. The conflict-repair technique that was used here inserts the widest conflict repair and solves the "hidden" and "visible" conflicts. Inserting the longest conflict-repair contributed to the robustness of the schedule created. Another result is the iterative process which was created; in the beginning the initial schedule is very non-optimal, however by inserting the conflict repairs (forward backward technique), it becomes better and better. This initial schedule is determined by tossing positive and negative random numbers. The activity should be to schedule as early (lately) as possible depending on the positive (negative) tossed value. However by using a forward- backward technique (as explained above), the makespan becomes shorter and the same is also true for other objectives. Eliezer and Csébfalvi G. (2009) developed a hybrid algorithm to deal with the hammock cost; the resource-constraint problem was converted to an unconstrained problem as a result of/by using the conflict repair technique. Firstly, all conflicts were repaired, after solving them, an unconstrained procedure was applied. The research on which this dissertation is based offers a very good solution for large scale problems by using an effective metaheuristic algorithm. The principal of the solution is based on SoS paradigm. Both of them present a new approach to the RCPSP that deals with

scheduling hammock and regular activities under constraints by using metaheuristic algorithms.

This new tool provides an answer to large-scale RCPSP and RCHCP by finding a heuristic and effective solution. Firstly, in the proposed primary-secondary criteria approach a project is characterized by its "best" schedule where “best” means a makespan minimal resource-constrained schedule for which the total hammock cost (the distance of the hammock hangers, multiplied by cost per time unit) is minimal. The immunity of activities, after inserting the conflict repairs, enables an achievement optimum of a secondary objective quite easily. Some other examples of secondary objectives (besides hammock cost) are minimal NPV, and maximal robustness (Csébfalvi et al., 2008a, 2008b).

This research suggests an innovative attitude based on the SoS; SoS helps us find the shortest makespan, and uses a thumb rule to repair all conflicts. The thumb-rule inserts the longest conflict repair; by applying it we get a better solution to the secondary criterion (the hammock cost). As a result of this combination two objectives are achieved: minimal makespan and minimal hammock cost.

Eliezer and Csébfalvi G. (2009) paper: ‘A Hybrid Method for the Resource-Constrained Project Scheduling Problem with Hammock Activities’ instigated the research for this dissertation.

Chapter 5: New model

5.1 The model principles

The presented new approach consists of N activities which represent N players – in the same manner as in the HS algorithm (Eliezer and Csébfalvi G., 2009) where every player represents an activity. As in the HS principal, every player tosses a sound out of his repertoire, and then tosses out a random number $-1 \leq \eta \leq 1$, in order to establish the order of playing (at the same time all players have the right to make a slight change in their chosen sound) – as explained in chapters 4.5 and 4.6., When η is negative (positive), it means that a player should play as late (early) as possible – in keeping with his/her time window. A conductor collects all the music of all of the N players (a new improvisation) and calculates the makespan of the project. He treats this improvisation as an initial solution which probably should be improved; his first task thereafter should be to schedule activities. The main target of the conductor is to systematically limit the players' freedom to decide, in order to reduce the melody's length (which is the project's makespan). Limiting the players' freedom is attained by solving an LP problem (4.9-4.11 above); its main objective is to establish a new schedule where most of the players there play as early as possible – according to their time windows.

The result of the optimization is a schedule (melody) which is used by the conductor to define the final starting (entering) order of the sounds (musicians). The conductor generates a soundless melody by taking the selected sounds one by one in the given order, and scheduling them at the earliest (latest) feasible start time. After that, by using the well-known forward-backward improvement (FBI) methods (Tormos and Lova, 2001) the conductor tries to improve the quality of the generated melody. The algorithm, in its initial phase, uses the latest finish times of all activities as a priority value, and gets the upper bound of the makespan. Following those values, all activities are ordered according to a list, which should be helpful in the scheduling process. In the iterative stage, the algorithm takes each of the activities out of the initial list one by one, and schedules each one in decreasing order of its finishing time: F_i when a starting time: S_i is assigned to each activity: i. At this stage the algorithm tries to shift each activity to its latest starting time: LST_i according to its forward free slack. The activities are rescheduled in decreasing order according to their finishing times. After

the first phase a backward scheduling process begins, while at the same time shifting each activity to its EST_i following its backward free slack. The activities are rescheduled in increasing order according to their starting times. This procedure is useful to reduce the "long break" between activities (see figures 14, 15, 16, 18, 21 below). Similar techniques were proposed by Li and Willis (1992) and used by özdamar and Ulusoy (1996). The forward-backward iterative procedure is a general technique that improves a feasible project schedule and minimizes the makespan of the project. The forward-backward passes are based on the concepts of the forward and backward free slack of the activities. The forward-backward free slack of an activity in a feasible schedule is the amount of time that the activity can be shifted right of left, allowing the remaining activities to start at their scheduled times while taking into consideration the resource constraints. By using this procedure every iteration produces a schedule which is then compared to the others, the conductor then memorizes the shortest feasible melody found till that time, and gets rid of the worst one in the repertoire. By using this method the repertoire becomes better and better with each step.

The conflict repairing version of the "SoS" was detailed above (chapter 4.7); it ensures that a robust, free of conflict, schedule is drawn up.

Clearly when we introduce a secondary criterion (e.g. hammock cost), for which the aesthetic value is a function of the starting times, the freedom of the players totally disappears. When a secondary objective is given, then in the improvisation phase the conductor firstly selects a promising makespan set, and later, from the chosen set, selects a schedule with a promising secondary criterion value.

In a "SoS algorithm", at every iteration, if a better solution is found it replaces the worst known one in the current repertoire, so that as a conclusion the quality of the population becomes better and better. According to the progress of the searching process, the size of the makespan minimal subset of the population increases. As the size of the makespan minimal subset becomes larger, the greater the chance of getting a good solution for the secondary criterion – which in our case is: the RCHCP.

5.2 The NP-Hard RCHC problem

In order to model hammock activities in projects, we consider the following resource constrained project-scheduling problem: a single project consists of N real activities

$i \in \{1,2,\ldots\ldots N\}$ with a non-preemptable duration of D_i periods. Furthermore, activity *i=0 (i=N+1)* and is defined as the unique dummy source (sink). The activities are interrelated by precedence and resource constraints. Precedence constraints prevent an activity from being started before all its predecessors are finished.

Let:

$$PS = \{i \to j, i \neq j, i \in \{0,1,\ldots.N\}, j \in \{1,2,\ldots\ldots N+1\}, FS_{ij} = 0\} \quad (1)$$

denote the set of immediate predecessor-successor relations. Resource constraints arise as follows: in order to be a processed activity *i* require R_{ir} units of resource type $r \in \{1,2,\ldots..R\}$ during every period of its duration. Since resource *r*, $r \in \{1,2,\ldots..R\}$ is only available with the constant period availability of R_r units for each period, activities might not be scheduled at their earliest (network-feasible) start time but later.

Let $\overline{T}$ denote the resource-constrained project's makespan (or its upper bound) and fix the position of the unique dummy sink in period $\overline{T}+1$.

Remark: A project as a whole is considered a hammock activity. It is a degenerate case, the upper bound of the project's makespan $\overline{T}$ is considered to be an initial hammock duration i.e. the first iteration of the algorithm (5.2)-(5.13) below, is to find the project's makespan from a standard RCPSP problem. Then from this schedule, with an upper bound makespan of: $\overline{T}$ the algorithm finds a schedule with the "best" hammock cost.

Let H denote the number of hammock activities. A hammock activity:

$\vec{H}_h, h \in \{1,2,\ldots\ldots H\}$, can be represented by a dummy activity pair:$\{\overrightarrow{H_h}, \overleftarrow{H_h}\}$ with zero duration (hammock hangers), and a membership function: $M_h(i)$ to identify the hammock members. Following the constraint (5.4) below, the membership function$M_h(i)$ is a Boolean one, which returns the true value when an activity *i* is found to be the inner activity of the hammock.

The NP-hard RCHC problem can be written as follows:

$$min\left[\sum_{h=1}^{H} C_h \cdot \vec{H}_h\right] \quad (5.2)$$

$$\overleftarrow{H}_h = \overline{H}_h - \vec{H}_h + 1, h \in \{1,2,\ldots..H\} \quad (5.3)$$

$$H_h = \{i|M_h(i) = true, i \in \{1,2,\ldots.,N\}\}, |H_h| \geq 2, h \in \{1,2,\ldots.,H\} \quad (5.4)$$

$$\vec{H}_h \leq X_i, i \in H_h, h \in \{1,2,\ldots.,H\} \quad (5.5)$$

$$X_i \leq \overline{H}_h, i \in H_h, h \in \{1,2,\ldots.,H\} \quad (5.6)$$

$$X_i + D_i \leq X_j, i \to j \in PS \quad (5.7)$$

$$X_{N+1} = \overline{T} + 1 \quad (5.8)$$

$$X_i = \sum_{t \in T_i} X_{it} \cdot t, T_i = \{EST_i, EST_i + 1, \ldots., LST_i\}, i \in \{1,2,\ldots.N\} \quad (5.9)$$

$$\sum_{t \in T_i} X_{it}, X_{it} = 1, X_{it} \in \{0,1\}, i \in \{1,2,\ldots.N\} \quad (5.10)$$

$$A_t = \{i|X_i \leq t \leq X_i + D_i, i \in \{1,2,\ldots.N\}\}, t \in \{1,2,\ldots.T\} \quad (5.11)$$

$$U_{tr} = \sum_{i \in A_t} R_{ir}, \ t \in \{1,2,\ldots.T\}, r \in \{1,2,\ldots.R\} \quad (5.12)$$

$$U_{tr} \leq R_r t \in \{1,2,\ldots.T\}, r \in \{1,2,\ldots.R\} \quad (5.13)$$

Objective (5.2) minimizes the total hammock cost subject to the precedence relations and the limited resource availabilities. Constraints (5.3-5.6) define the distance of the hammock hangers, identify the hammock members, and describe the relations between the hammock hangers and members for each hammock activity h, $h \in \{1,2,\ldots\ldots H\}$. Constraints (5.7) represent the precedence relations. Constraint (5.8) defines the deadline (latest completion time) of the project. Constraints (5.9-5.10) define the activity starting times in the function of the binary indicator variables, and ensure that each activity has exactly one starting time within its time window $T_i = \{EST_i, EST_i + 1, \ldots\ldots LST_i\}$ where $\mathrm{EST_i}(\mathrm{LST_i})$ is the early (late) starting time for activity i, $i \in \{1,2,\ldots\ldots N\}$according to the precedence constraints and the latest project completion time $\overline{T}$. Constraints (5.11-5.13) ensure that resources allocated to activities at any time during the project do not exceed the resources' availability.

