Focusing Search by Using Problem Solving Experience

Eyke Hüllermeier[1]

Abstract. Case-based reasoning (CBR) aims at using experience from the past in order to guide future problem solving rather than "starting from scratch" every time. We propose a CBR strategy particularly suitable for realizing this principle if heuristic search is used as a problem solving method: Given a new problem, a CBR method exploits previously solved problems in order to predict a region of the search space which is (provably) probable to contain the solution. The efficiency of a search method applied afterwards for actually finding the solution is then improved by focusing on this region. Our results provide a formal basis for the intuitively meaningful (even though not always justified) idea to concentrate on those parts of the search space where solutions to similar problems have already been found. The approach outlined in this paper either can be seen as one of CBR-supported heuristic search or as a formal framework of search-oriented CBR.

1 INTRODUCTION

Heuristic search is undoubtedly one of the most important problem solving methods in operations research and artificial intelligence (AI). Unfortunately, the size of search spaces can already be huge for toy problems, let alone problems originating from practice. Even though sophisticated heuristics can greatly reduce the amount of computation, an exponential (average) time complexity of (global) tree-search algorithms is often unavoidable. The range of applications of heuristic search is hence limited, at least if one is interested in finding optimal solutions, possibly under real-time conditions. Still, efficiency can be improved at the cost of solution quality in many cases. Besides, local (iterative improvement) search techniques such as, e.g., SIMULATED ANNEALING can be used when giving up the optimality requirement. Such methods often find near-optimal solutions with reasonable computational effort [13].

A further problem solving method of heuristic nature, namely case-based reasoning (CBR), has recently received considerable attention in AI. The guiding principle underlying case-based problem solving is the "CBR hypothesis" which, loosely speaking, assumes that "similar problems have similar solutions." More precisely, the idea of CBR is to exploit the experience from (attempts at) solving similar problems in the past and to adapt then successful solutions to the current situation [11]. The similarity-guided inference principle of CBR is closely related to methods of instance-based learning [2].

The objective of combining the merits of different methods has given rise to the emergence of hybrid (integrated) approaches in several fields of AI, notably problem solving and machine learning. The idea of integrating CBR and heuristic search presents itself if similar problems have to be solved repeatedly and if heuristic search techniques turn out to be adequate for doing so. Indeed, realizations of this idea can already be found in literature (e.g. [7]). The methods proposed use CBR mainly for guiding the search process, e.g., by choosing search operators based on the success of their application to similar search states. Even if this strategy might be effective for individual problems, we doubt that it can be successful in general. As a major reason let us mention that CBR belongs to the so-called *lazy* learning methods [1]: It simply stores observed cases but defers processing until receiving the request for an information. Answering the request is then accomplished by somehow combining the stored data, a process which requires at least the searching of the case-base. Thus, CBR can learn very efficiently by simply storing observations but causes high computational costs when answering requests. This, however, does generally exclude the application of CBR while search is in progress: Even if it might be able to support single search decisions, a CBR strategy will probably increase the overall time complexity due to the large number of node expansions which have to be made. In fact, alternative (model-based) approaches such as, e.g., probabilistic models [3] seem more suitable for this kind of ("on-line") decision support.

Yet, we suspect that a combination of CBR and heuristic search can be very efficient, provided that both methods are used appropriately. In this paper, we follow the idea of using CBR and heuristic search not simultaneously but in succession. Loosely speaking, CBR makes use of previously solved problems for constraining the solution to a new problem, i.e., for restricting the search space. A search algorithm is then applied for actually solving the problem, i.e., for finding a solution among the most promising candidates. This approach is in line with the idea of solving problems by *transformational* (rather than by *derivational*) analogy [5].

Interestingly enough, case-based problem solving itself can be cast as a search process according to the view of transformational adaptation taken in [4]. Within the related model, (potential) cases correspond to search states and adaptation operators play the role of search operators. The method proposed here complements this model in a reasonable way. In fact, in [4] the authors note that, according to their approach, CBR could principally be realized by completely enumerating the search space. Understandably, they look at this idea with reservation, immediately pointing to the enormous complexity it brings about. Our approach exactly applies to this problem: It supports CBR by focusing search to promising cases.

A method particularly suitable for realizing the CBR-related part of the approach outlined above has been proposed in [10]. Here, we shall put this approach, referred to as case-based inference (CBI), into practice: By predicting the solution to a new problem it will do preparatory work in the context of CBR, at which we look as *repetitive search problems*. Such kind of problems are introduced in Section 2. Section 3 gives a brief review of CBI and provides some extensions and new results. The aforementioned idea of exploiting CBI in order to focus search is discussed in Section 4.

