Expert Systems With Applications 62 (2016) 131–147

Contents lists available at ScienceDirect

Expert Systems With Applications

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

Correlation of job-shop scheduling problem features with scheduling efficiency

Sadegh Mirshekarian, Dušan N. Šormaz*

Department of Industrial and Systems Engineering, Ohio University, Athens, Ohio, USA

ARTICLE INFO

Article history:
Received 26 August 2015
Revised 8 June 2016
Accepted 8 June 2016
Available online 9 June 2016

Keywords:
Job-shop scheduling
Scheduling efficiency
Makespan prediction
Machine learning
Support vector machines

ABSTRACT

In this paper, we conduct a statistical study of the relationship between Job-Shop Scheduling Problem (JSSP) features and optimal makespan. To this end, a set of 380 mostly novel features, each representing a certain problem characteristic, are manually developed for the JSSP. We then establish the correlation of these features with optimal makespan through statistical analysis measures commonly used in machine learning, such as the Pearson Correlation Coefficient, and as a way to verify that the features capture most of the existing correlation, we further use them to develop machine learning models that attempt to predict the optimal makespan without actually solving a given instance. The prediction is done as classification of instances into coarse lower or higher-than-average classes. The results, which constitute cross-validation and test accuracy measures of around 80% on a set of 15,000 randomly generated problem instances, are reported and discussed. We argue that given the obtained correlation information, a human expert can earn insight into the JSSP structure, and consequently design better instances, design better heuristic or hyper-heuristics, design better benchmark instances, and in general make better decisions and perform better-informed trade-offs in various stages of the scheduling process. To support this idea, we also demonstrate how useful the obtained insight can be through a real-world application.

1. Introduction

The Job-Shop Scheduling Problem (JSSP) is one of the most difficult combinatorial optimization problems considered (Lawler, Lenstra, Rinnooy Kan, & Shmoys, 1993). Lenstra and Rinnooy Kan (1979) categorize it as NP-hard, while many of its variations have been proven to be NP-complete (Garey, Johnson, & Sethi, 1976; Gonzalez & Sahni, 1978; Lenstra, Rinnooy Kan, & Brucker, 1977). Different solution techniques including exact methods, heuristics, estimation methods and metaheuristics have been suggested for each JSSP variation (Błażewicz, Domschke, & Pesch, 1996; Hochbaum & Shmoys, 1987; Jain & Meeran, 1999; Pinedo, 2008). The variation on which we focus in this paper is the conventional JSSP with no sequence-dependent setup times, no due dates, no operation preemption, one operation per machine at a time, one machine per job at a time and one operation per machine for each job. However, most of the ideas and practices introduced in this paper can be used for other variations as well.

Analytical solution methods can quickly lose their applicability as problem size increases (Aytug, Bhattacharyya, Koehler, & Snowdon, 1994), and even very fast techniques for moderately-sized shops may not be useful in a dynamic real-life environment where changes in processes and machines are the order of the day. For this reason, Operation Research (OR) practitioners resort to *dispatching rules* or heuristics to solve practical-sized instances in reasonable time. Blackstone, Phillips, and Hogg (1982) provide a survey of such heuristics, while Aytug et al. (1994) noted that even though some dispatching rules give reasonable results for some problem instances, it is difficult to predict when or for what type of instances they give good results. On the other hand, Thesen and Lei (1986) observed that expert human intervention and adjustment of these heuristics can often improve their performance. Since being an expert implies having the necessary insight into problem structure and the pathways that exist between problem configuration and a desired output, we believe there is a justified need to research into the ways of improving that insight and discovering more pathways. In effect, in this paper, we do not attempt to design new heuristics or directly suggest improvements over the old ones. We instead investigate the use of inductive machine learning techniques in evaluating the relationship between various problem features and its optimal makespan, in order to provide more insight for the expert practitioners to use.

Aytug et al. (1994) present a preliminary review of the use of machine learning in scheduling. They mention such AI methods as expert systems, which were the earliest attempts made to incorporate intelligence in scheduling. Such systems however, relying

* Corresponding author.
E-mail addresses: sm774113@ohio.edu (S. Mirshekarian), sormaz@ohio.edu (D.N. Šormaz).

