Expert Systems with Applications 41 (2014) 544–562

Contents lists available at ScienceDirect

Expert Systems with Applications

journal homepage: www.elsevier.com/locate/eswa

A GRASP × ELS approach for the job-shop with a web service paradigm packaging

Maxime Chassaing, Jonathan Fontanel, Philippe Lacomme, Libo Ren, Nikolay Tchernev *, Pierre Villechenon

Laboratoire d'Informatique (LIMOS, UMR CNRS 6158), Campus des Cézeaux, 63177 Aubière Cedex, France

ARTICLE INFO

Keywords:
Job-shop
GRASP
Web Service
Java EE
Glassfish
C#
IIS

ABSTRACT

The Job-Shop Scheduling Problem (JSSP) is well known for its complexity as an NP-hard disjunctive scheduling problem. The problem addressed in this paper is JSSPs with an objective of minimizing makespan while satisfying a number of hard constraints. An efficient GRASP × ELS approach is introduced for solving this problem. The efficiency is evaluated using the widely known 40 Laurence's instances which encompass medium and large scale instances. The computational results prove that the proposed method competes with the best published methods in both quality of results and computational time. Recently, Web services have generated great interest in researchers. Such application architecture is based on the client–server model using existing Internet protocols and open standards. It provides new approaches to optimization methods. The proposed GRASP × ELS is packaged into a Web Service (WS), i.e., it offers for the research community an open access to our optimization approach. Moreover, the proposed web service can be even included in research future works with a very small programming effort.

To favor utilization of the web service and to prove the facility in which the service could be used, we provide an example in Java proving that it is possible to obtain in less than 10 min a client application using the different methods exposed by this web service. Such usage extends to classical library inclusion in program with the difference that a method is called in the client side and represents an execution on the server.

The Web Service paradigm is a new approach in spreading algorithms and therefore this paper stands at the crossroads of optimization research community and the web service community expectations. The GRASP × ELS provided in the web service, is a state of the art method which competes with previously published ones and which has the advantage of being available for free, in any languages, everywhere contributing in spreading operational research contribution.

1. Introduction

Optimization is a key paradigm for modeling and solving in Operations Research (OR) and in related aspects of engineering, science, economics, and business. There are numerous optimization problem classes including planning, scheduling, routing, etc. All these problems can be solved by means of different methods like linear programming or metaheuristics as well as using latest Information Technology (IT). As stressed by Mike Trick in his widely read blog (http://mat.tepper.cmu.edu/blog/?p=301), wrote, "Within OR, we often do not track IT concepts such as Service Oriented Architectures or business intelligence, but we should: it can have a great effect on how our work is used in organizations."

In this paper we concentrate more particularly on the job-shop scheduling problem. The goal of our research is: (1) to define an efficient GRASP × ELS approach taking advantages of an indirect representation of solutions; (2) to package the operational research algorithm as a software service based on Web services paradigm that facilitate distributed computing.

1.1. Job-shop definition

The Job-Shop Scheduling Problem (JSSP) is a well-known optimization problem often used in practical scheduling applications in the manufacturing sector. The JSSP can be formulated as follows: a set of n jobs (*index* $i = 1,2,\ldots,n$) has to be processed on a set of m machines (*index* $j = 1,2,\ldots,m$). Each job is fully defined by an ordered (linear) sequence of operations that are associated with a particular machine. Therefore, the dimension of the problem is often denoted as $n \times m$. In addition, the JSSP must satisfy other constraints such as: (i) no more than one operation of any job can be

* Corresponding author. Tel.: +33 473407585.
E-mail addresses: chassaing@isima.fr (M. Chassaing), fontanel@isima.fr (J. Fontanel), placomme@isima.fr (P. Lacomme), ren@isima.fr (L. Ren), tchernev@isima.fr (N. Tchernev), villechenon@isima.fr (P. Villechenon).

0957-4174/$ - see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.eswa.2013.07.080

executed simultaneously; and (ii) no machine can process more than one operation at the same time; (iii) the job operations must be executed in a predefined sequence and once an operation is started, no pre-emption is permitted.

Each operation $O_{i,j}$ is related with a specific job i and a rank j (in the sequence of operations) and has a duration $p_{i,j}$. The objective is to schedule each operation on the machines, taking the precedence constraints into account such that the total makespan ($C_{\max}$) is minimized. According to the $\alpha|\beta|\gamma$ notation introduced by Garey, Johnson and Seth (1976) the problem can be represented by $J||C_{\max}$ and is known to be NP-hard (Garey et al., 1976). JSSP is widely considered as one of the most difficult problems over the last few decades. A helpful problem representation is the disjunctive graph model due to Roy and Sussmann (1964)).

