A COMPUTATIONAL FRAMEWORK FOR PROJECT SCHEDULING, ANALYSIS AND OPTIMIZATION

Geraldo Luis da Silva Ribeiro, geraldo.ribeiro@vsesa.com.br
Rogério Rodrigues dos Santos, rogerio.santos@vsesa.com.br
Vale Solutions in Energy – VSE
Rodovia Presidente Dutra, km 138 Eugênio de Melo
São José dos Campos - SP, Brazil

Abstract. The ability to develop complex products is a feature commonly required from modern engineering. The increasing complexity and size of some projects requires a suitable tool to address the subject effectively. Some authors define "engineering" as the application of mathematical and scientific principles. In this context, the ability to propose, execute and improve complex planning schedules is of paramount importance. A basic schedule is a general guidance for preparing and releasing various types of technical plans. These documents include technical management plans (such as hardware development plans, software development plans, configuration and data management plans, and risk management plans), supporting technical plans (such as system safety plans, manufacturing plans, and system support plans) and test plans (such as system integration, test, and verification plans and hardware and software test plans), to cite some. Due the relevance of the theme when dealing with complex projects, in this paper a methodology for task planning, analysis and optimization is proposed. A draft planning containing task dependence, time duration and used resource may be provided by the manager. The proposed framework is able to import such information from commercially available management software. The schedule is then translated as an optimization problem, where the overall time is the objective to be minimized. Resource allocation and risk management are considered by means of weighting factors. Pareto framework is considered in order to provide visibility of alternative scenarios. The proposed formulation considers different task priorities of the optimal planning. As a result, relevant information about optimal scheduling and sensitivity of alternative scenarios is provided. Numerical results show the viability of the proposed methodology.

Keywords: *task scheduling, project optimization, project management.*

1. INTRODUCTION

Nowadays, more than ever, companies want to deliver products and services better, faster, and cheaper. At the same time, in the high-technology environment of the twenty-first century, nearly all organizations have found themselves building increasingly complex products and services.

In this context, a single company usually does not develop all the components that compose a product or service. More commonly, some components are built in-house and some are acquired; then all the components are integrated into the final product or service. Organizations must be able to manage and control this complex development and maintenance process.

A key point to achieve better performance is the project planning.

According to Capability Maturity Model Integration for development (CMMI, 2006), at maturity level 2, the projects of the organization have ensured that processes are planned and executed in accordance with a given policy. The process discipline reflected by maturity level 2 helps to ensure that existing practices are retained during times of stress. At the last level, the maturity level 5, an organization continually improves its processes based on a quantitative understanding of the common causes of variation inherent in processes. Maturity level 5 focuses on continually improving process performance.

Quantitative process improvement objectives for the organization are established, continually revised to reflect changing business objectives, and used as criteria in managing process improvement.

It motivates the development of a tool for fast analysis of changes in planning. The current paper proposes a new formulation for optimization of an arbitrary planning project. Based on a linear programming model, the current purpose ensures the global optimality independent of the initial configuration. Low CPU time is another attractive feature of the current purpose.

The remainder of the paper is organized as follows. Section 2 reviews some works in this field and presents a widely adopted classification for planning and scheduling problems. Linear programming framework, multiple objective optimization and the proposed model to scheduling problems are presented in section 3. Section 4 analyses performance results of the proposed strategy. Concluding remarks and directions for future works are given in section 5.

2. PROJECT PLANNING

The theory of scheduling is characterized by large number of problem types (Baker, 1974), (Blazewicz et al., 1996), (Coffman, 1976), (Conway et al., 1967), (French, 1982), (Lenstra, 1977), (Pinedo, 1995), (Rinnooy Kan, 1976), (Tanaev et al., 1994), (Tanaev et al., 1994). A classification scheme widely used in the literature is presented by (Lawler et al., 1993).

Suppose that *n* jobs J_i (*i=1,...,n*) have to be processed on *m* machines M_j (*j=1,...,m*). It is admitted that each machine can process one job at a time and that each job can be processed on at most one machine at a time. Job, machine and scheduling characteristics are reflected by a 3-field problem classification α | β | γ (Graham et al, 1979).