[1] Institut de Recherche en Informatique de Toulouse, Université Paul Sabatier, France, email: eyke@irit.fr

2 REPETITIVE PROBLEM SOLVING

Repetitive problem solving means the process of successively solving problems sharing a common structure. Here, we concentrate on problems which can be cast in the framework of (heuristic) search.

Definition 1 (RSP) *A repetitive search problem* (RSP) *is a (countable) parametrized class $\mathcal{P} = \{p_\theta \mid \theta \in \Theta\}$ of well-defined search problems sharing a common search space $\mathcal{S}$. More precisely, these problems share a related search graph in which nodes s and s' are connected by a directed edge if a search operator can be applied to s and yields s' as a successor. Search states $s \in \mathcal{S}$ provide complete information about respective solutions.*[2] *Each problem $p \in \mathcal{P}$, also called an* instance *of the* RSP*, defines an evaluation function $v_p : \mathcal{S} \to \mathbb{R}$ measuring the quality of solutions, where $v_p(s) = -\infty$ if s is not feasible. The objective is to find a solution of high quality, or even an optimal one. We assume problems to be chosen repeatedly and independently according to some probability measure over $\mathcal{P}$.*

The need for repeatedly solving problems sharing some common characteristics arises quite often in practice. In manufacturing or transportation, for instance, special types of (combinatorial) optimization problems have to be solved very frequently, say, several times a day, or even an hour. For obvious reason, such problems are interesting from the viewpoint of CBR [12]. Besides, it has already been mentioned above that CBR itself might be viewed as an RSP.

As an illustrative example let us consider *resource-based configuration* (RBC), a special approach to knowledge-based configuration [8]. It proceeds from the idea that a (technical) system is assembled of a set of primitive *components.* A resource-based description of components is a special type of property-based description in which each component (e.g. a lamp) is characterized by some set of *resources* or *functionalities* it provides (e.g. light) and some other set of resources it demands (e.g. electric current). The relation between components is modelled in an abstract way as the exchange of resources. A configuration problem consists of minimizing the price of a configuration while satisfying an external demand of functionalities. In its simplest form it corresponds to an integer linear program $A \times x \geq d, c \times x \to min$, where the matrix A specifies the quantities of functionalities offered and demanded by the components, the vector d quantifies the external demand, and the vector c contains the prices of the components. A configuration is identified by the vector x, where the jth entry is the number of occurences of the jth component.

A manufacturer offering client-specific products has to solve configuration problems repeatedly while using the same set of components. Mathematically, this means that different problems share the same *knowledge base* $\langle A, c\rangle$ while the external demand d changes. This gives rise to an RSP which is parametrized by the demand vector d. That is, $\mathcal{P} = \{p_d \mid d \in \mathcal{D}\}$, where p_d corresponds to the RBC problem defined by the triple $\langle A, c, d\rangle$ and $\mathcal{D}$ denotes the set of possible demands. Figure 1 shows the knowledge base of an exemplary configuration problem to which we shall return in Section 4.

Since an RBC problem is equivalent to an integer linear program, one could think of using standard methods from operations research for solving it. However, this equivalence is already lost under slight (but practically relevant) generalizations of the basic model. Realizing a heuristic search in the *configuration space*, i.e., the set $\mathcal{S}$ of possible configurations (identified by integer-valued vectors), seems to be a reasonable alternative which is more robust against extensions of the model. Besides, this approach is better suited for incorporating *experience* from already solved problems.

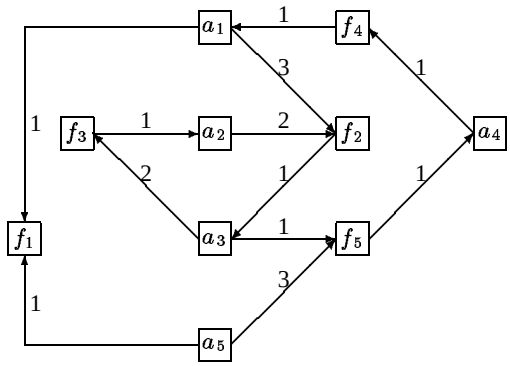

Figure 1. Dependency graph of an RBC problem indicating the offer (directed edge from component a_i to functionality f_j) and demand (directed edge in the reverse direction) of functionalities.

