MISTA 2015

LAHC applied to The Multi-Mode Resource-Constrained Multi-Project Scheduling Problem

**Janniele A. Soares · Haroldo G. Santos ·
Davi D. Baltar · Túlio A. M. Toffolo**

1 Introduction

The Project Scheduling Problem (PSP) consists in to schedule jobs over time in such a way that precedence relations are satisfied and resource consumption limits are respected [5]. In this paper a comprehensive variation of the PSP is considered: the Multi-Mode Resource Constrained Multi-Project Scheduling Problem (MMRCMPSP). In this problem jobs can be processed in different modes, with varying execution speeds and consuming different amounts of resources. The objective function also considers project delays.

As a generalization of the PSP, the MMRCMPSP is NP-Hard and can be used to model many problems in several areas, such as project management in information technology companies, scheduling instructions to processor architecture, civil engineering, ingot production scheduling, among others. Even the generation of an initial feasible solution for the MMRCMPSP is also NP-Hard, a feasible selection of modes most be selected, which corresponds to solving the multi-dimensional knapsack problem.

The model of MMRCMPSP [6] has a set P projects, where each $p\ inP$ consists of a set $J_p = 1, ..., |J_p|$ jobs. Each project p has a start time, where jobs can be initiated. The beginning and the end of a project are delimited by fictitious jobs.

Furthermore, the MMRCMPSP comprises a set of constraints, related with the precedence between jobs J and the consumption of resources. These resources, can be renewables R, in way local or global (shared with another projects), and non-renewable K, may to deplete throughout each project $p \in P$. Each job $j \in J$ can be performed in a certain mode $m \in M$, it determines the time taken for the execution d_{jm} and the amount of renewable and non-renewable resources consumed, respectively v_{rjm} and u_{kjm}. This consumption can not exceed the amount available of renewable q_r and non-renewable o_k resources. All relations of precedence $Pred$ and $Pred_j$ must be guaranteed.

The objective function adopted in this paper, refers to the default in MISTA 2013 Challenge [1], having two components: the TPD, main goal, which is the sum of the

J. A. Soares[1] · H. G. Santos[2] · D. D. Baltar[3] · T. A. M. Toffolo[4] · [1,3] Computing and Information Systems Department, Federal University of Ouro Preto, Brazil · [2,4] Computing Department, Federal University of Ouro Preto, Brazil · E-mail: janniele@decsi.ufop.br, haroldo@iceb.ufop.br, davibaltarx@gmail.com,tulio@toffolo.com.br

delays of the projects in relation to the duration of the critical path and the TMS, secondary objective, which is the difference between the maximum completion time of a project and the beginning minimum time of a project.

In this work, we propose and evaluate computationally a solver based in the Late Acceptance Hill-Climbing (LAHC) [4] metaheuristic. The techniques implemented consider as input specific instances of multi-projects available on the MISTA2013 Challenge site, these instances are composed of specific projects from PSPLib.

2 Solution Approach

Our approach, differently from most approaches used in the MISTA 2013 competition, always navigates in the search space of feasible solutions. The generation of different feasible mode sets is accomplished by the solution of a series of multi-dimensional knapsack problems and the use of indirect solution representations. After building an initial solution pool, several LAHC threads start local search around those solutions.

To increase diversity, we first build a pool of different mode sets which satisfy the consumption of non-renewable resources. Since the selection of processing modes also determines the duration of jobs, a greedy strategy to prioritize the selection of fast processing modes is employed. One must observe, however, that it does not guarantees a smaller TPD, because renewable resources constraints can delay the starting time of jobs.

The binary program to built this initial set of modes considers: J jobs with respective processing times p_{jm} and N non renewable resources. Each job has a set M_j of possible modes and the non-renewable resource n consumption of job j in mode m is denoted as r_{jmn}. Thus, the binary program is solved to select which mode m job j will be allocated, considering its respective decision variables x_{jm} and resource availability q_n for each non renewable resource. Which corresponds to the NP-Hard problem of the 0-1 multidimensional knapsack problem.

The local search procedures which will be described later operate over an indirect solution representation: a solution is stored in a $(\mathit{\Gamma}, \mathit{\Pi})$ ordered pair of vectors where $\mathit{\Gamma}_j \in M_j$ indicates the selected mode for job j and $\mathit{\Pi}_j \in \{1, \ldots, |J|\}$ indicates the desired position for job j in the sequence of allocations. $\mathit{\Gamma}$ always respects non-renewable resources consumption. As most of the processing time of this algorithm is spent in the local search phase, the ability to quickly decode a $(\mathit{\Gamma}, \mathit{\Pi})$ pair is a fundamental aspect in the performance of the algorithm.

