

**List-scheduling and column-generations for scheduling of  $n$  job-groups with set up time and due date through  $m$  identical parallel machines to minimize makespan**

Seekharin Sukto

*Department of Industrial Engineering  
Faculty of Engineering  
Khonkaen University  
Khonkaen, 40002  
Thailand*

Peerayuth Charnsethikul \*

*Operations Research and Management Science Unit  
Department of Industrial Engineering  
Faculty of Engineering  
Kasetsart University  
Bangkok, 10903  
Thailand*

---

**Abstract**

This paper presents an optimization based heuristic for minimizing makespan on scheduling of  $n$  job-groups with  $q_i$  as the number of identical jobs,  $p_i$  as the processing time,  $su_i$  as the setup time required and  $dd_i$  as the due date, in job-group  $i$  through a set of  $m$  identical parallel machines. This heuristic, referred as *LSCCG*, is based on *LS* (List-Scheduling) heuristic and *CG* (Column Generation) which are hybrid between the well-known in scheduling problems such as the *LPT* (Longest Processing Time) heuristic or in packing problems such as the *BFD* (Best-Fit Decreasing) heuristic and column generation procedure in a cutting stock problem, respectively. The first algorithm, referred to as *LPTCCG* which are hybrid between *LPT* heuristic, is constructed using the initial pattern and adjusted to a lower bound by *ALSP* (Adjusted List-Scheduling Pattern); while the *CG*, is modified to *CCG* (Continuous Column Generation) attempting to strengthen makespan bound and collectively generated patterns to minimize a number of machines required. This model is repeatedly looped by *LPT*, *ALSP* and *CCG* to reach an upper bound of minimum

---

\*E-mail: fengprc@ku.ac.th

*Journal of Information & Optimization Sciences*

Vol. 27 (2006), No. 3, pp. 511–535

© Taru Publications

0252-2667/06 \$2.00 + 0.25

makespan. The average performance of this proposed algorithm is considerably better than that of the *LPT* heuristic. The better algorithm, referred to as *BFDCCG*, is based on *BFD* heuristic and *CG*. As the same principle, the *BFD* heuristic is generated the initial pattern and adjusted to a lower bound by *ALSP* and then attempted to strengthen makespan bound with a number of available machines by *CCG*. *BFDCCG* heuristic is repeatedly looped by *BFD*, *ALSP* and *CCG* to reach an upper bound of minimum makespan. The average performance of this proposed algorithm, *BFDCCG* heuristic is considerably better than that of the *LPTCCG* heuristic and computational time also. In set up time case, referred to as *LPTsuCCG* and *BFDsuCCG* which are similar to *LSCCG*, the only difference is in the set up time addition after List-scheduling the initial pattern and sub problem formulation in *CCG* procedure. As expected, the average performance and computational time of the *BFDsuCCG* heuristic is considerably better than the *LPTsuCCG* heuristic. In the set up time and due date case, implemented as *EDDsuddCCG*, *EDD* (Earliest Due Date) is used to generate the initial pattern. This algorithm is appropriate for the case of a few job-groups.

---

*Keywords* : Parallel machines, multi-processor, scheduling, bin packing, cutting stock, minimize makespan, completion time, LP-based heuristic, column generation, longest processing time.

## 1. Introduction

Scheduling problems have developed over some thirty-five years. In 1968, most certainly, this condition greatly influenced the literature on scheduling as well as the way the scheduling subject was perceived. Scheduling involved the optimal allocation or assignment of resources, overtime, to a set of tasks or activities or jobs, where a "job" consists of one or more activities and a "machine" is a resource that can perform at most one activity at a time. Scheduling problems include important problems in the obvious setting of manufacturing, transportation and logistics as well as in fields such as communications, media management and sports. Moreover, effective scheduling solutions accordingly can produce substantial economic dividends (Parker (1995)).

Parallel machines scheduling problems require the construction of machine processing schedules for a given set of jobs to optimize a given measure of performance. For the example of this problem, each job consists of only one activity including scheduling of computer systems on one hand and the scheduling of bottle-neck workstations involving multiple machines on the other. The parallel machines scheduling problem is also an important generalization of the single machine problem and can