In fact, there are different ways of realizing the idea of *learning* from a set of (optimally) solved problems in connection with heuristic search. One possibility is to learn an appropriate evaluation function for guiding the search process, i.e., for controlling the choice of search operators. In RBC, for instance, (intermediate) configurations x can be chosen on the basis of their cost and an estimation of the cost of satisfying the demand $d - A \times x$ which remains of the original demand d. In this paper, we shall consider a second possibility. As already suggested in Section 1, the idea is not to learn a model for supporting search decisions, but to use the already solved instances by more direct means, namely for obtaining information about promising regions of the search space. It goes without saying that this approach complements rather than excludes other methods of learning in heuristic search. On the contrary, it suggests a combination of instance-based and model-based learning: CBR determines the search region, i.e., *where* to search, and a model supporting individual search decisions determines the strategy, i.e., *how* to search.

The success of an instance-based approach to learning from experience assumes the CBR hypothesis to be somewhat valid, of course. In connection with combinatorial optimization this is definitely not always the case.[3] Still, for many types of problems (or problem variations) the CBR assumption applies rather well, for some of them even provably. For instance, in connection with RBC or, more generally, integer linear programming, it can be shown that the (Euclidean) distance between (optimal) solutions is bounded by some function of the distance between demand vectors [14]. Sensitivity results of such kind can be seen as a formal justification of applying CBR to optimization. Due to their generality, however, corresponding estimations usually turn out to be rather imprecise. The approach discussed in the subsequent section can be considered as an "empirical" counterpart to related theoretical results. As will be seen, we obtain more precise predictions by adapting a model to a *restricted* problem class or even to individual problems of that class.

3 CASE-BASED INFERENCE

Let $\mathcal{P}$ be a (countable) set of problems, and $\mathcal{S}$ a set of solutions. A case is a tuple $\langle p, s\rangle$, where $p \in \mathcal{P}$ and $s = \varphi(p) \in \mathcal{S}$ denotes the

[2] In tree-search algorithms, the solution associated with a node is generally identified by the path from the root to that node.

[3] In fact, there are many problems for which a slight variation of an instance can have a tremendous effect on the optimal solution.

associated (unique) solution. We assume the concept of *similarity*, which lies at the heart of CBR, to be formalized by means of (reflexive and symmetric) similarity measures $\sigma_{\mathcal{P}} : \mathcal{P} \times \mathcal{P} \to [0,1]$ and $\sigma_{\mathcal{S}} : \mathcal{S} \times \mathcal{S} \to [0,1]$ over the set of problems and the set of solutions, respectively. Moreover, we suppose a (finite) memory $\mathcal{M} = (\langle p_1, s_1\rangle, \langle p_2, s_2\rangle, \ldots, \langle p_n, s_n\rangle)$ of $n \geq 1$ cases to be given. $\mathcal{M}^{\downarrow}$ denotes the projection of $\mathcal{M}$ to $\mathcal{P}$, i.e., $\mathcal{M}^{\downarrow} = (p_1, \ldots, p_n)$. The 6-tuple $\Sigma = \langle \mathcal{P}, \mathcal{S}, \varphi, \sigma_{\mathcal{P}}, \sigma_{\mathcal{S}}, \mathcal{M}\rangle$ is called a CBI setup.

Case-based inference [10] proceeds from a precise interpretation of the CBR hypothesis, according to which the similarity of problems imposes a constraint on the similarity of associated solutions in the form of a lower bound. This idea is formalized (below) by means of a *similarity hypothesis*, which is an approximation of the *similarity profile* of a CBI setup. The latter states in a precise way the conclusions which can be drawn from case-based information: Given, the similarity of two (arbitrary) problems, it provides a lower bound to the similarity of the respective soltions.

Definition 2 (similarity profile) *The function h_Σ defined by*

$$h_\Sigma(x) \doteq \inf_{p,p' \in \mathcal{P}, \sigma_{\mathcal{P}}(p,p')=x} \sigma_{\mathcal{S}}(\varphi(p), \varphi(p'))$$

for all $x \in D_{\mathcal{P}} = \{\sigma_{\mathcal{P}}(p,p') \mid p, p' \in \mathcal{P}\}$ is called the similarity profile of the setup Σ.