We implemented all optimizations proposed in [2], such as prefix detection and early exploration of resource insufficiency. Differently from [2] we do not guarantee a valid topological sort in $\mathit{\Pi}$. This speeds up the generation of valid movements, since some validations may be disabled but the cost saved is moved to the decoding phase. To transform the sorting in $\mathit{\Pi}$ into a valid topological sorting, at each new allocation one has to check the available job with highest priority (smallest desired position), which yields an $O(n^2)$ algorithm to decode $\mathit{\Pi}$. Fortunately we devised a simple heap to speed up this to $O(n \log n)$. Initially, all jobs are inserted into the heap with priority $|S_j^-| \times |J| + \mathit{\Pi}_j$.

The complete neighborhood structure $\mathcal{N}(s)$ is composed by fourteen types of movements: Change One Mode (COMS), Change Two Modes (CTMS), Change Three Modes (CTRMS), Change Four Modes (CFMS), Invert Subsequence (INVS), Shift Jobs (SJS), Swap Jobs (SWJS), Compact Project (CP), Shift Project (SPS), Swap

Two Projects (SWPS), Mutation One Extreme (MOE), Move Projects (MP), Swap Jobs FILS (SJF), Insert Jobs FILS (IJF).

The Late Acceptance Hill-Climbing (LAHC) is a new metaheuristic proposed by [3], which consists of an adaptation of the classic Hill-Climbing method. To accept or not a new candidate, its cost is compared with some cost previous belonging the list. It stores a list c of size l with values of cost, this size is the number of the previous iterations i. The list c is initialized to the cost of the initial solution s received as a parameter. At each iteration i, a candidate solution s' is generated and its cost is compared to a virtual position in the c. The solution s' is accepted if their cost is smaller than the contents of the virtual position on c, or is smaller than the cost of current solution s. If accepted, the solution s will be updated, if even better than the best solution found so far s^*, it will also be updated. Subsequently the contents of the virtual position of c will be updated with the new cost of s

To prevent stagnation and increase diversification, we store how many movements were performed for each job j. Whenever number 1000 of non-improvement iterations is reached, diversification is activated and a series of diversification movements is performed. The selection of these movements uses an adapted objective function which incentives the selection of movements involving jobs which were not commonly selected in previous iterations.

3 Computational Experiments

All algorithms were coded in C++ and the binary programming models were solved by CPLEX 12.6. The code was compiled with GCC 4.7.1 using flag -O3. All tests ran on a computer with an Intel(R) Core(TM) i7-4960X CPU @ 3.60GHz processor and 32 Gb of RAM, running OpenSUSE Linux 13.2. The developed method ran in parallel using 4 threads.

Table 1 shows the best results found by the proposed approach, as well the average and standard deviation, after 10 runs within 300 seconds of runtime, with 4 threads with l size of 500. The last column of the table represents the gap obtained by dividing the cost of the best known solution B for the cost of the best solution obtained C by approach. Instances were the obtained results were better or equal than the best results known in the literature are emphasized. It is important to emphasize that the results presented by the technical report [2] were obtained in 2500 different executions for each instance, while the results presented by MISTA2013 Challenge were obtained with only 10 different executions, the same as the results of the present work.

4 Conclusions

In this work we presented the application of LAHC metaheuristic for MMRCMPSP. Integer programming is used to build an initial feasible solution. The LAHC algorithm was enhanced to perform informed diversification using long term memory. Efficient algorithms to decode indirect solution representations were also implemented. Our solver was able to improve two best known solutions for instances used in the MISTA 2013 Challenge and provided solutions very close to the best known ones for the remaining instances.

