Problem Difficulty for Tabu Search in Job-Shop Scheduling

Jean-Paul Watson [a,*,1] J. Christopher Beck [b,2] Adele E. Howe [a,1]
L. Darrell Whitley [a,1]

[a] *Department of Computer Science, Colorado State University, Fort Collins, CO 80523-1873 USA*

[b] *Cork Constraint Computation Centre, University College Cork, Cork, Ireland*

Abstract

Tabu search algorithms are among the most effective approaches for solving the job-shop scheduling problem (JSP). Yet, we have little understanding of why these algorithms work so well, and under what conditions. We develop a model of problem difficulty for tabu search in the JSP, borrowing from similar models developed for SAT and other NP-complete problems. We show that the mean distance between random local optima and the nearest optimal solution is highly correlated with the cost of locating optimal solutions to typical, random JSPs. Additionally, this model accounts for the cost of locating sub-optimal solutions, and provides an explanation for differences in the relative difficulty of square versus rectangular JSPs. We also identify two important limitations of our model. First, model accuracy is inversely correlated with problem difficulty, and is exceptionally poor for rare, very high-cost problem instances. Second, the model is significantly less accurate for structured, non-random JSPs. Our results are also likely to be useful in future research on difficulty models of local search in SAT, as local search cost in both SAT and the JSP is largely dictated by the same search space features. Similarly, our research represents the first attempt to quantitatively model the cost of tabu search for *any* NP-complete problem, and may possibly be leveraged in an effort to understand tabu search in problems other than job-shop scheduling.

Key words: Problem Difficulty, Job-Shop Scheduling, Local Search, Tabu Search

* Corresponding author. E-mail: watsonj@cs.colostate.edu

[1] The authors from Colorado State University were sponsored by the Air Force Office of Scientific Research, Air Force Materiel Command, USAF, under grant number F49620-00-1-0144. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon.

[2] This work was performed while the second author was employed at ILOG, SA.

1 Introduction

The job-shop scheduling problem (JSP) is widely acknowledged as one of the most difficult NP-complete problems encountered in practice. Nearly all well-known optimization and approximation techniques have been applied to the JSP, including linear programming, Lagrangian relaxation, branch-and-bound, constraint satisfaction, local search, and even neural networks and expert systems [19]. Most recent comparative studies of techniques for solving the JSP conclude that local search algorithms provide the best overall performance on the set of widely-available benchmark problems; for example, see the recent surveys by Blażewicz et al. [6] or Jain and Meeran [19]. Within the class of local search algorithms, the strongest performers are typically derivatives of tabu search [6] [19] [34], the sole exception being the guided local search algorithm of Balas and Vazacopoulos [2]. The power of tabu search for the JSP is perhaps best illustrated by considering the computational effort required to locate optimal solutions to a notoriously difficult benchmark problem, Fisher and Thompson's infamous 10×10 instance [12]: Nowicki and Smutnicki's algorithm [22] requires only 30 seconds on a now-dated personal computer, while Chambers and Barnes' algorithm [9] requires less than 4 seconds on a moderately powerful workstation. In contrast, a number of algorithms for the JSP still have significant difficulty in finding optimal solutions to this problem instance.

Despite the relative simplicity and excellent performance of tabu search algorithms for the JSP, very little is known about *why* these algorithms work so well, and under what conditions. For example, we currently have no answers to fundamental, related questions such as "Why is one problem instance more difficult than another?" and "What features of the search space influence search cost?" No published research has presented problem difficulty models of tabu search algorithms for the JSP. Further, only one group of researchers, Mattfeld et al. [20], have analyzed the link between problem difficulty and local search for the JSP in general.

In contrast to the JSP, problem difficulty models do exist for several other well-known NP-complete problems, such as the Traveling Salesman Problem (TSP) and the Boolean Satisfiability Problem (SAT). Models of local search cost in SAT have received significant recent attention, and are able to account for much of the variability in problem difficulty observed for a particular class of random problem instances commonly known as Random 3-SAT [10] [24] [27]. The SAT models relate individual features of the search space to search cost, and model accuracy is generally quantified as the r^2 value of the corresponding linear regression model. We refer to such models as *static cost models* of local search; the goal of such models is to account for a significant proportion (ideally all) of the variability in local search cost observed for a set of problem instances. The 'static' modifier derives from the fact these models are largely independent of particular algorithm dynamics, relying instead on static features of the search space.

Static cost models of local search in SAT are based on three search space features: the number of optimal solutions, the backbone size, and the mean distance between random near-optimal solutions (i.e., those with only a few unsatisfied clauses) and the nearest optimal solution. Clark et al. [10] introduced the first static cost model for SAT, and demonstrated that the logarithm of the number of optimal (i.e., satisfying) solutions is highly and negatively correlated with the logarithm of search cost. In SAT, the *backbone* of a problem instance is the set of Boolean variables that have the same truth value in all optimal solutions. Both Parkes [24] and Singer et al. [27] demonstrated that the size of the problem backbone is inversely proportional to search cost. Most recently, Singer et al. [27] demonstrated a very strong positive correlation between the logarithm of local search cost and the mean distance between random near-optimal solutions and the nearest optimal solution.

Static cost models for problems other than SAT have received relatively little attention, the sole exception being the related and more general Constraint Satisfaction Problem (CSP) [10]. However, the factors underlying the SAT models are very intuitive, suggesting the potential for their applicability to other NP-complete problems, including the JSP; for example, most researchers would be surprised if the number of optimal solutions did *not* influence the cost of local search. However, the search spaces of the JSP and SAT are thought to be qualitatively dissimilar, and local search algorithms for SAT differ in many ways from tabu search algorithms for the JSP: e.g., local search algorithms for SAT have a strong stochastic component, while tabu search algorithms for the JSP are largely deterministic. Consequently, it is unclear a priori whether the SAT models can be leveraged in an effort to understand problem difficulty for tabu search in the JSP.

In this paper, we develop a static cost model of tabu search in the JSP, drawing heavily from existing static cost models of local search in SAT. The resulting model accounts for a significant proportion of the variability in the cost of finding optimal solutions to random JSPs with tabu search. We then use the model to provide explanations for two well-known but poorly-understood qualitative observations regarding problem difficulty in the JSP, and identify two important limitations of the model. More specifically, our research makes the following contributions:

(1) We show that analogs of those search space features known to influence local search cost in SAT, specifically the number of optimal solutions ($|\mathit{optsols}|$) and the mean distance between random local optima and the nearest optimal solution ($d_{\mathit{lopt\text{-}opt}}$), also influence the cost of locating optimal solutions using tabu search in the JSP. Further, the *strength* of the influence of these two features is nearly identical in both problems. As in SAT, we find that $d_{\mathit{lopt\text{-}opt}}$ has a much stronger influence on search cost than $|\mathit{optsols}|$, and ultimately accounts for a significant proportion of the variability in the cost of finding optimal solutions to random JSPs. This result was somewhat unexpected given differences in the search spaces and local search algorithms for the JSP and SAT.
(2) Our experiments indicate that for random JSPs with moderate to large back-

bones, the correlation between backbone size and the number of optimal solutions is extremely high. As a direct consequence, for these problems, backbone size provides no more information than the number of optimal solutions, and vice versa: one of the two features is necessarily redundant. Given the recent surge of interest in the link between backbone size and problem difficulty, the nearly one-to-one correspondence between these two features was completely unanticipated.

(3) In contrast to Singer et al. [27], we find *no* interaction effect between the backbone size and $d_{lopt\text{-}opt}$. Further, we find that more complex static cost models based on multiple search space features, or those that consider interaction effects between search space features, are no more accurate than the simple model based solely on $d_{lopt\text{-}opt}$.

(4) A simple extension of the $d_{lopt\text{-}opt}$ model accounts for most of the variability in the cost of finding *sub-optimal* solutions to the JSP. This extension is the first quantitative model of the cost of locating sub-optimal solutions to any NP-complete problem, and provides an explanation for the existence of 'cliffs' in the cost of finding sub-optimal solutions of varying quality (e.g., see [31]).

(5) For some time, researchers have observed that 'square' JSPs are generally more difficult than 'rectangular' JSPs. We show that this phenomenon is likely due to differences in the distribution of $d_{lopt\text{-}opt}$ for the two problem types. For square JSPs, the proportion of problem instances with large values of $d_{lopt\text{-}opt}$ is substantial, while most rectangular JSPs have very small values of $d_{lopt\text{-}opt}$.

(6) We identify two important limitations of the $d_{lopt\text{-}opt}$ model. First, we show that model accuracy is inversely proportional to problem difficulty, and is exceptionally poor for very high-cost problem instances. Second, we demonstrate that the $d_{lopt\text{-}opt}$ model is significantly less accurate when we consider more structured JSPs, specifically those with workflow partitions of the job routing orders.

Because local search cost in both the JSP and SAT is influenced by the same search space features, our results also identify likely deficiencies in the static cost models of local search in SAT. Specifically, we conjecture that the following also hold in SAT: (1) the high correlation between backbone size and the number of optimal solutions, (2) the extreme inaccuracies of the $d_{lopt\text{-}opt}$ model on very high-cost problem instances, and (3) the degradation in the accuracy of the $d_{lopt\text{-}opt}$ model for structured problem instances.

Although tabu search algorithms have been successfully applied to a number of NP-complete problems, very little is known in general about which search space features influence problem difficulty, and to what degree. Our research provides a preliminary answer to this question for one particular problem, the JSP, and only for a relatively simple form of tabu search. Consequently, our results may be useful to researchers developing problem difficulty models of tabu search in NP-complete problems other than the JSP, or for models of more advanced tabu search algorithms for the JSP.

In the following section, we briefly review prior research on models of problem difficulty and identify the subset of models that form the basis of our analysis. In Section 3, we define the job-shop scheduling problem and introduce the tabu search algorithm used in our experiments; the section then concludes with a discussion of prior work on problem difficulty for local search in the JSP. In Section 4, we develop our static cost model of tabu search in the JSP. Section 5 details two important applications of the resulting model: accounting for the cost of locating *sub-optimal* solutions to the JSP, and providing an explanation for differences in the relative difficulty of square versus rectangular JSPs. In Section 6, we expose two important limitations of the model, both of which suggest new directions in research on problem difficulty models. Finally, we conclude by discussing the implications of our results in Section 7.

2 Models of problem difficulty

One key lesson from the research into models of problem difficulty is the often 'universal' nature of these models, in that they typically apply to a wide range of NP-complete problems. For example, phase transitions have been observed in problems ranging SAT to Graph K-colorability [17]. Similarly, the distribution of local optima in many problems exhibit a 'Big-Valley' structure, for example in both the Traveling Salesman and Graph Bi-Partitioning Problems [8]. Given the apparent pervasiveness of these phenomena, the obvious first step in our research is to determine whether existing models can be extended to the JSP. However, before investigating individual models, we first consider the following questions:

- What type of information should our model provide?
- What existing models provide this type of information?
- What is our success criteria?

Our immediate goal is to develop an understanding of existing tabu search algorithms for the JSP; subsequently, we intend to leverage such knowledge to improve the performance of existing algorithms. A detailed understanding of how an algorithm interacts with the search space is clearly required to propose enhancements in a principled manner. Consequently, our goal is to produce quantitative models that relate search space features to search cost. We refer to such models as *static cost models*; the 'static' modifier derives from the fact that algorithm dynamics are not explicitly considered. An accurate static cost model should account for a significant proportion of the variability in difficulty observed for a set of problem instances.

