An Introduction to Fitness Landscape Analysis and Cost Models for Local Search

Jean-Paul Watson

Abstract Despite their empirical effectiveness, our theoretical understanding of metaheuristic algorithms based on local search (and all other paradigms) is very limited, leading to significant problems for both researchers and practitioners. Specifically, the lack of a theory of local search impedes the development of more effective metaheuristic algorithms, prevents practitioners from identifying the metaheuristic most appropriate for a given problem, and permits widespread conjecture and misinformation regarding the benefits and/or drawbacks of particular metaheuristics. Local search metaheuristic performance is closely linked to the structure of the fitness landscape, i.e., the nature of the underlying search space. Consequently, understanding such structure is a first step toward understanding local search behavior, which can ultimately lead to a more general theory of local search. In this paper, we introduce and survey the literature on fitness landscape analysis for local search, placing the research in the context of a broader, critical classification scheme delineating methodologies by their potential to account for local search metaheuristic performance.

1 Introduction

Despite widespread success, very little is known about *why* local search metaheuristics work so well and under what conditions. This situation is largely due to the fact that researchers typically focus on demonstrating, and not analyzing, algorithm performance. Most local search metaheuristics are developed in an ad-hoc manner. A researcher devises a new search strategy or a modification to an existing strategy, typically arrived at via intuition. The algorithm is implemented, and the resulting performance is compared with that of existing algorithms on sets of widely avail-

Jean-Paul Watson
Sandia National Laboratories, Discrete Math and Complex Systems Department, P.O. Box 5800, MS 1318, Albuquerque, New Mexico, USA 87185-1318, e-mail: jwatson@sandia.gov

able benchmark problems. If the new algorithm outperforms existing algorithms, the results are published, advancing the state-of-the-art. Unfortunately, most researchers (including, often, the author) fail to actually prove that the proposed enhancement(s) actually led to the observed performance increase (as typically, multiple new features are introduced simultaneously) or whether the increase was due to fine-tuning of the algorithm or associated parameters, implementation tricks, flaws in the comparative methodology, or some other factor(s). Hooker [10] refers to this approach to algorithm development as the *competitive testing* paradigm, arguing that "this *modus operandi* spawns a host of evils that have become depressingly familiar to the algorithmic research community" and that the "emphasis on competition is fundamentally anti-intellectual" [10], (p. 33). However, although few would argue with Hooker's criticisms, the competitive testing paradigm remains the dominant paradigm for developing new algorithms – independent of whether they are exact or approximate, or based on constructive or local search.

Due largely to the widespread practice of competitive testing, theoretical results concerning the operation of local search metaheuristics are very limited. In particular, we currently lack fundamental models of local search metaheuristic behavior. The importance of behavioral models cannot be understated. Ideally, behavioral models enable practitioners to identify those problems for which a particular local search metaheuristic is likely to be effective and those problem instances that are likely to be more difficult than others. The broad availability of behavioral models would enable researchers to identify fundamental similarities and differences between different local search metaheuristics and identify new research directions in order to improve the performance of existing local search algorithms. In contrast, the current lack of behavioral models has led to several undesirable side-effects, including widespread conjecture and mythology regarding the benefits and/or operation of particular local search metaheuristics.

A foundational concept in the development of behavior models of local search metaheuristics is the notion of a *fitness landscape*, i.e., the topological structure over which search is being executed. Given a specific landscape structure – defined by a search space, objective function, and neighborhood operator, a local search metaheuristic can be viewed as a strategy for navigating this structure in the search for optimal or near-optimal solutions. Given this context, the effectiveness of a particular metaheuristic is a function of how "well tuned" the navigation strategy is to the given landscape. Consequently, knowledge of the fitness landscape structure is key to developing effective metaheuristics, and consequently has been a primary focus in the theoretical analysis of local search metaheuristics. The objective of this chapter is to survey the prior research on fitness landscape analysis for local search metaheuristics, and assess the potential of these efforts to explain one or more aspects of metaheuristic behavior / performance.

To facilitate focus and brevity, our emphasis is on local search metaheuristics, e.g., tabu search, simulated annealing, and variable neighborhood search. These algorithms, which iteratively modify a single solution via repeated perturbations, contrast to both constructive (e.g., GRASP and ant colony optimization) and evolutionary (e.g., genetic algorithms and genetic programming) metaheuristics. However,

fitness landscape analysis does play a role in these paradigms as well. For example, local search is widely used in both GRASP and ant colony optimization to post-process solutions generated by the core constructive procedures. Similarly, fitness landscape analysis has long been central in genetic algorithm theory, although the analysis is significantly complicated by the presence of a population of solutions and multi-parent recombination. Where appropriate, we cite relevant literature in both of these sub-fields. Subsequently, where there is no risk of confusion, we refer to local search metaheuristics simply as metaheuristics.

We begin in Section 2 by formally introducing the concept of a fitness landscape and a local search metaheuristic. Next, we introduce in Section 3 a classification scheme for models of local search metaheuristic behavior, specifically focusing on the objectives of any particular analysis; such objectives are often implicit or unspecified in the literature, and making the objectives explicit allows for an assessment of the ultimate utility of a given methodology. We survey the broad range of landscape structures identified by various researchers in Section 4, discussing their role in and ability to account for metaheuristic performance. Most models of local search metaheuristic performance are implicit, in that they do not propose specific models of metaheuristic run-time dynamics. In Section 5, we survey the limited exceptions – which are exemplars of the types of models that ultimately will inform a rigorous theory of local search. Finally, we conclude in Section 6.

2 Combinatorial Optimization, Local Search, and the Fitness Landscape

We begin by formally defining a combinatorial optimization problem (COP), before introducing the concepts of a fitness landscape and a local search metaheuristic. First, we explicitly differentiate a *problem* (e.g., Boolean Satisfiability) from a *problem instance* (e.g., a 100-variable, 300-clause instance of Random k-SAT with $k = 3$). We denote a combinatorial optimization problem by Π and an instance of Π by Ω. Ω is drawn from some (possibly infinite) universe of possible problem instances, which we denote U_Π.

An instance Ω of a combinatorial optimization problem Π is fully specified by two components: the state space and the objective function. The state space S is a finite, although typically astronomically large, set of possible solutions to Ω. The objective function F assigns a numeric "worth" to each state $s \in S$. The only formal restriction on F is that there must exist a total order of the co-domain, such that a maximal or minimal value is well-defined. Typically, $F : S \rightarrow R^+$ or $F : S \rightarrow Z^+$. The objective function is commonly referred to as a *fitness function*.

Given a COP instance Ω, the ultimate objective is to locate a solution $s \in S$ such that $F(s)$ is optimal, i.e., minimal or maximal. Without loss of generality, we assume the objective is minimization unless otherwise noted.

Local search proceeds via iterative modifications to complete solutions $s \in S$, in contrast to constructive and population-based metaheuristics. We further restrict

our attention to the subset of local search metaheuristics that operate via iterative modifications to a *single* complete solution s, i.e., we ignore more complicated local search metaheuristics that employ elite pools and strategies such as optima linking [30] and path relinking [8]. In doing so, our goal is pragmatic: to discuss the state of local search theory with respect to the simplest (albeit still effective) class of local search metaheuristic before tackling more complex powerful derivatives.

All single-solution local search metaheuristics consist of the following four core components: the state space, the objective function, the move operator, and the navigation strategy. Search begins from an initial solution $s = s_{init}$ that is generated either at random or via some heuristic procedure, and proceeds via a sequence of iterations. The *move operator* specifies the set of allowable modifications (i.e., the neighborhood) to the current solution s at any given iteration, one of which is selected by the *navigation strategy* to serve as the basis for the next iteration. The best solution encountered in any iteration is stored and returned when the algorithm terminates, typically after a time limit is exceeded or maximal number of iterations is completed. We now explore each of these four components in more detail, providing simple examples of each as applied to both the well-known Traveling Salesman Problem (TSP) and Maximum Satisfiability Problem (MAX-SAT).

2.1 The State Space and the Objective Function

Both the state space S and the objective function F are taken directly from Ω, the problem instance under consideration. For example, in an n-city TSP instance, the state space consists of the set of $n!$ permutations, each representing a possible tour; the objective function simply returns the total length of the input tour. In an n-variable, m-clause MAX-SAT instance, the state space consists of the set of 2^n Boolean vectors of length n; the objective function returns the number of the m clauses that are satisfied in a solution $s \in S$. In general, although we do not consider the issue here, the details of how solutions are represented can impact the definition of both the move operator and the navigation strategy. However, this is typically not the case for many well-known combinatorial optimization problems – including the TSP and MAX-SAT.

