INTERNATIONAL JOURNAL OF OPTIMIZATION IN CIVIL ENGINEERING
Int. J. Optim. Civil Eng., 2012; 2(4):545-556

AN EXPERIMENTAL INVESTIGATION OF THE SOUNDS OF SILENCE METAHEURISTIC FOR THE MULTI-MODE RESOURCE-CONSTRAINED PROJECT SCHEDULING WITH PRE-OPTIMIZED REPERTOIRE ON THE HARDEST MMLIB+ SET

A. Csébfalvi [*,†,a] and E. Szendrői [b]
[a] *Department of Structural Engineering, University of Pécs*
H-7624 Pécs, Boszorkány u. 2. Hungary
[b] *Department of System and Software Engineering, University of Pécs*
H-7624 Pécs, Boszorkány u. 2. Hungary

ABSTRACT

This paper presents an experimental investigation of the Sounds of Silence (SoS) harmony search metaheuristic for the multi-mode resource-constrained project scheduling problem (MRCPSP) using a pre-optimized starting repertoire. The presented algorithm is based on the time oriented version of the SoS harmony search metaheuristic developed by Csébfalvi et al. [1] for the single-mode resource-constrained project scheduling problem (RCPSP). The multi-mode SoS version exploits the fact that using a state-of-the art solver a small mixed integer linear programming problem (MILP) or a large linear programming problem (LP) can be solved within reasonable time. In order to illustrate the viability of the pre-optimized starting repertoire we present computational results for the hardest and largest MMLIB+ benchmark set developed by Van Peteghem and Vanhoucke [2]. The computational result reveals the fact, that the pre-optimized repertoire drastically increases the efficiency of the problem solvong process.

Received: 9 August 2012; Accepted: 15 October 2012

KEY WORDS: project management; project scheduling; heuristics; metaheursitics; hybrid methods; population-based heuristics; harmony search

[*] Corresponding author: A. Csébfalvi, Department of Structural Engineering, University of Pécs H-7624 Pécs, Boszorkány u. 2. Hungary
[†]E-mail address: csebfalvi@pmmik.pte.hu (A. Csébfalvi)

1. INTRODUCTION

The problem considered in this paper involves the scheduling of N activities. Each activity i , $i \in \{1,2,...,N\}$ may be executed in one out of M modes. Activity $i = 0$ ($i = N + 1$) is defined to be the unique dummy source (sink). The activities are interrelated by precedence and resource constraints. Precedence constraints force an activity not to be started before all its predecessors are finished.

Let $PS = \{ i \to j \,|\, i \neq j, i \in \{0,1,\ldots,N\}, j \in \{1,2,\ldots,N+1\}\}$ denote the set of immediate predecessor-successor relations. Resource constraints arise as follows: Performing activity i, $i \in \{1,2,...,N\}$ in mode m , $m \in \{1,2,...,M\}$ takes D_{im} periods and is supported by a set of R renewable, a set of C of non-renewable resources. The per-period availability of the renewable resource r , $r \in \{1,2,...,R\}$ is R_r . The overall capacity of the non-renewable resource c , $c \in \{1,2,...,C\}$ is C_c . Let denote the per-period renewable (the overall non-renewable) resource requirements of activity i in mode m from resource r (c) by R_{imr} (C_{imc}), respectively. Let $\overline{T}$ denote an upper bound of precedence- and resource-feasible makespan:

$$\overline{T} = \sum_{i=1}^{N} \max\left(D_{im} \,|\, m \in \{1,2,\ldots,M\}\right) \tag{1}$$