Much of the research on models of problem difficulty does not share our goal of relating search space features to search cost, and as a consequence generally fail to account for much, if any, of the variability in problem difficulty. Within the AI community, phase transitions [17] are the dominant model of problem difficulty. Phase

transition models partition the 'universe' of problem instances into a large number of subclasses, and are able to account for mean differences in subclass difficulty. However, the variability within a subclass remains unaccounted for, and is typically largest in the most difficult subclasses (i.e., those near the transition region). For example, the cost of solving instances in the most difficult subclass of 100-variable Random 3-SAT instances varies over 5 orders of magnitude [10]. Outside of AI, the most widely studied problem difficulty models are correlation length and the Big-Valley local optima distribution. A correlation length model measures the 'smoothness' of a search space by analyzing the autocorrelation of a time-series of solution quality generated by a random walk. However, the correlation length for a large number of problems is *strictly* a function of the problem size (e.g., the number of cities in the Traveling Salesman Problem) [25]. Consequently, correlation length models fail to account for *any* of the often large variance in problem difficulty observed for an ensemble of fixed-size problem instances. Similarly, Big-Valley local optima distributions are found in both easy and hard problems; the model generally fails to account for differences in relative difficulty for individual problem instances.

To date, researchers have only produced static cost models of local search for SAT and the closely related Constraint Satisfaction Problem. Static cost models of local search in SAT are based on three search space features: the number of optimal solutions, the backbone size, and the mean distance between random near-optimal solutions and the nearest optimal solution. In Section 4, we define each of these features, discuss the properties of existing static cost models that are based on these features, and analyze the applicability of these features to static cost models of tabu search in the JSP.

As we discuss in Section 4, the accuracy of a static cost model can be quantified by the r^2 value of the corresponding linear or multiple regression model. We can also quantify worst-case model accuracy by analyzing the magnitude of the residuals under the regression model. Ultimately, our goal is to produce a static cost model with (1) $r^2 \geq 0.8$ and (2) the actual search cost varying no more than $1/2$ an order of magnitude from the predicted search cost. Although somewhat arbitrary, any static cost model satisfying these two criteria would conclusively identify those search space features that largely dictate the cost of tabu search in the JSP. Further, more stringent criteria are likely to leave insufficient room for measurement and/or sampling error. Finally, as we observe in Section 4, the task of producing static cost models satisfying the two proposed criteria is sufficiently challenging.

3 The job-shop scheduling problem, tabu search, and problem difficulty

We now introduce the job-shop scheduling problem and detail the specific tabu search algorithm that forms the basis of our analysis. We then briefly review prior

research on problem difficulty and job-shop scheduling.

3.1 The job-shop scheduling problem

We consider the well-known $n \times m$ static job-shop scheduling problem (JSP), in which n jobs must be processed exactly once on each of m machines. Each job i ($1 \leq i \leq n$) is routed through each of the m machines in some pre-defined order π_i, where $\pi_i(j)$ denotes the jth machine ($1 \leq j \leq m$) in the routing order. The processing of a job on a machine is called an *operation*, and the processing of job i on machine $\pi_i(j)$ is denoted by o_{ij}. An operation o_{ij} must be processed on machine $\pi_i(j)$ for an integral duration of $\tau_{ij} > 0$. Once processing is initiated, an operation cannot be pre-empted, and concurrency is not allowed. Finally, for $2 \leq j \leq m$, o_{ij} cannot begin processing until o_{ij-1} has completed processing.

A solution s to an instance of the $n \times m$ JSP specifies a processing order for all of the jobs on each machine, and implicitly specifies an earliest start time $est(x)$ and earliest completion time $ect(x)$ for each operation x [33]. Although a number of objective functions have been defined for the JSP, most research addresses the problem of makespan minimization [6]. The *makespan* $C_{max}(s)$ of a solution s is the maximum earliest completion time of the last operation of any job: $C_{max}(s) = max(ect(o_{1,m}), ect(o_{2,m}), ..., ect(o_{n,m}))$. We denote the optimal (minimal) makespan of a problem instance by C^*_{max}. The decision problem of finding a solution to the JSP with a makespan less than or equal to some constant L is known to be NP-complete for $m \geq 2$ and $n \geq 3$ [14]. Further, the JSP is widely regarded as one of the most difficult NP-complete problems encountered in practice [6] [19].

As we discuss in Section 3.3, extraction and manipulation of the critical paths of a solution s is a key component of tabu search algorithms for the makespan minimization form of the JSP. A *critical path* of a solution s consists of a sequence of operations $o_1, o_2, ..., o_l$ such that (1) $est(o_1) = 0$, (2) $ect(o_l) = C_{max}(s)$, and (3) $est(o_i) = ect(o_{i-1})$ for $1 \leq i \leq l$, where $ect(o_0) = 0$ by convention. The operations o_i are known as *critical operations*. A *critical block* consists of a contiguous sub-sequence of operations on a critical path that are processed on the same machine. A solution s may possess more than one critical path. If multiple critical paths exist, they may share common sub-sequences of critical operations.

3.2 Generating problem instances

An instance of the $n \times m$ JSP is uniquely defined by the set of nm operation durations τ_{ij} and the n job routing orders π_i ($1 \leq i \leq n$ and $1 \leq j \leq m$). Typically, the τ_{ij} are independently and uniformly sampled from a fixed-width interval, such

as $[1, 99]$ (e.g., see Taillard [32]). Most often, the job routing orders π_i are produced by generating independent random permutations of the integers $[1..m]$. We refer to problem instances in which both the τ_{ij} and π_i are independently and uniformly sampled as *general* JSPs.

Well-known specializations of the JSP impose non-random structure on the job routing orders. One common specialization organizes the job routing orders into *workflow partitions*: the set of machines is divided into two equal-sized partitions containing machines 1 through $m/2$ and $m/2+1$ through m, respectively, and every job must be processed on all machines in the first partition before any machine in the second partition. Within the partitions, the job routing orders are produced by generating independent random permutations of the integers $[1..m/2]$ and $[m/2 + 1..m]$, respectively. We refer to problem instances resulting from this process as *workflow* JSPs, which we analyze in Section 6.2.

3.3 Algorithm description

The analyses presented in Sections 4 through 6 are based on a tabu search algorithm for the JSP introduced by Taillard [33], which we denote $TS_{Taillard}$. We note that $TS_{Taillard}$ is *not* the best available tabu search algorithm for the JSP; the tabu search algorithms of Nowicki and Smutnicki [22] and Barnes and Chambers [3] [9] provide stronger overall performance. Rather, we have selected $TS_{Taillard}$ for three reasons. First, $TS_{Taillard}$ provides reasonable performance of the set of widely-used benchmark problems, and out-performs many other local and constructive search algorithms for the JSP. Second, state-of-the-art tabu search algorithms for the JSP are somewhat more complex than $TS_{Taillard}$, complicating analysis; the salient differences are in choice of move operator, the use of re-intensification mechanisms around high-quality solutions, and the method for constructing the initial solution. Instead of tackling the most complex algorithms first, our goal is to develop a static cost model for a straightforward implementation of tabu search in the JSP, and then to *systematically* assess the influence of more complex algorithmic features on the static cost model of the basic algorithm. Third, as we now discuss, certain features of $TS_{Taillard}$ make it particularly amenable to analysis, especially in comparison to some of the more advanced tabu search algorithms for the JSP.

At the core of any local search algorithm is a move operator, which defines the set of solutions that can be reached in a single step from the current solution; elements of this set are called *neighbors* of the current solution. In the JSP, neighbors are generally produced by re-ordering the sequence of operations on a critical path; only through such re-ordering is it possible to produce a neighbor with a makespan better than that of the current solution. The first successful move operator for the JSP was introduced by van Laarhoven et al. [35], and is often denoted by $N1$. The neighborhood of a solution s under the $N1$ move operator consists of the set of

solutions obtained by swapping the order of a single pair of adjacent operations on the same critical block in s. An important property of $N1$ is that it induces search spaces that are provably *connected*, in that it is always possible to move from an arbitrary solution to a global optimum. Consequently, it is possible to construct a local search algorithm based on $N1$ that will eventually locate an optimal solution, given sufficiently large run-times. Hoos [18] refers to algorithms with this property as being *probabilistically approximately complete*, or PAC: the probability of the algorithm locating an optimal solution approaches 1 as the run-time approaches ∞. A primary reason we consider $TS_{Taillard}$ in our analysis is that it is based on the $N1$ operator. Further, $TS_{Taillard}$ is, at least empirically, PAC: in generating the results presented in Sections 4 through 6, no trial of $TS_{Taillard}$ failed to locate an optimal solution. In contrast, some other tabu search algorithms for the JSP employ move operators that induce disconnected search spaces, and are consequently not PAC: e.g., Nowicki and Smutnicki's tabu search algorithm. Our primary goal is to model the cost of locating optimal solutions to the JSP, and as we discuss later in this section, the measurement of this cost is straightforward only if an algorithm is PAC.

The static cost models we develop in Section 4 are based in part on search space features that involve distances between pairs of solutions: e.g., the average distance between local optima or the mean distance between random local optima and the nearest optimal solution. Ideally, the distance between two solutions is defined as the minimum number of applications of a particular move operator that are required to transform one solution into the other. Unfortunately, computation of this measure is generally intractable, and operator-independent measures are typically substituted. The most widely used operator-independent distance measure in the JSP is defined as follows [20]. Let $preceeds_{ijk}(s)$ be a Boolean-valued function indicating whether job i is processed before job j on machine k in a solution s. The distance $D(s_1, s_2)$ between two solutions s_1 and s_2 to an $n \times m$ JSP instance is then given by:

$$\sum_{i=1}^{m} \sum_{j=1}^{n-1} \sum_{k=j+1}^{n} preceeds_{ijk}(s_1) \oplus preceeds_{ijk}(s_2) \tag{1}$$

where the symbol $\oplus$ denotes the Boolean XOR operator. We denote the normalized distance $2D(s_1, s_2)/mn(n-1)$ by $\overline{D}(s_1, s_2)$; clearly, $0 \leq \overline{D}(s_1, s_2) \leq 1$. Another important property of the $N1$ operator is the fact that Equation 1 provides a relatively tight lower bound on the number of applications of the $N1$ move operator to transform solution s_1 into solution s_2.

$TS_{Taillard}$ is a relatively 'vanilla' implementation of tabu search [16]. As with most tabu search algorithms for the JSP, recently swapped pairs of jobs are prevented from being re-established for a particular duration, called the tabu tenure. In each *iteration* of $TS_{Taillard}$, all $N1$ neighbors are generated. The neighbors are then classi-

fied as tabu (the pair of jobs was recently swapped) or non-tabu, and the best non-tabu move is taken; ties are broken randomly. All runs are initiated from randomly generated local optima, produced using a standard steepest-descent algorithm initiated from a random 'semi-active' solution [33]. The only long-term memory mechanism is a simple aspiration criterion, which over-rides the tabu status of any move that results in a solution that is better than any previously encountered during the current run. As Taillard indicates ([33], p. 100), long-term memory is only necessary for problems that require a very large (> 1 million) number of iterations, which is not the case for the test problems we consider. The only parameters of $TS_{Taillard}$ involve computation of the tabu tenure, which is uniformly sampled from the interval $[6, 14]$ every 15 iterations. Empirically, $TS_{Taillard}$ fails to be PAC without such a dynamic tabu tenure, or if the tabu tenure is sampled from a smaller interval (e.g., $[5, 10]$).

The cost required to solve a given problem instance using $TS_{Taillard}$, or any PAC algorithm, is naturally defined as the number of iterations required to locate an optimal solution. However, the number of iterations is stochastic (with an approximately exponential distribution [33]), due to both the randomly generated initial solution and random tie-breaking when more than one 'best' non-tabu move is available. Consequently, we define the local search cost for a problem instance as the median number of iterations required to locate an optimal solution over 5000 independent runs, which we denote $cost_{med}$. With 5000 independent trials, the estimate of $cost_{med}$ is relatively stable.

3.4 Prior research on problem difficulty in the JSP

A number of qualitative observations regarding the relative difficulty of various types of JSPs have emerged over time [19]:

(1) For both general and workflow JSPs, 'square' ($n/m \approx 1$) problems are generally more difficult than 'rectangular' ($n/m \gg 1$) problems.
(2) Given fixed n and m, workflow JSPs are generally more difficult than general JSPs.
(3) The relative difficulty of particular problem instances is largely algorithm independent.