2.2 The Move Operator

Given a state space S, the notion of locality in a local search algorithm is provided by the move or neighborhood operator N. N defines the set of allowable modifications to the current solution s in any given iteration. In single-solution local search algorithms, $N : S \rightarrow P(S)$, where $P(S)$ denotes the power set of S. More complicated move operators, e.g., those whose domain and/or codomain are cross-products of S and $P(S)$, respectively, are found primarily in evolutionary algorithms and other re-

lated approaches such as optima linking, path relinking, and scatter search; see [14] for a discussion of these and related issues. Given a solution s, the set $N(s)$ is known as the *neighborhood* of s. Similarly, if $s' \in N(s)$, then s' is said to be a *neighbor* of s.

Local search metaheuristics often employ rather simple move operators. For example, the most widely used move operator for MAX-SAT maps each solution $s \in S$ to the subset of n solutions (where n is the total number of Boolean variables) in S that differ from s in the value of a single variable assignment; this is known as the "1-flip" neighborhood. Similarly, most local search metaheuristics for the TSP are based in part on the 2-opt move operator [19], where the neighbors of a solution $s \in S$ are defined as the subset of $\binom{n}{2}$ solutions in S obtained by inverting the subtour between any pair of distinct cities on the tour specified by s. More complex move operators can be obtained via straightforward generalization of these basic operators, e.g., k-flip move operators in MAX-SAT and k-opt move operators in the TSP.

Move operators can vary significantly in their attempts to maintain "logical" locality. Both the 1-flip and 2-opt move operators induce minimal disruptions to the current solution s: 1-flip inverts the value of a single Boolean variable, while 2-opt changes exactly 2 edges. However, in local search algorithms such as iterated local search [20], the differences between neighboring solutions can be much more substantial, e.g., under the generalized k-opt move operator for the TSP [13]. In both cases, however, the perturbation is local in the sense that a neighboring solution is obtained via a *single* application of a move operator. We raise this issue to note that a "local" search metaheuristic may in fact proceed by making drastic modifications to individual solutions.

Mathematically, the move operator N induces a relation on the space $S \times S$, and the properties of this relation can influence the performance of local search. For simplicity, we refer to the relation induced by N simply as N. Although both the 1-flip and 2-opt move operators are symmetric, in that $s' \in N(s) \Rightarrow s \in N(s')$, this is generally not required. Further, N is necessarily transitive and anti-reflexive. Beyond defining the immediate neighborhood, a move operator also defines the connectivity of the search space, i.e., what solutions can be reached via a finite sequence of moves from an initial solution. A move operator N is said to induce a *connected* search space if from an arbitrary solution there always exists a sequence of moves to an optimal solution. N is said to induce a *fully connected* search space if there exists a sequence of moves between any two arbitrary solutions. Both the 1-flip and 2-opt move operators induce fully connected, and consequently connected, search spaces. However, many powerful, problem-specific move operators induce disconnected search spaces, e.g., see [25].

2.3 The Navigation Strategy

The mechanism for selecting some neighbor $s' \in N(s)$ at each iteration of local search is embodied in the navigation strategy, which we denote by Δ. One of the simplest navigation strategies follows the basic principle of gravity: select a neighbor $s' \in N(s)$ with $F(s') < F(s)$. Two well-known variants of this greedy strategy form the core of nearly all navigation strategies. In *next-descent* search, the neighbors $N(s)$ are randomly ordered, and the first neighbor $s' \in N(s)$ such that $F(s') < F(s)$ is selected. In *steepest-descent* search, the neighbor that provides the maximal decrease in the objective function value ($argmin_{s' \in N(s)} F(s')$) is selected, with ties broken randomly.

By iterating greedy descent, local search will eventually arrive at a solution $s \in S$ from which no immediate improvement in the value of the objective function is possible; s is known as a local optimum, such that $\forall s' \in N(s), F(s') \geq F(s)$. Unless s is also a globally optimal solution, the navigation strategy must then guide search to unexplored regions of the search space. When there exists a neighbor $s' \in N(s)$ such that $F(s) = F(s')$, s is actually contained in a plateau, which may or may not be locally optimal; this issue is discussed further in Section 2.4.

Within the local search community, strategies for escaping or avoiding local optima are commonly referred to as *metaheuristics*. Formally, a metaheuristic is a heuristic that dynamically alters the behavior of a core local search heuristic, typically in response to the properties of recently visited solutions. In most local search metaheuristics, the core heuristic is greedy descent; a metaheuristic is activated when the descent strategy becomes trapped in a local optimum, and deactivated once search is successfully directed toward new regions of the search space. Conceptually, metaheuristics and greedy descent are distinct forms of navigation strategies, each operating at a different level of abstraction. However, in practice, the boundary between the two is often fuzzy, for example in simulated annealing. Consequently, metaheuristics are often viewed as atomic entities, such that the distinction between the core heuristic and the metaheuristic is ignored. We present them as such, while at the same time acknowledging any intended distinction between the core and metaheuristic.

Perhaps the most obvious way to escape a local optimum is to generate a new starting solution s_{init} and then re-initiate greedy descent. This process can be iterated until a global optimum is located. The resulting metaheuristic is commonly referred to as *iterated descent*, which is distinct from the next-descent and steepest-descent procedures. In practice, iterated descent is a simple way to improve the performance of a core greedy descent strategy. Further, iterated descent can locate very high-quality solutions for some combinatorial optimization problems, e.g., see Beveridge et al. [2].

Clearly, the probability of iterated descent locating a global optimum approaches 1 as the number of greedy descents N approaches ∞. However, from a practical standpoint, iterated descent is only effective if the fitness distribution of the local optima assumes a certain form, i.e., one in which the left tail of the distribution is non-negligible. For many well-known combinatorial optimization problems, the

fitness distribution of local optima in small problem instances satisfies this requirement. At the same time, it has been empirically demonstrated that such tails typically vanish at larger problem sizes (for example in the TSP), causing iterated descent to perform poorly due to what has come to be known as a "central limit catastrophe" [13].

2.4 The Fitness Landscape

Given a local search metaheuristic A and a combinatorial optimization problem Π, we are interested in determining what makes a particular instance $\Omega \in U_\Pi$ easy or difficult for A. Problem difficulty, or equivalently search cost, is dictated by the interaction of A with the underlying search space. For example, suppose all globally optimal solutions to Ω reside in a small region of the search space containing otherwise poor local optima. If A consistently biases search toward regions of the search space containing generally high-quality local optima, then the cost (on average) of locating optimal solutions to Ω using A is likely to be large. In contrast, if A intensifies search in regions of the search space with poor local optima, then A is more likely to locate optimal solutions to Ω in shorter run-times.

Due to the central role of the search space in determining problem difficulty, much of the research on models of problem difficulty for local search has concentrated identifying structural features of the search space that are likely to influence the cost of local search. Given a local search metaheuristic A, the search space is defined by the combination of (1) the state space S, (2) the move operator N, and (3) the objective function F. Formally, we define the search space $L = (S, N, F)$ as a vertex-weighted directed graph $G = (V, E)$ in which:

1. $V = S$
2. $\forall v \in V$, the weight w_v of v is equal to $F(v)$
3. $E = \{(i, j) | i \neq j \wedge \exists i, j : i \in N(j)\}$

Within the local search community, the graph G is known as a *fitness landscape*, a concept first introduced by the theoretical biologist Sewall Wright in 1932 [44].

