HEURISTICS FOR PARTIAL AND TOTAL DYNAMIC W-T PROBLEMS IN SINGLE MACHINE ENVIRONMENTS

Lasso M., Pandolfi D., De San Pedro M., Villagra A. Vilanova G.
Proyecto UNPA-29/B032[1]
División Tecnología
Unidad Académica Caleta Olivia
Universidad Nacional de La Patagonia Austral
Ruta 3 Acceso Norte s/n
(9011) Caleta Olivia – Santa Cruz - Argentina
e-mail: {mlasso,dpandolfi,edesanpedro,avillagra}@uaco.unpa.edu.ar; gvilanov@satlink.com
Phone/Fax : +54 0297 4854888

Gallard R.
Laboratorio de Investigación y Desarrollo en Inteligencia Computacional (LIDIC)[2]
Departamento de Informática
Universidad Nacional de San Luis
Ejército de los Andes 950 - Local 106
(5700) - San Luis -Argentina
e-mail: rgallard@unsl.edu.ar
Phone: +54 2652 420823
Fax : +54 2652 430224

Abstract

In dynamic scheduling arrival times as well as some or all job attributes are unknown in advance. Dynamism can be classified as partial or total. In simplest partially dynamic problems the only unknown attribute of a job is its arrival time r_j. A job arrival can be given at any instant in the time interval between zero and a limit established by its processing time, ensuring to accomplish it before the due date deadline. In totally dynamics problems, other job attributes such as processing time p_j, due date d_j, and tardiness penalty w_j, are also unknown.

Our research proposes different approaches for resolution of Weighted Tardiness dynamic problems (partial and total) in a single machine environment. A first approach uses, as a list of dispatching priorities a final schedule, found as the best by another heuristic for a similar static problem: same job features, processing time, due dates and weights. A second approach uses as a dispatching priority the order imposed by a partial schedule created, at each decision point, by another heuristic. The details of implementation of the proposed algorithms and results for a group of selected instances are discussed in this work.

[1] The Research Group is supported by the Universidad Nacional de La Patagonia Austral.
[2] The LIDIC is supported by the Universidad Nacional de San Luis and the ANPCYT (National Agency to Promote Science and Technology).

1. Introducción

Manufacturing organizations are frequently subject to several sorts of changes, such as new job releases, machine breakdowns, job cancellation and due date or processing time changes. Due to their dynamic nature, real scheduling problems are computationally complex and the time required to compute an optimal solution increases exponentially with the size of the problem [10]. In *Single Machine Scheduling Problems* (SMSP) a set of jobs should be planned on a single machine. Quite often the single-machine problem appears as an elementary component in a larger scheduling problem [3]. As other scheduling problems, SMSP can be classified as *static* or *dynamic*. In *static* problems, all job attributes are known before scheduling starts, while in *dynamic* problems, job release times are unknown (partially dynamic) or all job properties are unknown (totally dynamic).

Evolutionary algorithms have been successfully applied to solve scheduling problems [8, 9, 11]. Current trends in evolutionary algorithms make use of multiparent [4] and multirecombined approaches [5, 6]. The latter, known as MCMP (Multiple-Crossovers-on-Multiple-Parents), allows a better balance between exploration and exploitation of the search space. A new variant of this approach applied to static scheduling problems [8, 9] is known as MCMP-SRI. Here, an individual selected from the old population and designated as the *stud* (S), provides to the multirecombination process good features of the evolved population while a set of random immigrants (RI) provides genetic diversity to avoid premature convergence.

Dispatching rules are techniques that dictate job priorities and provide a reasonably good solution in relatively short time. An elementary rule is a function of attributes of jobs and machines. An attribute can be any property associated with a job or a machine and can also be constant or time-dependent [10]. A composition of a dispatching rule is a ranking expression that combines a number of elementary dispatching rules.

Our research proposes two approaches to lead with partial and total dynamism. For partial dynamism, to decide which job to process *Dyna-S* make use of schedules previously found for the static case as a clue to build good schedules for the dynamic problem, where changes are related only to the unpredicted job arrival times. For total dynamism *Dyna-H* establishes the priority of jobs in the waiting queue each time the resource becomes available resorting to different heuristics.

2. Dynamic scheduling for Single Machine Problems

In the weighted-tardiness single machine problem n jobs should be planned without interruption. For each job j (j = 1,...,n) with processing time p_j and due date d_j, exists a penalty w_j for each tardy unit. The objective of this problem is to find a sequence that minimizes:

$$\sum_{j=1}^{n} w_j T_j$$

where the tardiness of a job, is given by $T_j = \max\{C_{j-1} + p_j - d_j, 0\}$. Even with this simple formulation, this model leads to an optimization problem that is NP-Hard [7, 10].
In the static weighted-tardiness problem all jobs and their properties are simultaneously available for processing in time zero, which represents in most cases a not very common situation. In scheduling problems involved with real production, the environments are dynamic, at least in the sense that jobs arrivals can occur at unpredicted times.
However, once the jobs arrive to the system for their processing producing a waiting queue, this can be considered as a static case for the determination of the next task to be allocated to the

machine. Due to this characteristic, the study of the static case is important since the approach that provide good solutions can be a suitable surrogate for the cases of dynamism. We can consider the static weighted tardiness problem, as a relaxation of the dynamic problem, where all arrival times are equal to zero, that is $r_j = 0$ for all j.