The job data J_i is usually specified through a number of operations m_i, processing time p_i, a release date r_i, on which J_i becomes available for processing, a due date d_i, by which J_i should ideally be completed and a weight w_i, indicating the relative importance of J_i.

The measuring the cost $f_i(t)$ incurred if J_i is completed at time *t* is given by a nondecreasing real cost function f_i. In general, m_i, p_i, r_i, d_i and w_i are integer variables.

A batch processing machine is one that can handle up to *B* jobs simultaneously all the time. The jobs that are processed together form a batch, and all jobs contained in the same batch start and complete at the same time since the completion time of a job is equal to the completion time.

The works of Santos and Magazine (1985) and Tang (1990) purpose integer programming formulations and some procedures to determine optimal batches of jobs for a single-stage production. Ikura and Gimple (1986) address the problem of scheduling batch processing machines from a deterministic scheduling perspective. Ahmadi, al et. (1992) examine a class of problems defined by a two or three machine flowshop where one of the machines is a batch processing machine. Another works on batching and scheduling includes Coffman, Nozari and Yannakakis (1989), Julien and Magazine (1989), to cite some.

The Job-Shop Scheduling Problem (JSP) is one of the most popular manufacturing optimization models (Jain and Meeran, 1999).

The interest is partially justified by its wide applicability and inherent difficulty (Carlier and Pinson, 1989), (Kolonko, 1999), (Nowicki and Smutnicki, 1996), (Yamada and Nakano, 1996).

It is known also that the JSP is NP-hard (Garey et al., 1976), and hence generally difficult to solve as the problem size grows larger.

The $n \times m$ classical JSP involves *n* jobs and *m* machines. Each job is to be processed on each machine in a pre-defined sequence, and each machine processes only one job at a time.

Some features of the JSP significantly increase the complexity of finding optimal solutions (Pinedo and Chao, 1999).

Heuristics search methods, such as Genetic Algorithms (Kacem et al., 2002), Simulated Annealing (Kolonko, 1999) and Tabu Search (Nowicki and Smutnicki, 1996) are some approaches to overcome such difficulty. The counterpart of such approach is the expense of a big computational cost, particularly when the problem is large in size.

A kind of dispatching rule has been proposed to reduce the computational cost (Panwalkar and Iskander, 1977), (Blackstone et al., 1982), (Holthaus and Rajendran, 1997). Although the quality of solutions produced by dispatching rules is usually no better than the local search methods, they are attractive due to their ease of implementation and low CPU time.

The shop scheduling problems, such as open shop problems, flow shop problems, job shop problems, and mixed shop problems, are widely used for modeling industrial production processes. All of these problems are special cases of the general shop problem.

The general shop problem may be defined as follows. Consider *n* jobs *i* = 1, ..., *n* and *m* machines $M_1, ..., M_m$. Each job *i* consists of a set of operations O_{ij} ($j = 1, . . . , n_i$) with processing times p_{ij} . Each operation O_{ij} must be processed on a machine $\mu_{ij} \in \{M_1, ..., M_m\}$. There may be precedence relations between the operations of all jobs. Each job can only be processed only by one machine at a time and each machine can only process one job at a time. The objective is to find a feasible schedule that minimizes some objective function of the finishing times C_i of the jobs *i* = 1, ... , *n*. The objective functions are assumed to be regular.

Some scheduling problems can be solved efficiently by reducing them to well-known combinatorial optimization problems, such as linear programs, maximum flow problems, or transportation problems. Others can be solved by using standard techniques, such as dynamic programming and branch-and-bound methods.

In the current paper is proposed a linear programming formulation to solve the scheduling problem, as presented in the following.

3. OPTIMIZATION

3.1 Linear programming

A linear program (LP) is an optimization problem in which the objective function is linear in the unknowns and the constraints consist of linear equalities and linear inequalities. Any linear program can be transformed into the following standard form:

$$\min c_1x_1 + c_2x_2 + \cdots + c_nx_n \tag{1}$$

Subject to

$$\begin{aligned} &a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n = b_1 \\ &a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n = b_2 \\ &\ldots \\ &a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n = b_m \end{aligned} \qquad (2)$$

$$x_1 \geq 0, x_2 \geq 0, \ldots, x_n \geq 0 \qquad (3)$$

where a_{ij}, b_i, and c_i are real constants and x_i are the design variables.

