Available at www.ComputerScienceWeb.com
POWERED BY SCIENCE DIRECT®

Parallel Computing 29 (2003) 393–430

PARALLEL COMPUTING

www.elsevier.com/locate/parco

Parallel GRASP with path-relinking for job shop scheduling

R.M. Aiex [a], S. Binato [b], M.G.C. Resende [c,*]

[a] *Catholic University of Rio de Janeiro, Rio de Janeiro, RJ 22451, Brazil*
[b] *Electrical Energy Research Center (CEPEL), P.O. Box 68007, Rio de Janeiro RJ 21944-970, Brazil*
[c] *AT&T Labs Research, Internet and Network Systems Research Center, 180 Park Avenue, Room C241, Florham Park, NJ 07932-0971, USA*

Received 1 March 2002; received in revised form 22 August 2002

Abstract

In the job shop scheduling problem (JSP), a finite set of jobs is processed on a finite set of machines under certain constraints, such that the maximum completion time of the jobs is minimized. In this paper, we describe a parallel greedy randomized adaptive search procedure (GRASP) with path-relinking for the JSP. Independent and cooperative parallelization strategies are described and implemented. Computational experience on a large set of standard test problems indicates that the parallel GRASP with path-relinking finds good-quality approximate solutions of the JSP.

Keywords: Combinatorial optimization; Job shop scheduling; GRASP; Path-relinking; Probabilistic algorithm

1. Introduction

The job shop scheduling problem (JSP) is a well-studied problem in combinatorial optimization. It consists in processing a finite set of jobs on a finite set of machines. Each job is required to complete a set of operations in a fixed order. Each operation is processed on a specific machine for a fixed duration. Each machine can process at most one job at a time and once a job initiates processing on a given machine it must

* Corresponding author. Tel.: +1-973-360-8444; fax: +1-973-360-8178.
E-mail addresses: rma@inf.puc-rio.br (R.M. Aiex), silvio@cepel.br (S. Binato), mgcr@research.att.com (M.G.C. Resende).

0167-8191/03/$ - see front matter
doi:10.1016/S0167-8191(03)00014-0

complete processing on that machine without interruption. A schedule is a mapping of operations to time slots on the machines. The makespan is the maximum completion time of the jobs. The objective of the JSP is to find a schedule that minimizes the makespan.

Mathematically, the JSP can be stated as follows. Given a set $\mathcal{M}$ of machines (where we denote the size of $\mathcal{M}$ by $|\mathcal{M}|$) and a set $\mathcal{J}$ of jobs (where the size of $\mathcal{J}$ is denoted by $|\mathcal{J}|$), let $\sigma_1^j \prec \sigma_2^j \prec \cdots \prec \sigma_{|\mathcal{M}|}^j$ be the ordered set of $|\mathcal{M}|$ operations of job j, where $\sigma_k^j \prec \sigma_{k+1}^j$ indicates that operation σ_{k+1}^j can only start processing after the completion of operation σ_k^j. Let $\mathcal{O}$ be the set of operations. Each operation σ_k^j is defined by two parameters: $\mathcal{M}_k^j$ is the machine on which σ_k^j is processed and $p_k^j = p(\sigma_k^j)$ is the processing time of operation σ_k^j. Defining $t(\sigma_k^j)$ to be the starting time of the kth operation $\sigma_k^j \in \mathcal{O}$, the JSP can be formulated as follows:

$$\begin{aligned}
&\text{minimize} \quad C_{\max} \\
&\text{subject to} \quad C_{\max} \geqslant t(\sigma_k^j) + p(\sigma_k^j) \quad \forall \sigma_k^j \in \mathcal{O}, \\
&\qquad t(\sigma_k^j) \geqslant t(\sigma_l^j) + p(\sigma_l^j) \quad \forall \sigma_l^j \prec \sigma_k^j, \qquad (1a) \\
&\qquad t(\sigma_k^j) \geqslant t(\sigma_l^i) + p(\sigma_l^i) \vee \qquad (1b) \\
&\qquad t(\sigma_l^i) \geqslant t(\sigma_k^j) + p(\sigma_k^j) \quad \forall \sigma_l^i, \sigma_k^j \in \mathcal{O} \text{ such that } \mathcal{M}_{\sigma_l^i} = \mathcal{M}_{\sigma_k^j}, \\
&\qquad t(\sigma_k^j) \geqslant 0 \quad \forall \sigma_k^j \in \mathcal{O},
\end{aligned}$$

where $C_{\max}$ is the makespan to be minimized.

A feasible solution of the JSP can be built from a permutation of $\mathcal{J}$ on each of the machines in $\mathcal{M}$, observing the precedence constraints, the restriction that a machine can process only one operation at a time, and requiring that once started, processing of an operation must be uninterrupted until its completion. Once the permutation of $\mathcal{J}$ is given, its feasibility status can be determined in $O(|\mathcal{J}| \cdot |\mathcal{M}|)$ time. The feasibility-checking procedure determines the makespan $C_{\max}$ for feasible schedules [1]. Since, each set of feasible permutations has a corresponding schedule, the objective of the JSP is to find, among the feasible permutations, the one with the smallest makespan.

The JSP is NP-hard [2] and has also proven to be computationally challenging. Exact methods [3–7] have been successful in solving small instances, including the notorious 10×10 instance of Fisher and Thompson [8], proposed in 1963 and only solved 20 years later. Problems of dimension 15×15 are still considered to be beyond the reach of today's exact methods. For such problems there is a need for good heuristics. Surveys of heuristic methods for the JSP are given in [9,10]. These include dispatching rules reviewed in [11], the shifting bottleneck approach [3,12], local search [10,13,14], simulated annealing [13,15], tabu search [1,14,16], and genetic algorithms [17]. Recently, Binato et al. [18] described a greedy randomized adaptive search procedure (GRASP) for the JSP. A comprehensive survey of job shop scheduling techniques can be found in Jain and Meeran [19]. In this paper, we present a new parallel GRASP with path-relinking for the JSP.