Definition 3 (similarity hypothesis) *A similarity hypothesis is identified by a function $h : [0,1] \to [0,1]$. The intended meaning of the hypothesis h is that $\sigma_{\mathcal{P}}(p,p') = x \Rightarrow \sigma_{\mathcal{S}}(\varphi(p), \varphi(p')) \geq h(x)$ holds true for all $p, p' \in \mathcal{P}$. A hypothesis h is called* stronger *than a hypothesis h' if $h' \leq h$ and $h \not\leq h'$. A hypothesis h is* admissible *for the* CBI *setup Σ if $h(x) \leq h_\Sigma(x)$ for all $x \in D_{\mathcal{P}}$.*

Consider a CBI setup Σ and a new problem $p_0 \in \mathcal{P}$, and let h be an admissible hypothesis. Moreover, let the *α-neighborhood* of a solution $s \in \mathcal{S}$ be defined as the set of all solutions which are at least α-similar to s: $\mathcal{N}_\alpha(s) \doteq \{s' \in \mathcal{S} \mid \sigma_{\mathcal{S}}(s,s') \geq \alpha\}$.[4] Then, the (set-valued) prediction

$$\widehat{\varphi}_{h,\mathcal{M}}(p_0) \doteq \bigcap_{\langle p,s\rangle \in \mathcal{M}} \mathcal{N}_{h(\sigma_{\mathcal{P}}(p,p_0))}(s), \tag{1}$$

is *correct* in the sense that $s_0 = \varphi(p_0) \in \widehat{\varphi}_{h,\mathcal{M}}(p_0)$, i.e., it covers the true solution.

Of course, knowledge about the similarity profile of a CBI setup will generally be incomplete, which means that we principally cannot guarantee the admissibility of a hypothesis h (except for $h \equiv 0$ which leads to trivial predictions) and, hence, the correctness of the prediction (1). In [9], we have developed an efficient learning algorithm $\mathcal{L}$ which estimates a hypothesis $h_* = \mathcal{L}(\mathcal{M})$ from the cases observed so far. This hypothesis is defined as the strongest hypothesis $h \in \mathcal{H}$ consistent with $\mathcal{M}$, where $\mathcal{H}$ is the class of step functions defined on a fixed partition $\mathcal{A}$ of $[0,1]$:

$$h_*(x) = \min_{p,p' \in \mathcal{M}^{\downarrow} : \sigma_{\mathcal{P}}(p,p') \in \kappa(x)} \sigma_{\mathcal{S}}(\varphi(p), \varphi(p')) \tag{2}$$

with $\kappa : [0,1] \to \mathcal{A}$ being defined by $\kappa(x) = A \Leftrightarrow x \in A$. Interestingly enough, h_* allows for deriving (non-trivial) predictions which are *probably correct*. More precisely, we can prove that the probability of an incorrect prediction is inversely related to the size of the memory $\mathcal{M}$ and, hence, can be made arbitrarily small [9]:

[4] In the context of (local) search, the sets $\mathcal{N}_\alpha(s)$ define a parametrized neighborhood of the search state s.

Theorem 1 *Consider a* CBI *setup with a memory $\mathcal{M}$ of n cases. Moreover, suppose that problems are chosen repeatedly and independently according to some probability measure over $\mathcal{P}$ and let p_0 be a new problem. The following estimation holds true:*

$$\Pr(\varphi(p_0) \notin \widehat{\varphi}_{h_*,\mathcal{M}}(p_0)) \leq 2m/(1+n),$$

where h_* is given by (2) and $m = |\mathcal{A}|$ is the size of the partition of $[0,1]$ underlying h_*. That is, the probability of an incorrect prediction is bounded from above by $2m/(1+n)$.

The use of h_* in (1) leads to the most precise among the predictions which are compatible with the data observed so far. Yet, it is not possible to guarantee a certain degree of precision. In fact, the accuracy of predictions depends on the suitability of the CBI setup under consideration and, hence, is strongly influenced by the choice of the similarity measures $\sigma_{\mathcal{S}}$ and $\sigma_{\mathcal{R}}$. Loosely speaking, precise predictions cannot be expected if the *similarity structure* of the setup is poorly developed, i.e., if the application at hand does hardly satisfy the CBR hypothesis. In this connection, it is noteworthy that the formalization of the similarity structure by means of a similarity profile is rather restrictive. In fact, the enforced global validity of the similarity bounds specified in Definition 2 might prevent from defining tight bounds for those (sub)regions of the instance space $\mathcal{S} \times \mathcal{R}$ where the CBR hypothesis applies well and, hence, from deriving precise predictions. A way of avoiding this effect is to maintain an individual similarity profile for each case in the memory.