Clearly, any model of problem difficulty for the JSP needs at least to be consistent with, and should ultimately provide explanations for, each of these three observations.

Large differences in the difficulty of square versus rectangular JSPs are easily illustrated by considering the status of the problem instances in Taillard's JSP benchmark suite, which is available from the OR-Library [4]. Specifically, the optimal makespans of all the relatively small 20×20 and 30×20 instances are currently

unknown, while optimality has been established for all but two of the larger 50×20 and 100×20 instances, despite the astronomical difference in the sizes of the search spaces. Taillard [33] studied the impact of changes in the ratio of n/m on search cost for $TS_{Taillard}$. His experiments demonstrated that for $n/m \geq 6$, the growth in the cost of locating optimal solutions grows *polynomially* with increases in n and m, despite an exponential growth in the size of the search spaces. In contrast, for problems with $n/m \approx 1$, the search cost grows exponentially with increases in n and m, as expected given the proportionate growth in the size of the search space. Although intuitive explanations have been proposed for why the growth in problem difficulty changes with increases in n/m, a complete understanding of this phenomenon remains elusive. No research has analyzed the changes in search space features as n/m is varied, which is of particular interest when developing static cost models of local search.

The second observation stems in part from computational experiments on two sets of 50×10 general and workflow JSPs introduced by Storer et al. [30]: for a number of algorithms, it is significantly more difficult to find high-quality solutions to the workflow instances. Further, the optimal makespans of all the general instances are known, while optimality has only been established for one of the workflow instances. The sole quantitative study of problem difficulty in the JSP is due to Mattfeld et al. [20], and is largely devoted to providing an explanation for the differences in relative difficulty of general and workflow JSPs. Mattfeld et al.'s study also provides a possible explanation for why genetic algorithms generally perform poorly on the JSP. Mattfeld et al. identified significant differences between the search spaces of Storer et al.'s general and workflow instances, specifically by demonstrating that the extension of the search space (as measured by the average distance between random local optima) is larger in workflow JSPs than in general JSPs. These differences suggest a cause for the relative differences in problem difficulty. Similar differences were observed for two other quantitative search space measures: entropy and correlation length.

Finally, the third observation results from the fact that easy (difficult) benchmark problem instances are generally easy (difficult) for *all* search algorithms, including those based on branch-and-bound, constraint programming, and local search. A causal basis for this phenomenon is lacking, although we hypothesize an explanation for the class of local search algorithms in Section 4.5.

4 Modeling the cost of locating optimal solutions

We now introduce and analyze several static models of the cost required by $TS_{Taillard}$ to find optimal solutions to general JSPs. Each model is based on an individual feature of the search space: the number of optimal solutions, the backbone size, the average distance between local optima, or the mean distance between random lo-

cal optima and the nearest optimal solution. Similar static cost models have been developed for other NP-complete problems, primarily SAT, and we analyze the applicability of these models to tabu search in the JSP. We then consider models based on aggregations of these features, specifically analyzing the impact of additive and interaction effects.

For a number of reasons, the extension of static cost models of local search in SAT to tabu search in the JSP is unclear a priori. For example, the SAT search space is dominated by plateaus of equally-fit 'quasi-solutions', each containing an identical, small number of unsatisfied clauses. The main challenge for local search is to either find an exit from a plateau to an improving quasi-solution, or to escape the plateau by accepting a short sequence of dis-improving moves [13]. In contrast, the JSP search space is dominated by local optima with variable-sized and variable-depth attractor basins, which local search algorithms must either escape or avoid. Further, local search algorithms for SAT are largely stochastic, while tabu search algorithms such as $TS_{Taillard}$ are largely deterministic. On the other hand, the features underlying the SAT models are very intuitive, and would appear to influence the difficulty of local search in a wide range of NP-complete problems.

In this section, we demonstrate that despite significant differences in both search space topologies and local search algorithms, a straightforward adaptation of a static cost model for SAT yields a surprisingly accurate model of the cost required by $TS_{Taillard}$ to locate optimal solutions to general JSPs. Specifically, we show that (1) the mean distance between random local optima and the nearest optimal solution accounts for a substantial proportion of the variability in local search cost, (2) backbone size, the number of optimal solutions, and the average distance between local optima account for far less of the variability in search cost, (3) simultaneous consideration of multiple search space features, through the inclusion of either additive or interaction terms, does not result in substantially more accurate models, and (4) the correlation between the number of optimal solutions and the backbone size is extremely high, with one feature providing no more information than the other.

The material presented in this section is a significant extension of an analysis we previously reported [36]. In our prior work, we directly replicated the methodology introduced by Singer et al. [27], and demonstrated that the static cost models for SAT also apply to $TS_{Taillard}$ for the general JSP. In this section, instead of controlling for various search space features a priori, we adopt a different methodology. Instead, we explicitly focus on model accuracy for 'typical' instances of the general JSP. The new methodology also enables our new insights concerning the existence of interactions between the various search space features.

Finally, as discussed in Section 2, we quantify the accuracy of each static cost model using linear or multiple regression techniques. Unless otherwise noted, the assumptions concerning model errors (e.g., the errors are normally distributed and

homogeneous) are approximately satisfied, and the F-statistics are significant at $p < 0.0001$. When the regression assumptions are not satisfied, we additionally report the non-parametric Spearman's rank correlation coefficient.

4.1 Test problems

For a number of reasons, problem difficulty models are typically generated by analyzing relatively small problem instances. We develop our static cost models using 6×4 and 6×6 general JSPs; we selected these two groups because they represent rectangular and square JSPs, respectively (see Section 3.4). For both groups, we generated 1000 instances using the procedures discussed in Section 3.2. Three of the four static cost models we analyze require computation of *all* optimal solutions to a problem instance, which can number in the tens of millions even for 6×4 and 6×6 general JSPs. Further, $cost_{med}$ for each problem instance is defined as the median search cost over 5000 independent runs of $TS_{Taillard}$, which requires considerable CPU time for even small JSPs. Consequently, extensive analysis of static cost models for larger general JSPs (e.g., 10×10) is currently impractical. For each of the 2000 problem instances, we used an independent implementation of Beck and Fox's [5] constraint-directed scheduling algorithm to compute the optimal makespan and to enumerate the set of optimal solutions. Finally, we note that the distribution of $log_{10}(cost_{med})$ is approximately normal for both problem groups, with any deviation due to the presence of a few very high-cost problem instances.

4.2 The number of optimal solutions

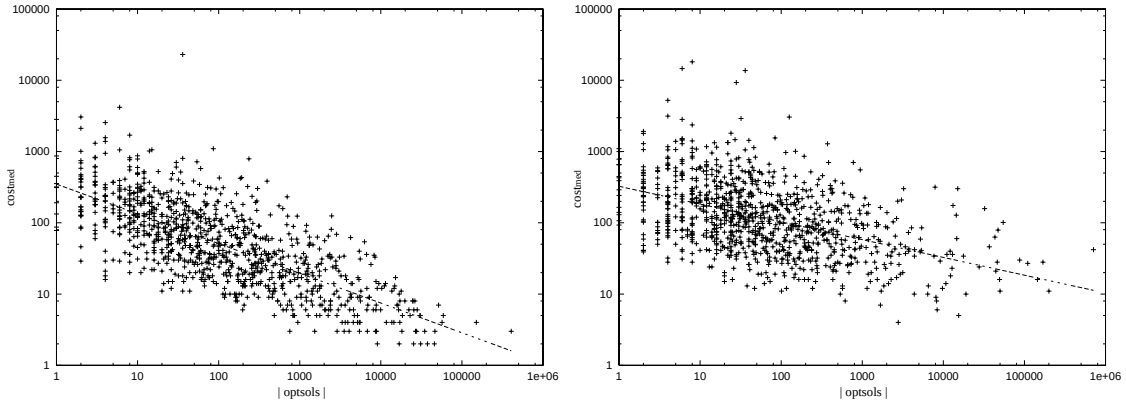

Fig. 1. Scatter-plots of $log_{10}(|optsols|)$ versus $log_{10}(cost_{med})$ for 6×4 (left figure) and 6×6 (right figure) general JSPs; the least-squares fit lines are super-imposed. The r^2 values for the corresponding regression models are 0.5307 and 0.2231, respectively.

The first static cost model we consider is based on the number of optimal solutions to a problem instance, which we denote $|optsols|$. Intuitively, a decrease in the

number of optimal solutions should yield an increase in local search cost. This observation formed the basis of the first static cost model of local search in both SAT and the CSP, first introduced by Clark et al. [10] (and later refined by Singer et al. [27]). Clark et al. demonstrated a relatively strong negative log_{10}-log_{10} correlation between the number of optimal solutions and search cost for three local search algorithms, with r-values ranging anywhere from -0.77 to -0.91. However, the model failed to account for the large cost variance observed for problems with small numbers of optimal solutions, where model residuals varied over three or more orders of magnitude. We have also observed very similar behavior for $TS_{Taillard}$ in the general JSP [36].

We show scatter-plots of $log_{10}(|optsols|)$ versus $log_{10}(cost_{med})$ for 6×4 and 6×6 general JSPs in Figure 1. The r^2 values for the corresponding regression models are 0.5365 and 0.2223, respectively. Although the model residuals are clearly heterogeneous [10] [36], the results are consistent with the computed rank correlation coefficients (-0.7277 and -0.4661, respectively). In comparing the results for the 6×4 and 6×6 general JSPs, it is important to note the large difference in size of the search spaces: 2^{60} versus 2^{90}, respectively. Consequently, although the range of $log_{10}(|optsols|)$ is nearly identical in both cases, the relative number of optimal solutions is, on average, much smaller for the 6×6 general JSP. Given that the accuracy of the $|optsols|$ model is poor for instances with small numbers of optimal solutions, the discrepancy between the r^2 values of the 6×4 and 6×6 general JSPs can be explained by noting that the frequency of instances with relatively small numbers of optimal solutions is larger in square general JSPs [36].

The results presented in Figure 1 indicate that for typical general JSPs, a static cost model based on $|optsols|$ is relatively inaccurate, accounting for roughly 50% of the variance in search cost in the *best* case. In the general JSP, as $n/m \rightarrow \infty$, the frequency of problem instances with a large number of optimal solutions increases. By extrapolation, we would then expect the accuracy of the $|optsols|$ model to increase as $n/m \rightarrow \infty$. In contrast, the accuracy of the model appears worst for the most difficult class of general JSP (i.e., those with $n/m \approx 1.0$), with model residuals varying over 2 to 3 orders of magnitude.