In the present context design variable is time for completing each task. Objectives are the time when the project is finished and the associated cost.

Since multiple objectives will be analyzed in a unified formulation, a theory about multiple objectives is considered.

3.2 Multi-objective optimization

A multi-objective optimization problem can be written in the form

$$\min[f_1(x), f_2(x), \ldots, f_k(x)] \qquad (4)$$

for k objective functions $f_i: \mathbb{R}^n \rightarrow \mathbb{R}$ subject to equality and inequality constraints. For the vector of decision variables, $x = [x_1, x_2, \ldots, x_n]^T$, the task is to determine the set F of all vectors which satisfy the constraints and the particular set of optimal values $x^* = [x_1^*, x_2^*, \ldots, x_n^*]^T$.

Since there are several objectives to be optimized simultaneously, usually there is no longer a single optimal solution but rather a whole set of solutions. When several objectives are optimized at the same time the search space becomes partially ordered. To obtain the optimal solution there will be a set of optimal trade-offs between the conflicting objectives.

In this context, best solution means a solution not worst in any of the objectives and at least better in one objective than the other. An optimal solution is the solution that is not dominated by any other solution in the search space. Such an optimal solution is called a Pareto-optimal and the entire set of such optimal trade-offs solutions is called a Pareto-optimal set.

The publication of Kuhn–Tucker (1951) is one of the first rigorous mathematical treatments about Pareto Optimality. The work of Koopmans (1951) initiated the use of Pareto optimality in operations research. Today a number of papers and books on the subject are found.

Even though there are several ways to approach a multi-objective optimization problem, most work is concentrated on the approximation of the Pareto set.

Given a set of alternatives, the problem of choosing the best alternative depends on the way the data is classified. One of the most popular evaluation methods is to associate to each alternative a real value, and the best alternative is chosen as the one with the largest or the smallest value.

In a higher dimension the notion of the smallest and the largest values is not available. In this case, the concept of partial order in a multidimensional space can be applied.

An up-to-date discussion about this subject is presented in Pardalos and Du (2008).

3.3 Weighting objective formulation

To formulate the performance criterion that takes into account all the objective functions in such a way that an overall multi-criterion objective function can be written, the Weighting Objective Method is used. The minimization process leads to a Pareto optimal solution or, alternatively, to a set of optimal solutions. The scalar objective function that represents the performance criteria altogether is written as:

$$F(x) = \sum_{i=1}^{k} \alpha_i f_i(x) \qquad (5)$$

where $\alpha_i \geq 0$ are weighting coefficients that represent the relative importance of each separate criterion. From the numerical point of view the minimization process depends also on the numerical values that express the objective functions. Due to scaling problems, the numerical values that express the objective functions should be adjusted. Otherwise, α_i will not represent the relative importance of the objective functions (Deb, 2001). Consequently, Eq. (5) should be rewritten as follows:

$$F(x) = \sum_{i=1}^{k} w_i f_i(x) \qquad (6)$$

where w_i are scaling factors. Usually, satisfactory results are obtained if $w_i = \frac{\alpha_i}{f_i^0}$, where f_i^0 represents the minimum of the objective function f_i calculated separately (Eschenauer *et al*, 1990). Eq. (6) was used in the optimization processes shown in this paper.

3.4 Proposed formulation

Suppose that m resources $M_j(j = 1, ..., m)$ have to perform n tasks $J_i(i = 1, ..., n)$. A schedule is for each task an allocation of one or more time intervals to one or more resources. Schedules may be graphically represented by Gantt charts, which may be resource-oriented or task-oriented. The corresponding scheduling problem is to find a schedule satisfying certain constraints.

Each task has a start time t_s and a time duration t_d. It is supposed that an amount of resource r is required to perform the task in a given period.