Definition 4 (local similarity profile) *Let $p \in \mathcal{P}$. The function $h^p_\Sigma : D_{\mathcal{P}} \to [0,1]$ defined by*

$$h^p_\Sigma(x) \doteq \inf_{p' \in \mathcal{P},\, \sigma_{\mathcal{P}}(p,p')=x} \sigma_{\mathcal{S}}(\varphi(p), \varphi(p'))$$

is called the local similarity profile associated with p, or the p-similarity profile of Σ. We call $\bigwedge_{p \in \mathcal{M}^{\downarrow}} h^p_\Sigma$ an $\mathcal{M}$-similarity profile,[5] *while the collection $h^{\mathcal{M}}_\Sigma = \{h^p_\Sigma \mid p \in \mathcal{M}^{\downarrow}\}$ of local profiles is referred to as the local $\mathcal{M}$-similarity profile.*

A local profile indicates the validity of the CBR hypothesis for *individual* cases. Loosely speaking, it reflects the extent to which a problem p is "typical" or "representative" of similar problems. Since typical cases with strongly developed similarity profiles contribute to precise predictions it seems reasonable to maintain a reduced memory $\mathcal{M}$ of *selected* cases. Thus, let $\mathcal{M}$ be a selection of the sequence $\mathcal{D}$ of cases which have been encountered so far. The above-mentioned learning algorithm can then be modified such that it derives a local hypothesis $h^{\mathcal{M}} = \{h^p \mid p \in \mathcal{M}^{\downarrow}\} = \mathcal{L}(\mathcal{D}, \mathcal{M})$ from $\mathcal{D}$ and $\mathcal{M}$ [9]. Predictions based on $h^{\mathcal{M}}$ are generally more precise than predictions (1). At the same time, however, the associated confidence level is smaller. Still, this level can again be made arbitrarily large by increasing the number of observed cases:

Theorem 2 *Suppose that a sequence $\mathcal{D}$ of n (independent and identically distributed) cases has been observed. For a subset $\mathcal{M}$ containing $|\mathcal{M}|$ cases let $h^{\mathcal{M}}$ be the local $\mathcal{M}$-hypothesis $\mathcal{L}(\mathcal{D}, \mathcal{M})$. Moreover, let $p_0 \in \mathcal{P}$ be a new problem (chosen at random from $\mathcal{P}$). The probability that*

$$\widehat{\varphi}_{h^{\mathcal{M}},\mathcal{M}}(p_0) = \bigcap_{\langle p,s\rangle \in \mathcal{M}} \mathcal{N}_{h^p(\sigma_{\mathcal{P}}(p,p_0))}(s) \tag{3}$$

does not cover the true outcome $s_0 = \varphi(p_0)$ is bounded from above by $|\mathcal{M}|m/(n+1)$.

[5] $f \wedge g$ denotes the function $x \mapsto \min\{f(x), g(x)\}$.

4 FOCUSING SEARCH

The idea of exploiting experience from previously solved problems in RSP can now be realized by combining the two frameworks which have been outlined in Section 2 and Section 3, respectively. For representing an RSP in the form of a CBI setup $\Sigma = \langle \mathcal{P}, \mathcal{S}, \varphi, \sigma_{\mathcal{P}}, \sigma_{\mathcal{S}}, \mathcal{M} \rangle$ we take $\mathcal{P}$ as the class of problem instances, $\mathcal{S}$ as the search space, and $\mathcal{M}$ as the memory of already solved instances.

Since we consider relatively well-structured (optimization) problems it is generally not difficult to define meaningful similarity measures $\sigma_{\mathcal{P}}$ and $\sigma_{\mathcal{S}}$, respectively. One should bear in mind, however, that $\sigma_{\mathcal{S}}$ determines the structure of the neighborhoods $\mathcal{N}_\alpha(s)$ and, hence, has a strong influence on the complexity of computing predictions (1). In RBC, for instance, problems and (search) states correspond to integer-valued vectors. Thus, the neighborhoods $\mathcal{N}_\alpha(s)$ are given in the form of (hyper-)rectangles when defining the similarity between two vectors x, y as a decreasing function of $|x - y|_\infty$. This approach allows for an efficient computation of (1).