We provide two examples of very simple fitness landscapes in Figure 1; in general, landscapes are high-dimensional and extremely difficult to visualize. In both examples, $S = \{1, 2, ..., 20\}$ and $N(x) = \{x-1, x+1\}$, subject to the boundary conditions $N(1) = \{20, 2\}$ and $N(20) = \{19, 1\}$. *Type I* fitness landscapes are characterized by deep, punctuated valleys with abrupt changes in the fitness of neighboring solutions. In contrast, *Type II* fitness landscapes are dominated by plateaus of equally fit neighboring solutions, with discrete jumps in fitness between the plateaus. We differentiate between the two types of fitness landscapes for three reasons. First, different terminology is associated with the two landscape types. Second, Type I and Type II landscapes have different implications for the design of metaheuristic navigation strategies. Third, these two types are representative of the

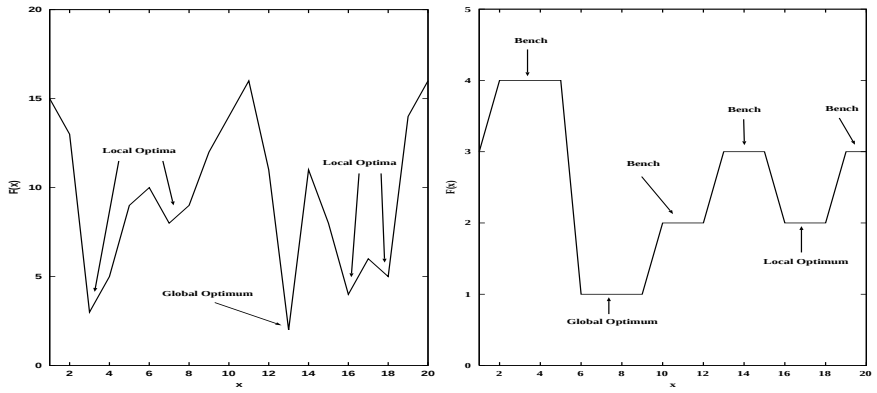

Fig. 1 Examples of Type I (left figure) and Type II (right figure) fitness landscapes.

fitness landscapes found in most *NP*-hard optimization problems. For example, the TSP and MAX-SAT respectively possess Type I and Type II fitness landscapes.

In a Type I fitness landscape, the two key features of interest are local optima and global optima. A *local optimum* is a point $x \in S$ such that $\forall y \in N(x), F(x) \leq F(y)$. In our example Type I landscape, the following vertices are local optima: 3, 7, 13, 16, and 18. A *global optimum* is a point $x \in S$ that is both locally optimal and $\forall y \in S$, $F(x) \leq F(y)$. In our example Type I landscape, vertex 13 is the sole global optimum. The *attractor basin* of a local optimum s consists of all $s' \in S$ such that s results with non-zero probability when a descent-based procedure is applied to s'; as first noted by Reeves [31], attractor basin membership may be stochastic due to the different forms of randomization commonly associated with descent procedures.

Plateaus are the dominant feature of Type II fitness landscapes. Informally, a plateau is simply an interconnected region of the fitness landscape where all points have equal fitness. Formally, a plateau is defined as a set $P \subseteq S$ such that:

1. $\forall x \in P, F(x) = C$ for some constant C
2. For any two points $x, y \in P$ there is a sequence of solutions $\{x, a_1, ..., a_n, y\}$ such that $\forall i$, $a_i \in S$ and $F(x) = F(a_1) = ... = F(a_n) = F(y) = C$
3. (a) $a_1 \in N(x)$, (b) $\forall i \neq n-1, a_{i+1} \in N(a_i)$, and (c) $y \in N(a_n)$

If for some $x \in P$ there exists a $y \in N(x)$ such that $F(y) < C$, the plateau is called a *bench*, and all such solutions y are called *exits*. If there are no exits from a plateau, then the plateau is locally optimal. If the plateau is locally optimal and $\forall x \in S, C \leq F(x)$, then the plateau is also globally optimal. All benches, local optima, and global optima are labeled in our example Type II fitness landscape. There are many additional nuances regarding the terminology of features found in Type II fitness landscapes; an overview is provided by Frank et al. [7].

The qualitative differences between Type I and Type II fitness landscapes have an important impact on the design of navigation strategies and metaheuristics for local search. For example, both next-descent and steepest-descent typically terminate once a solution s is located with no lower-fitness neighbors. The implicit, built-in

assumption is that the local optimum *s* is not a member of a plateau, or that if *s* is a member of a plateau, then the plateau itself is locally optimal. In general, these assumptions do not hold when dealing with Type II fitness landscapes; if greedy descent terminates at a local optimum, it is possible that the optimum resides on a bench, from which an exit may exist. Additionally, the attractor basins in Type II fitness landscapes are often very shallow. For example, Frank et al. [7] have shown that in MAX-SAT, it is often possible to escape a local optimum by accepting a single non-improving move. Consequently, the emphasis on navigation strategies in Type II fitness landscapes is on moving quickly from one plateau to another, either by finding an exit from a bench, or by temporarily accepting non-improving moves. In contrast, in Type I fitness landscapes the emphasis is on escaping local optima with potentially large and deep attractor basins.

3 Landscape Analysis and Cost Models: Goals and Classification

As discussed previously, most research on local search focuses on developing newer, better-performing algorithms. The goal in such research is clearly to *demonstrate* algorithm performance. Paul Cohen notes in his book *Empirical Methods for Artificial Intelligence* ([5], p. 249) that "It is good to demonstrate performance, but it is even better to *explain* [emphasis added] performance." The hard sciences advance via the development of accurate models of the object or objects of interest, models that are both consistent with existing observations and suggest new behavioral hypotheses. Currently, models of local search metaheuristics for *any* combinatorial optimization problem are rare to non-existent.

In developing a model of a given object, we generally concentrate on capturing specific behaviors or small sets of behaviors. In the context of local search metaheuristics, the behavior of interest is generally the cost required to locate an optimal solution (or, more generally, a solution with a given quality threshold) to a problem instance. Due to the stochastic nature of local search (with the sole noted exception of some variants of tabu search), search cost is a random variable with a particular distribution. *Cost models* of local search metaheuristics are behavioral models that capture various aspects of the cost distribution. Most often, we focus on the *average* or typical search cost, as defined by either the distribution mean or median. It is well-known that given a fixed problem size (e.g., 100-city TSPs), the average search cost across instances can vary by many orders of magnitude. One objective in developing cost models is to account for a significant proportion, and ideally all, of this variability. A more aggressive, penultimate objective is to develop cost models that account for the entire distribution of search cost.

In this section, we discuss a general classification scheme for cost models of local search metaheuristics. We consider three different types of cost models, differing in both the type of information upon which they are based and the extent to which they attempt to explicitly capture metaheuristic run-time dynamics. *Static* cost models (Section 3.1) are functions of one or more features of the fitness landscape, and

only implicitly consider metaheuristic dynamics. In contrast, *quasi-dynamic* and *dynamic* cost models (Sections 3.2 and 3.3, respectively) are based on analyses of metaheuristic run-time behavior. Quasi-dynamic cost models are functions of simple summary statistics of metaheuristic behavior. In contrast, dynamic cost models explicitly model low-level metaheuristic behavior using Markov chains.

3.1 Static Cost Models

Static cost models are strictly based on fitness landscape features; metaheuristic dynamics are completely and explicitly ignored. In a static cost model, the independent variables are fitness landscapes features, or combinations thereof, and the dependent variable is the mean or median search cost. To facilitate model evaluation, static cost models are expressed as linear or multiple regression models. Under this formulation, the accuracy of a static cost model can be naturally quantified as the r^2 value of the corresponding regression model, i.e., the proportion of the total variability accounted for by the model. Most static cost model considered to date are based on a single feature of the fitness landscape. For purposes of brevity, we often denote a static cost model based on the feature X as the X static cost model, or simply the X model. Similarly, given the close relationship between static cost models and regression models, we frequently use the two terms interchangeably. Finally, regression methods make certain assumptions (e.g., model errors are homogeneous across the range of the independent variable) in order to generate valid statistical inferences concerning model parameters. These assumptions are generally not satisfied in metaheuristic research. The motivation in using regression models is to (1) quantify overall model accuracy using the associated r^2 value and (2) analyze worst-case deviations from a predicted/expected value. Failure to satisfy regression assumptions does not impact our ability to achieve either of these objectives.

The quality of a static cost model is tied to the model r^2: models with larger r^2 values are more accurate. However, there are limits on the absolute level of accuracy that we can reasonably expect to achieve. As discussed in Section 4, the most accurate static cost models of local search only yield $r^2 \approx 0.5$ in the worst case, which is typically observed for the most difficult sets of problem instances. Although failure to develop more accurate static cost models, despite intense research effort, is not evidence for their impossibility, there does appear to be a practical limit on what can be achieved. Because static cost models ignore metaheuristic dynamics, the existence of models with even $r^2 \approx 0.5$ is in some sense surprising. In expressing fitness landscape features as atomic numeric quantities, there is also the obvious potential for loss of information. Further, there are practical (although not theoretical) limits on the accuracy with which we can measure various quantities, including search cost.

3.2 Quasi-Dynamic Cost Models

A first-order approach to improving static cost models is to incorporate coarse-grained information concerning metaheuristic run-time behavior. For example, we might track simple summary statistics that capture defining characteristics of the set of solutions generated by a metaheuristic. Given such summary statistics, we can then construct regression models relating these summary statistics to search cost, and quantify model accuracy as the resulting r^2. We refer to such cost models as *quasi-dynamic* cost models. The "quasi-dynamic" modifier derives from the fact that the model is based on aggregate statistics relating to run-time behavior, as opposed to an explicit model of metaheuristic run-time dynamics. The sole difference between static and quasi-dynamic cost models is in the nature of the information captured in the independent variable(s).