Furthermore, precedence between tasks are also taken into account, that is, $t_{s,i} \geq t_{s,j} + t_{d,j}$ for some indexes i and j. All these facts can be modeled through the equations

$$\min t_{s,n+1} \tag{7}$$

Subject to

$$t_{s,i} \geq t_{s,j} + t_{d,j}, \text{ for some } i \text{ and } j \tag{8}$$
$$t_{s,n+1} > t_{s,j} + t_{d,j}, \; j = 1, \dots, n \tag{9}$$
$$t_{s,j} \geq 0, \; j = 1, \dots, n \tag{10}$$

By considering the cost w_c of each task inversely proportional to its time duration, that is,

$$w_{c,j} = \frac{k_j}{t_{d,j}} \tag{11}$$

it is possible to reduce the cost by increasing its time duration. The parameter k_j is a given constant. Such expression means that larger the time, smaller the cost, and on the other hand, smaller the time, bigger the cost. The optimization problem can be rewritten as

$$\min t_{s,n+1} - \sum_{j=1}^{n} t_{d,j} \tag{12}$$

Subject to

$$t_{s,i} \geq t_{s,j} + t_{d,j}, \text{ for some } i \text{ and } j \tag{13}$$
$$t_{s,n+1} > t_{s,j} + t_{d,j}, \; j = 1, \dots, n \tag{14}$$
$$t_{s,j} \geq 0, \; j = 1, \dots, n \tag{15}$$

This multi-objective formulation considers two conflicting objectives in a single equation. The first objective

$$f_1 = t_{s,n+1} \tag{16}$$

represents the project termination and will be minimized. Simultaneously, the second objective

$$f_2 = \sum_{j=1}^{n} t_{d,j} \tag{17}$$

compute the duration of individual tasks and will be maximized. This quantity is maximized to implicitly reduce the associated cost.

From objective functions f_1 and f_2 follows that the design variable is the time t_d to perform each task. They are bounded by the early time, t_e, and the late time, t_l. As a result, $t_e \leq t_d \leq t_l$, and the optimization problem is

$$\min t_{s,n+1} - \sum_{j=1}^{n} t_{d,j} \tag{18}$$

Subject to

$$t_{s,i} \geq t_{s,j} + t_{d,j}, \text{ for some } i \text{ and } j \tag{19}$$

$$t_{s,n+1} > t_{s,j} + t_{d,j},\ j = 1, \dots, n \tag{20}$$
$$t_{s,j} \geq 0,\ j = 1, \dots, n \tag{21}$$
$$t_{e,j} \leq t_{d,j} \leq t_{l,j},\ j = 1, \dots, n \tag{22}$$

It is necessary to include scaling factors in the objective function because values may have different magnitudes. Priority of some tasks may also be considered in a similar fashion. According to Eq. (6), these parameters can be taken into account through a weighing factor *w*. Then, the optimization problem is

$$\min w_{n+1} t_{s,n+1} - \sum_{j=1}^{n} w_j t_{d,j} \tag{23}$$

Subject to

$$t_{s,i} \geq t_{s,j} + t_{d,j},\ \text{for some } i \text{ and } j \tag{24}$$
$$t_{s,n+1} > t_{s,j} + t_{d,j},\ j = 1, \dots, n \tag{25}$$
$$t_{s,j} \geq 0,\ j = 1, \dots, n \tag{26}$$
$$t_{e,j} \leq t_{d,j} \leq t_{l,j},\ j = 1, \dots, n \tag{27}$$

The proposed formulation is in the field of linear program with real variables.

4. NUMERICAL RESULTS

Numerical simulations were carried out to evaluate the viability of the proposed formulation. An extra weighting factor α was included in the formulation to set different priorities to project conclusion time

$$f_1 = w_{n+1} t_{s,n+1} \tag{28}$$

and project cost

$$f_2 = \sum_{j=1}^{n} w_j t_{d,j} \tag{29}$$

The corresponding objective function is

$$F = \alpha f_1 - (1 - \alpha) f_2 \tag{30}$$

The optimization problems were solved by means of a *lp_solve* software (lp_solve, 2009). Customizations where included by using a GNU C++ compiler. A PC Core Duo ® 1.6 GHz computer running Linux was used.

Three sets of experiments were carried out. In the first case, 10 test problems of dimension n=500 and 10 test problems of dimension n=1000 were generated with arbitrary precedence relationship, cost and priorities. Objective function f_1 and f_2 have the same weight α=0.5. Results are shown in Tab. 1.