In this connection, it should also be noted that (1) remains correct if the intersection is taken over $k < n$ of the problems $p \in \mathcal{M}^\downarrow$. Indeed, deriving a prediction based on k problems (maximally similar to the new problem p_0) might be reasonable if the derivation of (1) is computationally complex. Besides, it is interesting to note that (1) can be approached efficiently by means of parallel computation techniques. In fact, the sets which have to be combined (via intersection) can be derived independently of each other. Moreover, the (associative and commutative) combination itself can be realized in an arbitrary order.

It Section 3, we have assumed a *functional* relation $\varphi : \mathcal{P} \to \mathcal{S}$. For the sake of simplicity we therefore suppose that each problem instance $p \in \mathcal{P}$ has a unique (optimal) solution, namely $\varphi(p)$. Of course, a case $\langle p, s \rangle \in \mathcal{M}$ does not guarantee s to be an optimal solution to p when using a non-admissible (e.g. local) search method. One may then also think of $\varphi(p)$ as some near-optimal solution. In the context of RSP, the CBR hypothesis should hence be understood in the sense that "similar search problems have similar (near-)optimal solutions." Let us mention that the framework in Section 3 can be generalized such that φ is a set-valued function, thereby allowing to take the non-uniqueness of (optimal) solutions into account.

Having represented an RSP in the framework of CBI, a process of repeated problem solving can be sketched as follows:
(a) The process maintains the memory $\mathcal{M}$ and a (local) hypothesis h. Each time a new problem has been solved, the memory and the hypothesis are updated. Note that in general not all cases will be added to $\mathcal{M}$.
(b) Having to solve a new problem p_0, CBI is used for predicting the solution $s_0 = \varphi(p_0)$. To this end, $\mathcal{M}$ and h are called in for deriving a prediction $\widehat{\varphi}_{h,\mathcal{M}}(p_0)$ based on (1) resp. (3).
(c) The prediction covers the true outcome s_0 with high probability (provided that enough problems have as yet been solved). For a heuristic search method which is used in order to find s_0, it is hence advisable to focus on the reduced search space $\mathcal{S}_0 = \widehat{\varphi}_{h,\mathcal{M}}(p_0)$.
(d) Even if unlikely, it can happen that $\mathcal{S}_0$ fails to cover s_0. In this case, it might be necessary to start a second search process which seeks s_0 in $\mathcal{S} \setminus \mathcal{S}_0$.

The above outline leaves several important details open. Notably, it is not clear what is meant by *focusing* the search on $\mathcal{S}_0$. The most obvious idea, of course, is to search $\mathcal{S}_0$ systematically before eventually inspecting $\mathcal{S} \setminus \mathcal{S}_0$. A systematic search of a (complete) state space can even be realized by simple (uninformed) strategies such as breadth-first (BFS) and best-first search (DFS). Iterative deepening search (ID) is particularly interesting in connection with the configuration problem in Section 2, since it is the method of choice if the search space is large and the depth (cost) of the solution is not known. ID combines the merits of both, breadth-first search (optimality, completeness) and depth-first search (linear space complexity). Still, the size of the search tree and, hence, the search effort is generally exponential in the size of the solution. A combination of CBI and ID could hence greatly reduce the search effort: Rather than proceeding from the root of the original search tree and searching the complete state space, one starts at the node which corresponds to the least expensive state in $\mathcal{S}_0$. Moreover, a search path is cut off not only when exceeding the current cost limit, but also when reaching the boundary of $\mathcal{S}_0$. That is, a single ID search phase is not only cost-limited but also "similarity-bounded."

The size of the reduced state space $\mathcal{S}_0$ depends on several factors such as, e.g., the similarity structure of the RSP at hand and the availability of cases similar to the target problem. The diameter of $\mathcal{S}_0$ may also grow with the size of the problem class $\mathcal{P}$ (or even with the size of individual solutions if CBI uses local similarity profiles). However, it will generally do so to a much smaller extent than the original state space. Besides, it should be noted that the search effort itself may grow much stronger than the diameter of the state space (size of the solution), which is partly due to the problem of generating repeated states. In fact, there are many optimization problems (of simple structure) for which it can be proved that the relation between the expected search effort of ID and that of its combination with CBI is of exponential order in the size of the solution. Needless to say, there are as well applications for which a similarity-based prediction hardly improves efficiency. Besides, it deserves mentioning that an optimal search method (such as BFS) becomes *probably optimal* in combination with CBI, in the sense that $\mathcal{S}_0$ is not guaranteed to cover the (overall) optimal solution. Thus, it might become necessary to go beyond $\mathcal{S}_0$ in order to prove (with probability 1) that the best solution in $\mathcal{S}_0$ is also globally optimal.