Using the disjunctive graph model any job shop problem instance can be visualized by a directed graph $G = (V, A, E)$, where V represents the set of nodes, A the set of conjunctive arcs and E the set of pairs of disjunctive arcs. The set of nodes contains one element for each operation $O_{i,j}$, a source node, denoted by 0 connected to the first operation of each job and a sink node, denoted by $*$, linked with the last operation of each job. Conjunctive arcs are used to represent the routings of the different operations of the jobs and connect each pair of consecutive operations of the same job. Each pair of disjunctive arcs connects two operations, belonging to different jobs, which are to be processed on the same machine. A feasible solution corresponds to an acyclic subgraph that contains all conjunctive arcs and that contains exactly one disjunctive arc for each pair of disjunctive arcs. An optimal solution corresponds to the feasible subgraph with the minimal makespan $C_{\max}$.

1.2. Job-shop assumptions

The classical job-shop scheduling problem makes the following assumptions:

- Each job consists of a finite number of operations.
- The processing time for each operation using a particular machine is defined.
- There is a sequence of operations that has to be achieved to complete each job.
- Each job is performed on each machine only once.
- A machine can process only one job at a time.
- The system cannot be interrupted until each operation of each job is finished.
- Preemption is not allowed.
- Due date constraints are not defined.
- There is no setup or tardiness cost.

1.3. Graph modeling

Let us consider an example of JSSP composed of 3 jobs all of them having 3 operations defined in Table 1. For each job, this table gives the set of operations and for each operation the machine needed (m1, m2 or m3) and the processing time on this machine.

The disjunctive graph illustrating this example is given in Fig. 1. In this graph, an arc (in full line) between two successive operations (($O_{i,j}$, $O_{i,j+1}$) represents the routing constraint of this job i. It is valuated by the minimal distance between the start times of these two operations: $st_{O_{i,j+1}} - st_{O_{i,j}} \geqslant p_{O_{i,j}}$.

Each pair of disjunctive arcs is represented with a dotted edge and represents the resource constraint between two operations sharing the same resource.

The acyclic conjunctive graph of the solution modelling is given in Fig. 2 In this graph, all pairs of disjunctive arcs are reduced to one arc and represent the sequencing of operations processed on the same machine.

A comprehensive survey of scheduling techniques can be found in, for example Blazewicz, Domschke, and Pesch (1996) and Jain and Meeran (1999). Due to the complexity of the problem, a wide number of approaches are based on heuristics and meta-heuristics and provide reasonably good, rather than optimal solutions within reasonable computational effort. Great research effort is directed in such area to obtain framework with adequate ratio between computational time and solution quality which is summarized in the words of "efficient resolution".

1.4. Job-Shop key points for efficient resolution

The key points are the following:

- A Quasi-Direct Representation of Solution
- An efficient local search taking advantages of the longest path analysis

The approaches must guarantee an efficient exploration of the solution search space avoiding premature convergence and trying to favor non visited search space area (detection of clones, memory management avoiding analogous curves in the search space...).

The importance of definition of an ad hoc quasi direct representation of solution has been highlighted for years. Let us note the sequel publication of Cheng, Gen, and Tsujimura (1996) where the authors clearly defined that a quasi-direct representation permits: (1) a coding space and (2) a solution space. According to Cheng et al. (1996) an efficient mapping function must assign to any object of the coding space a solution which must satisfy the feasibility constraints. Authors also note that there are three types of mapping function (Fig. 3):

- 1-to-1 mapping function where one and only one object of the coding space is linked to one and only one solution;
- n-to-1 mapping function where several objects in the coding space can be linked to the same solution;
- 1-to-n mapping function where one object in the coding space can be linked to several solutions.

Table 1
Example of JSSP instance data.

Op. Jobs	$O_{i,1}$	$O_{i,2}$	$O_{i,3}$
$i = 1$	(m_1, 10)	(m_2, 35)	(m_3, 25)
$i = 2$	(m_1, 15)	(m_3, 16)	(m_2, 12)
$i = 3$	(m_3, 11)	(m_1, 12)	(m_2, 21)

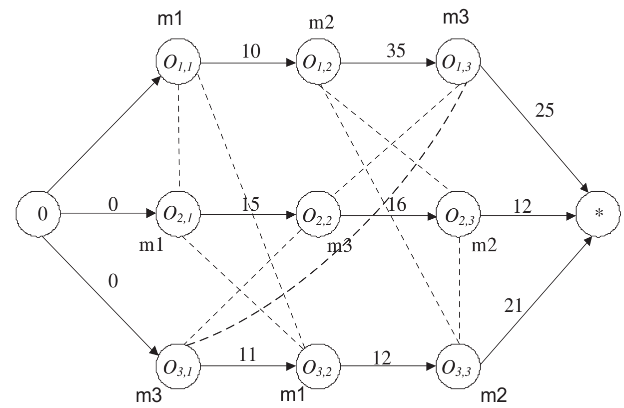

Fig. 1. Disjunctive graph of the job-shop (problem modelling).