4.3 Backbone size

Recently, researchers have introduced several problem difficulty models based on the concept of a backbone. Informally, the backbone of a problem instance is the set of solution attributes that have identical values in *all* optimal solutions to the instance. For example, in SAT the backbone is the set of Boolean variables whose value is identical in all satisfying assignments; in the TSP, the backbone consists of the set of edges common to all optimal tours. The recent interest in backbones stems largely from the discovery that backbone size (as measured by the fraction

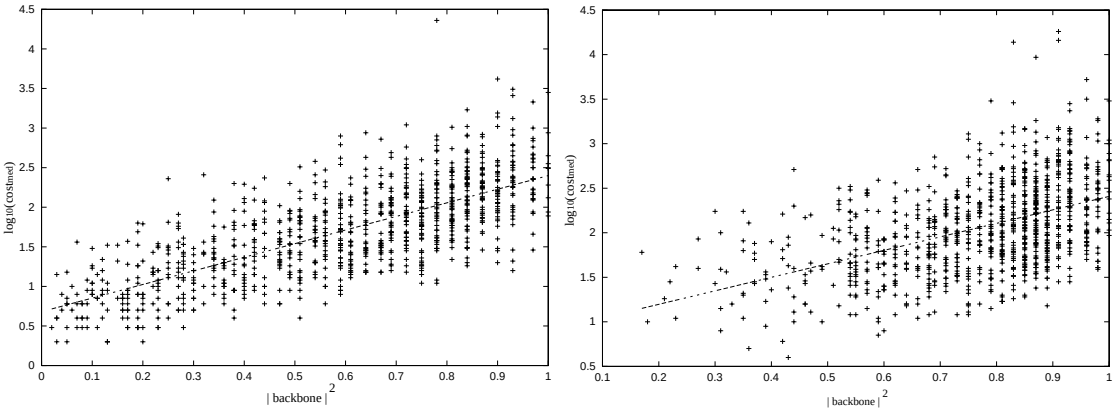

Fig. 2. Scatter-plots of $|\textit{backbone}|^2$ versus $\textit{log}_{10}(\textit{cost}_{\textit{med}})$ for 6×4 (left figure) and 6×6 (right figure) general JSPs; the least-squares fit lines are super-imposed. The r^2 values for the corresponding regression models are 0.5307 and 0.2231, respectively.

of solution attributes appearing in the backbone) is correlated with search cost in SAT (e.g., see Monasson et al. [21]). Specifically, Parkes [24] showed that large-backboned SAT instances begin to appear in large quantities in the critical region of the phase transition (for a more detailed investigation into the relationship between backbone size and the SAT phase transition, see Singer et al. [27] or Singer [26]). Similarly, Achlioptas et al. [1] demonstrated a rapid transition from small to large-backboned instances in the phase transition region. While researchers have demonstrated a correlation between backbone size and problem difficulty in SAT, the *degree* to which backbone size accounts for the variability in problem difficulty remains largely unknown.

Only Slaney and Walsh [28] have studied the influence of backbone size on search cost in problems other than SAT. Focusing on constructive search algorithms, they analyzed the cost of both finding optimal solutions and proving optimality for a number of NP-complete problems, including the TSP and the number partitioning problem. For these two problems, Slaney and Walsh report a weak-to-moderate correlation between backbone size and the cost of finding an optimal solution (0.138 to 0.388). No studies to date have directly quantified the correlation between backbone size and problem difficulty for local search algorithms.

The definition of a backbone clearly depends on how solutions are represented. The most common solution encoding employed by local search algorithms for the JSP, including $\textit{TS}_{\textit{Taillard}}$, is the *disjunctive graph* [7]. In the disjunctive graph representation, there are $\binom{n}{2}$ Boolean 'order' variables for each of the m machines, where the variables represent precedence relations between distinct pairs of jobs on the same machine. Consequently, we define the backbone of a JSP as the set of Boolean ordering variables that have the same value in all optimal solutions. We define the backbone size in the JSP as the fraction of the possible $m\binom{n}{2}$ order variables that are fixed to the same value in all optimal solutions; we denote the resulting value by $|\textit{backbone}|$.

Following Slaney and Walsh, our initial analysis considered the influence of $|\textit{backbone}|$ on $\textit{log}_{10}(\textit{cost}_{\textit{med}})$. We observed a significant quadratic component in the relationship, and the linear term in the quadratic regression model is statistically insignificant. We show scatter-plots of $|\textit{backbone}|^2$ versus $\textit{log}_{10}(\textit{cost}_{\textit{med}})$ for 6×4 and 6×6 general JSPs in Figure 2. The r^2 values for the corresponding regression models are 0.5307 and 0.2331, respectively. As with the $|\textit{optsols}|$, the model errors are heterogeneous, although the results are consistent with the computed rank correlation coefficients (0.7275 and 0.4701, respectively). In both instances, the r-values (0.7285 and 0.4828, respectively) are significantly larger than that reported by Slaney and Walsh for constructive search algorithms. Further, we found absolutely no evidence that the most difficult instances possess medium-sized backbones, as conjectured by Achlioptas et al. [1] for SAT.

Of more interest is the exceptional similarity between the r^2 values of the $|\textit{backbone}|$ and $|\textit{optsols}|$ models: the absolute differences for the 6×4 and 6×6 general JSPs are only 0.0058 and 0.0108, respectively. Upon closer examination, this phenomenon is due to an extremely high correlation between $|\textit{backbone}|^2$ and $\textit{log}_{10}(|\textit{optsols}|)$: -0.9337 and -0.9103 for 6×4 and 6×6 problems, respectively. Within each problem group, the correlation is near-perfect for instances with large backbones, and gradually decays as $|\textit{backbone}| \rightarrow 0.0$. Our results indicate that, quite unexpectedly, for problem instances with moderate-to-large backbones, the backbone size is essentially a proxy for the number of optimal solutions, and vice-versa. From the standpoint of static cost models for reasonably difficult general JSPs (i.e., those with moderate-to-large backbones), the two features are largely redundant. In retrospect, this observation is not surprising given what is implied by a large backbone – as more order variables are fixed, fewer solutions can satisfy the constraints of the backbone. We conjecture that a similar phenomenon can be observed in SAT.

4.4 The average distance between random local optima

Search in algorithms with a strong bias toward local optima, such as tabu search, iterated local search, and certain hybridized genetic algorithms, is largely constrained to the sub-space of local optima. Consequently, we would expect search cost in these algorithms to be at least somewhat correlated with the size of this sub-space. A similar observation led Mattfeld et al. [20] to consider whether differences in the size of the local optima sub-space could account for relative differences in the difficulty of general and workflow JSPs. Using Equation 1 (presented in Section 3.3), Mattfeld et al. define the size of the local optima sub-space as the average normalized distance $\overline{D}(s_1, s_2)$ between distinct pairs of random local optima s_1 and s_2; we denote this measure by $\overline{\textit{loptdist}}$. Although Mattfeld et al. did find mean differences in $\overline{\textit{loptdist}}$ between general and workflow JSPs, they did not analyze the ability of $\overline{\textit{loptdist}}$ to account for the variability in problem difficulty observed within a given set of general or workflow JSP instances.

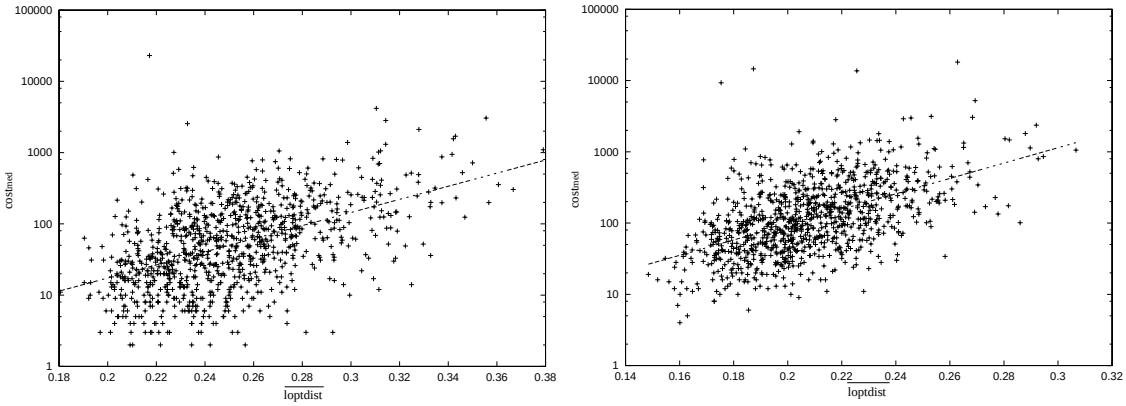

Fig. 3. Scatter-plots of $log_{10}(\overline{loptdist})$ versus $cost_{med}$ for 6×4 (left figure) and 6×6 (right figure) general JSPs; the least-squares fit lines are super-imposed. The r^2 values for the corresponding regression models are 0.2415 and 0.2744, respectively.

For each of our general JSPs, we computed $\overline{loptdist}$ using a set of 5000 random local optima produced using the steepest-descent procedure documented in Section 3.3. Scatter-plots of $\overline{loptdist}$ versus $log_{10}(cost_{med})$ for 6×4 and 6×6 general JSPs are shown in Figure 3. The r^2 values for the corresponding regression models are 0.2415 and 0.2744, respectively. These results confirm the intuition that the size of the local optima sub-space is correlated with the cost of finding optimal solutions under $TS_{Taillard}$, albeit more weakly than either $|optsols|$ or $|backbone|$ in 6×4 general JSPs (r^2 values of 0.2415 versus 0.5365 and 0.5307, respectively). The strength of the correlation is roughly identical to that of $|optsols|$ and $|backbone|$ for 6×6 general JSPs (r^2 values of 0.2744 versus 0.2223 and 0.2331, respectively). Finally, in contrast to both $|optsols|$ and $|backbone|$, the strength of the $\overline{loptdist}$ model appears largely insensitive to relatively small changes in the problem dimensions.

To summarize, the static cost model based on $\overline{loptdist}$ fails to account for a significant proportion of the variability in the cost required by $TS_{Taillard}$ to locate optimal solutions to general JSPs. Further, the $|optsols|$ and $|backbone|$ models are at least as accurate as the $\overline{loptdist}$ model. Later in Section 6.2, we re-visit and ultimately refute Mattfeld et al.'s original claim regarding the ability of differences in $\overline{loptdist}$ to account for differences in the difficulty of general and workflow JSPs.

4.5 The distance between random local optima and the nearest optimal solution

In both the JSP and SAT, the accuracy of the $|optsols|$ model decreases as the number of optimal solutions approaches 0. Analogously, the $|backbone|$ model is more accurate on problem instances with small backbones. Singer et al. [27] recently introduced a static cost model for SAT that largely corrects for these deficiencies. Local search algorithms for SAT, such as GSAT or Walk-SAT [18], quickly locate sub-optimal 'quasi-solutions', in which relatively few clauses are unsatisfied. These quasi-solutions form a sub-space that contains all optimal solutions, and is largely

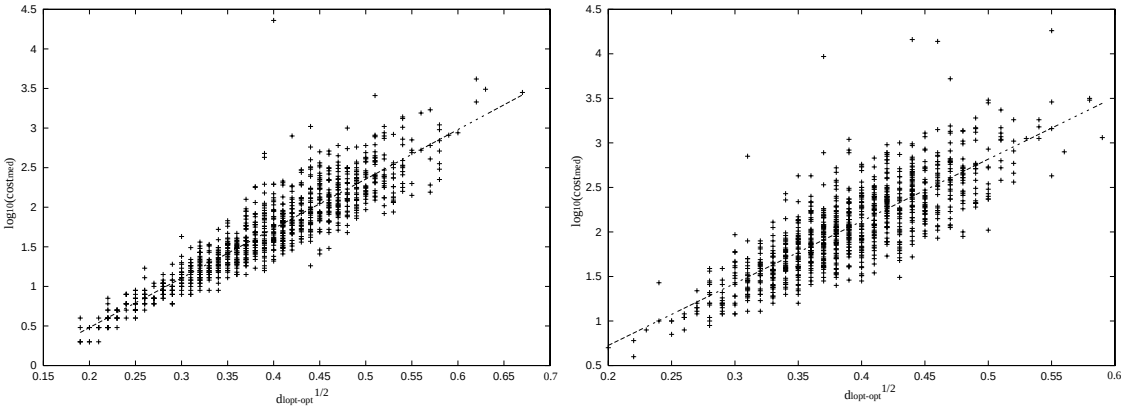

Fig. 4. Scatter-plots of $\sqrt{d_{\textit{lopt-opt}}}$ versus $log_{10}(cost_{med})$ for 6×4 (left figure) and 6×6 (right figure) general JSPs; the least-squares fit lines are super-imposed. The r^2 values for the corresponding regression models are 0.826 and 0.6541, respectively.

interconnected; once a solution in this sub-space is identified, local search is typically restricted to this sub-space. This observation led Singer et al. to hypothesize that the distance between the first quasi-solution encountered and the nearest optimal solution, which we denote $d_{\textit{quasi-opt}}$, largely dictates the cost of local search in SAT.