http://dx.doi.org/10.1016/j.eswa.2016.06.014
0957-4174/

Abbreviations and symbols

OPT	Operation processing time
JPT	Job processing time
MPT	Machine processing time
OSPT	Operation slot processing time
OSMM	Operation slot missing machines
OSRM	Operation slot repeated machines
OSRMA	OSRM amplified
OSCOMB	Operation slot repeated machines combined with processing times
OSCOMBA	OSCOMB amplified
MLDU	Machine load uniformity
MLDV	Machine load voids
MLDVA	MLDV amplified
STD	Standard deviation
JCT	Job completions time
MCT	Machine completion time
PCC	Pearson correlation coefficient
SNR	Signal-to-noise ratio
n	Number of jobs
m	Number of machines
P	Matrix of processing times
M	Matrix of machine allocations
S	Solution schedule
Φ	Feature vector
L	Label (for supervised machine learning)
F	Number of features
φ_f	Feature f in the feature vector
I_i	Total idle time of machine i
p_{jk}	Processing time of operation k of job j
m_{jk}	Machine allocated to operation k of job j
s_{it}	tth operation of machine i in a schedule
C	Makespan
C_{min}	Optimal makespan
C'	Normalized makespan
C'_{min}	Normalized optimal makespan
SPT	'Shortest processing time' heuristic
LPT	'Longest processing time' heuristic
MWRM	'Most work remaining' heuristic
LWRM	'Least work remaining' heuristic
FIFO_MWRM	'First-in-first-out with least work remaining conflict resolution' heuristic

heavily on the wit of the human expert, were criticized for not being effective for most dynamic environments. Other AI methods such as various "search techniques" followed to fill the gap, but they were not adaptive and were very slow. Machine learning was the latest technique to be used, and it overcame many of the early problems, by introducing adaptability and versatility without adding too much computation. Aytug et al. (1994) argue for the importance of automated knowledge acquisition and learning in scheduling, and cite researchers like Yih (1990) and Fox and Smith (1984) to support this premise.

Because most research on scheduling has been on designing and improving techniques that solve a given problem instance and find an optimal or near-optimal schedule in terms of a certain cost function such as makespan, most machine learning research has also followed the same lines and has been around automating and improving this process. Adaptive heuristics and adaptive hyper-heuristics are perhaps the two main categories of machine learning research on scheduling. In the first category, the aim is to use machine learning models (and models obtained through other AI methods like rule-based systems, evolutionary algorithms, etc.) 'as a heuristic' that is more adaptive than conventional dispatching rules and can incorporate more knowledge into the scheduling process. See for example Lee, Piramuthu, and Tsai (1997) who combined decision trees and genetic algorithms to develop adaptive schedulers, Zhang and Rose (2013) who used artificial intelligence techniques to develop an individual dispatcher for each machine, or Li, Zijin, Jiacheng, and Fei (2013) who developed an adaptive dispatching rule for a semiconductor manufacturing system. In the second category which is more active in recent literature, the focus is on designing models that can help 'design heuristics' or help choose the best dispatching rule for a given system state. See Branke, Nguyen, Pickardt, and Zhang (2015) and Burke et al. (2013) for a recent review, Nguyen, Zhang, Johnston, and Tan (2013) and Nguyen et al (2013b) who used genetic programming to discover new dispatching rules, Pickardt (2013) who used evolutionary algorithms to automate the design of dispatching rules for dynamic complex scheduling problems, Olaffson and Li (2010) and Li and Olaffson (2005) who used data mining and decision trees to learn new dispatching rules, and also Priore, de la Fuente, Gomez, and Puente (2001), Priore, de la Fuente, Puente, and Pareño (2006) and Burke, MacCarthy, Petrovic, and Qu (2003) for other applications.

Despite the existence of smarter scheduling algorithms, many practitioners still use their own expertise to make the final decision or make a decision in critical conditions. Their expertise usually comes from experience, and little theoretical work is done in the literature to support the understanding of scheduling problem structure in a meaningful and easy-to-relate way. Smith-Miles, James, Giffin, and Tu (2009) and Smith-Miles, van Hemert, and Lim (2010) investigated the use of data mining to understand the relationship between scheduling and the travelling salesman problem[1] structure and heuristic performance, while Ingimundardottir and Runarsson (2012) used machine learning to understand why and predict when a particular JSSP instance is easy or difficult for certain heuristics. We believe such work is valuable, because without a practical understanding of problem structure, moving towards a goal of better scheduling practice might just be a tedious campaign of trial-and-error. Practitioners also need supporting insight in a less cryptic and more palpable form. We try to achieve this in our paper, by developing a set of descriptive features that characterize a job-shop scheduling problem, and establishing their correlation with the optimal makespan.

Another aspect of scheduling that is not often addressed in the literature is at the shop design level, where the JSSP instance to be solved is actually defined. Researchers usually neglect the fact that JSSP instances are not always designed without any flexibility. Sometimes there is the option of choosing different machines for certain jobs (as for the example application given in Section 5), or even more commonly, the option of ordering job operations differently. Current research mostly supports the "solution" of a given JSSP instance, and so the shop design practitioners' only options are to trust their own insight and/or to run approximation algorithms repeatedly to find the best option. Since the practices of shop design and shop scheduling are inherently entwined, there is a justifiable need for work that can support both. The only work we are aware of that addresses these practices simultaneously is Mirshekarian and Šormaz (2015), and this paper is a continuation of that work.

The type of characterizing features that we define and evaluate in this paper, can support both practices (shop design and shop scheduling). For example, intuition can tell us that if one job has a much higher total processing time than other jobs, or more

[1] The travelling salesman problem is a special case of JSSP when the number of machines is 1 and there are sequence-dependent setup times.