Let X_i denote the start time of activity i , $i \in \{1,2,...,N\}$ and let $\left[\underline{X}_{im}, \overline{X}_{im}\right]$ denote its time window in mode m , $m \in \{1,2,...,M\}$ where $\underline{X}_{am}$ ($\overline{X}_{am}$) denotes the earliest (latest) starting time of activity i in the unconstrained (only precedence-feasible) case according to the fixed latest completion time. The objective of MRCPSP is to find an assignment of modes to activities as well as precedence- and resource-feasible starting times for all activities, such that the makespan of the project is minimized. Defining the decision variable of the model as binary variables X_{ims} which equal 1 if activity i is scheduled in mode m , $m \in \{1,2,...,M\}$ to start at s , and 0 otherwise, the problem can be formulated as follows:

$$min\left[X_{N+1}\right] = X^*_{N+1} \tag{2}$$

$$X_{N+1} \leq \overline{T} + 1 \tag{3}$$

$$\sum_{m=1}^{M} \sum_{s=\underline{X}_{im}}^{\overline{X}_{im}} X_{ims} = 1, \; i = 1,2,\ldots,N \tag{4}$$

$$\sum_{m=1}^{M} \sum_{s=\underline{X}_{im}}^{\overline{X}_{im}} s * X_{ims} \leq X_{N+1}, \; i \to N+1 \in \boldsymbol{PS} \tag{5}$$

$$\sum_{m=1}^{M}\sum_{s=\underline{X}_{im}}^{\overline{X}_{im}}(s+D_{im})*X_{ims}\le\sum_{m=1}^{M}\sum_{s=\underline{X}_{jm}}^{\overline{X}_{jm}}s*X_{jms}\ ,\ i\to j\in\boldsymbol{PS} \tag{6}$$

$$\sum_{i=1}^{N}\sum_{m=1}^{M}\sum_{s=\underline{X}_{im}}^{\overline{X}_{im}}W_{imst}*R_{imr}*X_{ims}\le R_r\ ,\ t=1,2,\ldots,T\ ,\ r=1,2,\ldots,R \tag{7}$$

$$W_{imst}=\begin{cases}1 & s\le t\ \wedge\ t<s+D_{im}\\ \quad if & \\ 0 & t<s\ \vee\ t\ge s+D_{im}\end{cases} \tag{8}$$

$$\sum_{i=1}^{N}\sum_{m=1}^{M}\sum_{s=\underline{X}_{im}}^{\overline{X}_{im}}C_{imc}*X_{ims}\le C_c\ ,\ c=1,2,\ldots,C \tag{9}$$

$$X_i=\sum_{m=1}^{M}\sum_{s=\underline{X}_{im}}^{\overline{X}_{im}}s*X_{ims}\ ,\ i=1,2,\ldots,N\ , \tag{10}$$

$$M_i=\sum_{m=1}^{M}\sum_{s=\underline{X}_{im}}^{\overline{X}_{im}}m*X_{ims}\ ,\ i=1,2,\ldots,N\ , \tag{11}$$

$$\boldsymbol{X}=\{X_1,X_2,\ldots,X_N\}, \tag{12}$$

$$\boldsymbol{M}=\{M_1,M_2,\ldots,M_N\}, \tag{13}$$

$$X_{ims}\in\{0,1\},\ i=1,2,\ldots,N\ ,\ m=1,2,\ldots,M\ ,\ s=\underline{X}_{im},\ldots,\overline{X}_{im}\ . \tag{14}$$

Constraint set (4) ensures that each activity i, $i=1,2,\ldots,N$ is performed in one mode and is started within its mode dependent time window. Constraints (4-5) represent the precedence relations. The availability per-period of the renewable resource types is maintained by constraints (7-8). Constraints (8) limit the total resource consumption of non-renewable resources to the available amount. The definition of all decision variables is considered in constraints (10-14). Finally, (2) is the objective function considered in this work, to minimize the project completion time.

2. SOS FOR MRCPSP WITH PRE-OPTIMIZED REPERTOIRE

Harmony search (HS) algorithm was recently developed by Lee and Geem [3] in an analogy

with music improvisation process where music players improvise the sounds of their instruments to obtain better harmony. In HS, the optimization problem is specified as follows:

$$\boldsymbol{max}\ \left\{ f(\boldsymbol{X}) \,\middle|\, \boldsymbol{X} = \left\{ X_i \,\middle|\, \underline{X}_i \le X_i \le \overline{X}_i,\ i \in \{1,2,...,N\} \right\} \right\} \tag{15}$$