An obvious analog of the quasi-solutions in SAT are local optima in the JSP. For each of our general JSPs, we generated 5000 random local optima using the steepest-descent procedure documented in Section 3.3. We then computed the mean normalized distance between the resulting local optima and the nearest optimal solution using Equation 1; we denote the result by $d_{\textit{lopt-opt}}$. The distances are normalized to enable comparisons between 6×4 and 6×6 general JSPs; Singer et al. did not perform normalization because the problem size is held constant in their experiments. An initial regression model of $d_{\textit{lopt-opt}}$ versus $log_{10}(cost_{med})$ indicated a slight curvature in the residual plots for small values of $d_{\textit{lopt-opt}}$, which is corrected via substitution by the term $\sqrt{d_{\textit{lopt-opt}}}$. Scatter-plots of $\sqrt{d_{\textit{lopt-opt}}}$ versus $log_{10}(cost_{med})$ for 6×4 and 6×6 general JSPs are shown in Figure 4. The r^2 values for the corresponding regression models are 0.826 and 0.6541. As we discuss in detail below and in Section 6.1, the model errors are heterogeneous; however, the results are consistent with the computed rank correlation coefficients (0.9162 and 0.8072, respectively).

Clearly, the $d_{\textit{lopt-opt}}$ model is significantly more accurate than any of the $|\textit{optsols}|$, $|\textit{backbone}|$, or $\overline{\textit{loptdist}}$ models. In both 6×4 and 6×6 general JSPs, there is strong evidence that the model residuals are heterogeneous, generally growing larger with increases in $\sqrt{d_{\textit{lopt-opt}}}$. Consequently, the $d_{\textit{lopt-opt}}$ model is on average less accurate for problem instances with large $d_{\textit{lopt-opt}}$, or equivalently, with large $cost_{med}$. Singer et al. report a similar phenomenon holds for the analogous static cost model of local search in SAT, and these results are consistent with our prior research on local search cost in the general JSP [36].

The discrepancy between the r^2 values for the 6×4 and 6×6 general JSPs is due to two factors. First, there are more very high-cost 6×6 instances (i.e., those with $log_{10}(cost_{med}) > 3.5$), and large model residuals are typically associated with such instances. Second, although the range of $d_{lopt\text{-}opt}$ is nearly identical in 6×4 and 6×6 general JSPs, the relative frequency of instances for which $\sqrt{d_{lopt\text{-}opt}} \leq 0.3$ is much larger in 6×4 general JSPs (161 versus 67). We further analyze the relationship between the $d_{lopt\text{-}opt}$ model and very high-cost general JSPs in Section 6.1, and consider the influence of the ratio of jobs to machines (n/m) on the accuracy of the $d_{lopt\text{-}opt}$ model in Section 5.2.

To summarize, the static cost model based on $d_{lopt\text{-}opt}$ accounts for a substantial proportion of the variance in the cost required by $TS_{Taillard}$ to locate optimal solutions to 'typical' general JSPs. With few exceptions, the model residuals vary over roughly 1 to 1.5 orders of magnitude in the 6×4 and 6×6 problems, respectively; the improvement is substantial in comparison to the residuals for the models based on either $|optsols|$, $|backbone|$, or $\overline{loptdist}$. Finally, we observe that the $d_{lopt\text{-}opt}$ model is also consistent with the observation that hard (easy) problem instances tend to be hard (easy) for all local search algorithms, as discussed in Section 3.4. Intuitively, if the distance between random local optima and the nearest optimal solution for a particular problem instance is very large, we would expect the instance to be difficult for *any* algorithm based on local search, as search in such algorithms clearly progresses in small increments.

4.6 Models based on multiple search space features

Table 1
The correlation (Pearson's) between search space features for 6×4 general JSPs.

	$log_{10}(\|optsols\|)$	$\|backbone\|$	$\overline{loptdist}$	$d_{lopt\text{-}opt}$
$log_{10}(\|optsols\|)$	1.0	-0.921	-0.039	-0.751
$\|backbone\|$	-0.921	1.0	0.006	0.722
$\overline{loptdist}$	-0.039	0.006	1.0	0.571
$d_{lopt\text{-}opt}$	-0.751	0.722	0.571	1.0

We now consider whether we can improve the accuracy of the $d_{lopt\text{-}opt}$ model by *simultaneously* considering $d_{lopt\text{-}opt}$ in conjunction with the three other search space features considered earlier in this section. We proceed via well-known multiple regression methods. Ideally, the independent variables in a multiple regression model are highly correlated with the dependent variable, but not with each other; if the independent variables are highly correlated, they are said to be collinear. Collinearity is known to cause difficulties for multiple regression model selection techniques, in part because the regression coefficients are not unique, making interpretation very difficult [23]. In Table 1, we show the pair-wise correlation (for 6×4 general JSPs) between the four search space features that serve as the independent variables in our multiple regression model. Similar correlations hold for 6×6 general

JSPs, indicating a high degree of collinearity exists among the four search space features we have considered. Finally, we note that when the sample size is large, terms may be statistically significant due to high power, but in reality may have very small practical effect: i.e., dropping these terms yields a negligible reduction in the model r^2.

We first consider multiple regression models with only additive terms. In both 6×4 and 6×6 general JSPs, the models resulting from forward selection, backward elimination, and step-wise model selection methods [23] [11] are very different, as expected given collinear independent variables and a large sample size. However, the $d_{\textit{lopt-opt}}$ term was present in all of the resulting models, and was consistently the most statistically significant term. For 6×4 and 6×6 general JSPs, the best multiple regression models we obtained yielded r^2 values of 0.8296 and 0.6589, respectively; further, the r^2 values for all models were very similar. Given that the corresponding r^2 values for the basic $d_{\textit{lopt-opt}}$ model are 0.8260 and 0.6541, we conclude that the addition of the $\overline{\textit{loptdist}}$, $|\textit{optsols}|$, and $|\textit{backbone}|$ features fails to enhance the accuracy of the $d_{\textit{lopt-opt}}$ model. Similarly, we found *no* statistically significant interaction terms. Further, the r^2 values for any models with interaction terms were no larger than those obtained by models without interaction terms.

Interestingly, although Singer et al. control for backbone size in their experiments, they do not explicitly indicate whether an interaction effect between the backbone size and the mean distance between random quasi-solutions and the nearest optimal solution ($d_{\textit{quasi-opt}}$) was observed. However, their results do suggest a lack of interaction effect, in that the regression slopes observed for their $d_{\textit{quasi-opt}}$ model are largely homogeneous across a wide range of backbone sizes and clause-to-variable ratios (e.g., see Singer et al. (2000), Table 2, p. 249); the intercepts are slightly more variable, which is likely due in part to the presence of high-residual problem instances.

4.7 A note on backbone robustness

In addition to introducing the $d_{\textit{quasi-opt}}$ static cost model for SAT, Singer et al. also posited a causal model to account for the variability in $d_{\textit{quasi-opt}}$ observed for different problem instances. Their model is based on the notion of *backbone robustness*. A SAT instance is said to have a *robust* backbone if a substantial number of clauses can be deleted before the backbone size is reduced by at least half. Conversely, an instance is said to have a *fragile* backbone if the deletion of just a few clauses reduces the backbone size by half or more. Singer et al. argue that "backbone fragility approximately corresponds to how extensive the quasi-solution area is" ([27], p. 251), by noting that a fragile backbone allows for large $d_{\textit{quasi-opt}}$ because of the sudden drop in backbone size, while $d_{\textit{quasi-opt}}$ is necessarily small in problem instances with robust backbones.

As evidence of this hypothesis, Singer et al. measured a moderate (≈ -0.5) negative correlation between backbone robustness and the log of local search cost for large-backboned SAT instances. Surprisingly, this correlation degraded as the backbone size was decreased, leading to the conjecture that "finding the backbone is less of an issue and so backbone fragility, which hinders this, has less of an effect" ([27], p. 254); this conjecture was never explicitly tested. We have previously reported very similar results for general JSPs [36]. As indicated in Section 4.5 and more fully in Section 6, we have since discovered relatively serious deficiencies in the $d_{lopt\text{-}opt}$ model (and by analogy, likely deficiencies in the $d_{quasi\text{-}opt}$), and feel it is somewhat premature to posit causal hypotheses before the source of these deficiencies is completely understood. As a consequence, we have not pursued further analyses of backbone robustness in the JSP.

5 Applications of the $d_{lopt\text{-}opt}$ model

The analyses presented in Section 4 demonstrate that the $d_{lopt\text{-}opt}$ static cost model accounts for a substantial proportion of the variance in the cost of finding optimal solutions to typical general JSPs using $TS_{Taillard}$. Further, more complex models that consider $d_{lopt\text{-}opt}$ in conjunction with backbone size, the number of optimal solutions, and the size of the local optima sub-space fail to yield even marginal improvements in accuracy. In this section, we additionally show that the $d_{lopt\text{-}opt}$ model accounts for both (1) a substantial proportion of the variance in the cost of finding *sub-optimal* solutions to typical general JSPs using $TS_{Taillard}$ and (2) differences in the relative difficulty of general JSPs with different job-to-machine ratios.

5.1 Modeling the cost of locating sub-optimal solutions

Because they are incomplete, local search algorithms are only used to find solutions to satisfiable SAT instances, where the evaluation of the global optima is known, and is equal to the total number of clauses m. Such a priori knowledge leads to the obvious termination criterion: keep searching until a global optimum is located. Consequently, analyses of problem difficulty for local search in SAT only consider the cost required to locate globally optimal solutions. For most NP-complete problems, however, the evaluation of the global optimum is not known a priori. Armed only with the knowledge that larger run-times generally lead to higher-quality solutions, local search practitioners generally use the following termination criterion: allocate as much CPU time as possible, and return the best solution found.

Although larger run-times generally yield higher-quality solutions, the relationship is typically discontinuous, non-linear, or both. Often, small or moderate increases in run-time fail, on average, to improve solution quality; for example, Stützle ([31], p.

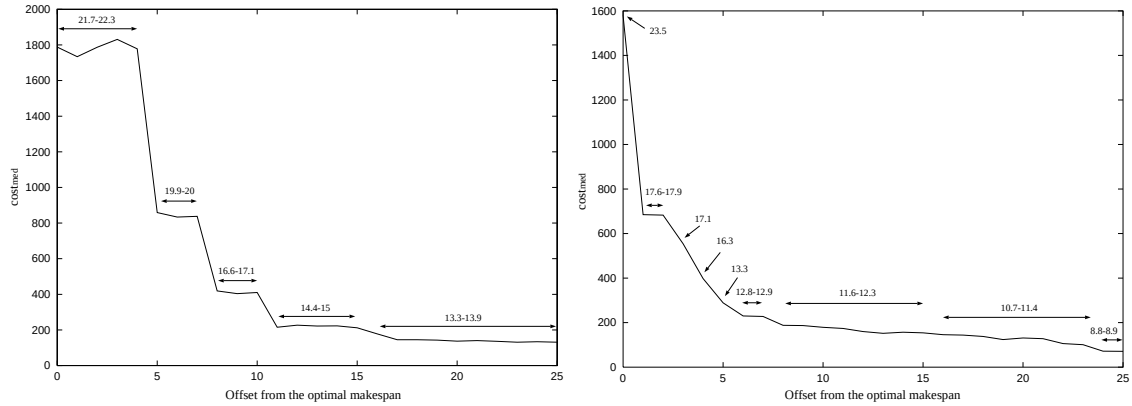

Fig. 5. The offset x from the optimal makespan C^*_{max}, $0 \leq x \leq 25$, versus the cost $cost_{med}(x)$ required to locate a solution with $C_{max} \leq C^*_{max} + x$ for two 6×6 general JSPs. The numeric annotations indicate either $d_{lopt-T(x)}$ for a specific x, or the range of $d_{lopt-T(x)}$ over a contiguous sub-interval of x.

47) notes that in the Traveling Salesman Problem "...instances appear to have 'hard cliffs' for the local search algorithm, corresponding to deep local minima, which are difficult to pass." Similar observations have been reported for a variety of NP-complete problems, including the JSP. Another manifestation of this phenomenon has been observed by several researchers, including ourselves. Here, multiple independent trials of a particular local search algorithm typically yield sub-optimal solutions that can be partitioned into a very small number of subsets (often 1), with each subset containing solutions with identical evaluations.