Most of the issues relating to possible limitations on the accuracy of static cost models equally apply to quasi-dynamic cost models. However, because they account for some aspects of run-time behavior, we would expect in some sense the accuracy of quasi-dynamic cost models to be higher than that of static cost models, although less than the fully dynamic cost models considered below. Empirical evidence supports this observation: some of the most accurate cost models of local search metaheuristics developed to date are quasi-dynamic [35], and achieve a worst-case accuracy of $r^2 \approx 0.65$.

3.3 Dynamic Cost Models

Because they are respectively based on fitness landscape features and summary statistics of run-time behavior, static and quasi-dynamic cost models yield no *direct* insight into the dynamical behavior of local search. To gain insight as to why particular landscape or run-time statistics are highly correlated with search cost, we turn to *dynamic* cost models. Dynamic cost models are high-resolution models (e.g., Markov models) of the run-time behavior of local search metaheuristics. Research on dynamic cost models can be traced to Hoos [11], who used Markov models to posit an explanation for certain run-time behaviors observed for Walk-SAT and other local search algorithms in the Random 3-SAT phase transition region. However, the ability of these models to account for variability in problem difficulty was not considered.

To date, dynamic cost models are represented as Markov chains which coarse-grain the search space in some way. In one common approach, each state of the Markov chain captures the distance i to the nearest *target* (e.g., optimal) solution, in addition to other algorithm-specific attributes. Transitions in the Markov chain correspond to iterations of the local search metaheuristic. A dynamic cost model is constructed by specifying a set of states, and then estimating the various transition probabilities between the states. The details of the estimation process are algorithm-dependent. The search cost predicted by a dynamic cost model is defined as the

mean number of iterations until an absorbing state (i.e., a state with $i = 0$) is encountered. For some Markov chains, analytic formulas for the mean time-to-absorption are easily derived. When analytic formulas are not immediately available, it is pragmatic to resort to simulation of the cost model to estimate mean search cost.

To quantify the accuracy of a dynamic cost model, straightforward linear regression models can be used, in which the predicted and actual search costs serve as the independent and dependent variables, respectively; model accuracy is then quantified by the r^2 value of the linear model. Dynamic cost models differ from their static counterparts in that they explicitly consider the metaheuristic, and move beyond simple numeric characterizations of either fitness landscape features or runtime behavior. Consequently, we *a priori* anticipate higher levels of accuracy than are possible for static and quasi-dynamic cost models. This conjecture is supported in practice; r^2 values in excess of 0.90 are reported in the literature. However, the near-perfect accuracy does not come without costs: dynamic cost models are generally more expensive to construct than static or quasi-dynamic cost models, and are generally far less intuitive.

3.4 Descriptive Versus Predictive Cost Models

For all practical purposes, the cost models we discuss are purely descriptive, in that they provide *a posteriori* explanations for why one problem instance is more difficult than another for a given local search metaheuristic. In principle, cost models could be used to compute a relatively tight confidence interval, via standard regression techniques, for the expected cost required to locate an optimal solution to a new problem instance. However, because the most accurate cost models to date (as discussed below) are functions of the set of *all* optimal solutions to a problem instance, the effort required to generate the prediction actually exceeds that of simply locating an optimal solution. Given an accurate cost model, the problem of runtime prediction is essentially equivalent to the problem of estimating the value of the model parameters. The nature of the cost-accuracy trade-off in model parameter estimation is currently an open research question.

This does *not* imply that cost models are a scientific curiosity, useless in practice. Cost models have been used to make specific predictions regarding the behavior of local search metaheuristics (e.g., see [42]). Further, and perhaps most importantly, cost models can explicitly identify those features of the fitness landscape that are overwhelmingly responsible for problem difficulty in local search. By identifying such features, we are enabling algorithm designers to focus on the areas most likely to yield performance improvements, and to move beyond the ad-hoc, benchmark-driven design methodology that is current employed [10].

4 Fitness Landscape Features and Static Cost Models

The performance of any local search metaheuristic is dictated by the interaction of the metaheuristic with the underlying fitness landscape. Toward understanding this interaction, researchers have initiated numerous investigations of the structural characteristics of the fitness landscapes of various combinatorial optimization problems. As a result, several fitness landscape features have been identified that have been shown, via abstract argument or in concrete inference, to influence problem difficulty for local search. Examples of such features include[1]:

- The number and/or distribution of local optima
- The strength and size of local optima attractor basins
- The size and extension of the search space

Although the importance of these features is widely acknowledged, little or no empirical evidence exists to substantiate the *extent* to which any of these features, or combination thereof, is actually correlated with local search cost. Because the strength of the relationships have not in general been quantified, it is possible or even likely that the prime factor(s) dictating problem difficulty for local search have either yet to be identified or remain largely unexplored.

Structural features of the fitness landscape also have, or at least *should* have, a major influence on the design of metaheuristics. Local search metaheuristics differ largely in their approach to escaping the attractor basins of local optima, and the complexity of the proposed escape mechanisms – in terms of algorithmic details – is highly variable. Ideally, designers tailor a metaheuristic to the class of fitness landscapes that the algorithm is likely to encounter. Yet, very few concrete details are known about attractor basin strength, i.e., the expected computational effort required to escape local optima. This is true for nearly all combinatorial optimization problems. Consequently, it is unclear whether further attention on novel escape mechanisms is warranted, or if researchers should shift their focus to designing more effective high-level search strategies, such as those associated with advanced implementations of tabu search.

While important, the study of factors such as local optima attractor basin strength are not necessarily a driver in overall problem difficulty. In particular, we observe that once the problem of escaping local optima is solved, the broader issue of how to perform effective global search remains open. We now survey those global structural features of the fitness landscape that have been proposed to account for the variability in problem difficulty for local search. We present the motivation behind each feature, summarize prior research, and identify limitations. It should be noted that not all of these features have been investigated *explicitly* in the context of a cost model of local search. However, the objective of correlating the presence of particular features with search difficulty is a common, necessary theme. For illustrative purposes, we additionally provide in some cases graphics illustrating cost model accuracy drawn from the author's own research on job-shop scheduling (JSP). Finally,

[1] Kauffman (p. 44, [16]) provides a more comprehensive list, developed for adaptive local search algorithms.

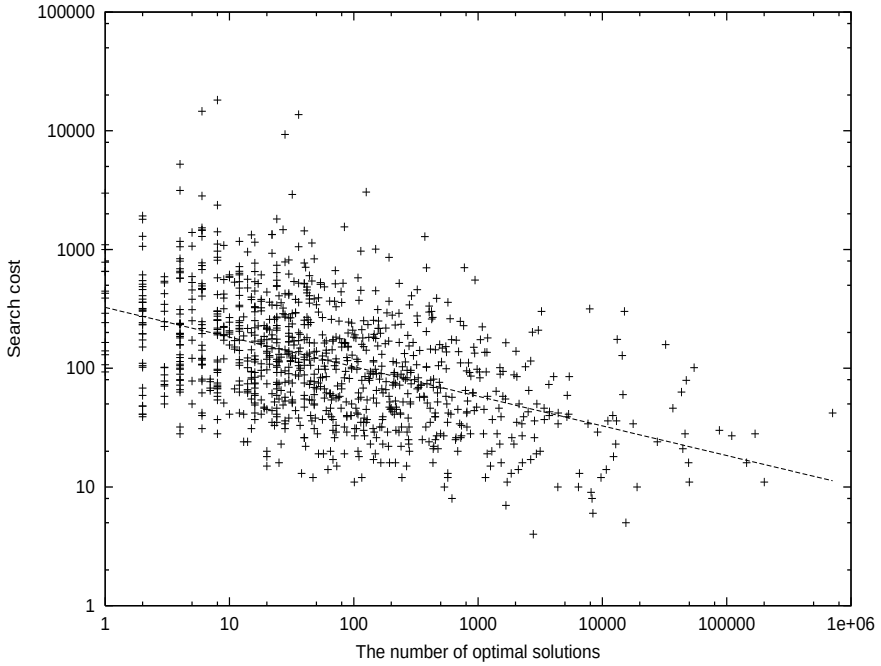

Fig. 2 Scatter-plot of the number of globally optimal solutions versus search cost for 6 job, 6 machine random job-shop problems; the least-squares fit line is super-imposed.

we note that an alternative, complementary perspective on fitness landscape analysis is provided in [12].