In the language of music, $\boldsymbol{X}$ is a melody, which aesthetic value is represented by $f(\boldsymbol{X})$. Namely, the higher the value $f(\boldsymbol{X})$, the higher the quality of the melody is. In the band the number of musicians is N, and musician i, where $i \in \{1,2,...,N\}$ is responsible for sound X_i. The improvisation process is driven by two parameters: (1) According to the *repertoire consideration rate* (*RCR*), each musician is choosing a sound from his/her "personal 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* where modification means random perturabtion. The algorithm starts with a totally random "repertoire uploading" phase, after that, the band begins to improvise. During the improvisation, when a new melody is better than the worst in the memory, the worst will be replaced by the better one. Naturally, the two most important parameters of HS algorithm are the *repertoire size* (*S*) and the *number of improvisations* (*I*).

In order to improve the fine-tuning characteristic of HS algorithm, an improved harmony search algorithm (IHS) was developed by Mahdavi et al. [4]. The essence of the modification is very simple: in the function of the progress, the *sound adjusting rate* (*SAR*) is increasing, but the *sound adjusting scope* (*SAS*) is decreasing step by step. Naturally the HIS algorithm increase the number of the tunable parameters therefore the time requirement of the "fine-tuning" phase may become larger.

We have to note that the original HS (IHS) algorithm is an explicit one, because it operates directly on the sounds. In the case of (M)RCPSP, we can only define an implicit algorithm, and we will show that without introducing a conductor we are unable to manage the problem. In the original HS (IHS) algorithm, the definition of improvisation is far from the reality because the improvisation a perturbation of randomly selected "more or less asthetic" sounds. In SoS, the improvisation is a random perturbation of a randomly selected "more or less asthetic" melody, therefore, in our algorithm; the improvisation is really close to the real one (The fundamental differences between HS (IHS) and SoS is shown in Figure 1, where in the repertoire matrix the rows are melodies, the columns are sounds, and the melodies are ordered by their asthetic values. The lighter the gray color the higher asthetic value of melody is). To select a good melody will be the task of the introduced conductor. But this is not enough, in our approach, each of the decisions will be the conductor's responsibility, therefore the musicians form only a decision support system. In the naming of the method, the authors were inspired by the album *Sounds of Silence* (Simon & Garfunkel, 1966). The name is nice and able to express the essence of theharmony search from the (M)RCPSP point of view. In our approach, to simplify the comparison of effectiveness with other population-based approaches we arranged the improvisations into generations, where one generation means *repertoire size* (*S*) improvisations and the *number*

of generations (*G*) is an input parameter of SoS.