The unconstrained hammock cost problem (HCP) can be formulated as a LP problem therefore it can be solved in polynomial time. Although the heuristic above is resource constrained, the analysis of the hammock is carried out in the unconstrained case. The reason is: by using the metaheuristics of SoS that was developed by Csébfalvi et al. (2008a, 2008b), and Eliezer and Csébfalvi G. (2009) and which is based on the forbidden set principle, all resource conflicts are solved in advance, as a result we get a robust (flexible) schedule in such a way that all activities can be moved without a negative effect to its feasibility. The following algorithm can be applied on polynomial time which facilitates matters, and a hammock activity can be analyzed by using the unconstrained case below:

$$HC^* = min\left[\sum_{h=1}^{H} C_h \cdot \overleftrightarrow{H}_h\right] \tag{5.14}$$

$$\overleftrightarrow{H}_h = \overleftarrow{H}_h - \overrightarrow{H}_h + 1, h \in \{1,2,\ldots..H\} \tag{5.15}$$

$$H_h = \{i|M_h(i) = true, i \in \{1,2,\ldots.,N\}\}, |H_h| \geq 2, h \in \{1,2,\ldots.,H\} \tag{5.16}$$

$$\overrightarrow{H}_h \leq X_i, i \in H_h, h \in \{1,2,\ldots.,H\} \tag{5.17}$$

$$X_i \leq \overleftarrow{H}_h, i \in H_h, h \in \{1,2,\ldots.,H\} \tag{5.18}$$

$$X_i + D_i \leq X_j, i \rightarrow j \in PS \tag{5.19}$$

$$X_{N+1} = \overline{T} + 1 \tag{5.20}$$

$$EST_i \leq X_i \leq LST_i \tag{5.21}$$

The model (5.14)-(5.21), is similar to (5.2)-(5.13) however there are no resource constraints at all.

5.3 The unconstrained Hammock Cost Problem (HCP)

In the unconstrained case the earliest (latest) schedule of the HCP can be formulated as a simple LP problem. This is a formal mathematical model – which is similar to (5.14-5.21):

$$\min(max)\left[\sum_{i=1}^{N} X_i\right] = \underline{X}^* = \overline{X}^* \tag{5.22}$$

$$\overleftrightarrow{H}_h = \overleftarrow{H}_h - \vec{H}_h + 1, h \in \{1,2,\ldots..H\} \tag{5.23}$$

$$H_h = \{i|M_h(i) = true, i \in \{1,2,\ldots.,N\}\}, |H_h| \geq 2, h \in \{1,2,\ldots.,H\} \tag{5.24}$$

$$\vec{H}_h \leq X_i, i \in H_h, h \in \{1,2,\ldots.,H\} \tag{5.25}$$

$$X_i \leq \overleftarrow{H}_h, i \in H_h, h \in \{1,2,\ldots.,H\} \tag{5.26}$$

$$X_i + D_i \leq X_j, i \to j \in PS \tag{5.27}$$

$$X_{N+1} = \overline{T} + 1 \tag{5.28}$$

$$EST_i \leq X_i \leq LST_i \tag{5.29}$$

$$\overleftrightarrow{H}_h = \overleftrightarrow{H}_h^{\,*}, h \in \{1,2,\ldots.,H\} \tag{5.30}$$

Objective (5.22) minimizes (maximizes) the starting times by using the precedence relations and the given deadline's artificial objective subject.

Constraints (5.23-5.29) are the same as (5.3-5.11). Constraint (5.30) fixes the duration of each hammock activity according to its optimal duration i.e. after obtaining the minimal makespan, the hammock hangers should be fixed according to it.

***Comment no.1**: The heuristic produces (5.2-5.13) resource-feasible schedules (where every possible activity's movement is resource-feasible). In order to investigate its hammock cost we have to use only the unconstrained model. The early (late) model with fixed hammock durations is very important from a managerial point of view, because it describes the flexibility of the hammock activities (a hammock activity should move without affecting the resource feasibility at any time period).*

<u>Comment no. 2:</u> The schedules which are generated by the heuristics are resource-feasible, and differ only in the project's makespan and the hammocks' durations. Improvement means a better schedule for primary (project makespan) and secondary (hammock cost) criteria.

In order to illustrate the RCHCP, we present a project instance with 30 non-dummy activities that require four renewable resources. More specifically, the project instance J30-1-1 will be presented, which is the first instance of the "easiest" single-mode benchmark set J30 of PSPLIB produced by Kolisch and Sprecher (1996). Figures 17, 18 hereunder represent a precedence-feasible early start schedule, for instance: J30-1-1 is shown with one ($H = 1, C_1 = 1$) randomly generated hammock activity. This means that this project contains one hammock activity and a cost of 1- uniformly for each hammock member. As a result the hammock's cost is a minimal hammock's time-length:

$$\mathrm{M}_1(\mathrm{i}) = RandomUniform(0,1) < 0.5, i \in \{1,2, \dots \dots N\} \qquad (5.31)$$

The activities are represented by bars and the precedence relations by lines. The real activities are identified in a form. The hammock members are represented by dark grey bars. The duration of the hammock activity is represented by a dark grey bar. The unique dummy source (sink) is represented by the $>$ $(<)$ symbol. In the resource usage histograms the over-utilisation is represented by dark grey bars. The early schedule is not resource feasible $\overleftrightarrow{H_1} = 36$.

In figures 19 and 20 the "best" schedule is shown, where "best" means a makespan minimal ($\overline{T} = 44$) resource-constrained schedule for which the hammock duration (cost) is minimal ($\overleftrightarrow{H_1} = 44$.) according to the generated hammock structure.

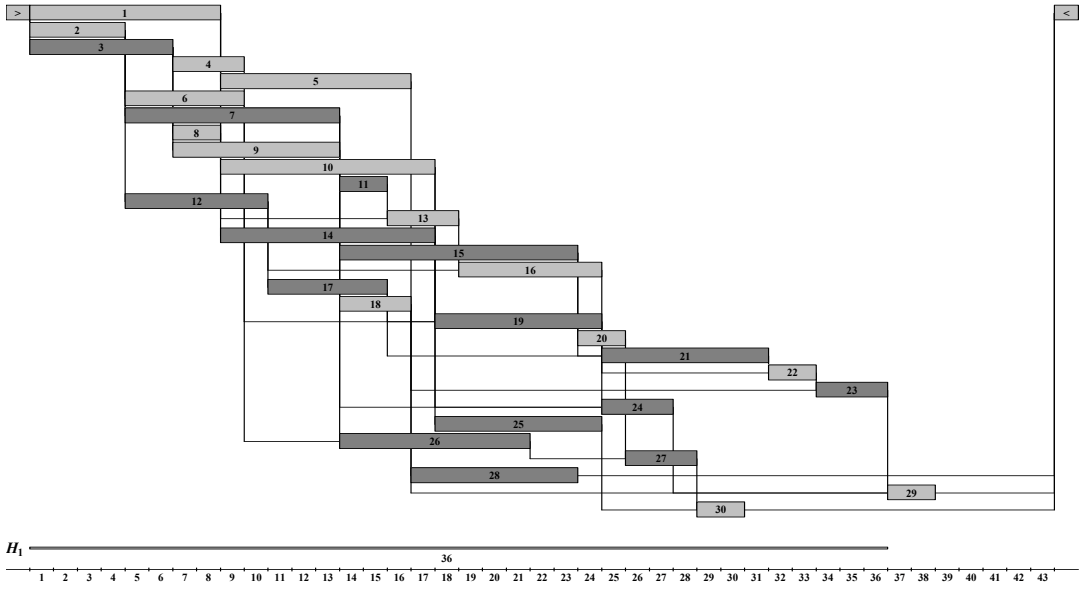

Figure 17: An early schedule of J30MM-1-1 (Eliezer and Csébfalvi G., 2009)