4.1 The Number of Optimal Solutions

One of the most intuitive measures of problem difficulty is the number of globally optimal solutions in a fitness landscape. It should be difficult to locate a global optimum if they are relatively rare. Conversely, if global optima are numerous, then it should be relatively easy for local search to find one.

The relationship between the number of globally optimal solutions and problem difficulty for local search was originally analyzed in the context of MAX-SAT and the more general MAX-CSP [4]. The motivation behind this research was to develop an explanation for the easy-hard-easy pattern in problem difficulty observed in the phase transition regions of these problems [17, 28]. It was initially conjectured that the peak in search cost was due to changes in the number of optimal solutions. Yokoo [45] proved that this was not the case, by showing that the mean number of optimal solutions varies in no special way near the phase transition region. In a more refined analysis, Clark et al. demonstrated a relatively strong negative log_{10}-

log_{10} correlation between the number of globally optimal solutions and search cost for three MAX-SAT metaheuristics, with r-values ranging anywhere from -0.77 to -0.91. However, the cost 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. A similar situation is exhibited in the JSP for tabu search.

In Figure 2, we show a scatter-plot of the mean search cost required by tabu search (see [40] for details) to locate an optimal solution to 6 job, 6 machine random JSPs, as a function of the number of optimal solutions to a problem instance. The model is not overly accurate, with $r^2 = 0.2223$, and the graphic clearly demonstrates the very large residuals common to cost models based on the number of optimal solutions.

The distribution of the number of optimal solutions depends in large part on the nature of the objective function, specifically whether all or a fraction of solution attributes dictate solution fitness. For example, the number of optimal solutions to instances of the 2-D integer Euclidean Traveling Salesman Problem is generally very small, and is frequently equal to 1 [36]. The reason is straightforward: tour length is a function of *all* the cities in the instance, and the likelihood of two tours having identical lengths is relatively small given randomly sampled inter-city distances. The likelihood of a single optimal solution is even higher if real-valued city coordinates are allowed. A similar situation is observed in the Permutation Flow-Shop Problem [39]. In contrast, the fitness of solutions in the JSP is dictated by a subset of job orderings, i.e., those on the critical path. Consequently, large plateaus of solutions are common in the JSP [41].

While intuitive, the accuracy of static cost models based on the number of optimal solutions is clearly limited. Further, there is anecdotal evidence that accuracy decreases as larger problem instances are considered.

4.2 The Distance Between Local Optima

The cost of local search is also influenced by the size of the search space. Search in most metaheuristics is heavily biased toward local optima, suggesting that the size of the sub-space of local optima may be strongly correlated with problem difficulty. A straightforward approach to quantifying the size of the local optima sub-space is to simply measure the mean distance between a sample of random local optima; large distances should be indicative of large sub-spaces. This notion of quantifying search space size was first introduced by Mattfeld et al. [21] in the context of the JSP. However, Mattfeld et al. did not investigate the ability of the metric to account for the variability in problem difficulty across a fixed set of instances; rather, they used the metric to account for differences between distinct types of JSP instances.

In Figure 3, we show a scatter-plot of the mean search cost required by tabu search (again see [40] for details) to locate an optimal solution to 6 job, 6 machine random JSPs, as a function of the mean distance between random local optima. Ac-

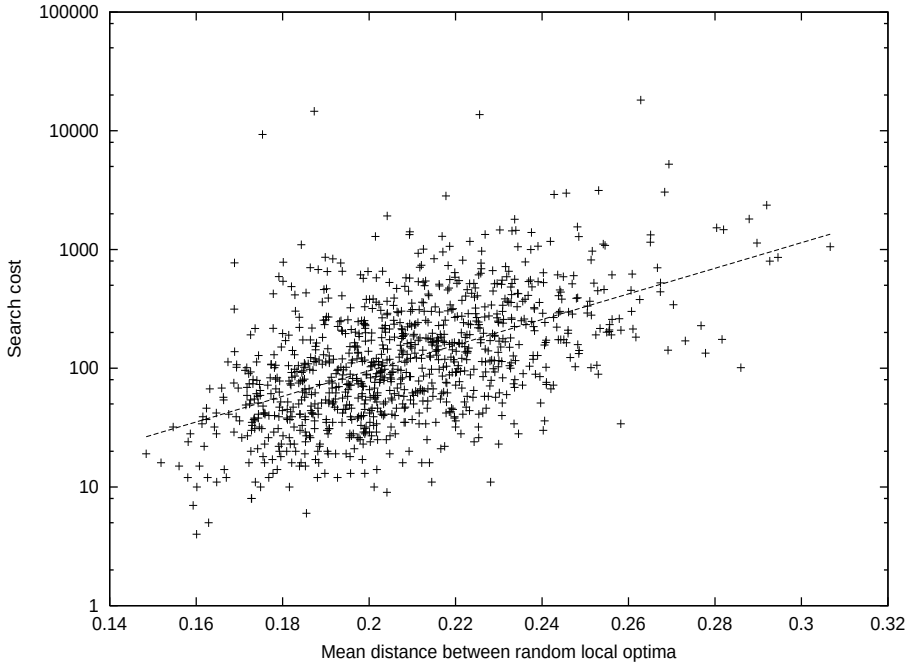

Fig. 3 Scatter-plot of the mean distance between random local optima versus search cost for 6 job, 6 machine random job-shop problems; the least-squares fit line is super-imposed.

curacy is similar to the static cost model based on the number of optimal solutions, with $r^2 = 0.2744$. Similarly, the accuracy of such models tends to decrease with increases in problem size and the graphic exhibits very large residuals, varying over several orders of magnitude.

4.3 The Distance Between Local and Global Optima

The number of globally optimal solutions and the size of the search space S are conceptually independent; it is possible to embed as many as $|S|$ optimal solutions within a search space S. Undoubtedly, both factors influence problem difficulty for local search. If we fix $|S|$ and assume that attractor basin strength and size remain relatively constant, we expect problems to become easier as the number of optimal solutions grows. Analogously, if we fix the number of optimal solutions, it should be more difficult to locate an optimal solution as the number of optimal solutions shrinks. It follows that both the number of optimal solutions and the distance between local optima are, in isolation, unlikely to account for a significant proportion of the variability in problem difficulty for local search.

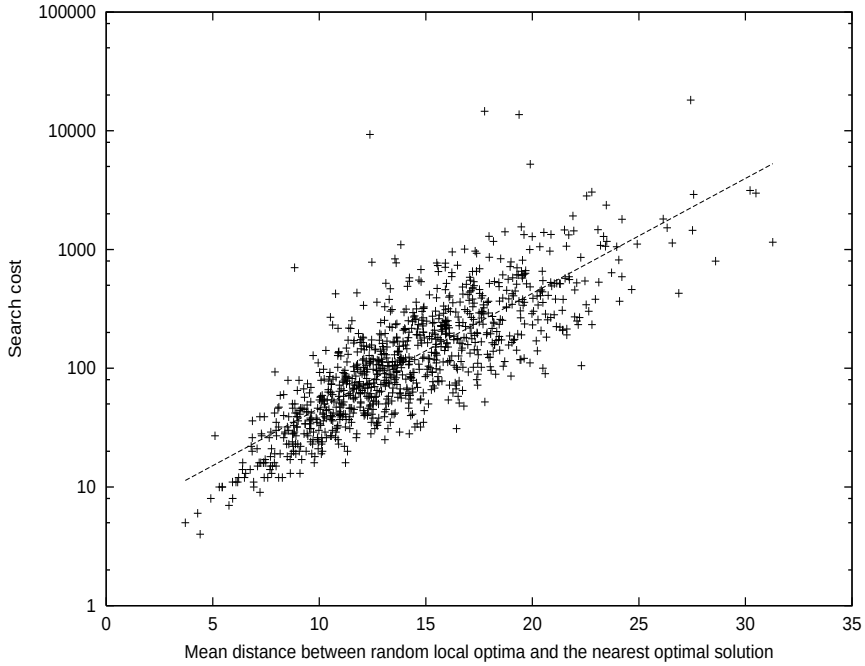

Fig. 4 Scatter-plot of the mean distance between random local optima and the nearest optimal solution versus search cost for 6 job, 6 machine random job-shop problems; the least-squares fit line is super-imposed.