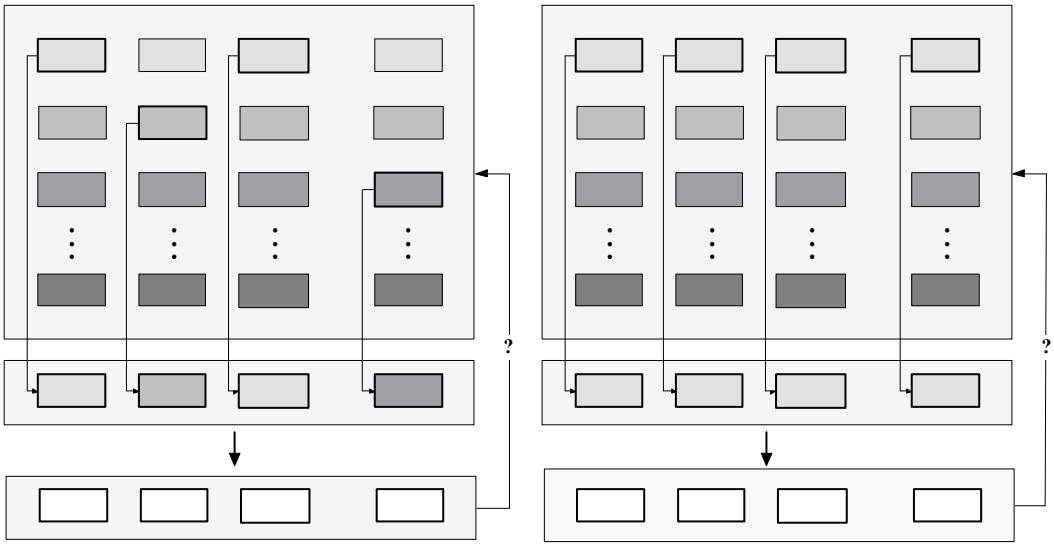

Figure 1: Selection and Improvisation

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

In the case of MRCPSP each $i \in \{1,2,...,N\}$ musician is characterized by a set of disjunctive polyphonic sounds, and a nonrenewable resource can be interpreted as the "energy" requirement of the performance from a given energy type. A nonrenewable resource can be interpreted as the "physical energy" needed to sound a sound, or it may be the "spiritual energy" needed to control the "quality" of the performance. It is also a natural assumption, that the "total energy" of the band is limited from each type, and that the total energy will be consumed by the performance.

In each step, each musician has the right to present (modify) a probabilistic idea IS_i, $IS_i \in [1, M]$ about the "best" sound M_i, and an idea IP_i, $IP_i \in [-1,+1]$ about its "best" starting position X_i, where a large positive (negative) value means that the musician want to enter into the melody as early (late) as possible.

In the random starting repertoire uploading phase IS_i is generated randomly from a truncated normal distribution:

$$IS_i = \textbf{Gauss}\left(\mu^s, \sigma^s\right),\ 1 \le IS_i \le M\ ,\ i \in \{1,2,...,N\},\ \mu^p = 1,\ \sigma^p = 1. \qquad (17)$$

The pre-optimized starting repertoire is generated from the relaxed solution by random

perturbation:

$$IS_i = \textbf{Gauss}\left(\widetilde{M}_i, \sigma^s\right),\ 1 \le IS_i \le M\ ,\ i \in \{1,2,...,N\} \tag{18}$$

where $\widetilde{M}_i,\ 1 \le \widetilde{M}_i \le M\ ,\ i \in \{1,2,\ldots,N\}$ is the relaxed mode value from the relaxed solution. In the pre-optimized version σ^s is a "golden number" of the algorithm because the diversity of the starting repertoire is highly affected by this number. According to our preliminary results, a god setting may be the following: $\sigma^s = 0.1$.

In our approach, independently from the applied starting mode set generation process (which may be totally random or based on random perturbation around the relaxed mode) the starting sound position set is always generated randomly:

$$IP_i = \textbf{Gauss}\left(\mu, \sigma^p\right),\ -1 \le IP_i \le +1\ ,\ i \in \{1,2,...,N\},\ \mu = 0\ ,\ \sigma^p = 1 \tag{19}$$

In the improvisation phase the old $IS_i\,(IP_i)$ will be the mean of the distribution, from which the perturbated new $IS_i\,(IP_i)$ will be generated (see Figure 2-3). During the improvisation the standard deviation $\sigma_g^s\left(\sigma_g^p\right)$ is an exponentially decreasing function of the progress:

$$\sigma_g^{\bullet} = \underline{\sigma}^{\bullet} * \textbf{exp}\left(\textbf{log}\left(\frac{\overline{\sigma}^{\bullet}}{\underline{\sigma}^{\bullet}}\right) * \frac{g-1}{G-1}\right),\ g = 1,2,\ldots,G \tag{20}$$