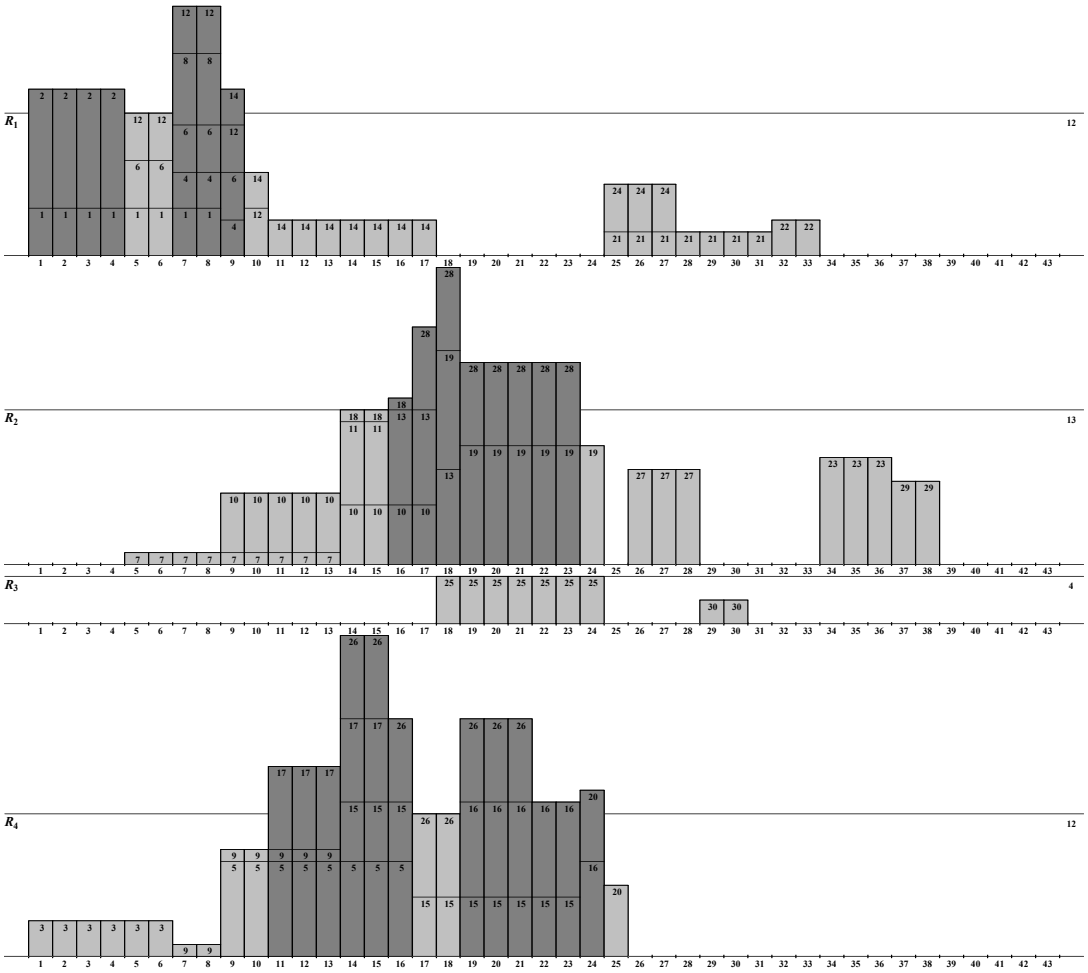

Figure 18: Resource profiles in the early schedule (Eliezer and Csébfalvi G., 2009)

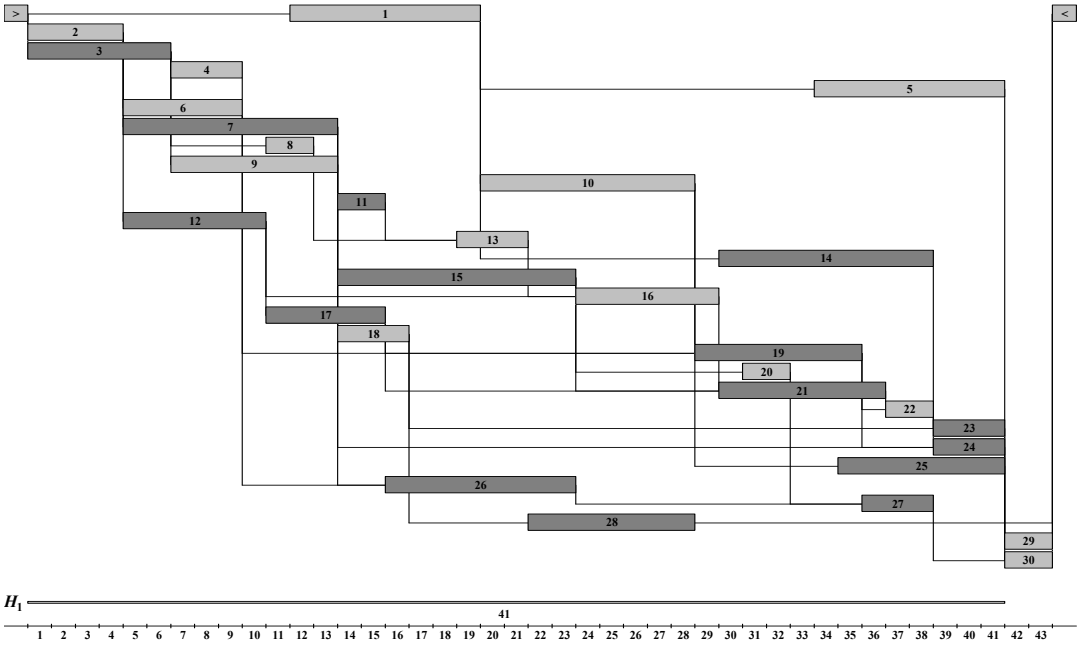

Figure 19: The optimal solution of J30MM-1-1(Eliezer and Csébfalvi G., 2009)

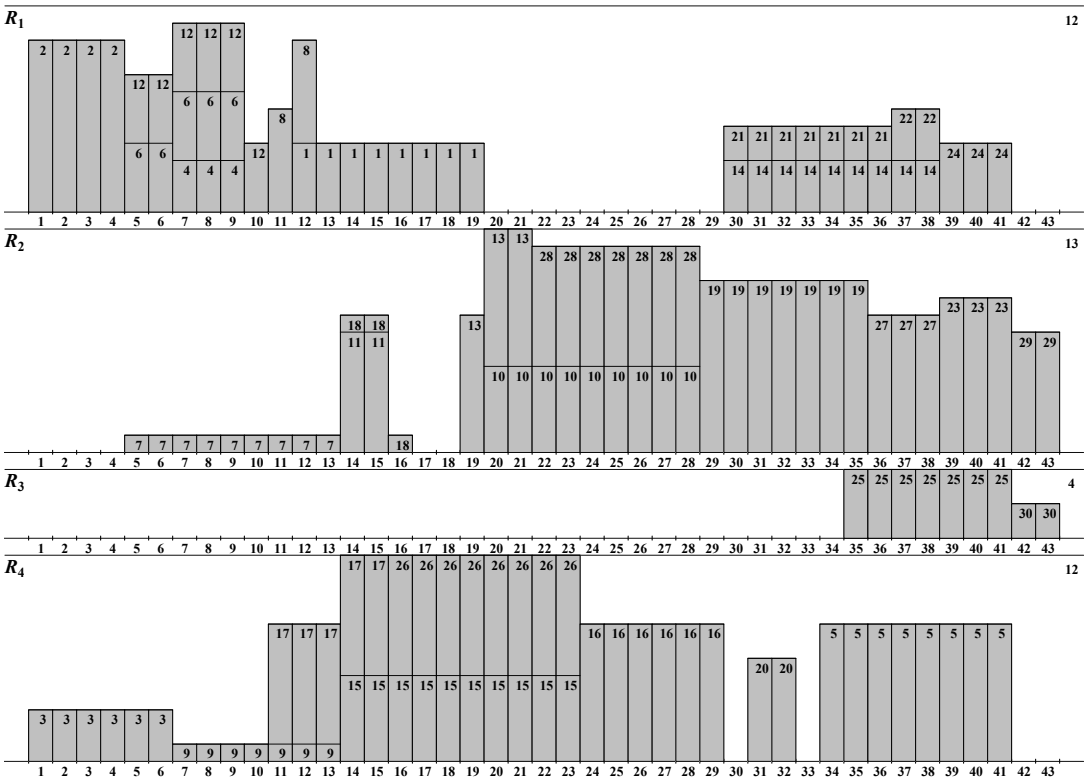

Figure 20: Resource profiles of the "best" schedule (Eliezer and Csébfalvi G., 2009)

The next step after fixing the position of the finishing milestone according to the optimal makespan and the hammock durations according to the optimal durations and inserting the optimal conflict repairing relations is smoothing out the resource usage histograms on the set of possible activity movements.. In the presented improved algorithm the harmony search is combined with a resource levelling-assigning procedure based on a MILP formulation developed by Csébfalvi and Konstantinidis (2002). To minimize the computation cost, the conductor calls the procedure only when the currently best solution is changing, and solves the resource levelling problem on the "repaired" resource-feasible solution set fixing only the distance of the hammock hangers according to local the optimal solution. In other words, a hammock activity can be moved as a whole in the levelling process. After fixing the length of the hammock activities, we have some freedom to smooth out the resource profiles on the "repaired" local solution set. (Eliezer O. and Levi R. (2010))/

The improved algorithm exploits the fact, that the resource leveling and the dedicated resource assigning problem can be managed separately, because the given resource leveling measure is invariant to the permutation of the resource units and the dedicated resource assignment is unable to change this measure value. The applied resource leveling procedure, preferring the continuous work, minimizes the number of starting-restarting events of resource units on the set of resource-feasible activity movements fixing the makespan and the length of the hammock activities.

From a managerial point of view, leveling procedure is quite important, since is ensures fluent working and usage of resources for a long term.

Let $T_i = \{\underline{X}_i, \underline{X}_i + 1, \ldots\ldots, \overline{X}_i\}$ denote the time window of activity i after conflict repairing, where $\underline{X}_i$ $(\overline{X}_i)$ is the early (late) starting time for activity i according to the fixed modes and makespan. The locally best hammock duration is denoted by $\widetilde{H}_h^*$ where $h \in \{1,2, \ldots\ldots H\}$.