According to Tab. 1, columns "Initial" and "Optimal" shows the values of f_1 and f_2 before and after the optimization, respectively. Column "Deviation" indicates the improvement achieved by f_1 and f_2, respectively. A deviation of 0.3 indicates that f_1 was decreased to 30% of its initial value. A deviation of 0.7 indicates that f_2 was decreased to 70% of its initial value.

This data shows that a reduction about 70% in the total time required to finish the project can be achieved while the cost (or resource) is decreased by 30%.

A problem with n=500 tasks was evaluated in 5 seconds, while a problem with n=1000 tasks was evaluated in 20 seconds.

Table 1. Numerical experiments where cost and time have the same priority.

	Initial		Optimal		Deviation			
N	*f1*	*f2*	*f1*	*f2*	*f1*	*f2*	α	CPU [s]
500	1084135	3511.6	323622.3	2381.0	0.30	0.68	0.5	4.960
500	1096060	3646.4	273764.3	2369.5	0.25	0.65	0.5	4.970
500	1070226	3401.5	374383.0	2424.1	0.35	0.71	0.5	4.960
500	1066090	3286.7	374083.9	2359.1	0.35	0.72	0.5	5.010
500	1087743	3560.7	327005.2	2478.3	0.30	0.70	0.5	4.970
500	1085918	3373.6	342365.9	2358.5	0.32	0.70	0.5	4.980
500	1087486	3549.3	324435	2365.8	0.30	0.67	0.5	4.970
500	1091314	3683.9	281564	2344.9	0.26	0.64	0.5	4.990
500	1097285	3574.9	306978.6	2326.1	0.28	0.65	0.5	4.970
500	1144058	3544.6	327508.7	2311.6	0.29	0.65	0.5	4.960
1000	4338819	7120.0	1206007	4763.6	0.28	0.67	0.5	19.610
1000	4427215	6946.9	1384693	4752.8	0.31	0.68	0.5	19.620
1000	4168668	6966.4	1086398	4555.4	0.26	0.65	0.5	19.750
1000	4296253	6827.4	1308107	4708.0	0.30	0.69	0.5	19.740
1000	4059817	6542.2	1221632	4470.7	0.30	0.68	0.5	19.650
1000	4064180	6834.9	1082149	4622.0	0.27	0.68	0.5	19.620
1000	4030020	6785.3	1111422	4619.5	0.28	0.68	0.5	19.740
1000	4165613	6864.0	1127778	4590.4	0.27	0.67	0.5	19.760
1000	4289263	6913.5	1263862	4784.2	0.29	0.69	0.5	19.720
1000	4178773	6646.1	1278270	4597.2	0.31	0.69	0.5	19.750

In the second set of experiments, the effect of changes on the weighting factor α is considered. Results obtained from 100 experiments, where n=500 and $\alpha \in (0,1)$, are presented in Fig. 1.

