A New Framework for Dynamic Deterministic Job-Shop Scheduling Problems Using Genetic Algorithms

Ana Madureira and Carlos Ramos
Dept. de Engenharia Informática, Instituto Superior de Engenharia do Porto, Portugal
anamadur@dei.isep.ipp.pt, csr@dei.isep.ipp.pt

Sílvio do Carmo Silva
Dept. de Produção e Sistemas, Universidade do Minho, Portugal.
scarmo@dps.uminho.pt

Abstract

The problem of finding good solutions to scheduling problems is very important to real manufacturing systems, since the production rate and production costs are very dependent on the schedules used for controlling the work in the system. Most research in scheduling focuses on optimisation of static problems, where all problem data are known before scheduling starts. However many real world optimisation problems are dynamic, in which changes may occur continually.

This paper presents a scheduling system, based on Genetic Algorithms for the resolution of the deterministic Job-Shop Scheduling Problem (JSSP), which considers the existence of different job release dates and job due dates, and different assembly levels. This approach is based on a decomposition of the Job-Shop Scheduling Problem into a series of deterministic Single Machine Scheduling Problem (SMSP). A Genetic Algorithm (GA) solves each SMSP, and the obtained solutions are integrated at the end. A coordination mechanism is proposed.

1. Introduction

So far, most research on scheduling has been focused mainly on optimising one particular performance measure, like makespan or tardiness, normally reflecting some kind of cost. It is assumed that all problem data are known before scheduling has to take place and no change ever happens. However real world applications operate in dynamic environments frequently subject to several kinds of random occurrences and perturbations, such as new job arrivals, machine breakdowns, employees sickness, jobs cancellation and due date and time processing changes, causing that the original schedule becomes unfeasible.

Due to their dynamic nature, real scheduling problems have an additional complexity in relation to static ones. In many situations these problems, even for apparently simple situations, are hard to solve, i.e. the time required to compute an optimal solution increases exponentially with the size of the problem [4].

If all jobs are known and ready to process before processing starts, the scheduling problem is called *static*, while if job release times are not fixed at a single point in time, i.e. jobs arrive to the system at different times and its parameters can change, the problem is called *dynamic*. Dynamic scheduling problems can also be classified as *deterministic*, when release times and all other parameters are known and fixed, and *non-deterministic*, when jobs can arrive over time and some or all parameters are uncertain.

The proposed approach deals with these two cases: deterministic and non-deterministic scheduling problems. For such class of problems, the goal is no longer to find a single optimum, but rather to continuously adapt the solution to the changing environment.

When a change in the environment happens rescheduling is needed, and the existence of a good near-optimal schedule, which is easy to modify will be in some situations preferable to an optimal, which cannot be modified.

It is our conviction that, the more randomness the sys-tem is subject to, the simpler the scheduling approach should be.

The purpose of this paper is to describe an approach based on Genetic Algorithms for solving a class of real time scheduling problems, where the products (jobs) to be processed have due dates, release dates, different assembly levels. A coordination mechanism is presented.

The paper is structured as follows: Section 2 provides a description of the considered scheduling problem. The proposed approach for dynamic scheduling is described on section 3. Finally, the paper presents conclusions and some ideas for future work.

2. Problem Definition

Almost all practical scheduling problems can be described in terms of the Job-Shop Scheduling Problem. Usually as restricted or relaxed versions of this classic combinatorial optimisation problem. The general Job-Shop Scheduling Problem (JSSP) of size n x m can generally be described as a decision-making process concerning about the allocation of a limited set of m re-sources over time to perform a set of n tasks or jobs. In manufacturing systems, scheduling typically concerns allocating a set of machines to perform a set of jobs within a certain time period. Each job has a specified processing order through the machines, i.e. a job is composed of an ordered list of operations, which are characterised by the machine required, and the processing time. Several constraints on jobs and machines can be defined:

- machines are always available and never break down;
- there are no precedence constraints among operations of the different jobs;
- the operations processing can not be interrupted and each machine can process only one job at a time;
- each job can be processed only on one machine at a time;
- setup times are independent of the schedules and are included in processing times
- processing times, due dates and technological constraints are deterministic and known in advance.