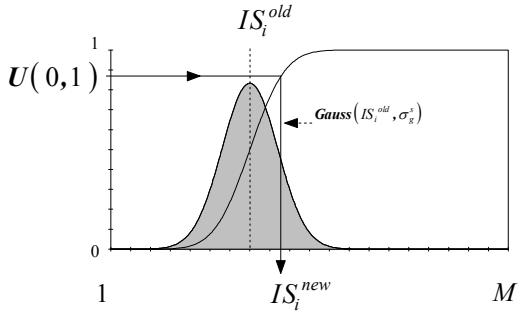

Figure 2: Perturbation of IS_i

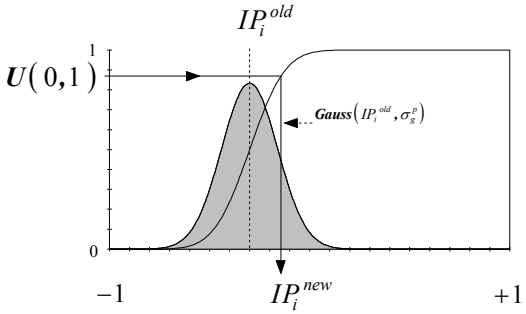

Figure 3: Perturbation of IP_i

The essence of SoS is very simple: The algorithm starts with the repertoire uploading phase. After that, in each improvisation step, the conductor select a melody (the shorter the melody, the higher the chance that it will be selected by the conductor), the musicians modify their own ideas, after the conductor collects the modified ideas, and solve a MILP and a LP problem to balance the effect of the more or less opposite ideas about the better harmony:

The MILP can be formulated as following:

$$\boldsymbol{max}\left[\sum_{i=1}^{N}\sum_{m=1}^{M} IS_{im} * Y_{im}\right] \tag{21}$$

$$\sum_{m=1}^{M} Y_{im} = 1,\ i = \{1,2,\ldots,N\} \tag{22}$$

$$\sum_{i=1}^{N}\sum_{m=1}^{M} C_{imk} * Y_{im} \le C_k,\ k \in \{1,2,\ldots,C\} \tag{23}$$

$$Y_{im} \in \{0,1\} \tag{24}$$

The result of the MILP is an energy-feasible sound combination $M = \{M_1, M_2, \ldots, M_N\}$ which maximizes the satisfaction of the musicians. Theoretically, this MILP is a so-called multiple-choice multi-dimension knapsack problem (MMKP) with several fast and effective problem solving possibilities. One simple problem-specific heuristic, which is competitive with the state-of-the-art MILP solvers was developed by Csébfalvi and Csébfalvi [5].

The LP problem, which maximizes the satisfaction of the musicians with the sound positions, is the following:

$$\boldsymbol{min}\left[\sum_{i=1}^{N} IP_i * X_i\right] \tag{25}$$

$$X_i + D_i \le X_j, i \to j \in \boldsymbol{PS},\ D_i = D_{iM_i} \tag{26}$$

$$\underline{X}_i \le X_i \le \overline{X}_i,\ i \in \{1,2,\ldots,N\} \tag{27}$$

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 generate a soundless energy-feasible 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, using the well-known forward-backward improvement (FBI) methods (see Tormos and Lova [6]) the conductor tries to improve the quality of the generated melody. Naturally, the conductor memorizes the shortest silent melody found so far.

3. COMPUTATIONAL RESULTS

The SoS for MRCPSP has been programmed in Compaq Visual Fortran 6.5. The algorithm, as DLL, was built into the ProMan system (Visual Basic® Version 6.0) developed originally by Ghobadian and Csébfalvi [7]. The projects presented graphically in this paper are Windows® meta-files, which have been generated automatically by ProMan. To solve the large relaxed problems a fast and sparse primal-dual interior point LP solver BPMPD (Mészáros [8]) was used. To solve the small MILP problems a state-of-the-art callable MILP solver (CPLEX 12.0) was used in default mode. The computational results were obtained by running ProMan on a 2.16 GHz HP desktop PC with 1.0 GB of memory under Microsoft Windows XP® operation system. To solve the statistical problems the world's leading statistical software (IBM SPSS 20.0) was used.

The MMLIB+ benchmark set for MRCPSP (*http://www.projectmanagement.UGent.be*) was proposed by Van Peteghem and Vanhoucke [2] to replace the traditional and popular (but not so hard) PSPLIB dataset developed by Kolisch and Sprecher [9]. The MMLIB+ dataset contains projects with 50 and 100 activities. Each project activity has 2 or 4 renewable and nonrenewable resources. For each activity, 3, 6 or 9 modes are defined. The resource strength is set at 0.25, 0.50 or 0.75. In order to keep the number of instances to a reasonable level, the renewable and nonrenewable resource factor is set to 1. Using 5 instances from each problem class, the MMLIB+ dataset contains 2*2*2*3*3*3*5=3,240 instances. The problem is presented by an illustrative example (see Figure 6).