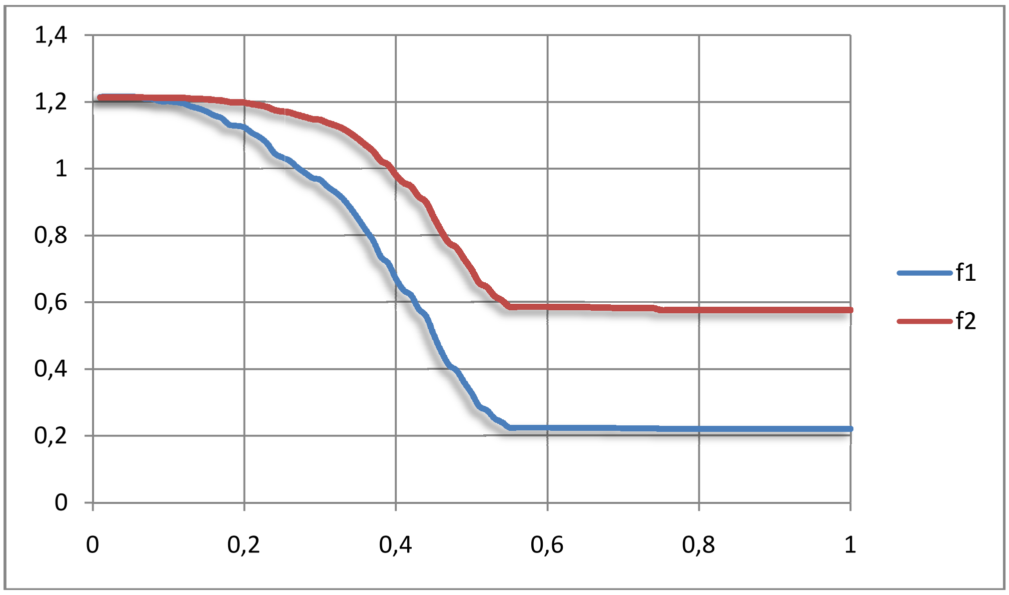
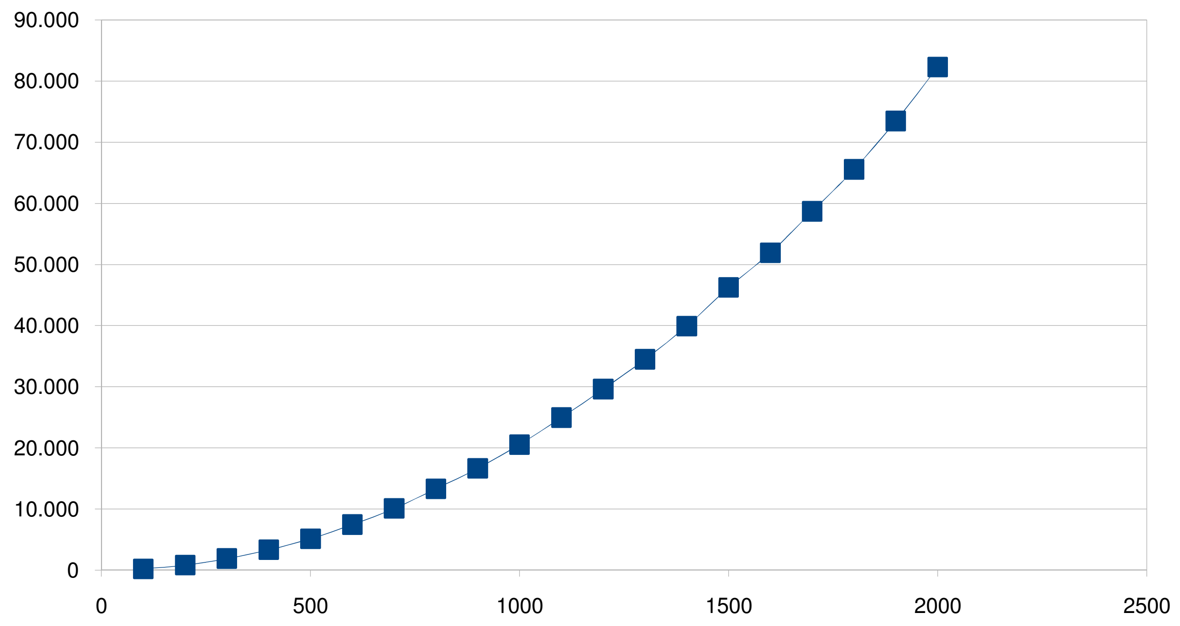


Figure 1. f_1 and f_2 deviation for $\alpha \in (0,1)$.

Figure 1 shows that better optimization is achieved if α has a value greater or equal 0.6. This behavior was found in all the problems evaluated by the authors, that is, optimization is more effective when the value of α is near 1. However, the parameter can be tuned in the interval (0, 1) to adjust the relevance of each objective. Furthermore, the evaluation of different parameters also gives a macroscopic view that may support decisions of a manager.

A third set of experiments was proposed to evaluate the CPU time required to solve problems of different dimensions. Problems of size between n=100 and n=2000 were considered. Results are shown in Fig. 2.



According Fig. 2, a problem of dimension *n*=500 is solved in 5 seconds. A problem of dimension *n*=2000 is solved in 82 seconds. The CPU time is closely related to the size of the problem and does not have a large deviation, since a deterministic algorithm is used.