A schedule is an assignment of jobs over time in the respective machines. The objective is to find a schedule, which optimises some performance measure. The most extensively investigated is the makespan, the time elapsed from the beginning of processing until the last operation has been finished. In our approach, the makespan objective is not realistic, since it does not consider due dates, and it is not well suited for dynamic environments where jobs can arrive continuously over time. For literature on this subject see for example [2][3][4].

Real-world scheduling problems are very different from the mathematical models studied by researchers in academy. In reality many scheduling problems are not so well defined. The environment may be dynamic with new jobs arriving at unpredictable intervals, machine breakdowns, jobs cancellation and due date and time processing changes.

Additional constraints will be considered in our approach for the JSSP: the existence of different job release dates, prior to which no processing of the job can be done; the existence of job due dates, a time by which the processing of the job is supposed to be finished and different assembly levels for the jobs.

3. GA-based Scheduling System for Deterministic Scheduling Problems

An operation O_{ijk} is described by the triplet (i, j, k), where i defines the machine where the operation is processed, j the job which belongs, and k the graph precedence operation level (level 1 correspond to initial operations, without precedents).

In a previous work the adequacy and efficiency of the Genetic Algorithms (GA), for the static Single Machine Scheduling Problem (SMSP) was studied [8][9]. This work starts by studying the

performance of two interrelated GA for the minimisation of the static weighted tardiness. One is a single start GA, the other, called MetaGA, is a multi-start version GA. The performance was evaluated, on the basis of the quality of scheduling solutions obtained for a limit on computation time. The obtained results show that these GA perform well for the studied cases, being possible to find good solutions in a short period of time, i.e. a few minutes of CPU time. Substantial performance improvements with the MetaGA were obtained in relation to single start GA.

The proposed approach considers additional constraints, which characterise real manufacturing systems: the existence of different job release dates; the existence of job due dates and different assembly levels for the jobs.

We select Genetic Algorithms because they have proven an acceptable performance on Job-Shop problems, see for example [6][7]. Other traditional scheduling techniques [1][5], such as, Shifting Bottleneck and Branch-and-Bound have sometimes presented better results than GA on static scheduling problems, but they are not well suited for dynamic scheduling problems.

The proposed approach consists on a centralised GA-based scheduling system for each machine involved in the process. The original deterministic Job-Shop Scheduling Problem is decomposed into a series of deterministic Single Machine Scheduling Problems solved one at a time, consecutively. Each machine is considered as $1|r_j|L_{max}$ problem with release dates and due dates and solved by a Genetic Algorithm. The obtained solutions are then incorporated into the main problem.

Table 1. Scheduling algorithm for deterministic JSSP

Step 1	Define the completion times estimates (due dates) for all operations of each job.
Step 2	Define the interval of estimates starting times (release times) for all operations of each job.
Step 3	Define all SMSP $1\|r_j\|L_{max}$ based on information defined on Step1 and Step 2.
Step 4	Solve all SMSP $1\|r_j\|L_{max}$ with those release and due dates using a GA.
Step 5	Integrate all the optimal or near-optimal solutions into the main problem.
Step 6	Verify if they constitute a feasible solution and terminate with a local optimum; otherwise it is necessary to apply a repair mechanism.

The operation completion times C_{ijk} are defined considering job due dates and the respective processing times. The operation completion times C_{ijk} are calculated subtracting the operation processing time from the operation completion times of the immediately succeeding operation. This procedure is started from final operation, which correspond as completion time the respective job due date.

$$C_{ijk-1} = C_{ijk} - p_{ijk} \quad (1)$$

where k is the operation level on the precedence graph, i is the machine where the operation is processed and j the job which operation belongs.