To correct for these flaws, we now discuss a measure that simultaneously accounts for the impact of both features on problem difficulty: the mean distance between random local optima and the *nearest* optimal solution. The intuition is that problem difficulty for local search is proportional to the total distance that must be traversed between an initial solution (e.g., a random local optima) and a target solution (e.g., an optimal solution). This measure was first introduced in the context of MAX-SAT by Singer et al. [35]. Well-known local search algorithms for MAX-SAT rapidly descend from poor-quality initial solutions to near-optimal "quasi-solutions", and subsequent search is restricted to the space of such quasi-solutions. Singer et al. hypothesized that the search cost was proportional to the size of the quasi-solution sub-space, which in turn could be estimated by the mean distance between the first quasi-solution encountered and the nearest optimal solution. Their experimental results demonstrated a very strong ($r \approx 0.95$) correlation between this metric and search cost for easy MAX-SAT instances; for more difficult instances, the accuracy degraded only slightly to $r \approx 0.75$.

In Figure 4, we show a scatter-plot of the mean search cost required by tabu search (again see [40] for details) to locate an optimal solution to 6 job, 6 machine random JSPs, as a function of the mean distance between random local optima and the nearest optimal solution. The r^2 value for the corresponding static cost model is

equal to 0.6424, similar to the accuracy obtained by Singer et al. With few exceptions, residuals vary over roughly 1 to 1.5 orders of magnitude; the improvement is significant relative to static cost models based strictly on the number of optimal solutions or the mean distance between local optima. Clearly, the static cost model based on the measure proposed by Singer et al. is a landmark achievement, representing the first reasonably accurate cost model of any local search metaheuristic, for any combinatorial optimization problem. Previously proposed models achieved accuracy of at most $r^2 \approx 0.3$ in the worst case, in contrast to the $r^2 \approx 0.6$ achieved by Singer et al.

4.4 Fitness-Distance Correlation

Another factor hypothesized to influence problem difficulty for *adaptive* local search algorithms is the correlation between solution fitness and the distance to an optimal solution, often simply denoted as FDC (Fitness-Distance Correlation) [18, 22, 37, 24]. In a problem instance with high FDC, good solutions tend to be tightly clustered or, equivalently, share many solution attributes in common. Consequently, an adaptive search algorithm should be able to exploit these similarities during search. For example, Schneider et al. [33] introduce an adaptive local search algorithm for the Traveling Salesman Problem that, after identifying the edges common to a set of high-quality local optima, restricts subsequent search to the subspace of solutions with *only* those edges. Similarly, Sourlas [37] introduced an adaptive simulated annealing algorithm for the TSP that determines those edges appearing infrequently in high-quality solutions, and prevents subsequent search from generating tours containing those edges. FDC has also been used to account for differences in the relative difficulty of problem instances, e.g., see [15]. As with correlation length, we do not further consider FDC in the context of cost models for local search due to most (basic forms of) local search metaheuristics being non-adaptive in the evolutionary algorithm sense. Further, there is little evidence that FDC can account for a significant proportion of variability in search cost observed for a set of fixed-sized problem instances.

4.5 Solution Backbones

Recently, a number of researchers (e.g., [1] and [36]) have hypothesized that the *backbone* of a problem instance may be correlated with problem difficulty. Informally, the backbone of an instance is the set of solution attributes or variables that possess identical values in *all* optimal solutions; as a consequence, the definition of a backbone depends on the representation scheme used to encode solutions. The intuition behind the backbone measure is that the majority of effort in local search may be spent assigning correct values to backbone variables. Non-backbone vari-

ables appear to be significantly less constrained, enabling search to quickly locate an optimal solution once the backbone is located.

The recent interest in backbones is due in large part to the observation that large-backboned problem instances begin to appear in large quantities near the critical region of the Random 3-SAT phase transition [34] [27] [23] [35]; the coincidence of the two observations immediately leads to the hypothesis that backbone size is correlated with problem difficulty. More recently, Achlioptas et al. [1] argue that the shift from small to large-backboned instances in the phase transition region suggests that the most difficult instances may in fact have a backbone size of 0.5, although this hypothesis has not been verified. Slaney and Walsh [36] analyze the correlation between problem difficulty and backbone size for constructive search algorithms for a number of *NP*-hard optimization problems. For the Traveling Salesman and Number Partitioning Problems, they report a weak-to-moderate correlation (e.g., r between 0.138 and $r = 0.388$) between backbone size and the cost of locating an optimal solution.

4.6 Landscape Correlation Length

A number of researchers have hypothesized that the "ruggedness" of a fitness landscape is likely to be highly correlated with problem difficulty for adaptive search algorithms such as genetic and other evolutionary algorithms [43, 16, 38]. A fitness landscape is said to be rugged if there is a rapid change in the fitness between nearby solutions in the landscape. If the fitness of nearby solutions is uncorrelated, we cannot expect adaptive search to outperform a random walk, i.e., there is no structure to exploit. Ruggedness is frequently quantified as the landscape correlation length, which captures the maximal distance between two arbitrary solutions for which there still exists significant correlation between their fitness values [43]. We do not consider correlation length in the context of static cost models of local search for two reasons. First, most local search metaheuristics are not adaptive, such that correlation length is unlikely to have a major impact on problem difficulty. Second, and more importantly, the extensive research on landscape correlation lengths indicate that for a wide range of well-known *NP*-hard optimization problems, the correlation length is *strictly* a function of problem size [32]. For example, the correlation length in an n-city TSP is given by $n/2$, while in an n-vertex Graph Bi-Partitioning Problem, it is given by $(n-3)/8$ [38]. Perhaps most dramatically, Rana [29] showed that the landscape correlation length is effectively constant over the easy-hard-easy pattern in problem difficulty observed for Random 3-SAT. Consequently, correlation length fails to account for *any* of the variability in problem difficulty observed in sets of fixed-sized problem instances.

4.7 Phase Transitions

Much of research on problem difficulty within the artificial intelligence and computer science communities has focused on the identification of so-called *phase transitions* in problem difficulty [9]. A phase transition in a combinatorial optimization problem identifies an order parameter that partitions the universe of problem instances into subsets with differing degrees of expected difficulty. For example, the clause-to-variable ratio m/n in Random 3-SAT induces a clear pattern: as m/n ranges from 0 to ∞, the degree of problem difficulty exhibits a well-known easy-hard-easy pattern [3]. While successful in identifying inter-partition differences in problem difficulty, phase transitions fail to account for the often considerable variability *within* a partition; the latter can vary over many (e.g., 6 or more) orders of magnitude, even for small problem instances. The failure to explain intra-partition variance in problem difficulty should not, however, be viewed as a deficiency of phase transition models; phase transition research was initially motivated by the desire to generate difficult test problems, and this goal has been achieved.

5 Fitness Landscapes and Run-Time Dynamics

Fitness landscape analysis and the associated static cost models are only a first step toward a more general theory of local search metaheuristics. Rather, the ultimate objective is to develop models linking the fitness landscape structure with models of metaheuristic search dynamics. In contrast to work on fitness landscape analysis, research addressing metaheuristic dynamics is very limited. This is due in part to the difficulty of the modeling, and to the relatively recent emphasis on dynamics modeling. In this section, we survey this research, highlighting the accuracy of the models and the insights they facilitate.

As discussed previously, static cost models only correlate fitness landscape structures with search cost; metaheuristic dynamics are completely ignored. An intermediate between static and dynamic cost models are quasi-dynamic cost models, which are based on summary information concerning metaheuristic dynamics. One example is introduced by Watson et al. [42] in the context of tabu search and the JSP. In Section 4.3, we introduced a related static cost model based on the mean distance between random local optima and the nearest optimal solution. However, Watson et al. observe that random local optima are *not* representative of the solutions visited by tabu search during execution. Instead, they proposed a quasi-dynamic cost model based on the mean distance between solutions actually visited by tabu search and the nearest optimal solution.

