Bi-objective scheduling of multi-processor tasks on two dedicated processors

Adel KACEM[1] **and Abdelaziz DAMMAK**[2]
Modeling and Optimization for Decisional, Industrial and Logistic Systems Laboratory
Faculty of Economics and Management, Sfax, Tunisia.
[1] adel.kacem@gmail.com
[2] abdelaziz.dammak@fsegs.rnu.tn

Abstract: In this work, we study the problem of scheduling multi-processor tasks on two dedicated processors. The aim is to minimize the makespan and the total tardiness. This NP-hard problem requires using well-adapted methods. Thus, we constructed lower bounds for each criterion and a genetic algorithm adapted for the multi-criteria case was applied to solve this problem. Our numerical tests showed the effectiveness of the proposed algorithms.

Keywords: Scheduling, Optimization, Multi-Objective.

I. Introduction

We deal with the problem of scheduling multi-processor tasks on two dedicated processors on which the assignment of tasks is fixed. Indeed, we have three categories of tasks. The first class includes those processed only by the first processor. The second type of tasks contains those processed by the second processor, whereas the remaining ones need simultaneously both processors to be processed.

This problem represents a practical issue in the computer control systems where a task is executed in several copies on different processors in order to ensure better safety of the system. The various identical copies of the same task are treated as a single task using simultaneously many processors. In the production management, we can state the case where a task execution requires several operators. [Drozdowski, 1996].

In this study, we are interested in minimizing the makespan and the total of tasks tardiness in case where each task *j* has a due date d_j beyond which it is considered late. This NP- hard problem necessitates the use of well- adapted methods.

More specifically, we aim at developing new heuristic and meta-heuristic methods. We will particularly study the design of genetic algorithms efficiently used in the resolution of optimization problems in the logistics sector. This can be achieved by the good quality solutions obtained by applying such methods in a short computation time.

In this paper, we study the problem of bi-objective scheduling of multi-processor tasks on two dedicated processors. The aim is to minimize the makespan and the total tardiness. Thus, we operate and combine three ideas to build a new lower bound for the total of tardiness. Then, we exploit the aggregative technique to introduce a new selection method in order to adapt the genetic algorithm to the multi-objective case and to develop two heuristics to enrich and diversify the initial population. Finally, we compare and evaluate the obtained results with Pareto technique and we present some concluding remarks.

II. Lower bounds

In this section, we study the lower bound for the problem of minimizing the makespan and the total of the Tardiness of multi-processor tasks on two dedicated processors.

1) Lower bound Lb_C for the problem $P2\left| fix_j, r_j \right| C_{\max}$

Manaa et *al* proposed two ideas and combined them to construct a lower bound:

- The idea of dividing the problem into two sub-problems to a processor by relaxing the studied problem.
- The idea of Bianco et al (1997) to determine an optimal solution to minimize the makespan for one-processor problem.

The relaxation of the studied problem allows obtaining two simple problems:

a) Scheduling tasks that necessitate using simultaneously both processors and tasks that require first processor on the first processor.
b) Scheduling tasks that require employing simultaneously both processors and tasks that necessitate second processor on the second processor.

The optimal solutions of problems (a) and (b) can be found by scheduling tasks according to the order of their release dates.

The lower bound Lb_C for the studied problem corresponds to the solution among the solutions of problems (a) and (b).

2) Lower bound Lb_T for the problem $P2|fix_j, r_j|\sum T_j$

In this study, we operate and combine three ideas to build a lower bound:

- The idea of reducing the problem in two sub-problems on one processor by partitioning the bi-processor tasks.
- The idea of under-estimating the completion time of the tasks (initially suggested by Chu (1992)).
- The idea of calculating the lower bound by assigning the due dates to the reduced completion time (originally proposed by Rebai et al (2010) for another scheduling problem.

The first step of this lower bound is to divide the bi-processor tasks into two mono-processor tasks; each of which is executed on one of the two processors. Consequently, we obtain two independent problems on each processor. On the first processor P_1, we consider the n_1 mono-processor tasks J_j with a weight $\lambda_j^1 = 1$, and the n_{12} bi-processor sub-tasks J_j on processor P_1 having a weight $\lambda_j^1 = \lambda$ with $\lambda \in [0,1]$.

Similarly, we consider, on the second processor P_2, the n_2 mono-processor tasks J_j with a weight $\lambda_j^2 = 1$. However, the n_{12} bi-processor sub-tasks J_j on the processorP_2, have a weight $\lambda_j^2 = 1 - \lambda$

Thus, we obtain a problem on each processor $P_1|fix_j, r_j|\sum \lambda_j^1 T_j$ and $P_2|fix_j, r_j|\sum \lambda_j^2 T_j$.