Herein the MILP model of levelling :

$$\min LM = \sum_{r=1}^{R} \sum_{t=1}^{T} CU_{rt}^{+} = LM^* \tag{5.32}$$

$$\overrightarrow{H_h} = \overleftarrow{H_h}^*, h \in \{1,2,\ldots..H\} \tag{5.33}$$

$$H_h = \{i|M_h(i) = true, i \in \{1,2,\ldots.,N\}\}, |H_h| \geq 2, h \in \{1,2,\ldots.,H\} \tag{5.34}$$

$$\overrightarrow{H_h} \leq X_i, i \in H_h, h \in \{1,2,\ldots.,H\} \tag{5.35}$$

$$X_i \leq \overleftarrow{H_h}, i \in H_h, h \in \{1,2,\ldots.,H\} \tag{5.36}$$

$$X_i + D_i \leq X_j, i \to j \in PS \tag{5.37}$$

$$X_i = \sum_{t \in T_i} X_{it} \cdot t, \;\; T_i = \{\underline{X_i}, \underline{X_i} + 1, \ldots..,\overline{X_i}\}, i \in \{1,2,\ldots N\} \tag{5.38}$$

$$\sum_{t \in T_i} X_{it} = 1, \;\; X_{it} \in \{0,1\}, \quad i \in \{1,2,\ldots N\} \tag{5.39}$$

$$A_t = \{i|X_i \leq t \leq X_i + D_i, i \in \{1,2,\ldots N\}\}, t \in \{1,2,\ldots.T\} \tag{5.40}$$

$$U_{tr} - CU_{tr}^{+} + CU_{tr}^{-} = U_{t-1,r}\,, t \in \{1,2,\ldots.T\}, r \in \{1,2,\ldots.R\} \tag{5.41}$$

$$U_{1r} - CU_{1r}^{+} = 0, r \in \{1,2,\ldots.R\}$$

$$U_{tr} \leq R_r, t \in \{1,2,\ldots.T\}, r \in \{1,2,\ldots.R\} \tag{5.42}$$

The objective (5.32) minimize the total amount of resource using *r*, at time *t*, while $r \in \{1,2,\ldots.R\}$ and $t \in \{1,2,\ldots.T\}$. (5.33) determines the fact that the hammock is scheduled in its optimal position, (5.34-5.36) defines the hammock's inner activities, (5.37) determines the precedence relations , (5.38-5.40) define exactly one starting point of every activity, following its time window, (5.41) and (5.42) define the smoothness of usage resources, in such a way that the heights difference of each adjacent histograms should be minimized.

The dedicated resource assigning model can be formulated very easily, because the resource leveling procedure exactly defines the starting time of the activities, therefore we have to prescribe only that each resource demand unit have to be serviced, and each resource unit have to be assigned at most one resource

demand unit in each period. The problem is a variant of the so-called unit execution time (UET) model, so it can be solved in polynomial time.

Regarding the example of figure no. 17 above, the author describe hereunder (figures 21,22) the optimal solution. The latter contains a minimal makespan with minimal hammock cost and smooth usage of resources:

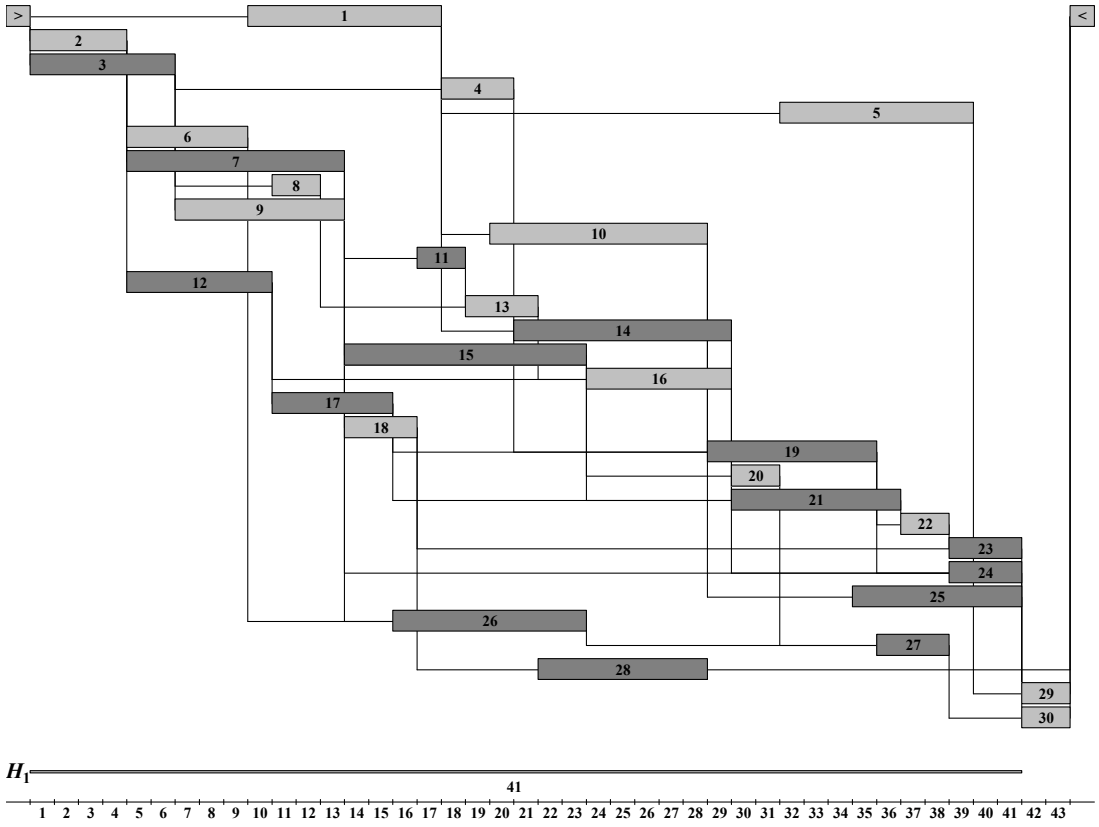

Figure 21: The "best" schedule after levelling (Eliezer and Levi, 2010).

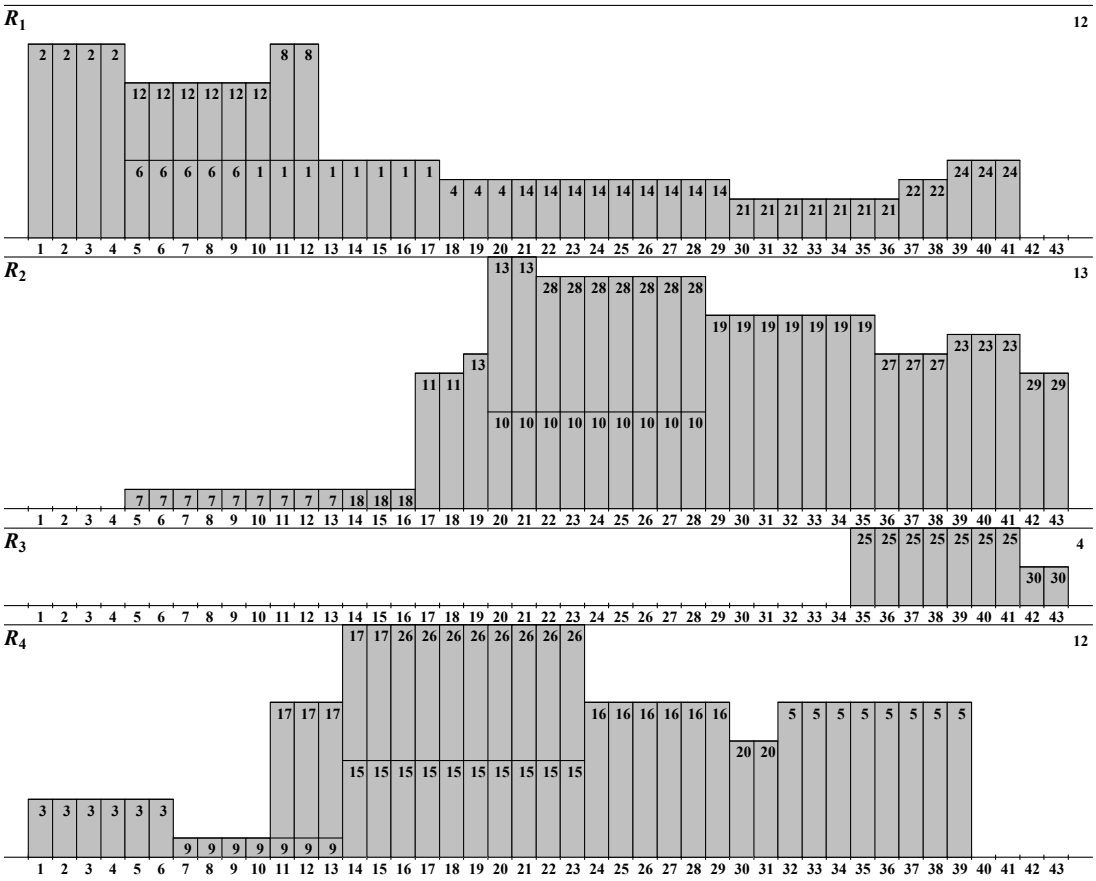

Figure 22: Resource profiles of the "best" schedule after leveling (Eliezer and Levi, 2010).