Despite what the authors promised, up to now we know nothing about the best found makespan for each project instance which, according to their original announcement, should have been uploaded more than two years ago. Therefore, according to the best of our knowledge, this is the first time when somebody presents computational results for the MMLIB+ dataset.

The solution of the MMLIB+ dataset is a challenging but sometimes really frustrating problem. According to the large number of activities and different modes and a not necessarily good makespan upper bound even the relaxed problem solution may cause result extra computational difficulties. In the presented experimental investigation we first used SoS without pre-optimized repertoire with setting $\{S, G\} = \{5, 5\}$ to get a relaxed problem with reasonable size and after solving it we used SoS width the pre-optimized repertoire with setting $\{S, G\} = \{100, 10\}$.

To illustrate the problem, we show the case of instance 648-5 $\{N, M, R, C\} = \{100, 9, 4, 4\}$ which is one of the hardest instances in MMLIB+. The estimated upper bound without pre-optimized repertoire was 218, the relaxed makespan was 81. To get the relaxed solution we had to solve a LP problem with 2,342 rows and 134,560 columns (density: 0.808 %). The solution time was 157.172 sec using a fast interior point solver (BPMPD) in sparse mode. After 10 independent runs with pre-optimized repertoire the best found makespan was 108 and the corresponding schedule is shown in Figure 6.