One simple way to visualize this phenomenon is to plot the cost required to achieve a solution with an evaluation of *at least* $C^*_{max} + x$ over a wide range of $x \geq 0$. In Figure 5, we provide examples of such plots for two moderately difficult 6×6 general JSPs. In both plots, the offset from the optimal makespan x is varied from 0 to 25, and the median cost (over 5000 independent runs of $TS_{Taillard}$) required to find a solution with an evaluation of at least $C^*_{max} + x$ is computed for each x, which we denote by $cost_{med}(x)$. In the left side of Figure 5, we see a typical example of a problem instance with discrete jumps in search cost at specific sub-optimal makespans, with plateaus in search cost in between the jump points. In the right side of Figure 5, we show a problem instance for which the decay in search cost is generally more gradual; a large, discontinuous jump in search cost occurs only between $x = 0$ and $x = 1$.

As shown in Section 4, the $d_{lopt\text{-}opt}$ static cost model accounts for a significant proportion of the variance in the cost of finding optimal solutions to general JSPs using $TS_{Taillard}$. Intuitively, this cost is large if $TS_{Taillard}$ is, on average, initiated from solutions that are very distant from the nearest optimal solution. We conjecture that this intuition extends to *any* subset of solutions, including sub-optimal solutions; we would expect local search cost to be proportional to the distance between the initial solutions and the nearest target solution. As evidence of this conjecture, we consider a set $T(x)$ containing all solutions with a makespan between C^*_{max} and

$C^*_{max} + x$, $x \geq 0$, and denote the mean distance between random local optima and the nearest solution in the set $T(x)$ by $d_{\textit{init}-T(x)}$; as with the computation of $d_{\textit{lopt-opt}}$, the statistics are taken over 5000 independent samples. We have annotated the plots in Figure 5 with the computed $d_{lopt-T(x)}$, $0 \leq x \leq 25$. In both instances, (1) large jumps in search cost clearly coincide with large jumps in $d_{lopt-T(x)}$, (2) intervals of roughly constant search cost correspond to contiguous sub-intervals of x with nearly identical values of $d_{lopt-T(x)}$, and (3) gradual drops in search cost coincide with gradual drops in $d_{lopt-T(x)}$. Consequently, we hypothesize that $d_{lopt-T(x)}$ accounts for a significant proportion of the variance in the cost of finding *both* optimal and sub-optimal solutions to typical general JSPs using $TS_{Taillard}$.

To test this hypothesis, we computed $\textit{cost}_{med}(x)$ and $d_{\textit{init}-T(x)}$ for both our 6×4 and 6×6 general JSPs, varying x from 1 to 25. Finding solutions to 6×4 and 6×6 general JSPs with $C_{max} > C^*_{max} + 25$ is generally quite easy for $TS_{Taillard}$, with $\textit{cost}_{med}(25) \leq 100$ in all but a few cases. Under this methodology, we are effectively creating 25 derivatives of each problem instance (one for each value of x), which results in new 'sub-optimal' 6×4 and 6×6 problem groups, each with $25\,000$ instances. For many of the derivative instances, especially those produced using large x, $\textit{cost}_{med}(x) = 0$, or equivalently $d_{\textit{init}-T(x)} \approx 0.0$. We observed 1293 6×4 zero-cost instances, and 60 6×6 zero-cost instances; in both cases, the zero-cost instances are excluded in the following analysis.

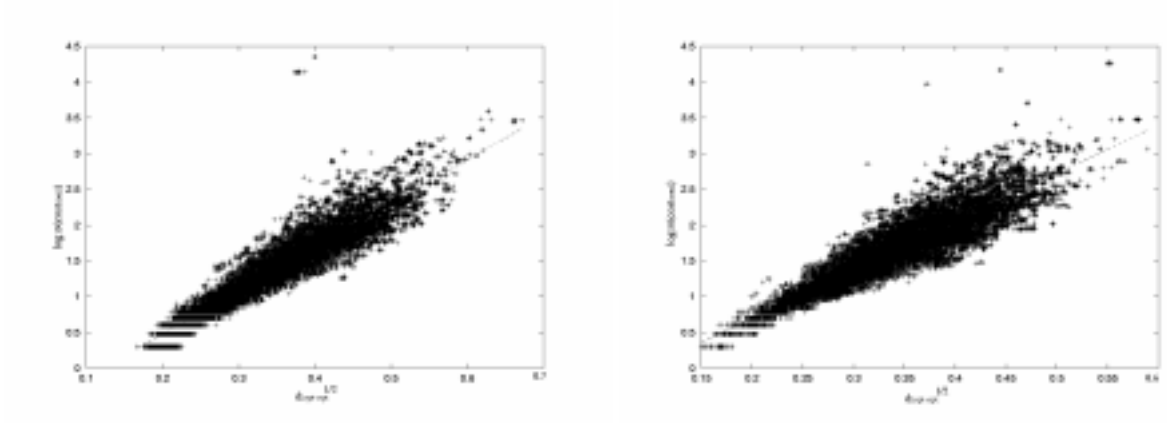

Fig. 6. Scatter-plots of $\sqrt{d_{\textit{lopt-opt}}}$ versus $\textit{log}_{10}(\textit{cost}_{med})$ for the sub-optimal 6×4 (left figure) and 6×6 (right figure) general JSP problem groups; the regression lines are super-imposed. The r^2 values for the corresponding regression models are 0.8866 and 0.8252, respectively.

In Figure 6, we show scatter-plots of $\sqrt{d_{\textit{lopt-opt}}}$ versus $\textit{log}_{10}(\textit{cost}_{med})$ for the sub-optimal 6×4 and 6×6 problem groups; the r^2 values for the corresponding regression models are 0.8866 and 0.8252, respectively. Clearly, the $d_{\textit{lopt-opt}}$ model accounts for most of the variance in the cost of finding sub-optimal solutions to typical general JSPs using $TS_{Taillard}$. We observed larger r^2 values in the sub-optimal 6×4 and 6×6 problem groups than for the corresponding problem groups analyzed in Section 4.5: 0.8866 versus 0.8260 for the 6×4 problems and 0.8252 versus 0.6541 for the 6×6 problems. We explain the greater accuracy of the $d_{\textit{lopt-opt}}$ model on the

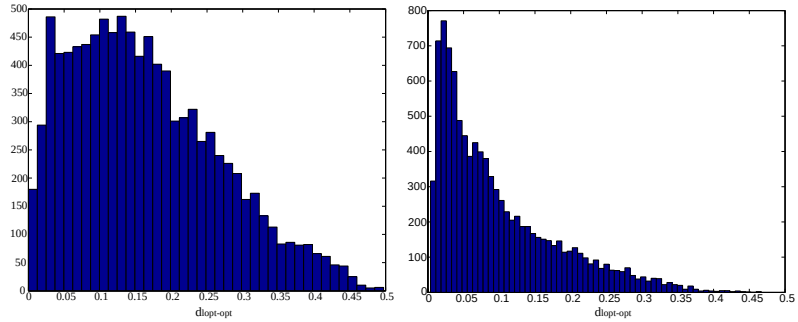

Fig. 7. Histograms of $d_{\textit{lopt-opt}}$ for $10\,000$ 4×3 (left figure) and 7×3 (right figure) general JSPs.

sub-optimal problem groups by noting that the proportion of instances with small values of $d_{\textit{lopt-opt}}$ is larger in the sub-optimal problem groups, which corresponds to the region where the $d_{\textit{lopt-opt}}$ model is most accurate.

We conclude by noting that the $d_{\textit{lopt-opt}}$ model provides the first quantitative explanation for 'cliffs' in local search cost observed at particular sub-optimal evaluations: abrupt changes in local search cost occur where there are abrupt changes in $d_{\textit{lopt-opt}}$. Similarly, the plateaus observed in Figure 5 occur because solutions on the plateau are equi-distant from random local optima; $TS_{\textit{Taillard}}$ is equally likely to encounter any of the solutions on the plateau, given a fixed run-time. Similarly, gradual increases in search cost occur when slightly better solutions are only marginally farther from random local optima.

5.2 Explaining differences in the relative difficulty of square versus rectangular JSPs

Given the accuracy of the $d_{\textit{lopt-opt}}$ model for both 6×4 and 6×6 general JSPs, it is natural to consider whether or not differences in the distribution of $d_{\textit{lopt-opt}}$ for problems with different ratios of n/m can account for the empirical observation that square JSPs are generally more difficult than rectangular JSPs.

Fixing $m = 3$, we generated $10\,000$ general JSPs for $n = 4$ through $n = 7$; although we initially considered larger values of n, the huge number of *optimal* solutions (> 1 billion in many cases) prevented us from efficiently computing $d_{\textit{lopt-opt}}$. We show histograms of $d_{\textit{lopt-opt}}$ for 4×3 and 7×3 general JSPs in Figure 7. In 4×3 general JSPs, the right-tail mass of the distribution is substantial (e.g., for $d_{\textit{lopt-opt}} \geq 0.3$), especially in comparison to the distribution for 7×3 general JSPs, where instances with $d_{\textit{lopt-opt}} \geq 0.3$ are relatively rare. We have also generated histograms for general JSPs with $n/m < 1$, observing a continued shift of the distribution mass toward 0.5.

Although not entirely conclusive, our results provide strong evidence that the right-

tail mass of the $d_{\textit{lopt-opt}}$ distribution vanishes as $n/m \rightarrow \infty$, suggesting a cause for the empirical observation that square JSPs are generally more difficult than rectangular JSPs. Further, we hypothesize that the shift from exponential to polynomial growth in search cost at $n/m \approx 6$ [33] is due to the disappearance of any significant mass in the right tail of the $d_{\textit{lopt-opt}}$ distribution. However, due to the huge number of optimal solutions in problem instances with $n/m \geq 4$, we are currently unable to empirically test this hypothesis. Finally, we note that the accuracy of the $d_{\textit{lopt-opt}}$ model should further improve as $n/m \rightarrow \infty$, due to the increasing frequency of instances with small values of $d_{\textit{lopt-opt}}$. Consequently, from the standpoint of static cost models, only general JSPs with $n/m \approx 1.0$ warrant significant attention in the future.

In a previous paper [36], we argued that a shift in the distribution of $|\textit{backbone}|$, and not $d_{\textit{lopt-opt}}$, was responsible for differences in the relative difficulty of square versus rectangular JSPs. While our original observation still holds (i.e., the proportion of instances with small backbones grows as $n/m \rightarrow \infty$), we have chosen to re-cast our original results in terms of the more accurate static cost model based on $d_{\textit{lopt-opt}}$.

6 Limitations of the $d_{\textit{lopt-opt}}$ model

Although the $d_{\textit{lopt-opt}}$ static cost model largely accounts for the cost of finding both optimal and sub-optimal solutions to typical general JSPs using $TS_{\textit{Taillard}}$, and provides an explanation for the differences in the relative difficulty of general JSPs with different job-to-machine ratios, the model is by no means perfect. As discussed in Section 4.5, the $d_{\textit{lopt-opt}}$ model is less accurate for problem instances with large values of $d_{\textit{lopt-opt}}$ (or, equivalently, large $\textit{cost}_{\textit{med}}$), and consequently fails to account for roughly 35% of the cost variance in our 6×6 general JSPs.

In this section, we identify two additional limitations of the $d_{lopt-opt}$ model. First, we conclusively demonstrate that the accuracy of the $d_{lopt-opt}$ model is exceptionally poor for very high-cost general JSPs (we provided some preliminary evidence for this conclusion in Section 4.5). Second, we show that the $d_{lopt-opt}$ model is unable to account for a significant proportion of the variance in the cost of finding optimal solutions to more structured JSPs: e.g., workflow JSPs. Although both of the results presented in this section are clearly 'negative', we feel it is important to identify and report such deficiencies, as research into why the $d_{lopt-opt}$ model fails in these circumstances is likely to lead to more general and accurate static cost models in the future.