In Figure 5, we show a scatter-plot of the mean cost required by tabu search to locate an optimal solution to 6 job, 6 machine random JSPs, as function of the mean distance between solutions visited by tabu search and the nearest optimal solution. The r^2 value for the corresponding quasi-dynamic cost model is 0.7808, representing a 21% increase over the corresponding static cost model; greater improvements

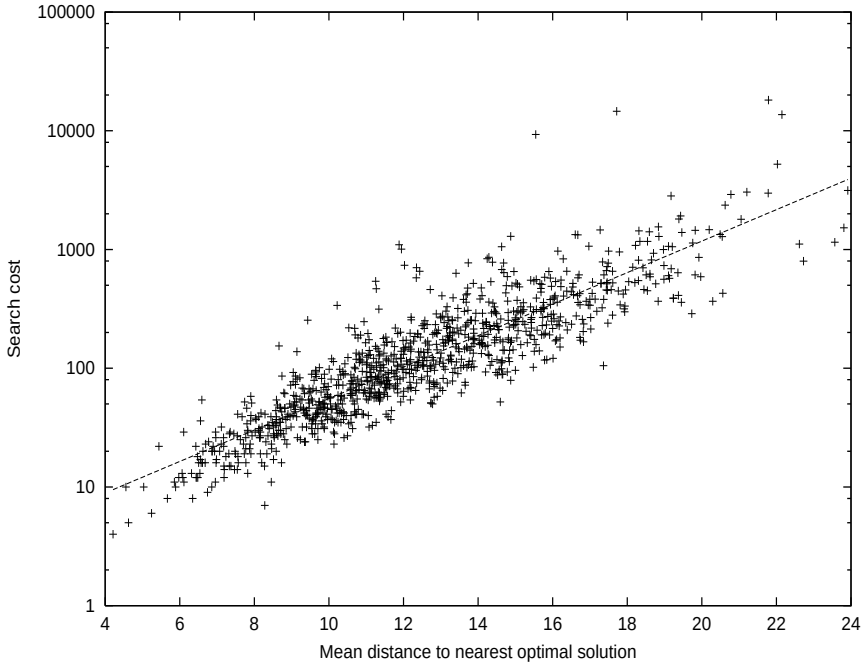

Fig. 5 Scatter-plot of the mean distance between solutions visited by tabu search and the nearest optimal solution versus search cost for 6 job, 6 machine random job-shop problems; the least-squares fit line is super-imposed.

in accuracy were obtained for larger problem instances. With few exceptions, the predicted costs are within a factor of five of the observed costs, representing a marked improvement in accuracy relative to the best available static cost models presented previously.

While quasi-dynamic cost models clearly illustrate the improved accuracy that is facilitated by linking fitness landscape structure and metaheuristics dynamics, they provide only indirect insight into metaheuristic dynamics, which is – we argue – the primary objective in developing any theory of local search metaheuristics. Most metaheuristics are randomized, if implicitly (e.g., through specification of an initial starting solution), such that Markov chains can be used to explicitly model search dynamics. There are two primary challenges in developing such Markov models: (1) aggregating elements of the search space, in order to avoid exponential numbers of states and (2) estimating the state transition probabilities.

To date, the majority of dynamic models of metaheuristics (which, as discussed below, are very limited) follow Pappadimitriou [26] in aggregating states based on their distance to the nearest optimal solution. Depending on the metaheuristic, it may further be necessary to replicate states in order to account for memory mechanisms, e.g., of the form found employed in tabu search [42].

Hoos [11] proposed a Markov model based on distance aggregation in the context of MAX-SAT. The objective of Hoos' analysis was to provide an explanation for specific stagnation behavior in (at the time) state-of-the-art local search metaheuristics for MAX-SAT, in which the run-time cost required to locate optimal solutions for certain instances could not be explained by existing models. Hoos hypothesized that the observed stagnation behavior was caused by the presence of sub-optimal "traps", which caused search to be drawn away from regions of the state space containing optimal solutions, and introduced a branched Markov model to represent the search dynamic. Using posited transition probabilities, Hoos demonstrated that the model accurately captured the search dynamics observed by MAX-SAT local search metaheuristics on this class of problem instance.

Watson et al. [42] introduced Markov models for a tabu search algorithm for the JSP. The contents of short-term memory were abstractly represented as the current "gradient" or change in the distance to the nearest optimal solution observed between the current and previous iteration of the tabu search metaheuristic, and embedded in the Markov state capturing distance to the nearest optimal solution. Transition probabilities were estimated by periodically observing the tabu search algorithm, computing the current search gradient and the distance to the nearest optimal solution; transition probabilities were then estimated using the aggregate sample, for each problem instance.

The resulting Markov models were then simulated to compute the distribution of the number of tabu search iterations required to locate an optimal solution. Comparison of the simulated and empirical results indicated that the proposed Markov model accurately predicts the observed search costs; predicted mean search cost was within a factor of five of the observed value, and the full search cost distribution was reasonably approximated by the Markov model. Although beyond the present scope, Watson et al. also discuss novel observations regarding the linkages between static, quasi-dynamic, and dynamic cost models in the context of tabu search. Further, the model allowed the authors to propose and test certain hypotheses regarding metaheuristic behavior for the JSP, including the lack of benefit due to alternate initialization strategies. For completeness and contrast with corresponding results for static and quasi-dynamic cost models, we shown in Figure 6 the predicted versus actual search costs for 6 job, 6 machine random JSPs. The r^2 for the model is a remarkable 0.96, with predictions in nearly all cases within a factor of two of the observed values.

Most recently, Fournier [6] introduced a Markov model to account for a simple stochastic local search metaheuristic for MAX-SAT. In contrast to Hoos and Watson et al. the search space is aggregated in terms of solution quality. Each discrete value of solution quality (the number of unsatisfied clauses) is represented by a state in the Markov chain. The state transition matrix then specifies the probability of transitioning from a state with quality q to a neighboring state with solution quality q'. The resulting Markov chain is then simulated and compared against experimental results to assess model quality, which in turn provides information concerning the accuracy of the proposed metaheuristic dynamics.

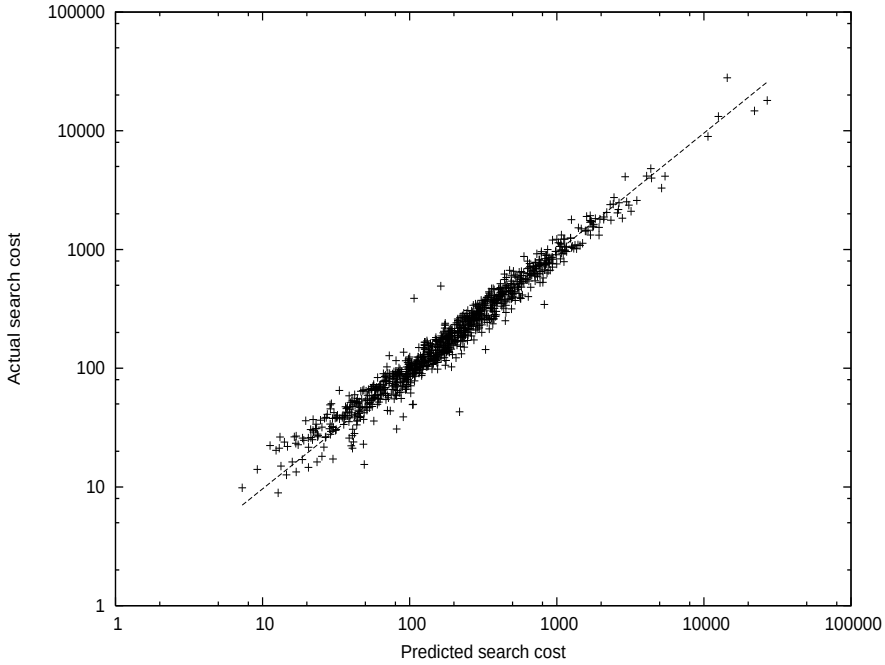

Fig. 6 Scatter-plot of the predicted versus actual search cost for 6 job, 6 machine random job-shop problems, using a Markov model; the least-squares fit line is super-imposed.

Fournier's analysis focuses on a simple metaheuristic for MAX-SAT, called RSAT, which simply selects a neighbor at each iteration with a probability in proportion to the neighbor's quality. This is in contrast to Hoos and Watson et al. who analyze metaheuristics closely related to the state-of-the-art for their respective problems. Further, Fournier's analysis is primarily concerned with average behavior over an ensemble of instances. In particular, the transition matrix is estimated from a large sample of instances, aggregated into a single ensemble estimate. Not unexpectedly, the accuracy of the model on a per-instance basis is limited, although the model (with some minor, noted exceptions) accurately captures metaheuristic dynamics at the ensemble level.

While limited, the research into dynamic models of metaheuristic behavior has lead to impressive advances – relative to simple models based on fitness landscape features – in cost model accuracy. This progression in accuracy is graphically illustrated in Figures 4 through 6. Although far from representing a general theory of local search metaheuristics, such dynamic models do provide the first steps in that general, key direction.