3. Algorithms for W-T Dynamic Scheduling

For the *partially dynamic case* we propose *Dyna-S*, which uses as a dispatching rule the job order provided by a total schedule *S* generated by an evolutionary algorithm (*MCMP-SRI*), or by conventional heuristics (*Rachamadagu and Morton Heuristic R&M,* and *Covert*) to solve similar static cases. These variants are called *Dyna-S-EA, Dyna-S-R&M* and *Dyna-S-Covert*, respectively. To schedule a job, an arrival queue is created with those jobs whose r_j are eralier or equal than the time *t* when the machine is available for the processing. From that waiting queue, the job that appears first in the ordering of the total schedule *S,* is selected to be allocated first. Once a job is planned, it is removed from the queue. This process is repeated each time when the resource becomes available and while there are jobs in the waiting queue.

For the totally dynamic case we propose *Dyna-H,* which applies different heuristics to determine which job to schedule when the machine becomes available. Essentially in *Dyna-H*, jobs in the waiting queue are planned according to some dispatching rule, which generates a partial schedule. Again, a waiting queue is generated with those jobs whose r_j are smaller or equal than the time t when the machine is available for the processing. Now we run the heuristic to produce a partial schedule (reordering the available jobs). The algorithm uses this partial schedule as a list of dispatching priorities to schedule the next job by choosing the job in the first position of this list. Once the job is planned, it is removed from the queue. This process is repeated each time when the resource is available and while there are jobs in the waiting queue. *Dyna-H* was conceived in three different versions: using R&M (*Dyna-H-R&M*), using Covert (*Dyna-H-Covert*), or using a hybridized MCMP-SRI (*Dyna-H-EA*), as a dispatching rule to establish job priorities. Also, a decision is made depending in queue length to apply an enumerative algorithm, a conventional heuristic or an evolutionary algorithm.

4. Current and future work

For both partial and total dynamism we built our own test suite with data (p_j ,d_j , w_j) extracted from 20 selected instances of the OR-library benchmarks for the static case, with 40-jobs problem size, [1, 2]. Two types of random arrivals for each instance were generated: early, that is in the interval [0,(*dj-pj*)/2], and tardy that is in the interval [(*dj-pj*)/2,(*dj-pj*)]. Many series of runs were performed for each algorithm on each instance.
The results obtained through our current work can be summarized as follows:

- For partial dynamism and early arrivals the order of performance is the following: *Dyna-S-EA, Dyna-S-R&M* and then *Dyna-S-Covert.* This case is the most similar to the static case, the schedules obtained are also similar and the evolutionary approach performs better.
- For partial dynamism and late arrivals the order of performance is the following: *Dyna-S-R&M, Dyna-S-EA,* and then *Dyna-S-Covert.* This case differs much more from the static case and the ability of R&M to discriminate jobs according to their slack factor provides better overall performance.
- For total dynamism and early or late arrivals the order of performance is the following: *Dyna-H-EA, Dyna-H-R&M* and then *Dyna-H-Covert.* For this difficult case, where a re-

scheduling is necessary at each decision point, the hybrid algorithm inserting the problem-specific-knowledge of conventional heuristics and combining them with the ability of the evolutionary algorithm provides the best results independently of the arrival being early or late.

At the light of these results future work will be devoted to larger problems, different job arrival distributions and variants of the *Dyna-S* and *Dyna-H* approaches.

5. References

[1] Beasley J.E. "Common Due Date Scheduling", OR Library, http://mscmga.ms.ic.ac.uk/

[2] Crauwels H.A.J., C.N.Potts and L.N.Van Wassenhove "Local search heuristics for the single machine total weighted tardiness scheduling problem", Informs Journal on Computing 10, 341-350. 1998.

[3] Baker K. R., "Introduction to sequencing and scheduling" Willey New York 1974.

[4] Eiben A.E., C.H.M. Van Kemenade, and J.N. Kok, "Orgy in the computer: Multi-parent reproduction in genetic algorithms". Proceedings of the 3rd European Conference on Artificial Life, Springer-Verlag, 1995, number 929 in LNAI, pages 934-945.

[5] Esquivel S., A. Leiva, R. Gallard, "Multiple Crossover per Couple in Genetic Algorithms", Proceedings of the Fourth IEEE Conference on Evolutionary Computation (ICEC'97), Indianapolis, USA, April 1997, pp 103-106.

[6] Esquivel S., H. Leiva,.R. Gallard, "Multiple crossovers between multiple parents to improve search in evolutionary algorithms", Proceedings of the Congress on Evolutionary Computation (IEEE). Washington DC, 1999, pp 1589-1594.

[7] Morton T., D. Pentico, "Heuristic scheduling systems", Wiley series in Engineering and technology management. John Wiley and Sons, INC, 1993.

[8] Pandolfi D., Vilanova G., De San Pedro M., Villagra A., Gallard R., "Adaptability of Multirecombined Evolutionary Algorithms in the single-machine common due date problem." Proceedings of the Multiconference on Systemics, Cybernetics and informatics. Orlando, Florida July 2001.

[9] Pandolfi D., De San Pedro M., Villagra A., Vilanova G., Gallard R.- "Studs mating immigrants in evolutionary algorithm to solve the earliness-tardiness scheduling problem" . In Cybernetics and Systems of Taylor and Francis Journal, Vol. 33 Nro. 4, pp 391-400 (U.K.) June 2002.

[10] M. Pinedo, "Scheduling: Theory, Algorithms and System." First edition Prentice Hall, 1995.

[11] R.V. Rachamadugu, T.E. Morton, "Myopic heuristics for the single machine weighted tardiness problem". GSIA, Carnigie Mellon University, Pittsburgh, PA. 1982., Working paper 30-82-83.