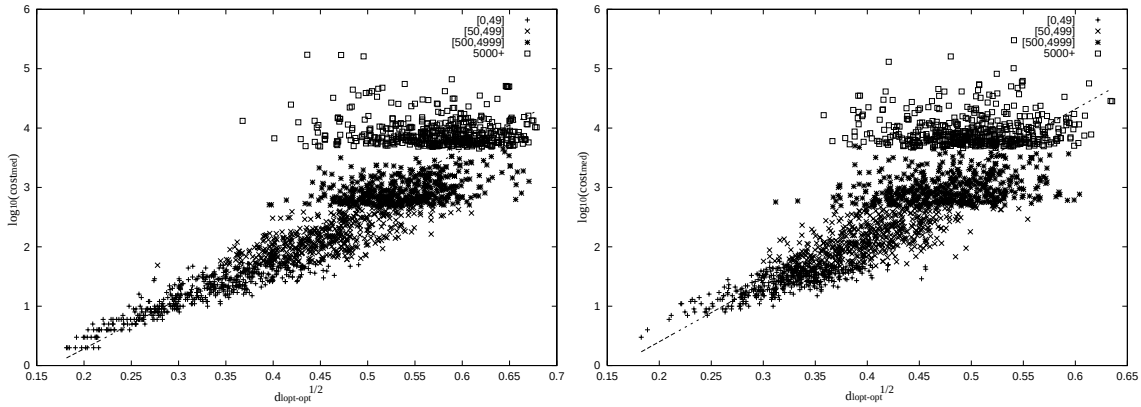

Fig. 8. Scatter-plots of $\sqrt{d_{lopt\text{-}opt}}$ versus $log_{10}(cost_{med})$ for easy ($cost_{med} \in [1, 49]$), moderate ($cost_{med} \in [50, 499]$), high ($cost_{med} \in [500, 4999]$), and very high-cost ($cost_{med} \in [5000, \infty]$) 6×4 (left figure) and 6×6 (right figure) general JSPs; the least-squares fit lines are super-imposed. The r^2 values for the corresponding regression models are 0.7742 and 0.6820, respectively.

6.1 Modeling search cost in exceptionally hard general JSPs

In Section 4.5, we provided evidence that the $d_{lopt\text{-}opt}$ model is less accurate for problem instances with large values of $d_{lopt\text{-}opt}$, or equivalently, large $cost_{med}$. Of particular concern are the rare, very high-cost ($cost_{med} \geq 10000$) instances appearing in both sides of Figure 4; in all but one case, these instances possess the largest residuals under the corresponding regression model. To determine whether large model residuals are typically associated with very high-cost general JSPs, we created groups of 6×4 and 6×6 general JSPs with equal proportions of problem instances over the range of $cost_{med}$. Specifically, we sub-divided the range of possible $cost_{med}$ values into the following four contiguous intervals: $[1, 49]$, $[50, 499]$, $[500, 4999]$, and $[5000, \infty]$. These intervals qualitatively correspond to easy, moderate, difficult, and very difficult problem instances, respectively. For both 6×4 and 6×6 general JSPs, we then produced 500 instances belonging to each interval using a generate-and-test procedure.

We provide scatter-plots of $\sqrt{d_{lopt\text{-}opt}}$ versus $log_{10}(cost_{med})$ for the two resulting problem groups in Figure 8. The r^2 values for the corresponding regression model are 0.7742 and 0.6820, respectively. First, we note that because the high-cost and very high-cost instances reside in the right-tail of the $log_{10}(cost_{med})$ distribution, the large relative frequencies of problem instances with $cost_{med}$ near the lower bounds of the corresponding intervals was expected. In both problem groups, we observe a substantial reduction in the accuracy of the $d_{lopt\text{-}opt}$ model for high-cost ($500 \leq cost_{med} \leq 4999$) instances. For very high-cost instances ($cost_{med} \geq 5000$), the degradation in accuracy is far more extreme, such that $\sqrt{d_{lopt\text{-}opt}}$ provides almost no information about $cost_{med}$. These results clearly reinforce the deficiencies of the $d_{lopt-opt}$ model discussed in Section 4.5: accuracy is inversely proportional to both

$d_{lopt\text{-}opt}$ and $cost_{med}$. As a direct consequence, although we are now able to account for much of the variability in search cost for 'typical' general JSPs, an understanding of the search space properties that make certain problems exceptionally difficult for $TS_{Taillard}$ remains elusive.

Several researchers have reported situations in which problems that are exceptionally difficult for one algorithm are much easier for other algorithms [29] [15]. To date, this phenomenon has only been observed in constructive search algorithms, and occurs when one algorithm makes a particular sequence of decisions that yields a very difficult sub-problem [29]. Although this phenomenon has *not* been observed for local search in any NP-complete problem, it does raise an obvious question: "Is the exceptional difficulty of our very high-cost general JSPs algorithm-independent?". To informally answer this question, we solved both the 500 very high-cost and 1000 'typical' 6×6 (i.e., those considered in Section 4) instances using two local search algorithms other than $TS_{Taillard}$, and a constructive heuristic search algorithm. Specifically, we considered the following local search algorithms: (1) Nowicki and Smutnicki's state-of-the-art tabu search algorithm [22] and (2) van Laarhoven et al.'s simulated annealing algorithm [35]. We selected Nowicki and Smutnicki's algorithm because it uses a more powerful move operator than $TS_{Taillard}$, and employs an intensification mechanism (see Section 3.3); van Laarhoven et al.'s algorithm provides a well-known alternative local search paradigm to tabu search. The constructive algorithm we consider is Beck and Fox's constraint-directed scheduling algorithm [5], which was selected because it shares little in common with local search algorithms for the JSP. In all three cases, the search cost (as measured by the median search cost over 1000 independent trials for the two local search algorithms, and the number of nodes visited by the constructive algorithm) was *generally* larger in the very high-cost instances. However, we did find some exceptional instances that were easily solved by the other algorithms. Upon closer examination, we found that these instances are extremely sensitive to the length of the tabu list of $TS_{Taillard}$. We conclude that, with a few exceptions, the difficulty of our very high-cost general JSPs is algorithm-independent.

Finally, we conjecture that the failure of the $d_{lopt\text{-}opt}$ model to account for local search cost in very difficult problem instances also extends to SAT. Although Singer et al. do not provide scatter-plots of $d_{lopt\text{-}opt}$ versus $log_{10}(cost_{med})$ for high-cost problem instances (i.e., those with large backbones), their analysis does indicate that the accuracy of the $d_{lopt\text{-}opt}$ model is inversely proportional to backbone size (e.g., see Singer et al. (2000), Table 2, p. 249), and as a consequence, to $cost_{med}$ (as in the general JSP, local search cost and backbone size are positively correlated in SAT). Further, very high-cost SAT instances possess the largest residuals under Singer et al.'s model of backbone robustness (e.g., see Singer et al. (2000), Figure 11, p. 255), which in turn is correlated with $d_{lopt\text{-}opt}$.

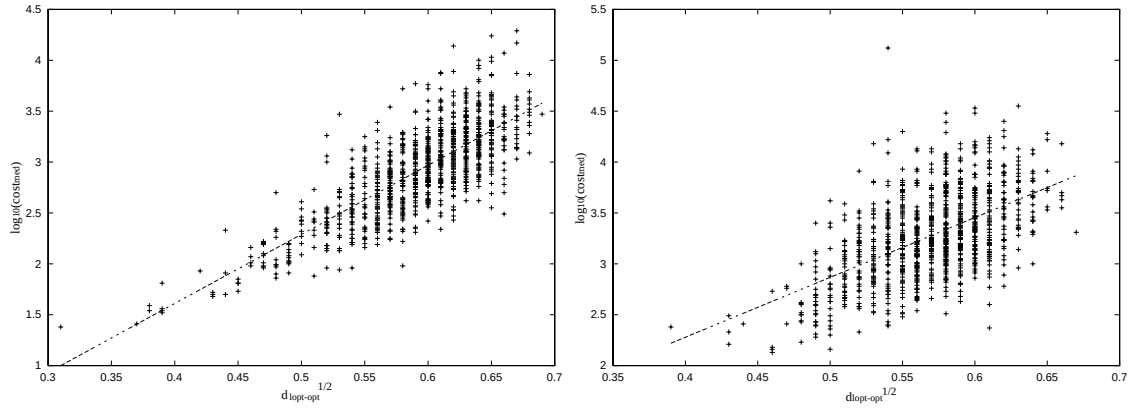

Fig. 9. Scatter-plots of $\sqrt{d_{\textit{lopt-opt}}}$ versus $log_{10}(cost_{med})$ for 6×4 (left figure) and 6×6 (right figure) workflow JSPs; the least-squares fit lines are super-imposed. The r^2 values for the corresponding regression models are 0.6082 and 0.3049, respectively.

In the JSP and SAT, the primary problem constraints are the job routing orders π_i and the disjunctive clauses, respectively. Generally, benchmark suites consist of problem instances in which these constraints are, in expectation, completely random. An important issue is then generalization: real-world problems have non-random constraints, and it is unclear whether static cost models developed for random instances are extensible to instances with more structured constraints. To study the effect of non-random constraints on the accuracy of the static cost models examined in Section 4, we apply the same analysis to JSPs with workflow–which impose a simple structure on the job routing orders. All results are produced using groups of 6×4 and 6×6 workflow JSPs, each containing 1000 problem instances; the details of the problem generation process are discussed in Section 3.2.

We first consider the results for 6×4 workflow JSPs. A scatter-plot of $\sqrt{d_{\textit{lopt-opt}}}$ versus $log_{10}(cost_{med})$ is shown in the left side of Figure 9. The r^2 value for the corresponding regression model is 0.6082, which is roughly 75% of the r^2 value observed for 6×4 general JSPs (see Section 4.5). In contrasting the left sides of Figures 4 and 9, it is clear that the presence of workflow partitions greatly increases the relative frequency of instances with large values of $d_{\textit{lopt-opt}}$, which partially explains the reduction in the observed r^2. Workflow partitions also have a substantial impact on the accuracy of the other models considered in Section 4. For example, we observed an r^2 value of only 0.0016 for the $\overline{\textit{loptdist}}$ model, in contrast to 0.2415 for 6×4 general JSPs: the size of the local optima sub-space has effectively no influence on the cost of search with $TS_{Taillard}$. In contrast to the results for $d_{\textit{lopt-opt}}$, we observed roughly a 20% *increase* in the r^2 values for the $|optsols|$ and $|backbone|$ models, to 0.6155 and 0.6107, respectively. Consequently, the accuracy of the $|optsols|$, $|backbone|$, and $d_{\textit{lopt-opt}}$ models is nearly identical for 6×4 workflow JSPs. Finally, we note that strong correlation between $log_{10}(|optsols|)$ and $|backbone|^2$ is maintained ($r = -0.8936$).

Next, we consider the results for 6×6 workflow JSPs; a scatter-plot of $\sqrt{d_{\textit{lopt-opt}}}$ versus $\textit{log}_{10}(\textit{cost}_{\textit{med}})$ is shown in the right side of Figure 9. Here, we see a further reduction in the accuracy of the $d_{\textit{lopt-opt}}$ model: the r^2 value is over 50% less than that observed for 6×6 general JSPs, dropping from 0.6541 to 0.3049. As with 6×4 workflow JSPs, the reduction in accuracy is partially due to dramatic increases in the relative frequency of problem instances with very large values of $d_{\textit{lopt-opt}}$. Similarly, the correlation between $\overline{\textit{loptdist}}$ and $\textit{log}_{10}(\textit{cost}_{\textit{med}})$ is insignificant ($r^2 = 0.002983$), and we observed an increase in the accuracy of the $|\textit{optsols}|$ and $|\textit{backbone}|$ models (to r^2 values of 0.3345 and 0.2974, respectively); the relatively strong correlation between $\textit{log}_{10}(|\textit{optsols}|)$ and $|\textit{backbone}|^2$ was again maintained ($r = -0.8346$).