Table 1 Best and average results after 10 runs of the algorithm sided with [1] and [2]

Inst.	Best		Avg.		Std.Dev.		Mista		Report [2]		B/C
	TPD	TMS	TPD	TMS	TPD	TMS	TPD	TMS	TPD	TMS	
A-1	**1**	**23**	**1.00**	**23.00**	**0**	**0**	**1**	**23**	**1**	**23**	**1.00**
A-2	**2**	**41**	**2.27**	**41.13**	**0**	**0**	**2**	**41**	**2**	**41**	**1.00**
A-3	**0**	**50**	**0**	**50**	**0**	**0**	**0**	**50**	**0**	**50**	**1.00**
A-4	**65**	**42**	**67.50**	**43.10**	**0.81**	**1.58**	**65**	**42**	**65**	**42**	**1.00**
A-5	157	105	169.00	109.40	6.03	2.06	153	105	150	103	0.96
A-6	145	96	159.00	99.60	9.26	3.38	147	96	133	99	0.92
A-7	608	195	630.90	206.80	11.63	5.74	596	196	590	190	0.97
A-8	286	152	308.90	155.20	10.99	1.99	302	155	272	148	0.95
A-9	207	126	220.50	129.50	6.25	2.91	223	119	197	122	0.95
A-10	896	312	943.10	320.70	18.45	3.66	969	314	836	303	0.93
B-1	352	125	366.70	130.80	6.96	2.68	349	127	294	118	0.84
B-2	444	167	460.90	168.70	8.09	2.41	434	160	431	158	0.97
B-3	557	213	579.30	213.10	10.87	0.83	545	210	526	200	0.94
B-4	1276	281	1355.70	291.10	31.37	4.64	1274	289	1252	275	0.98
B-5	845	250	871.90	257.40	16.06	4.20	820	254	807	245	0.96
B-6	911	226	938.20	229.60	12.22	3.17	912	227	905	225	0.99
B-7	807	233	863.00	240.50	23.49	4.15	792	228	782	225	0.97
B-8	**2974**	**541**	**3081.20**	**545.60**	**54.66**	**4.36**	**3176**	**533**	**3048**	**523**	**1.02**
B-9	4630	851	4730.30	852.30	52.19	7.24	4192	746	4062	738	0.88
B-10	**3050**	**448**	**3106.90**	**450.20**	**34.59**	**5.33**	**3249**	**456**	**3140**	**436**	**1.03**
X-1	393	143	426.50	148.50	13.38	4.10	392	142	386	137	0.98
X-2	367	166	394.20	171.40	12.16	2.94	349	163	345	158	0.94
X-3	325	191	342.30	193.50	10.59	2.50	324	192	310	187	0.95
X-4	915	208	958.10	211.90	21.16	3.39	955	213	907	201	0.99
X-5	1786	377	1862.80	382.50	36.02	5.20	1768	374	1727	362	0.97
X-6	700	234	745.00	239.50	23.86	3.56	719	232	690	226	0.99
X-7	861	236	902.20	236.80	15.65	5.90	861	237	831	220	0.97
X-8	1218	286	1277.50	292.40	25.46	2.97	1233	283	1201	279	0.99
X-9	3475	706	3520.90	697.60	27.89	7.79	3268	643	3155	632	0.91
X-10	1652	399	1695.90	396.90	18.29	4.81	1600	381	1573	383	0.95

Acknowledgements The authors thank CNPq and FAPEMIG for supporting this research.

References

1. Wauters, T. Kinable, J. Smet, P. Vancroonenburg, W. Berghe, G.V. and Verstichel, J. : Mista 2013 challenge (2013). URL `http://allserv.kahosl.be/mista2013challenge/`. Access date: 2 jan. 2015
2. Asta, S., Karapetyan, D., Kheiri, A., Ozcan, E., Parkes, A.J.: Combining Monte-Carlo and Hyper-heuristic methods for the Multi-mode Resource-constrained Multi-project Scheduling Problem, technical report. Tech. rep., University of Nottingham, School of Computer Science (2013)
3. Burke, E., Bykov, Y.: A late acceptance strategy in hill-climbing for exam timetabling problems. PATAT 08 Proceedings of the 7th International Conference on the Practice and Theory of Automated Timetabling (2008)
4. Burke, E., Bykov, Y.: The Late Acceptance Hill-Climbing Heuristic. Tech. rep., University of Nottingham, School of Computer Science (2012)
5. Kolisch, R., Hartmann, S.: Experimental investigation of heuristics for resource-constrained project scheduling: An update. European Journal of Operational Research **174**(1), 23 – 37 (2006)
6. Wauters, T., Kinable, J., Smet, P., Vancroonenburg, W., Berghe, G.V., Verstichel, J.: The multi-mode resource-constrained multi-project scheduling problem. Journal of Scheduling (2014)