Of course, the modification of simple (uninformed) search strategies as outlined above is only one example of using the information about the probable location of s_0. Combining CBI with other search methods might call for a less straightforward adaptation of an underlying search strategy. Local (iterative improvement) search methods seem to be particularly qualified for such an adaptation. For instance, the performance of such algorithms crucially depends on the state from which search starts. Thus, $\mathcal{S}_0$ might simply be used for defining a good initial state, or several initial states in algorithms such as, e.g., random-restart hill-climbing. Likewise, an initial collection of search states might be distributed over $\mathcal{S}_0$ when using genetic algorithms or parallel (local) search. Quite natural ways of integration exist in the case of TABU SEARCH [6]. In fact, this heuristic provides explicit control structures which support the focusing of the search process on promising (or still unexplored) regions of the search space.

Experimental results supplying evidence for the effectiveness of CBI, which are not presented in this paper due to space limitations, can be found in [9]. Still, let us reconsider the simple configuration problem in Section 2 in order to convey a first idea of how CBI performs. The corresponding knowledge base is specified by the dependency graph in Figure 1 and the price vector $c = (2, 1, 3, 1, 6)$.

For the problem class defined by the set $\mathcal{D} = \{d \in \mathbb{N}_0^5 \mid |d|_\infty \leq 6\}$ of external demands we have carried out the following experiment: A (rather small) subset of $k = 20$ problems is chosen at random and solved optimally. For the resulting memory $\mathcal{M}$, the $\mathcal{M}$-similarity profile as well as the local $\mathcal{M}$-profile are estimated. To

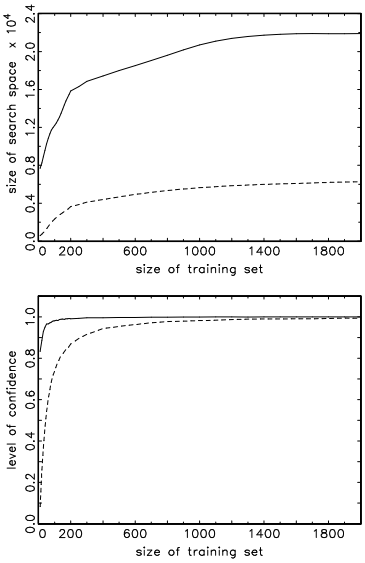

Figure 2. Expected precision (above) and correctness (below) of predictions for configuration problems when using the $\mathcal{M}$-similarity profile and the local $\mathcal{M}$-profile (dashed line), respectively.

this end, the learning method discussed in Section 3 is applied to n (randomly chosen) training examples. The average precision $\rho_{\mathcal{M}}$ (number of search states) and correctness $\varepsilon_{\mathcal{M}}$ (probability of correct prediction) of (1) resp. (3) are derived afterwards:

$$\rho_{\mathcal{M}} = \frac{1}{|\mathcal{D}|} \sum_{d \in \mathcal{D}} |\widehat{\varphi}_{h,\mathcal{M}}(d)|, \quad \varepsilon_{\mathcal{M}} = \frac{1}{|\mathcal{D}|} \sum_{d \in \mathcal{D}} 1_{\widehat{\varphi}_{h,\mathcal{M}}(d)}(\varphi(d)),$$

where $|A|$ denotes the cardinality of the set A. The expected values $\rho = \mathsf{E}_{\mathcal{M}}(\rho_{\mathcal{M}})$ and $\varepsilon = \mathsf{E}_{\mathcal{M}}(\varepsilon_{\mathcal{M}})$ have been approximated by repeating this experiment a large number of times and taking the respective average of the results obtained. Figure 2 shows these values as a function of the number n of training examples. As can be seen, the use of local hypotheses leads to more precise predictions at the cost of a lower level of confidence. Anyway, this level converges toward 1 in both cases, thus confirming our theoretical results of Section 3.

5 CONCLUDING REMARKS

We have proposed to combine heuristic search and case-based reasoning for improving efficiency in repetitive problem solving: CBR brings a promising subset of search states into focus, thereby providing important information to a search method which is applied for actually finding a solution. From the perspective of CBR, our method should not merely be seen as an application. In conjunction with [4], it contributes in a more general way to a formal framework of CBR in which (transformational) adaptation is realized as a search process and (case-based) experience is used in order to concentrate on promising regions of the related search space.