Our results cast serious doubt on Mattfeld et al.'s assertion that differences in the relative difficulty of general and workflow JSPs are due to differences in $\overline{\textit{loptdist}}$. While we observed statistically significant differences between the *mean* $\overline{\textit{loptdist}}$ of general and workflow JSPs (e.g., 0.2080 versus 0.3465 in 6×6 general and workflow JSPs, respectively), we also computed a relatively weak correlation between $\overline{\textit{loptdist}}$ and $\textit{log}_{10}(\textit{cost}_{\textit{med}})$ for general JSPs – for workflow JSPs, the correlation between these same variables is effectively 0. Statistically significant mean differences between general and workflow JSPs also exist for $|\textit{optsols}|$, $|\textit{backbone}|^2$, and $\sqrt{d_{\textit{lopt-opt}}}$. Further, each of these models is at least as accurate as the $\overline{\textit{loptdist}}$ model for *both* general and workflow JSPs. Consequently, we believe that any of the $|\textit{optsols}|$, $|\textit{backbone}|$, and $d_{\textit{lopt-opt}}$ models provide *at least* an equally likely explanation as the $\overline{\textit{loptdist}}$ model for the differences in relative difficulty between general and workflow JSPs.

Finally, we note that our results provide the first solid evidence that the static cost models for random and structured problem instances may in fact be quite different. No research to date has considered the impact of problem structure on the static cost models for SAT. Given the strong similarities between the models for the JSP and SAT, we conjecture that existing static cost models for SAT, because they are based on random problem instances, are likely to be significantly less accurate when applied to structured SAT instances.

7 Conclusion

Drawing from research on problem difficulty in SAT, we demonstrated that the $d_{\textit{lopt-opt}}$ static cost model accounts for much of the variability in the cost of finding optimal solutions to general JSPs using a straightforward tabu search algorithm, $\textit{TS}_{\textit{Taillard}}$. This result was somewhat unexpected, given the differences in both the search space topologies and local search algorithms for the general JSP and SAT. Further, the accuracy of the model is nearly identical in both problems. In the course

of our analyses, we also encountered several other important, unanticipated results: (1) backbone size and the number of optimal solutions are largely redundant search space features, (2) there is no significant interaction effect between $d_{\textit{lopt-opt}}$ and any of the other search space features we considered, and (3) multiple-factor models do not significantly improve upon the accuracy of the $d_{\textit{lopt-opt}}$ model.

We then used the $d_{\textit{lopt-opt}}$ model to provide explanations for two qualitative observations regarding problem difficulty in the JSP. First, we showed that the $d_{\textit{lopt-opt}}$ model accounts for much of the variability in the cost of locating *sub-optimal* solutions to general JSPs using $TS_{\textit{Taillard}}$. The resulting extension provides an explanation for the discontinuous jumps in search cost observed at particular offsets from the optimal makespan. Second, we demonstrated that strong differences in the distributions of $d_{\textit{lopt-opt}}$ provide a possible explanation for differences in the relative difficulty of square versus rectangular JSPs: problem instances with large values of $d_{\textit{lopt-opt}}$ are common in square JSPs, but are relatively rare in rectangular JSPs.

Finally, we showed that the $d_{\textit{lopt-opt}}$ model has some limitations. First, our analyses indicated that the accuracy of the model is inversely proportional to the magnitude of $d_{\textit{lopt-opt}}$, or equivalently, the difficulty of the problem instance. We also found that the accuracy is exceptionally poor on relatively rare, very high-cost problem instances. Second, we demonstrated that the accuracy of the $d_{\textit{lopt-opt}}$ model is significantly worse for a particular class of structured JSPs – those with workflow partitions.

We selected $TS_{\textit{Taillard}}$ precisely because it serves as a baseline for more advanced tabu search algorithms, such as the state-of-the-art algorithm of Nowicki and Smutnicki, which employ more advanced move operators and make more extensive use of long-term memory and intensification mechanisms. With a relatively accurate static cost model of Taillard's algorithm, we can begin to *systematically* assess the influence of these more advanced features on the model. One inherent limitation of our analysis is that it is only directly applicable to tabu-like search algorithms for the JSP. Because static cost models are tied to specific algorithms, it seems likely that other factors may be responsible for local search cost in algorithms such as iterated local search or genetic algorithms, which are based on principles quite different from tabu search. At the same time, it seems likely that variations on the basic $d_{\textit{lopt-opt}}$ model may account for the cost of tabu search in other NP-complete problems.

Because the static cost models for SAT and the JSP are very similar, it also seems likely that our results will be useful to researchers working on models of local search cost in SAT. For example, our analyses indicate that the backbone size and the number of optimal solutions are largely redundant, that simultaneous consideration of number of optimal solutions, backbone size, the average distance between local optima fail to improve the accuracy of the basic $d_{\textit{lopt-opt}}$ model, and that the accuracy of the $d_{\textit{lopt-opt}}$ model is exceptionally poor on very high-cost problem in-

stances. We conjecture similar observations hold in SAT. Similarly, we showed that the static cost models for random and structured problem instances can be very different. If similar results hold in SAT, they would provide some evidence that the best algorithms for solving random instances may be based on different principles than the best algorithms for solving structured instances.

References

[1] Dimitris Achlioptas, Carla Gomes, Henry Kautz, and Bart. Selman. Generating satisfiable problem instances. In Kenneth Ford, editor, *Proceedings of the 17th National Conference on Artificial Intelligence (AAAI-2000)*, pages 256–261. AAAI/MIT Press, 2000.

[2] E. Balas and A. Vazacopoulos. Guided local search with shifting bottleneck for job-shop scheduling. *Management Science*, 44(2):262–275, 1998.

[3] J.W. Barnes and J.B. Chambers. Solving the job shop scheduling problem with tabu search. *IIE Transactions*, 27:257–263, 1995.

[4] J.E. Beasley. OR-library: Distributing test problems by electronic mail. *Journal of the Operational Research Society*, 41(11):1069–1072, 1990.

[5] J. Christopher Beck and Mark S. Fox. Dynamic problem structure analysis as a basis for constraint-directed scheduling heuristics. *Artificial Intelligence*, 117(2):31–81, 2000.

[6] Jacek Blażewicz, Wolfgang Domschke, and Erwin Pesch. The job shop scheduling problem: Conventional and new solution techniques. *European Journal of Operational Research*, 93:1–33, 1996.

[7] Jacek Blażewicz, Erwin Pesch, and Malgorzata Sterna. The disjunctive graph machine representation of the job shop scheduling problem. *European Journal of Operational Research*, 127:317–331, 2000.

[8] Kenneth D. Boese, Andrew B. Kahng, and Sudhakar Muddu. A new adaptive multi-start technique for combinatorial optimization. *Operations Research letters*, 16:101–113, 1994.

[9] John B. Chambers and J. Wesley Barnes. New tabu search results for the job shop scheduling problem. Technical Report ORP96-10, Graduate Program in Operations Research and Industrial Engineering, The University of Texas at Austin, 1996.

[10] David A. Clark, Jeremy Frank, Ian P. Gent, Ewan MacIntyre, Neven Tomov, and Toby Walsh. Local search and the number of solutions. In *Proceedings of the Second International Conference on Principles and Practices of Constraint Programming (CP-96)*, pages 119–133, 1996.

[11] Paul R. Cohen. *Empirical Methods for Artificial Intelligence*. The MIT Press, 1995.

[12] H. Fisher and G.L. Thompson. *Probabilistic Learning Combinations of Local Job-Shop Scheduling Rules*, chapter 15, pages 225–251. Prentice-Hall, Englewood Cliffs, New Jersey, 1963.

[13] Jeremy Frank, Peter Cheeseman, and John Stutz. When gravity fails: Local search topology. *Journal of Artificial Intelligence Research*, 7:249–281, 1997.

[14] Michael R. Garey, David S. Johnson, and Ravi Sethi. The complexity of flowshop and jobshop scheduling. *Mathematics of Operations Research*, 1(2):117–129, 1976.

[15] Ian P. Gent and Toby Walsh. Easy problems are sometimes hard. *Artificial Intelligence*, 70(1–2):335–345, 1994.

[16] Fred Glover and Manuel Laguna. *Tabu Search*. Kluwer Academic Publishers, Boston, MA, 1997.

[17] Tad Hogg, Bernardo A. Huberman, and Colin P. Williams. Phase transitions and the search problem. *Artificial Intelligence*, 81(1–2):1–15, 1996.

[18] Holger H. Hoos. *Stochastic Local Search – Methods, Models, Applications*. PhD thesis, Darmstadt University of Technology, 1998.

[19] A.S. Jain and S. Meeran. Deterministic job-shop scheduling: Past, present and future. *European Journal of Operational Research*, 113:390–434, 1999.

[20] Dirk C. Mattfeld, Christian Bierwirth, and Herbert Kopfer. A search space analysis of the job shop scheduling problem. *Annals of Operations Research*, 86:441–453, 1999.

[21] R. Monasson, R. Zecchina, S. Kirkpatrick, B. Selman, and L. Troyansky. Determining computational complexity for characteristic 'phase transitions'. *Nature*, 400:133–137, 1998.

[22] E. Nowicki and C. Smutnicki. A fast taboo search algorithm for the job shop problem. *Management Science*, 42(6):797–813, 1996.

[23] R. Lyman Ott. *An Introduction to Statistical Methods and Data Analysis*. Duxbury Press, Belmont, California, 1993.

[24] Andrew J. Parkes. Clustering at the phase transition. In *Proceedings of the Fourteenth National Conference on Artificial Intelligence (AAAI-97)*, pages 340–345, 1997.

[25] Christian M. Riedys and Peter F. Stadler. Combinatorial landscapes. Technical Report 01-03-014, The Santa Fe Institute, 2001.

[26] Josh Singer. *Why solutions can be hard to find: A featural theory of cost for a local search algorithm on random satisfiability*. PhD thesis, University of Edinburgh, 2000.

[27] Josh Singer, Ian P. Gent, and Alan Smaill. Backbone fragility and the local search cost peak. *Journal of Artificial Intelligence Research*, 12:235–270, 2000.

[28] John Slaney and Toby Walsh. Backbones in optimization and approximation. In Bernhard Nebel, editor, *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence (IJCAI-01)*, pages 254–259. Morgan Kaufmann, 2001.

[29] Barbara Smith and Stuart Grant. Sparse constraint graphs and exceptionally hard problems. In Chris Mellish, editor, *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence (IJCAI-95)*, 1995.

[30] R. H. Storer, S. D. Wu, and R. Vaccari. New search spaces for sequencing problems with application to job shop scheduling. *Management Science*, 38:1495–1509, 1992.

[31] Thomas Stützle. *Local Search Algorithms for Combinatorial Problems – Analysis, Improvements, and New Applications*. PhD thesis, Darmstadt University of Technology, 1999.

[32] Eric D. Taillard. Benchmarks for basic scheduling problems. *European Journal of Operational Research*, 64:278–285, 1993.

[33] Eric D. Taillard. Parallel taboo search techniques for the job shop scheduling problem. *ORSA Journal on Computing*, 6(2):108–117, 1994.

[34] R.J.M. Vaessens, E.H.L. Aarts, and J.K. Lenstra. Job shop scheduling by local search. *INFORMS Journal on Computing*, 8(3):302–317, 1996.

[35] P.J.M van Laarhoven, E.H.L. Aarts, and J.K. Lenstra. Job shop scheduling by simulated annealing. *Operations Research*, 40:113–125, 1992.

[36] Jean-Paul Watson, J. Christopher Beck, Adele E. Howe, and L. Darrell Whitley. Toward an understanding of local search cost in job-shop scheduling. In Amadeo Cesta, editor, *Proceedings of the Sixth European Conference on Planning*. Springer-Verlag, 2001.