The applied ternary criterion of levelling (Konstantinidis (2002)) can be replaced by any other criterion which is able to rearrange the schedule according to a well-specified managerial goal without affecting the primary (secondary) optimality. The introduced ternary MILP approach can be replaced by a problem-specific heuristic (metaheuristic) algorithm, which generates resource-feasible activity movements and maintains the ternary criterion time to time.

5.4 The Algorithm

As mentioned above, after achieving an initial non-feasible early schedule, the new algorithm should be applied in order to solve the NP-hard RCHCP problem 5.2-5.13.

The first step of the algorithm is to use the HS metaheuristics as explained above. As previously mentioned, the conductor controls the degree of freedom of the players, and

as a result an initial schedule (melody) is obtained. This schedule is used by the conductor to define the final starting (entering) order of the sounds (musicians). The conductor generates a soundless melody by taking the selected sounds one by one in the given order, and scheduling them at the earliest (latest) feasible start time. Thereafter by using the well-known FBI methods (Tormos and Lova, 2001) the conductor tries to improve the quality of the generated melody. The conductor memorizes the shortest feasible melody found thus far.

As mentioned above the conflict repairing version of the "SoS" algorithm is based on the forbidden set concept. In the conflict repairing version the primary variables are conflict repairing relations, whereby a solution will be a makespan minimal resource-feasible solution set in which every movable activity can be shifted without affecting the resource feasibility. In the traditional "time oriented" model the primary variables are starting times, therefore an activity shift may be able to destroy the resource feasibility. The makespan minimal solutions of the conflict repairing model are immune to the activity movements, so we can introduce a not necessarily regular secondary performance measure to select the "best" makespan minimal resource feasible solution from the generated solution sets. In the "SoS" algorithm, according to the applied replacement strategy (whenever the algorithm obtains a solution superior to the worst solution of the current population, the worst solution will be replaced by the better one), the quality of the population is systematically improved. According to the progress of the searching process, the size of the makespan minimal subset of the population increases. As the makespan minimal subset size becomes larger, the chance of getting a good solution for the secondary criterion is greater. It is well-known that the crucial point of the conflict repairing model is the forbidden set computation. In the "SoS" algorithm the conductor uses a simple (but fast and effective) "thumb rule" to decrease the time requirement for the forbidden set computation. In the forward-backward list scheduling process the conductor (without explicit forbidden set computation) inserts a precedence relation $i \rightarrow j$ between an already scheduled activity i and the currently scheduled activity j whenever they are connected without lag ($S_i + D_i = S_j$) . The result will be a schedule without "visible" conflicts. Then the conductor (in exactly one step) repairs all of the hidden (invisible) conflicts, by always inserting the "best" conflict repairing relation for each forbidden set. In this context "best" means a relation $i \rightarrow j$ between two forbidden set members for which the lag ($S_i - S_j - D_i$) is maximal. In the language of music the result of the conflict repairing process will be a robust (flexible)

"SoS" melody in which the musicians have some freedom to enter the performance without affecting the aesthetic value of the composition.

Theoretically the resource-constrained case can be formulated as a MILP problem which can be solved for small-scale projects within a reasonable timeframe. It is important to note that in the HS algorithm, the RCHC measure of a "repaired" schedule can be evaluated in polynomial time. This evaluation is done by using the presented unconstrained LP formulation.

Literature dealing with research on the RCPSP problem and hammock cost is lacking as most companies keep their IT and logistics policies "top secret". Therefore there is no exposure to applied (and very necessary) data; the freezing machine described in the underground tunnel example is the only material on the subject which is described in the literature as an example which deals with realistic costs.

The new model (and metaheuristics) is the first "open" model that deals with the RCPSP problem and hammock cost of medium-large scale problems. This model dictates a new methodology for handling managerial problems connected to the hammock's cost; by using this model, we attain satisfactory levelling of resources, or hammock float (slack). This concept is illustrated in figure 23 hereunder:

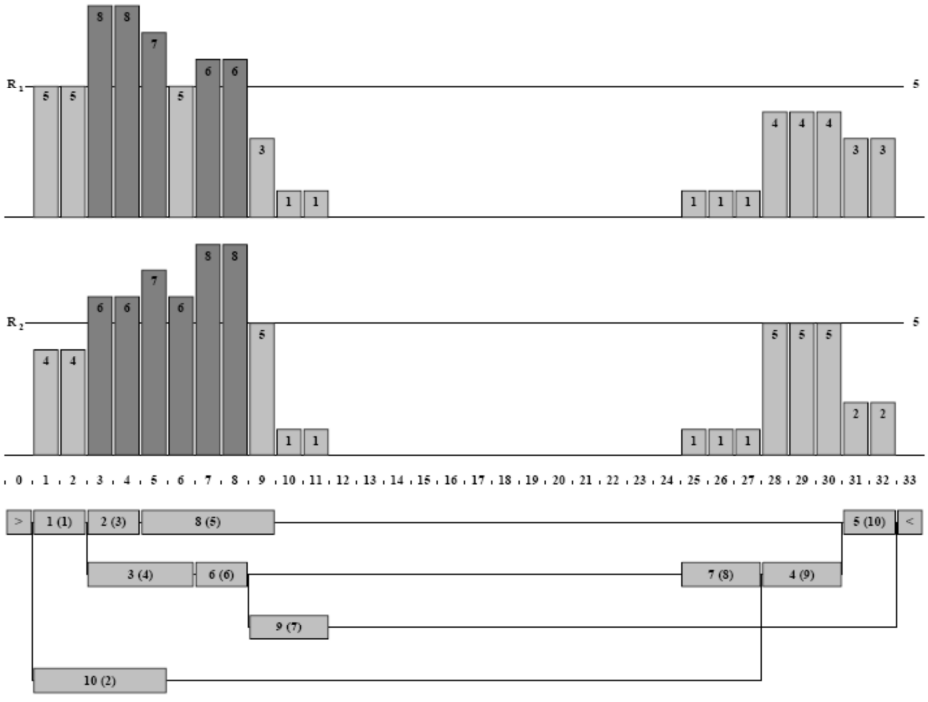

Figure 23: A random non-feasible scheduling (Csébfalvi, et al., 2008b)

Figure 23 represents a non-feasible schedule which is created as a free improvisation of the players. This schedule describes the initial state of the repertoire. This non-feasible schedule contains conflicts which cause a violation of resources: $2 \rightarrow 3, 3 \rightarrow 8, 8 \rightarrow 6$ and $9 \rightarrow 4$ and which are visible. The makespan is 32 time-units. After applying the "thumb rule" explained above, all the visible conflicts are repaired, and the makespan is reduced to 24 time-units as seen hereunder (figure 24):

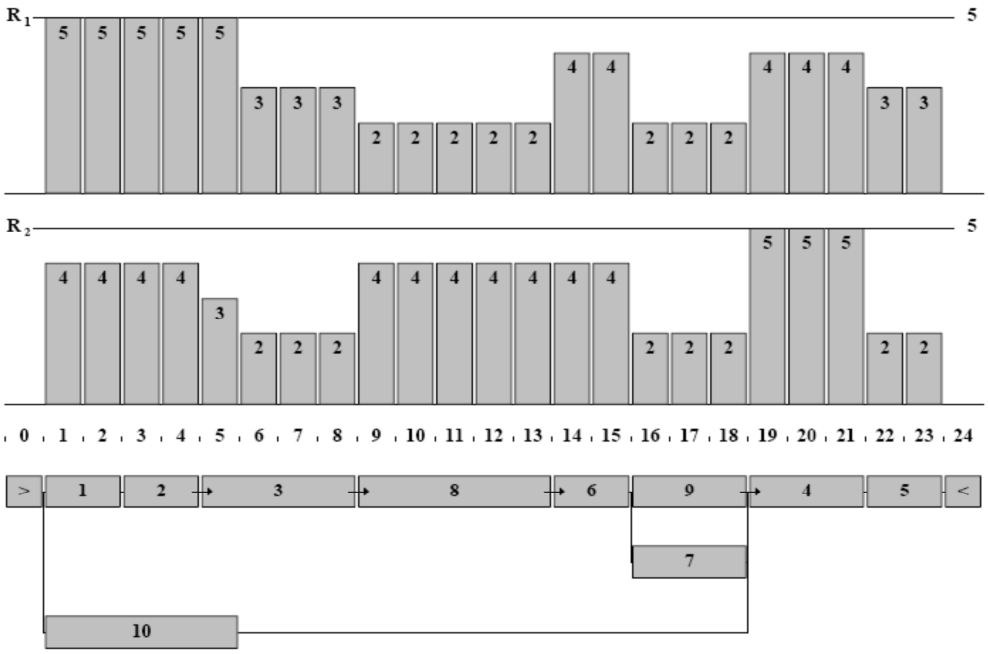

Figure 24: A feasible scheduling after inserting conflict repairs(Csébfalvi, et al., 2008b)

However with regard to the feasibility of the schedule above, activity no. 10 has a wide slack, so there is a possibility of a resource violation if activity no. 10 should be shifted to the right in such a way that it should be scheduled parallel to activity no. 6. In order to prevent this, a hidden conflict repair $10 \rightarrow 6$ should be the most effective answer as seen hereunder:

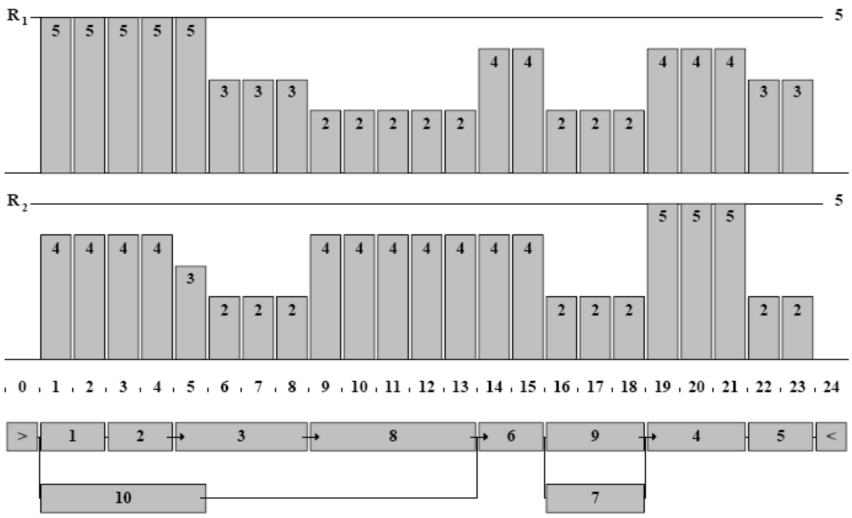

Figure 25: The network after entering the 10 → 6 hidden conflict repair (Csébfalvi, et al., 2008b)

Figures 24 and 25 are better solutions to the ILP problem, so the repertoire is extended by adding a new, feasible (and better) solution at each step of the algorithm. Figure 24 describes a better solution than the initial one, however the solution described by figure 25, is more flexible than 22.