5. CONCLUSION

In this paper a new formulation for task scheduling was proposed. Initially, a brief discussion about the importance of task scheduling and a review of related papers was presented. Next, the general formulation of linear program and a theory to deal with multiple objectives where addressed.

The proposed formulation considers two objectives: total time required to finish a project and cost associated to the execution of all tasks. It was supposed cost is inversely proportional to time spend in a task. This assumption permits the representation of the optimization as a linear programming problem. The absence of integer variables means that this formulation can be solved by a standard simplex method.

A strong point of the presented methodology is the low CPU time required to solve a problem with a large number of tasks. Due the simplicity of the formulation the global minimum is always found through a deterministic procedure, which is inherent to linear models in a convex set.

Numerical results demonstrated the effectiveness of the proposed methodology in reducing the overall time required to conclude the scheduling and simultaneously reducing the cost of the project. Better performance of time or cost can be manually tuned by adjusting weighting factors. This behavior is a feature of multiple objective problems, in agreement with Pareto theory.

Later, a comparative study about CPU time and problem size was shown. A problem with 2000 variables was solved in less than 2 minutes. The reasonable time required to solve a given problem enables the use of the proposed methodology as an interactive tool for scheduling. Furthermore, optimal solution does not depend of the initial configuration. Optimization process finds the optimal task duration independently of its initial value. Key information is the minimal and maximal time to perform each task (the early and late times, respectively).

Through changes in the value of the cost it is also possible to set different priorities to the realization of each task or a group of tasks. It enables the avoidance of undesired scenarios and a fine tuning of preferred situations.

Finally, further research directions are extending the analysis to deal with stochastic information, as uncertainty in parameters, and explicit constraint on resource usage.

Due the low computational cost, interactive behavior and quality of the answer, the authors believe this is a useful tool for scheduling analysis.

6. ACKNOWLEDGEMENTS

The authors are thankful to Dr. Gilberto da Cunha Trivelato and Dr. Sérgio Butkewitsch Choze. They proposed challenging ideas and motivating discussions on this topic.

7. REFERENCES

Ahmadi, J. h., Almadi, R. h., Dasu, S. and Tang, C. S., 1992, “Batching and Jobs on Batch and Discrete Processors”, Operations Research, Vol. 40, pp. 750-763.

Baker, K. R., 1974, “Introduction to Sequencing and Scheduling”, John Wiley& Sons, New York.

Blackstone, J.H., Phillips, D.T., Hogg, G.L., 1982, “A state-of-the-art survey of dispatching rules for manufacturing job shop operations”, Int J Prod Res, pp. 27-45.

Blazewicz, J., Ecker, K. H., Pesch E., Schmidt G., and Weglarz, J., 1996, “Scheduling computer and manufacturing processes”, Springer Verlag, Berlin.

Carlier, J., Pinson, E., 1989, “An Algorithm for Solving the Job-Shop Problem”, Manage Sci, Vol. 35, pp. 164-176.

CMMI, 2006, Capability Maturity Model Integration, Version 1.2, Carnegie Mellon University Software Engineering Institute. <http://www.sei.cmu.edu/publications/documents/06.reports/06tr008.html>

Coffman Jr, E. G.,1976, “Scheduling in Computer and Job Shop Systems”, J.Wiley, New York.

Coffman Jr, E., Nozari, A. and Yannakakis, M., 1989, “Optimal Scheduling of Products with Two Subassemblies of a Single Machine”, Operations Research, Vol. 37 pp. 426-436.

Conway, R. W., Maxwell, W.L. and Miller, L.W., 1967, “Theory of Scheduling”, Addison Wesley, Reading, Mass., USA.

Deb, K., 2001, “Multi-Objective Optimization using Evolutionary Algorithms”, John Wiley, New York.

Eschenauer, H. , Koski, J. and Osyczka, A., 1990, “Multicriteria Design Optimization”, Berlin, Springer-Verlag.

French, S., 1982, “Sequencing and scheduling: an introduction to the mathematics of the job-shop”, Ellis Horwood Limited, New York.

Garey, M.R., Johnson, D.S., Sethi, R., 1976, “The complexity of flow shop and job-shop scheduling”, Mathematics of Operations Research, Vol. 1, pp. 117-129.