6 Conclusions

Despite the high level of research activity in local search metaheuristics over the last two decades, comparatively little progress has been made in the theoretical foundations of the field. Most research focuses either on the application of existing metaheuristics to new problems or the development of new metaheuristics. Ideas and techniques are routinely re-introduced and re-invented, and it is often difficult to assess the novelty and/or contribution of new research. Recently, the roots of a theory of local search have begun to emerge. The type of model discussed in this paper, we believe, provides a basis for a more general theory of local search. Specifically, we have seen examples of how researchers have used fitness landscape analysis to better understand the mechanisms underlying metaheuristic search and how these mechanisms give rise to various observed behaviors. Model generalization to both other problems and a wider range of metaheuristics is a significant outstanding challenge. Similarly, the implications of these models for metaheuristic design are largely unknown and unexplored. Even with inefficient and ad-hoc development methodologies, researchers have continued to make significant advances in the effectiveness of local search metaheuristics. By developing a generalized theory of local search, it should be possible to more precisely focus future research and, as a consequence, significantly accelerate the rate of advances in the field.

Acknowledgements Sandia is a multipurpose laboratory operated by Sandia Corporation, a Lockheed-Martin Company, for the United States Department of Energy under contract DE-AC04-94AL85000.

References

1. Achlioptas, D., Gomes, C., Kautz, H., Selman, B.: Generating satisfiable problem instances. In: K. Ford (ed.) Proceedings of the Seventeenth National Conference on Artificial Intelligence (AAAI-00), pp. 256–261. AAAI/MIT Press (2000)
2. Beveridge, J., Graves, C., Steinborn, J.: Comparing random starts local search with key feature matching. In: Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence (IJCAI-97) (1997)
3. Cheeseman, P., Kanefsky, B., Taylor, W.: Where the *Really* hard problems are. In: Proceedings of the Twelfth International Joint Conference on Artificial Intelligence (IJCAI-91), pp. 331–337 (1991)
4. Clark, D., Frank, J., Gent, I., MacIntyre, E., Tomov, N., Walsh, T.: Local search and the number of solutions. In: E.C. Freuder (ed.) Proceedings of the Second International Conference on Principles and Practices of Constraint Programming (CP-96), pp. 119–133. Springer-Verlag (1996)
5. Cohen, P.: Empirical Methods for Artificial Intelligence. The MIT Press (1995)
6. Fournier, N.G.: Modelling the dynamics of stochastic local search on k-SAT. Journal of Heuristics **13**, 587–639 (2007)
7. Frank, J., Cheeseman, P., Stutz, J.: When gravity fails: Local search topology. Journal of Artificial Intelligence Research **7**, 249–281 (1997)
8. Glover, F., Laguna, M.: Tabu Search. Kluwer Academic Publishers, Boston, MA (1997)

9. Hogg, T., Huberman, B., Williams, C.: Special issue on frontiers in problem solving: Phase transitions and complexity. Artificial Intelligence **81**(1–2) (1996)
10. Hooker, J.: Testing heuristics: We have it all wrong. Journal of Heuristics **1**, 33–42 (1995)
11. Hoos, H.: A mixture-model for the behaviour of SLS algorithms for SAT. In: Proceedings of the Eighteenth National Conference on Artificial Intelligence (AAAI-02), pp. 661–667. AAAI Press/The MIT Press (2002)
12. Hoos, H., Stüzle, T.: Stochastic Local Search: Foundations and Applications. Morgan Kaufmann (2005)
13. Johnson, D., McGeoch, L.A.: The traveling salesman problem: A case study in local optimization. In: Local Search in Optimization, pp. 215–310. John Wiley and Sons (1997)
14. Jones, T.: Evolutionary algorithms, fitness landscapes, and search. Ph.D. thesis, Department of Computer Science, University of New Mexico (1995)
15. Jones, T., Forrest, S.: Fitness distance correlation as a measure of problem difficulty. In: L. Eschelman (ed.) Proceedings of the Sixth International Conference on Genetic Algorithms, pp. 184–192. Morgan Kaufmann (1995)
16. Kauffman, S.: The Origins of Order. Oxford University Press (1993)
17. Kirkpatrick, S., Selman, B.: Critical behavior in the satisfiability of random boolean expressions. Science **264**, 1297–1301 (1994)
18. Kirkpatrick, S., Toulouse, G.: Configuration space analysis of traveling salesman problems. Journal de Physique **46**, 1277–1292 (1985)
19. Lin, S., Kernighan, B.: An effective heuristic algorithm for the traveling salesman problem. Operations Research **21**, 498–516 (1973)
20. Lourenço, H., Martin, O., Stuützle, T.: Iterated local search. In: F. Glover, G. Kochenberger (eds.) Handbook of Metaheuristics. Kluwer Academic (2003)
21. Mattfeld, D., Bierwirth, C., Kopfer, H.: A search space analysis of the job shop scheduling problem. Annals of Operations Research **86**, 441–453 (199)
22. Mezard, M., Parisi, G.: A replica analysis of the traveling salesman problem. Journal de Physique **47**, 1285–1296 (1986)
23. Monasson, R., Zecchina, R., Kirkpatrick, S., Selman, B., Troyansky, L.: Determining computational complexity for characteristic 'phase transitions'. Nature **400**, 133–137 (1998)
24. Mühlenbein, H., Georges-Schleuter, M., Krämer, O.: Evolution algorithms in combinatorial optimization. Parallel Computing **7**, 65–85 (1988)
25. Nowicki, E., Smutnicki, C.: A fast taboo search algorithm for the job shop problem. Management Science **42**(6), 797–813 (1996)
26. Papadimitriou, C.: On selecting a satisfying truth assignment. In: Proceedings of the 33rd Annual IEEE Symposium on Foundations of Computer Science (FOCS 91) (1991)
27. Parkes, A.: Clustering at the phase transition. In: Proceedings of the Fourteenth National Conference on Artificial Intelligence (AAAI-97), pp. 340–345. AAAI/MIT Press (1997)
28. Prosser, P.: Binary constraint satisfaction problems: Some are harder than others. In: Proceedings of the 11th European Conference on Artificial Intelligence (ECAI-94), pp. 95–99 (1994)
29. Rana, S.: Local optima and genetic algorithms. Ph.D. thesis, Department of Computer Science, Colorado State University (1999)
30. Rana, S., Whitley, L.: Representation, search, and genetic algorithms. In: Proceedings of the 14th National Conference on Artificial Intelligence (AAAI-97). AAAI Press/MIT Press (1997)
31. Reeves, C.: Landscapes, operators and heuristic search. Annals of Operations Research **86**, 473–490 (1998)
32. Riedys, C., Stadler, P.: Combinatorial landscapes. Tech. Rep. 01-03-014, The Santa Fe Institute (2001)
33. Schneider, J., Froschhammer, C., Morgernstern, I., Husslein, T., Singer, J.: Searching for backbones - an efficient parallel algorithms for the traveling salesman problem. Computational Physics Communications **96**, 173–188 (1996)
34. Schrag, R., Crawford, J.M.: Implicates and prime implications in random 3SAT. Artificial Intelligence **88**, 199–222 (1996)

35. Singer, J., Gent, I., Smaill, A.: Backbone fragility and the local search cost peak. Journal of Artificial Intelligence Research **12**, 235–270 (2000)
36. Slaney, J., Walsh, T.: Backbones in optimization and approximation. In: B. Nebel (ed.) Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence (IJCAI-01), pp. 254–259. Morgan Kaufmann (2001)
37. Sourlas, N.: Statistical mechanics and the traveling salesman problem. Europhysics Letters **2**, 919–923 (1986)
38. Stadler, P.: Landscapes and their correlation functions. Journal of Mathematical Chemistry **20**, 1–45 (1996)
39. Stuützle, T.: Personal communication (2001)
40. Watson, J., J.C., Howe, A., Whitley, L.: Problem difficulty for tabu search in job-shop scheduling. Artificial Intelligence **143**(2), 189–217 (2003)
41. Watson, J.P.: Problem difficulty for local search in job-shop scheduling. Ph.D. thesis, Department of Computer Science, Colorado State University (2003)
42. Watson, J.P., Whitley, L.D., Howe, A.E.: Linking search space structure, run-time dynamics, and problem difficulty: A step toward demystifying tabu search. Journal of Artificial Intelligence Research **24**, 221–261 (2005)
43. Weinberger, E.D.: Correlated and uncorrelated fitness landscapes and how to tell the differences. Biological Cybernetics **63**, 325–336 (1989)
44. Wright, S.: The roles of mutation, inbreeding, crossbreeding and selection in evolution. In: D. Jones (ed.) International Proceedings of the Sixth International Congress on Genetics, vol. 1, pp. 356–366 (1932)
45. Yokoo, M.: Why adding more constraints makes a problem easier for hill-climbing algorithms: Analyzing landscapes of CSPs. In: Proceedings of the Third International Conference on the Principles and Practice of Constraint Programming (CP-97), pp. 356–370. Springer-Verlag (1997)