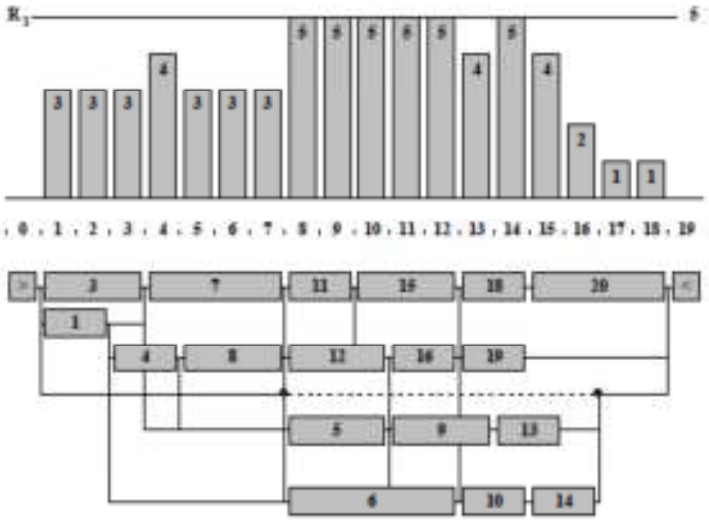

Figure 26: The optimal solution out of the repertoire (Csébfalvi, et al., 2008b)

In figure 26 the optimal solution out of the algorithm has a makespan of 19 and has a minimal hammock cost.

5.5 Conclusions

In this chapter a new algorithm that solves the RCHCP of large-scaled projects was introduced. The model is a MILP model which is a NP-hard problem. The steps of the algorithm are:

1. In the initial step, an early, non-feasible schedule is generated. The model is similar to figure 14: a non-feasible solution, which contains time gaps between activities and conflicts. However, it is an initial state of the repertoire. By generating this schedule, the order of activities which should be scheduled, and an upper bound of the schedule is calculated (the $\overline{T}$ in the MILP of 5.2-5.13 and 5.14-5.21**).** The motivation here is to minimize the makespan and at the same time obtain a feasible solution.
2. The HS algorithm is applied. Thereafter the conductor takes each activity, follows the initial order that was determined in step no. 1, and tries to schedule it according to its time window which is based on the principle of HS. The particular schedule of a single activity is arranged by tossing out a random number between [-1,+1]; then when a better schedule is achieved, it replaces the worst one in the repertoire. All conflicts are repaired by using the procedure of the forbidden set, as explained in section 4.4.
3. The SoS algorithm is now used. The conductor uses a simple (and effective) "thumb-rule" to decrease the time requirements of the forbidden set computation. He then uses a backward/forward list scheduling process by inserting a precedence relation $i \rightarrow j$ between an already scheduled activity i and a currently scheduled activity j whenever there is no lag

between them ($S_j = S_i + D_i$). The forward/backward technique is based on Tormos and Lova's (2001) FBI, (section 5.1). The shifting process depends on a stochastic variable which determines the exact shifting rate and timing (see (4.11)-(4.13) model). While using this variable, the freedom of activities is reduced gradually (the conductor controls the freedom of players in the orchestra). As a result all "visible" conflicts are repaired.

4. The conductor then repairs all hidden conflicts by inserting the longest conflict repair, meaning a relation: $i \rightarrow j$ between forbidden set members for which the lag $(S_j - S_i - D_i)$ is maximal. As a result we get a flexible (robust) schedule where every activity can be moved along its free slack without harming the feasibility of the solution.

5. Instead of dealing with the NP-hard problem of 5.2-5.13, the author deals with the unconstrained model of 5.14-5.21 which can be solved in polynomial time.

In conclusion, the algorithm above creates a new methodology which addresses RCPSP and large-scale hammock activities.

There are some new research directions that should be examined:

- The minimization of HC, as a second criterion to a RCPSP, can be replaced by other objectives for instance: the leveling cost (if we inspire to "smooth" the resources quite uniformly between activities).
- Calculation of the optimal number of hammocks in large-scale projects, and checking to see if by dividing a project to hammocks is indeed improving its makespan.

Chapter 6: Computational results

We tested our approach on a benchmark set: J30 generated with ProGen. In this set the number of non-dummy activities is 30. The projects in this test set consist of 30 activities that require four renewable resources. These instances were generated under a full factorial experimental design with the following three independent problem parameters: network complexity, resource factor, and resource strength. The set J30 consists of 480 instances (10 replications of 48 treatments). The hammock activities were generated randomly but in a reproducible form. Therefore the benchmark set with its' optimal solutions can be used for testing the quality of exact and heuristic algorithms that will be developed in the future; in order to provide this we used the ***Rand 2000*** random number generator (Microsoft Q86523) with a starting value of 0.5 for each instance:

$$M_1(i) = Rand2000 < 0.5, i \in \{1,2,\ldots..N\} \qquad (6.1)$$

A state-of-the-art callable MILP solver (CPLEX 8.1) was used in default mode with a 900 sec time limit to generate the exact solutions. Within the given time limit 380 instances were solved to optimality by CPLEX. The proposed algorithm was programmed in Visual C++® 6.0 and Visual Basic® 6.0. The computational results were obtained by running the algorithm on a 1.8 GHz Pentium IV IBM PC with 256 MB of memory under the Microsoft Windows XP® operating system. The results of the experiments are presented in tables 3-4. The quality of a solution is measured by the percentage gap of the primary criterion value with respect to the optimal primary criterion value. The percentage gap for a secondary criterion was estimated from the primary gap free solutions.

Table 3: Solution times using CPLEX 8.1 (Eliezer and Csébfalvi, 2009)

Solved Instances	Maximum Time	Minimum Time	Standard Deviation	Average
380	678.00	0.00	97.23	27.21

Table 4: SoS results (Eliezer and Csébfalvi G., 2009)

Solution Time		Secondary Criteria (Gap %)	Primary Criteria (Gap %)	Iterations
μ (sec)	σ (sec)			
0.132	**0.049**	**3.17**	**0.14**	**100**
1.273	**0.230**	**2.74**	**0.06**	**1000**

Further to table no. 3, after applying the algorithm to 380 instances (out of 480 instances of J30 problems), it can be concluded that on average it terminates after 27.21 seconds when the maximal termination time is 678 seconds.

The algorithm was examined after 100 and 1000 iterations. It was concluded from table no. 4, that the gap in the primary criteria (the minimal makespan) is between 0.06% and 0.14%. This means that the gap between the optimal and heuristic makespans is between 0.06% to 0.14%; however the results are mediocre when dealing with the secondary criteria (hammock cost), the gap is between 2.74% and 3.17% (dependent on number of iterations); however it cannot be calculated exactly, so it is estimated out of the primary gap free solutions.

As a conclusion the results are satisfactory. The average time for executing 1000 iterations is 1.273 seconds, with an approximate gap of 2.74% calculated for the secondary criteria (hammock cost). However, in order to emphasize the advantage of SoS over other algorithms a comparison is required. Table no. 5 shows that a SoS was

compared to the Valls et al. (2008) hybrid algorithm, and was found to be nearly three-times better for: J60.

Table 5: The Valls et al. hybrid algorithm versus "SoS" results for J60 (Csébfalvi G. 2007, Valls et al. , 2008)

Set		J60				
		DM – 1.75 (July 5, 2007)				
Iterations		100	500	1000	5000	50000
Valls et al.		-	-	**11.56**	**11.10**	**10.73**
	Average Distance (%)	**4.71**	**4.20**	**4.02**	-	-
	Standard Deviation	0.03	0.05	0.01	-	-
Sounds of Silence	Average Time (sec)	**0.088**	**0.411**	**0.805**	-	-
	Standard Deviation	0.002	0.004	0.006	-	-
	Independent Runs	**10**	**10**	**10**	-	-

Chapter 7: New results

7.1 Preface

The motivation for this research came from the work done by Harhalakis (1990) who formalized a definition for hammock activity. Harhalakis's work focused on the unconstrained case, and it motivated the author to develop a new tool to cope with constrained cases and medium-large scale problems. Harhalakis presented an abstract method which can be replaced by a realistic one: RCHCP, whereby its model 5.2-5.13 is an NP-hard model (discussed above), so a near-optimal solution is reached by solving a heuristic algorithm. The heuristic algorithm is a hybrid and was applied by using a metaheuristic of SoS, which is based on a forbidden set principle. At the beginning of the algorithm an initial and non-feasible schedule is generated that determines the activities' scheduling order. Iteratively, while treating the conflicts, and at each stage of the algorithm, the activities are shifted to the right or to the left depending on the forward/backward process. A forbidden set principle includes solving all visible and hidden conflicts by inserting the longest conflict repair (as a thumb-rule) between two

activities$i \rightarrow j$, where each of them is a member of a forbidden set. A flexible schedule results where every activity can be moved without negatively affecting the feasibility of the solution; from a managerial point of view it can be treated by the unconstrained model (5.22-5.30, the unconstrained formal model), which is solved by a polynomial algorithm.