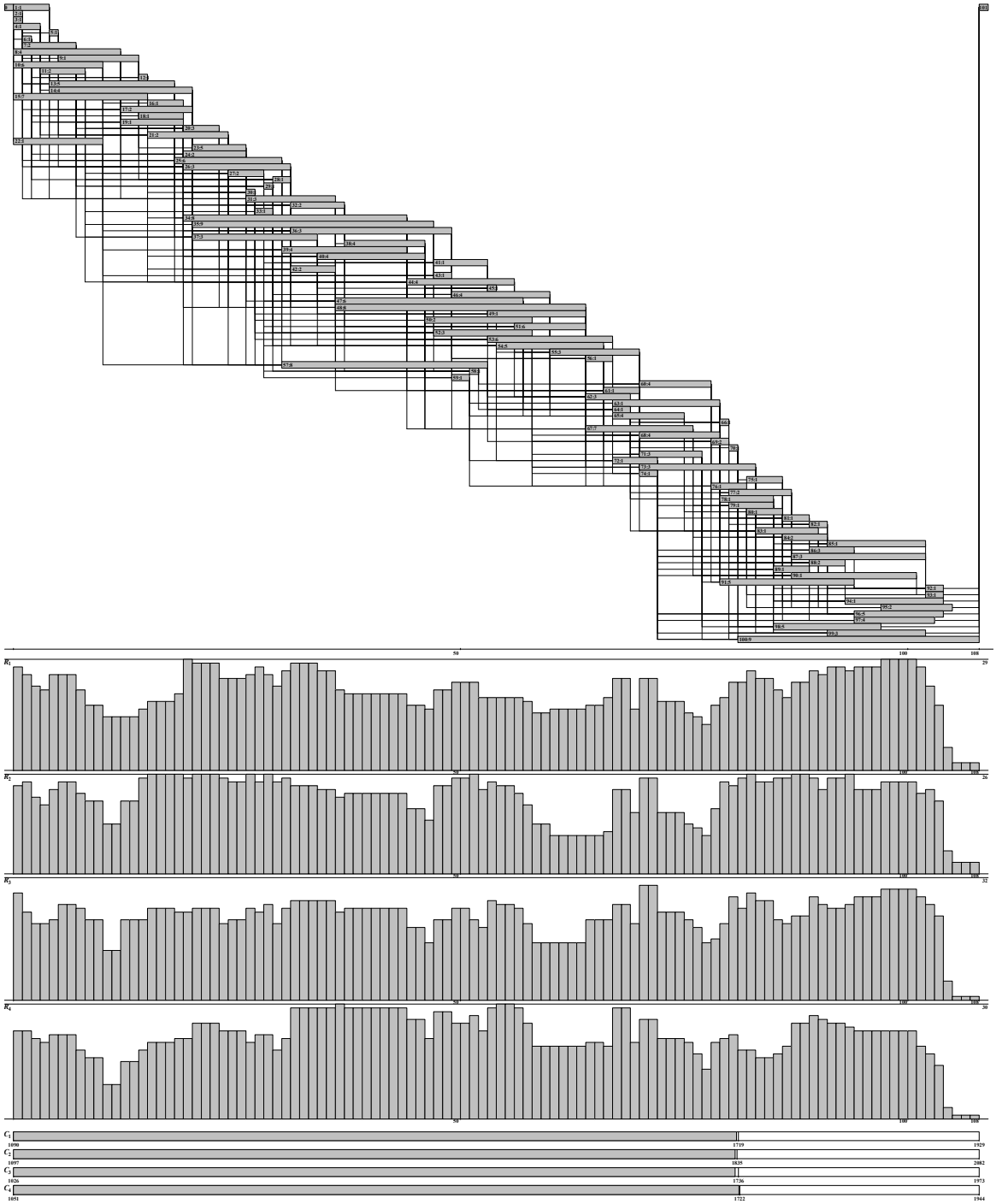

Figure 6: The best schedule for instance 648-5 from MMLIB+ after 10 independent runs

To analyse the effect of the pre-optimized repertoire we repeat the experiment without pre-optimized repertoire. The results revealed the fact that the pre-optimized repertoire significantly increases the quality of the solutions with exactly the same other settings:

$$\left\{S, N, \overline{\sigma}^{s}, \underline{\sigma}^{s}, \overline{\sigma}^{p}, \underline{\sigma}^{p}\right\} = \left\{100, 10, 0.1, 0.01, 1, 0.1\right\} \tag{28}$$

In Figures 7 and 8 we present the searching history with or without pre-optimized starting repertoire where the quality of a solution was measured by the percentage gap of the makespan value with respect to the minimal critical path lover bound.

We have to note, that in this example the statistically significant large difference in the efficiency between the pre-optimized and random starting repertoire cases can be explained by a simple fact: according to our preliminary investigations, the higher the number of the feasible modes the higher the chance that the pre-optimizisation gives better results.

Using the previously defined parameters, the results on the whole MMLIB+ set are presented in Table 1. According to the nonparametric paired comparison test the pre-optimized starting repertoire significantly increases the efficiency of the SoS algorithm developed for MRCPSP.

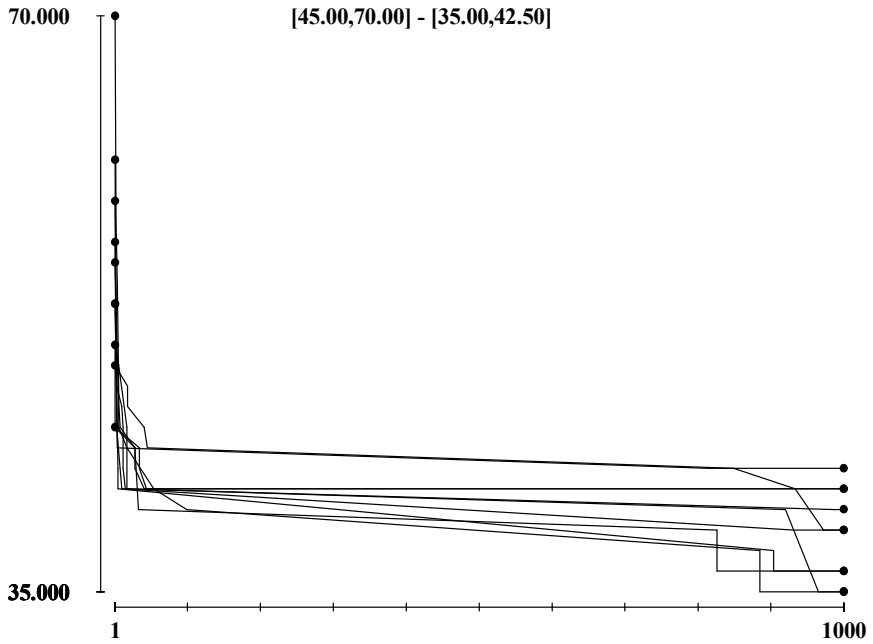

Figure 7: Searching history with pre-optimized starting repertoire for instance 648-5