Our method of learning from experience can be realized in a computationally efficient way. Even though it is principally instance-based, it also contains a model-based component: A similarity hypothesis is used for quantifying the (minimal) similarity of solutions, given the similarity of associated problems. In fact, this quantification can be seen as the basic prerequisite for *combining* the information provided by *several* cases (via intersection) and, hence, for deriving precise predictions. The CBR-related part of our approach is actually less heuristic than CBR in general, since it guarantees a certain correctness of such predictions. A successful application of case-based inference providing precise predictions still assumes a problem domain which is somehow in accordance with the CBR hypothesis. Fortunately, *local* models (hypotheses) are less demanding and only assume the existence of individual problem instances which are representative of similar instances.

In [4], the concept of similarity is integrated into problem solving by means of a, say, "ideal" similarity measure. By pointing to optimal initial search states, this measure somehow guarantees the retrieval of cases which can be adapted in an optimal way. Needless to say, that finding such measures will be difficult in practice, if possible at all. Here, we take a different (more pragmatic) approach: It is quite possible that a similarity measure has been learned or is successively adapted to the current application, but our prediction method takes it as a *given* input. It then derives a *set* of *promising* search states rather than *the optimal* initial state, and the precision of this prediction depends on how "ideal" the similarity measure actually is. In fact, this measure dictates the quality of predictions obtained. Loosely speaking, it is not the solutions which define the concept of similarity; rather it is the similarity measures that define the (quality of) solutions.

In this paper, the combination of case-based inference and basic search strategies such as iterative deepening has been outlined. Besides, we have briefly touched on the integration with other algorithms, particularly local search methods. A more thorough investigation and the realization of concrete (case-based) search algorithms is an important topic of ongoing work. Besides, we currently address further aspects which have an essential impact on the success of the problem solving approach outlined in Section 4. This concerns, e.g., a strategy for maintaining an optimal memory of cases and the question of how to define, or even learn, suitable similarity measures.

REFERENCES

[1] *Lazy Learning*, ed., D.W. Aha, Kluwer Academic Publ., 1997.
[2] D.W. Aha, D. Kibler, and M.K. Albert, 'Instance-based learning algorithms', *Machine Learning*, **6**(1), 37–66, (1991).
[3] S. Baluja and S. Davis, 'Fast probabilistic modeling for combinatorial optimization', in *Proceedings* AAAI-98, pp. 469–476, (1998).
[4] R. Bergmann and W. Wilke, 'Towards a new formal model of transformational adaptation in case-based reasoning', in *Proc.* ECAI-98, pp. 53–57, (1998).
[5] J.G. Carbonell, 'Derivational analogy: A theory of reconstructive problem solving and expertise acquisition', in *Machine Learning: An Artificial Intelligence Approach (Volume II)*, eds., R.S. Michalski, J.G. Carbonell, and T.M. Michell, 371–392, Morgan Kaufmann, (1986).
[6] F. Glover and M. Laguna, *Tabu Search*, Kluwer Academic Publ., 1997.
[7] S. Grolimund and J.G. Ganascia, 'Driving tabu search with case-based reasoning', *Eur. J. of Operational Research*, **103**(2), 326–338, (1997).
[8] M. Heinrich, 'Ressourcenorientiertes Konfigurieren', *Künstliche Intelligenz*, **1/93**, 11–14, (1993).
[9] E. Hüllermeier, 'Similarity-based inference as constraint-based reasoning: Learning similarity hypotheses', Technical Report 64, Department of Economics, University of Paderborn, (September 1999).
[10] E. Hüllermeier, 'Toward a probabilistic formalization of case-based inference', in *Proc.* IJCAI-99, pp. 248–253, (1999).
[11] J.L. Kolodner, *Case-based Reasoning*, Morgan Kaufmann, 1993.
[12] D.R. Kraay and P.T. Harker, 'Case-based reasoning for repetitive combinatorial optimization problems, part I: Framework', *Journal of Heuristics*, **2**, 55–85, (1996).
[13] *Modern Heuristic Search Methods*, eds., V.J. Rayward-Smith, I.H. Osman, C.R. Reeves, and G.D. Smith, John Wiley and Sons, New York, 1996.
[14] A. Schrijver, *Theory of Linear and Integer Programming*, Wiley and Sons Ltd., 1986.