The estimate operation starting times interval $[t_{ijk}, T_{ijk}]$ is defined considering the job due date and the operation processing times. The earliest starting time t_{ijk} corresponds to the time from which the operation processing can be started. The latest starting time T_{ijk} correspond to the time in which the operation processing must be started, in order to meet the estimate completion time (due date). When an operation has more than one precedent operation (multi-level structure) it corresponds to the interval intersection from precedent operations correlated by the respective processing times.

At this phase, it will be only considered technological precedent constraints of operations and job due dates, for defining completion and starting times.

The extensions for the resolution of non-deterministic version are in developing at this moment [10].

3.1. Coordination Mechanism

The objective of the coordination mechanism is to repair the schedule obtained up to the step 5 of the scheduling algorithm, described above. The repairing is done in order to obtain a feasible schedule through coordination of machine activity, having into account job operation precedence and all other problem constraints and, at the same time, keeping job allocation order in each machine as given by the schedule to be repaired. The objective is minimizing the maximum lateness.

Essential to the coordination mechanism is to establish the starting Ti and the completion times Tc for each operation. These times are related being that the starting time Ti for each operation must be equal to the larger of the following two values: completion time of the immediately precedent operation in the job and completion time of the immediately precedent operation in the machine.

4. Conclusions

In this paper it was proposed an approach based on GA for solving the dynamic deterministic scheduling problem, where the products (jobs) to be processed have due dates. The union of the obtained local optimal, one for each $1|r_j|Lmax$ problem could not produce a feasible solution. This is due to the processing overlapping operations belonging to the same job. In order to repair the obtained solution a coordination mechanism is proposed. The objective is to coordinate the machine occupation times with job precedence relationships.

The approach is already prepared to deal with dynamic non-deterministic problems, where random disturbances can occur and frequent rescheduling of work is necessary [10].

More research is needed in testing the performance of the proposed mechanisms under dynamic Job-Shop environments subject to several random perturbations.

References

[1] Adams, Joseph, Balas, Egon and Zawack, Daniel (1988), *The Shifting Bottleneck Procedure for Job Shop Scheduling*, Management Science, Vol. 34, nº 3 Março, USA.

[2] Blazewicz, J., et al (2001), *Scheduling Computer and Manufacturing Processes*, Springer, 2nd edition, New York.

[3] Brucker P. (2001), *Scheduling Algorithms*, Springer, 3rd edition, New York.

[4] Morton, E. T., and Pentico, D. W. (1993), *Heuristic Scheduling Systems*, John Wiley & Sons.

[5] Jain, A. S. and S. Meeran (1999), *Deterministic Job Shop scheduling: past, present and future*, European Journal of Operational Research 113, 390-434.

[6] Kumar, N. S. Hemant and Srinavasan, G. (1996), *A genetic algorithm for Job Shop scheduling- a case study*, Computers Industry, 155-160.

[7] Lee, C.Y., Piramuthu, S. and Tsai, Y.K. (1997), *Job Shop Scheduling With a Genetic Algorithm and Machine Learning*, Int. J. Prod. Res., vol.35, nº4, pp.1171-1191.

[8] Madureira, Ana M., Ramos, C. and Silva, S. C. (1999), *A Genetic Approach to Dynamic Scheduling for Total Weighted Tardiness Problem*, PLANSIG'99 (18th Workshop of the UK Planning and Scheduling Special Interest Group), Manchester,UK.

[9] Madureira, Ana M., Ramos, C. and Silva, S. C. (2000), *A Genetic Algorithm for The Dynamic Single Machine Scheduling Problem*, 4th IEEE/IFIP International Conference on Information Technology for Balanced Automation Systems in Production and Transportation, Berlin, Germany.

[10] Madureira, Ana M., Ramos, C. and Silva, S. C., "A Genetic Approach for Dynamic Job-Shop Scheduling Problems", 4th MetaHeuristics Int'l Conf. (MIC'2001), Porto, Julho 2001.