The calculation of the completion times' lower-bounds is based on the following theorem:

Theorem Chu (1992): Let $C_{[i]}(\sigma)$ be the completion time of the task in the *i*th position of a feasible schedule σ . $C_i^{'}$ is the completion time of the task in the *i*th position of a feasible schedule constructed by the SRPT (Shortest Remaining Processing Time) priority rule. Chu proved that for every σ is feasible, we will obtain $C_{[i]}(\sigma) \geq C_i^{'}$.

By applying the idea of Chu, we propose for the studied problem a lower bound value for each task completion time.

The next step of computing the lower bound is based on the idea of assigning the weight and the due date of each task to completion times' lower bounds. The total is minimized by the classic Hungarian algorithm.

Let $C_{i,j}$ be the cost of assigning a reduced $C_i^{'}$ to the task J_j supposed to end at the *i*th position of scheduling. This cost can be calculated according to the following formula: $C_{i,j} = \lambda_j * \max\{0; C_i^{'} - d_j\}$

This assignment technique, presented by Rebai et al, allows us to elaborate a new lower bound. We apply the Hungarian algorithm to determine, from the assignment matrix $C_{i,j}^{P_1}$, a lower bound (*Lb1)* to solve the following problem $P_1|fix_j, r_j|\sum \lambda_j^1 T_j$.

$$Lb1 = \min \sum x_{i,j} C_{i,j}$$

$$\text{s/c} \begin{cases} \sum_i x_{i,j} = 1 \\ \sum_j x_{i,j} = 1 \\ x_{i,j} \in \{0;1\} \end{cases}$$

Respectively, we calculate Lb2 for the problem $P_2 \left| fix_j, r_j \right| \sum \lambda_j^2 T_j$.

We consider $Lb_T = Lb1 + Lb2$ as a lower bound for the problem $P2 \left| fix_j, r_j \right| \sum T_j$

III. Bi-objective study

We consider simultaneously the problem of minimization makespan and the total of the tardiness of the multi-processor tasks on two dedicated processors. Our goal is to reduce both the C_{max} and the total of the tardiness. We adapt the genetic algorithm to the multi-objective case and we develop two heuristics to enrich and diversify the initial population. Finally, we compare and evaluate the obtained results with Pareto technique and we present some concluding remarks.In this section, we describe the introduced genetic algorithm.

A. Coding

To represent the data of the studied problem, we used the coding technique of N base. This coding consists in representing an individual with N distinct numbers that correspond to a basic dial integers N.

B. The initialization

To form the diversified initial population, we used the following heuristics:

- A heuristic method was applied to compute a feasible sequence where, at each stage, it will select the best task among unprocessed ones, which ensures the performance of that individual. This heuristic method was used to create the first individual of the initial population.
- A method of random assignment that allows creating a feasible sequence was used to create the other individuals of the initial population.

C. Assessment

To evaluate the quality of individuals in a population, we presented two methods to calculate the total of task tardiness and the makespan of a given sequence.

D. The selection

The literature presents several selection techniques, such as lexicographical, by rank, random, by weighted total, etc. For our algorithm, we developed a aggregative method of selection which consists to order the tasks according to the total of two criteria harmonized by the following formula:

$$H(s) = w\left(\frac{C_{\max}(s) - \min_{x \in P}\{C_{\max}(x)\}}{\max_{x \in P}\{C_{\max}(x)\} - \min_{x \in P}\{C_{\max}(x)\}} \right) + (1-w)\left(\frac{T(s) - \min_{x \in P}\{T(x)\}}{\max_{x \in P}\{T(x)\} - \min_{x \in P}\{T(x)\}} \right)$$

Such as $w_j = \begin{pmatrix} w \\ 1-w \end{pmatrix}$ and $w \in [0..1]$.

Let $(\sigma^k)_{w_j}$ be the best sequence found by ordering the kith population based on the formula. We selected from this population the non-dominated sequences $(\sigma^{k*})_{w_j}$.

Tests carried out by Mabed et al (2001) showed that the three selection strategies (NSGA, NDS and WAR) are nearly identical for the multi-criteria flow shop problem. For this reason, we chose to compare our method of selection with the Pareto technique: transformed the population by crossing the non-dominated sequences and by muting the dominated sequences.