Since Harhalakis dealt with the unconstrained case, it can be concluded that his problems can be solved by a LP algorithm (5.22-5.30 above) which is a polynomial algorithm. During this process, we deal with two main problems:

1) Minimization of hammock duration – a specific case of the hammock's cost.
2) The criticality of the hammock itself ─ at the hammock level. When we fix the hammock's hangers (after getting the optimal makespan), we can calculate the hammock's cost and estimate its robustness.

When dealing with hammock activities, the author considered hammock activity as a ***red activity***. A red activity is considered to be unique, and/or quite expensive, and which therefore needs special equipment and budget. Since this red activity consumes quite a significant part of the project's budget, managers are inspired to minimize its duration with regard to resource constraints. In light of this, we have to treat two important issues:

1) Finding the minimization of a project's makespan ─ this is the main objective when solving the problem. Once this makespan is known we fix the terminal and sink of the whole project, and take into consideration the resource constraints (like manpower, money etc).
2) Minimization of the HC under resource constraints while dealing with the criticality of the hammock. At this point: the hammock's hangers should be fixed according to the optimal makespan; hammock members should be ordered at their optimal operating times; and the total project's cost will be reduced. At this point resource constraints can be smoothed by applying the RLP procedure of Konstantinidis (2002).

7.2 The structure of the research

- The first part of the dissertation surveys known methods of dealing with RCPSP and RCHCP algorithms like: MILP and Branch and Bound algorithms which solve small-medium scale problems. Accordingly the author is convinced of a need to develop a tool to deal with large-scale problems. Indeed the MILP model was applied to small-medium scale problems where the scheduling process was completed in polynomial time. The new MILP that is represented here is a heuristic tool, which deals with large-scaled NP-hard scheduling problems. This heuristic looks quite similar to the traditional RCPSP model however some changes have been made.

- The second part of this research represents a new metaheuristic method, named SoS algorithms. The algorithm simulates a schedule problem i.e. determining an order of play in the orchestra, according to the instructions of a "conductor". Accordingly every activity has its "time window", ─ a period of time that dictates its beginning time. At this stage a conductor reduces the activity's freedom by controlling its random tossed value in such a way that when a better scheduling is created it replaces the worst one in the repertoire. By using a heuristic (and useful) tool for the widest conflict repair, all conflicts are solved. By using this technique, the collection of solutions becomes bigger and better, so the chance of getting the secondary objective (hammock cost), becomes better too (at each step a schedule with a reduced makespan is obtained as a primary objective however a minimal hammock cost is also achieved).

- This dissertation serves a useful purpose in solving conflicts. It arose from the forbidden set principle which is an idea that Csébfalvi G. (2007), and Csébfalvi G. and Csébfalvi A. (2005) had. According to their idea, after getting an initial non-feasible schedule (e.g. the earliest one) all conflicts are solved by inserting conflict repairs. Thus, at each step of the algorithm there is no fear that a shift should destroy the feasibility of the schedule. One of the main advantages is getting flexibility of the schedule: every activity, whether regular or hammock, can be shifted without negatively affecting the feasibility of the solution. Although the major problem is a RCHCP, the analysis of the

hammock's "behaviour" from a managerial point of view, is done on the unconstrained case, thanks to the flexibility gained by using this important advantage. ***The most important conclusion is that hammock activity as a whole can be moved without negatively affecting the feasibility of the solution.*** This flexibility is achieved by inserting a conflict repair with a very wide slack S_j-S_i-D_i. (However, the forbidden set has a small disadvantage in that the search for all the conflict repairs can be quite extensive). The hybrid algorithm represents a continuous augmenting process from which we get a minimal makespan with a minimal hammock cost in the end.

7.3 New results

1. I developed a new multi-phase MILP approach to manage the RCHCP, where the hammock hangers are represented by dummy activities with zero duration. Starting base: Harhalakis (1990) (HCP) and Csébfalvi and Csébfalvi (2005) (RCHCP).

   i. In the first phase, I solved the RCPSP to get the resource-feasible makespan.

   ii. In the second phase, after fixing the position of the finishing milestone according to the optimal makespan, I solved the RCHCP on the set of the possible conflict repairing relations, to get the optimal hammock durations. According to the applied forbidden-set-oriented approach, the optimal schedule of the second phase is totally invariant to the activity movements; therefore an activity move is unable to destroy the resource-feasibility.

   iii. In the ternary phase, after fixing the position of the finishing milestone according to the optimal makespan and the hammock durations according to the optimal durations and inserting the optimal conflict repairing relations, I introduced a ternary criterion as a MILP to smooth out the resource usage histograms on the set of possible activity movements. The applied

leveling/smoothing criterion (developed by Csébfalvi and Konstantinidis (2002)) can be replaced by any other ternary criterion. The MILP approach can be replaced by a problem-specific heuristic algorithm to manage larger problems.

2. I developed a harmony search metaheuristic algorithm to manage RCHCP. The algorithm is based on the time-oriented Sounds of Silence (SoS) algorithm developed by Csébfalvi for RCPSP. The new problem-specific forbidden-set-oriented SoS algorithm manages the primary and secondary criteria simultaneously using a FBI approach. The new forbidden-set-oriented elements are invariant to the SoS algorithm; therefore they may be inserted to any other population-based metaheuristic algorithms which generate resource-feasible solutions using a variant of FBI. The hammock-oriented secondary criterion can be replaced by any other criterion which can be optimized on the set of the resource-feasible activity movements. The applied ternary criterion (Konstantinidis) can be replaced by any other criterion which is able to rearrange the schedule according to a well-specified managerial goal without affecting the primary (secondary) optimality. The introduced ternary MILP approach can be replaced by a problem-specific heuristic (metaheuristic) algorithm, which generates resource-feasible activity movements and maintains the ternary criterion time to time.

7.4 Future investigations

Based on this research hereunder there are some interesting potential investigations for future consideration.

1) Since a flexible schedule is achieved, it will be interesting to investigate a case of buffers along the critical path – especially putting buffers on hammock activities along the critical path. This means that a project manager should allocate some spare time (which equals the activity's slack) in a case of complex activities, or alternatively, a maximal total slack, to the whole critical sequence.

The investigated issue here is the project's duration resulting from this change. In this case, the project manager should establish priority rules for determining the scheduling order of every activity i.e. by scheduling activities in a non-increasing order of total slacks for each activity.

- However regarding the above, the most pertinent question is how to deal with a hammock activity under the priority rule above i.e. if a hammock activity contains a complicated activity that should be delayed in quite high probability, how should this hammock activity be treated as a whole?
- Another way of treating this issue, is by looking at this problem as a performance problem: instead of getting a minimal hammock cost, we should classify the problem as a minimization of the project's lateness (i.e. the maximum lateness of the sub-projects or activities), or as a minimization of the project's tardiness (i.e. the maximum tardiness of the sub-projects or activities). These suggestions are naturally considered as specifications (with regard to the hammocks' cost), and as indicators to the robustness of the RCHCP model.

2) Examining the power of a hammock (the number of the hammock's members). It should be interesting to find out if the number of the hammock's members could affect the optimality of the solution.

3) In light of the papers of Eliezer and Levi (2010, 2011), and Eliezer and Csébfalvi G. (2009), it should be interesting to combine Alvarez-Valdés's and Tamarit's conflict repairing technique (1993) with Csébfalvi's and Konstantinidis's RLP procedure (2002) (used in Eliezer's and Levi's paper (2010)) By so doing, a better and more flexible scheduling, with smooth resource usage, should be created, and the algorithm should achieve a faster solution than that of Eliezer and Csébfalvi G. (2009).

References

- Ackoff, R.L. (1970). *"A concept of corporate planning*.", New York, Wiley.
- Al-Fawzan M.A., and Hoauari M. (2005). "A bi-objective model for robust resource-constrained project scheduling." ***International Journal of Production Economics,*** (96) 175-187.
- Alvarez-Valdés R., and Tamarit J.M. (1993). "The project scheduling polyhedron - dimension, facets and lifting theorems." ***European Journal of Operational Research*****, 96, 204-220.**
- Battersby, A. (1967). "*Network analysis for planning and scheduling.*" MacMillan, New York.
- Blazewicz J., Ecker K.M., Schmidt G., and Weglarz J. (1985). *"Scheduling in computer and manufacturing systems.*" Springer, Berlin.
- Bouleimen K. and Lecocq H. (2003). "A new efficient simulated annealing algorithm for the resource-constrained project scheduling problem and its multiple mode versions." ***European Journal of Operational Research***, 149, 268–281.
- Bowers, J.A. (1995). "Criticality in resource constrained networks." ***Journal of the Operational Research Society***, 46, 80-91.
- Bowers, J.A. (2000). "Interpreting float in resource constrained projects", ***International Journal of Project Management,*** 18, 385-392.
- Burgess R. and Killebrew J.B. (1962). "Variation in activity level on a cylindrical arrow diagram." ***Journal of Industrial Engineering,*** 13(2), 76-83.
- Clark, C.E. (1962). "The PERT model for the distribution of an activity time." ***Operations Research****,* 10, 405-406.
- Csébfalvi, G., and Csébfalvi A. (2005). "Hammock activities in project scheduling***", in Proceedings of the Sixteenth Annual Conference of POMS***, ***S. Panwalkar, J. Li, (Editors)***, POMS, USA.