Graham, R. L., Lawler, E. L., Lenstra, J. K., Rinnooy Kan, A. H. G., 1979, “Optimization and approximation in deterministic sequencing and scheduling: A survey”, Annals of Discrete Mathematics.

Holthaus, O., Rajendran, C., 1997, “Efficient dispatching rules for scheduling in a job shop”, Int J Prod Econ, pp. 87-105.

Ikura, Y. and Gimple, M., 1986, “Efficient scheduling algorithms for a single batch processing machine”, Operations Research Letter, Vol. 5, pp. 61-65.

Jain, A.S., Meeran, S., 1999, “Deterministic job-shop scheduling: Past, present and future”, Eur J Oper Res, Vol. 113, pp. 390-434

Julien, F., and Magazine, M., 1989, “Batching and Scheduling Multi-Job Orders”, CORS/TIMS/ORSA Vancouver Bulletin.

Kacem, I., Hammadi, S., Borne, P., 2002, “Approach by localization and multiobjective evolutionary optimization for flexible job-shop scheduling problems”, IEEE T Syst Man Cy C, 32, pp. 1-13.

Kolonko, M., 1999, “Some new results on simulated annealing applied to the job shop scheduling problem”, Eur J Oper Res, Vol. 113, pp. 123-136.

Koopmans, T.C., 1951, “Activity Analysis of Production and Allocation”, Cowles Commission for Research in Economics, Monograph, John Wiley and Sons, Vol. **13,** New York.

Kuhn, H.W., Tucker, A.W., 1951, “Nonlinear Programming”, Proceedings of the Second Berkeley Symposium on Mathematical Statistics and Probability, University of California Press, Berkeley, California, pp. 39–54.

Lawler, E. L., Lenstra, J.K., Rinnooy Kan, A. H. G., and Shmoys, D.B., 1993, “Sequencing and Scheduling: Algorithmus and Complexity”, volume 4 of Handbook in Operations Research and Managment Science, North-Holland, Amsterdam.

Lenstra, J. K., 1977, “Sequencing by enumerative methods”, Mathematical Centre Tracts, pp. 69.

lp_solve Reference Guide, 2008 < http://lpsolve.sourceforge.net/5.5/>

Nowicki, E., Smutnicki, C., 1996, “A fast taboo search algorithm for the job shop problem”, Manage Sci, Vol. 42, pp. 797-813.

Panwalkar, S.S., Iskander, W., 1977, “A Survey of Scheduling Rules”, Oper Res pp. 45-61.

Pardalos, P. M. and Du, Ding-Zhu., 2008, “PARETO OPTIMALITY, GAME THEORY AND EQUILIBRIA”, Springer Optimization and Its Applications, Springer Science.

Pinedo, M., 1995, “Scheduling: Theory, Algorithms and Systems”, Prentice Hall, Englewood Cliffs N.J.

Pinedo, M., Chao, X., 1999, “Operations scheduling with applications in manufacturing and services”, MCGraw- Hill, chapter 3.

Rinnooy Kan, A.H.G., 1976, “Machine Scheduling Problems: Classification, Complexity and Computations”, Martinus Nijhoff, The Hague.

Santos, C., and Magazine, M, 1985, “Batching in Single Operation Manufacturing System”, Operations Research Letter, Vol. 4, pp. 99-103.

Tanaev, V. S., Gordon, V. S., and Shafransky, Y. M., 1994, “Scheduling theory. Single-stage systems”, Kluwer Academic Publishers., Vol. 1, Dordrecht.

Tanaev, V. S., Sotskov, Y. N., and Strusevich, V. A., 1994, “Scheduling theory. Multi-stage systems”, Mathematics and its Applications, Kluwer Academic Publishers Group, Vol 285, Dordrecht.

Tang, C.S., 1990, “Scheduling Batches on Parallel Machines with Major and Minor Setups”, European Journal of Operational Research, Vol. 46, pp. 28-37.

Yamada, T., Nakano, R., 1996, “A fusion of crossover and local search”, Proceedings of The IEEE International Conference on Industrial Technology, pp. 426-430.

8. RESPONSIBILITY NOTICE

The authors are the only responsible for the printed material included in this paper.