E. Crossing

After the selection process, we implemented the process of crossing between two parents to give birth to two children. In this case, an exchange position is randomly determined. The first part of the first child is directly obtained from the first parent. The second part is provided by respecting the order of the remaining tasks as they appear in the second parent tasks. The same process is applied on the second child by reversing the parents. For our algorithm, we implemented the one-point cross-over.

F. Mutation

In the literature, there exist several mutation techniques, such as permutation, insertion and inversion. In our case, we used the first method which consists in permuting two positions of a given individual.

G. Replacement

When the population is not improved or when it ceases to evolve, it is necessary to integrate new individuals in the population to improve the obtained results.

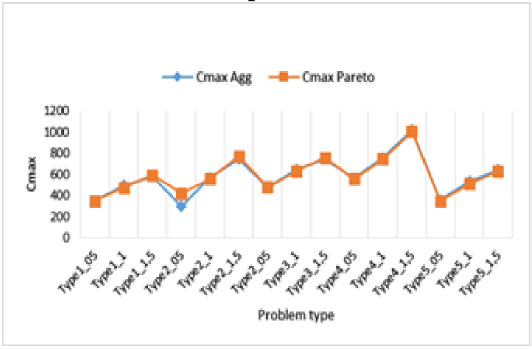
IV. Results evaluation

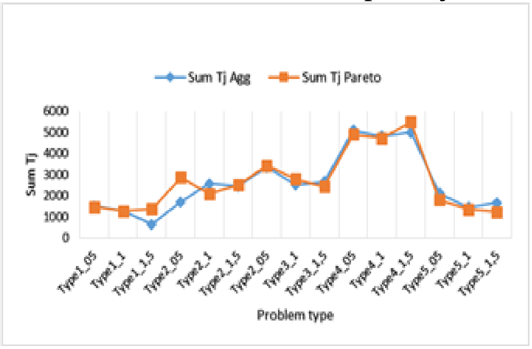
In this section, we present some experimental results provided on randomly-generated instances. Then, we evaluate the obtained results.

We used randomly generated instances by taking into account five types of problems illustrated in the table presented by Manna and Chu [Manaa, Chu, 2010].

The results found for both criterion listed in table 6. shows that the aggregative selection technique is more efficient for the problems type1 with (α=1.5), type 2 with (α=0.5; α=1.5) and type 3 with (α=0.5). For the problem type 1,2 with (α=1), type 4 with (α=0.5; α=1) and type 5, Pareto is more efficient. For other cases problems, each technique has an advantage over the other on a single criterion.

For the aggregative selection technique where the weights w = (1, 0) or w = (0, 1), we are actually studying a single criterion. The results found minimize the two criteria. This allows us to conclude that the makespan criterion and the total of tardiness tasks are implicitly linked.





Results makespan (with n =20) *Results total of tasks tardiness (with n =20)*

The graphical representation of the results found for the makespan criterion shows that the aggregative selection technique is more efficient for the problems type 2 with (α=0.5).

The results found for the total of tasks tardiness criterion shows that the aggregative selection technique is more efficient for the problems type 1 with (α=1.5) and type 2 with (α=0.5). For the other cases, both techniques are nearly identical.

V. Conclusions

In this paper, we studied a bi-objective scheduling problem, which consists in minimizing the makespan and the total of tasks tardiness for the problem having two dedicated processors with release dates. Several resolution methods allow solving this problem either approximately or exactly. Each method has advantages and disadvantages either in terms of the obtained solution or in terms of the calculation time. A good study of the problem allows obtaining good solution with a better complexity method.

We also adapted a genetic algorithm to solve a bi-objective scheduling problem. It is beneficial in terms of the computing time compared to exact method; branch and bound, for example, when the size of instances is important.

References

[1] Chu, C.(1992)'A branch and bound algorithm to minimize the total of tardness with different release date' Naval Researech Logistics 39, p256-283.

[2] Drozdowski, M. (1996) 'Scheduling multiprocessor tasks – an overview', European Journal of Operational Research, Vol. 94, pp.215–230.

[3] Manaa, A., Chu, C.(2010)'Scheduling multiprocessor tasks to minimise the makespan on two dedicated processors'. European J. Industrial Engineering, Vol. 4, No. 3, 2010.

[4] Rebai, M., Kacem, I., andAdjallah, (2010)'Earliness–tardiness minimization on a single machine to schedulepreventive maintenance tasks: metaheuristic and exact methods'J Intell Manuf (2012) 23:1207–1224.