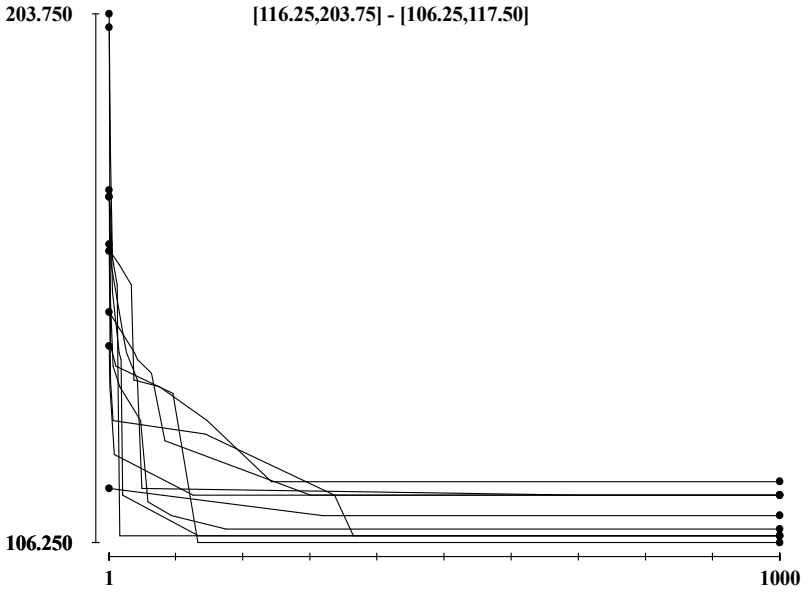

Figure 8: Searching history with random starting repertoire for instance 648-5

Table 1: SoS results on MMLIB+

Method	**AverageSolution Quality** (%)	**Average Solution Time** (sec)
Random Starting Repertoire	119.40	17.35
Pre-optimized Starting Repertoire	97.50	77.02

6. CONCLUSION

In this paper, we have presented a hybrid harmony search metaheuristic for MRCPSP with pre-optimized repertoire. The computational results on the hardest and largest MMLIB+ benchmark set revealed the fact, that the pre-optimized starting repertoire significantly increases the efficiency of the SoS algorithm developed for MRCPSP.

REFERENCES

1. Csébfalvi G, Csébfalvi A, Szendrői E. 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.

2. Van Peteghem V, Vanhoucke M. An Experimental Investigation of Meta-heuristics for the Multi-mode Resource-Constrained Project Scheduling on new Dataset instances. *Working Paper*, 2010, 11/758, *Ghent University*.
3. Lee KS, Geem ZW, A new meta-heuristic algorithm for continuous engineering optimization: harmony search theory and practice, *Compu. Methods Appl Mech Eng,* 2005, **194** 3902–3933
4. Mahdavi M, Fesanghary M, Damangir E, An improved harmony search algorithm for solving optimization problems. *Appl Match Comput*, 2007; **188**; 1567-79.
5. Csébfalvi G, Csébfalvi A. A new metaheuristic for the multidimensional 0-1 knapsack problem, in *Computational Management Science Conference* 2008, Imperial College, London, UK, 38-39.
6. Tormos P, Lova A, A competitive heuristic solution technique for resource-constrained project scheduling. *Ann Oper Res*, 2001; **102**; 65–81.
7. Ghobadian A, Csébfalvi G, Workshop on Developing Interactive Learning Material for Project Management, in *Proceedings of Decision Sciences Institute*, 1995, Boston, US.
8. Mészáros Cs. The Efficient Implementation of Interior Point Methods for Linear Programming and their Applications, *PhD Thesis*, Eötvös Loránd University of Sciences, 1996, Hungary.
9. Kolisch R, Sprecher A. "PSPLIB – a project scheduling library", *Eur J Oper Res*, 1996; **96**; 205-16.