- Csébfalvi, G. (2007). "Sounds of Silence: a harmony search metaheuristic for the resource-constrained project scheduling problem." ***European Journal of Operational Research*****,** (under review).
- Csébfalvi, G., Csébfalvi, A., and Szendrői E. (2008a). "A harmony search metaheuristic for the resource-constrained project scheduling problem and its multi-mode version". ***In Project management and scheduling 2008***, F. S. Serifoglu, and Ü. Bilge (Editors), 56-59, Istanbul, Turkey.
- Csébfalvi, G., Eliezer O., Láng B., and Levi R. (2008b). "A conflict repairing harmony search metaheuristic and its application for bi-objective resource-constrained project scheduling problems". ***In Project management and scheduling 2008,*** F. S. Serifoglu, and Ü. Bilge (Editors), 60–63. Istanbul, Turkey.
- Dean B., Mertel Jr. S., and Roepke L. (1969). "Research project cost distribution and budget forecasting." ***IEEE Transactions of Engineering Management,*** 16 (4), 176-191.
- Demeulemeester, E. (1995). "Minimizing resource availability costs in time-limited project networks". ***Management Science,*** 41, 1590-1598.
- Demeulemeester, E., and Herroelen W. (2002). *"Project scheduling: a research handbook."* Norwell, Massachusetts: Kluwer.
- Demeulemeester E., Herroelen W., and Van de Vonder S. (2008). "Proactive heuristics procedure for robust project scheduling: an experimental analysis." ***European Journal of Operational Research***, 189 (3), 723-733.
- Eliezer O. and Csébfalvi G. (2009). "A hybrid method for the resource-constrained project scheduling problem with hammock activities." ***in Proceedings of the first international conference on soft computing technology in civil, structural and environmental engineering***, B.H.V. Topping, and Y. Tsompanakis, (Editors), Civil-Comp Press, Stirlingshire, United Kingdom, paper: 1, 2009. doi:10.4203/ccp.92.1
- Eliezer O. and Levi R. (2010). "A hybrid method for the resource-constrained project scheduling problem with hammock activities and strip packing like resource constraints." ***in Proceedings of the seventh international conference on engineering computational technology***, B.H.V. Topping, J.M. Adam, F.J. Pallarés, R. Bru, and M.L. Romero, (Editors), Civil-Comp Press, Stirlingshire, UK, Paper 91, 2010. doi:10.4203/ccp.94.91.

- Eliezer O. and Levi R. (2011). "An improved hybrid algorithm for the resource-constrained project scheduling problem with hammock activities". ***Ref: CSC2011/2010/000020.***
- Elmaghraby S.E. (1977). *"Activity networks: project planning and control by network models"*, Wiley New York.
- Goncalves J.F., Resende M.G.C., and Mendes J.J.M. (2010). "A biased random-key genetic algorithm with forward-backward improvement for the resource constrained project scheduling problem". ***Journal of Heuristics*** **online:**DO1:10.1007/s10732-010-9142-2
- Harhalakis G. (1990). "Special features of precedence network charts." ***European Journal of Operational Research*** 49 (1), 50-60.
- Icmeli-Tukel O. and Rom W. (1997) "Ensuring quality in resource constrained project scheduling", ***European Journal of Operational Research,*** 103, 483-496.
- Kauffmann, A., and Desbazeille, G. (1964). "*La methode du chemin critique".* Dumond.
- Klein, R. (2000). *"Scheduling of resource constrained projects."* Boston: Kluwer.
- Kolisch R. and Sprecher A. (1996). "PSPLIB – a project scheduling problem library." ***European Journal of Operational Research***, 96, 205-216.
- Konstantinidis, P. (2002). "A model to optimize project resource allocation by construction of a balanced histogram." Unpublished Ph.D. dissertation, University of Pécs, Department of Business Administration and Economics.
- Lambrechts O., Demeulemeester E., and Herroelen V. (2007). "A tabu search procedure for developing robust predictive project schedules." ***International Journal of Production Economics,*** 111 (2), 493-508.
- Leashman R.C., Dinceler, and Kim S. (1990). "Research – constrained scheduling of projects of variable – intensity activities". ***IIE transactions,*** 22 (1) 31-40.
- Lee K.S. and Geem Z.W. (2005). "A new meta-heuristic algorithm for continuous engineering optimization: harmony search theory and practice." ***Computer Methods in Applied Mechanics and Engineering,*** 194, 3902-3933.

- Li K. and Willis R. (1992). "An iterative scheduling technique for resource-constrained project scheduling." ***European Journal of Operational Research,*** 56, (3) 370-379.
- Maylor, H. (2003). *"Project management."* 3rd ed. England: Pearson Education Limited.
- Muller, J.H., and Spinrad J. (1989). "Incremental modular decomposition***." Journal of ACM,*** 36, 1-19.
- Nai-Hsin P., Po-Wen H., and Kuei-Yen C. (2008). "A study of project scheduling optimization using tabu search algorithm." ***Engineering Applications of Artificial Intelligence,*** 21, (7), 1101-1112.
- Özdamar L. and Ulusoy G. (1996). "A note on an iterative forward/backward scheduling technique with reference to a procedure by Li and Willis." ***European Journal of Operational Research,*** 89, (2), 400-407.
- Patterson, J.H., Slowinski R., Talbot F., and Weglarz J. (1989). "An algorithm for a general class of precedence and resource constrained scheduling problems", in ***Advances in project scheduling,*** Slowinski, R. and J. Weglarz (Editors). The Netherlands: Elsevier.
- Pritsker, A.A.B., Watters L.J. (1968). "A Zero-One Programming Approach to Scheduling with Limited |Resources", ***The RAND Corporation***, RM-5561-PR.
- Rafael, F. (1990). "*Sistema de apoio a gestao e programacao financeira de projectos*." Investigacao Operacional, 10 (1), 97-108.
- Ragsdale, C. (1989). "*The current state of network simulation in project theory and practice*." Omega**,** 17, 21-25.
- Raz T. and Marshall B. (1996). "Effect of resource constraints on float calculations in project networks." ***International Journal of Project Management,*** 14, 241-248.
- Schonberger, R.J. (1981). "*Why projects are always late: a rationale based on manual simulation of PERT/CPM network*." Interfaces, 11, 17-66.
- Simon, M.A. (1977). "*The new science of management decision."* New York: Prentice – Hall.

- Speranza, G.M., and Vercellis C. (1993). "Hierarchical models for multi-project planning and scheduling." ***European Journal of Operational Research***, 64, (2), 312-325.
- Stinson, J.P., Davis E.W., and Khumawala B.M. (1978). "*Multi resource constrained scheduling using Branch-and-Bound*." AIIE Transactions, 10, 252-259.
- Tavares, L.V. (1984). "The TRIDENT approach to rank alternative tenders for large engineering projects." ***Foundations of Control Engineering***, 9, 181-191.
- Tavares L.V. (1986). "*Stochastic planning and control of program budgeting – the model MACAO, in OR models on microcomputers",* Coelho J.D. and Tavares L.V. (Editors). Amsterdam: North-Holland.
- Tavares, L.V. (1994). "A stochastic model to control project duration and expenditure. " ***European Journal of Operational Research***, 78, 262-266.
- Tavares L.V., Farreira J.A., and Cuelho, J.S. (1997). "The risk of delay of a project in terms of morphology of its network." ***EURO XV-INFORMS XXXIV Conference, Barcelona, Spain***.
- Tavares L.V., Farreira J.A., and Cuelho, J.S. (1997). "On the optimal management of project risk." ***European Journal of OR***, 107, 451-469.
- Tavares L.V. (1998). *"Advanced models for project management"***.** Dordrexht: Kluwer.
- Tavares, L.V. (2002). "A review of the contribution of operational research to project management." ***European Journal of Operational Research,*** 136, 1-18.
- Tormos P. and Lova A. (2001). "A competitive heuristic solution technique for resource-constrained project scheduling." ***Annals of Operations Research***, 102, 65–81.
- Tseng L.Y. and Chen S.C. (2005). "A hybrid metaheuristic for the resource-constrained project scheduling problem." ***European Journal of Operational Research*** 175, 707-721.
- Valls V., Ballestín F., and Quintanilla S. (2008). "A hybrid genetic algorithm for the resource-constrained project scheduling problem". ***European Journal of Operational Research,*** 185, (2), 495-508.

- Van de Vonder S., Ballestin F., Demeulemeester E., and Herroelen V. (2006). "Heuristic procedure for reactive project scheduling." ***Computer and Industrial Engineering***, 52, 11-28.
- Vanhoucke M. and Van Osselaer K. (2004). "Work continuity in a real-life schedule: the Westerschelde Tunnel", ***Working Papers of Faculty of Economics and Business Administration, Ghent University, Belgium Faculty of Economics and Business Administration*** , October, wp_04_271.
- Wiest J.D. (1964). "Some properties of schedules for large projects with limited resources." ***Operational Research,*** 12, 395-418.
- Willis, R. J. (1985). "Critical path analysis and resource constrained scheduling – theory and practice." ***European Journal of Operational Research,*** 23, 149-155.
- http://www2.cit.cornell.edu/computer/robohelp/cpmm/Phase3_Process_Descriptions.htm