ECAI 2010
H. Coelho et al. (Eds.)
IOS Press, 2010

doi:10.3233/978-1-60750-606-5-733

Contract Search : Heuristic Search under Node Expansion Constraints

Sandip Aine [1] and **P. P. Chakrabarti** and **Rajeev Kumar** [2]

Abstract. In this work, we present a heuristic search technique (Contract Search) which can be automatically adapted for a specified node expansion limitation. We analyze the node expansion properties of best first search and propose a probabilistic model (rank profile) to characterize heuristic search under restricted expansions. We identify the basic properties of the rank profile and establish its relation with the search space configuration. In Contract Search, we use the rank profile model to formulate an optimal strategy to choose level dependent restriction bounds maximizing the probability of obtaining the goal node under the specified contract. Experimental comparison with anytime search techniques like ARA* and beam search shows that Contract Search outperforms these techniques over a range of constraint specifications.

1 Introduction

The design of efficient anytime algorithms [4], i.e., algorithms which can work under any *time* limitations, has become a necessity for modern practitioners facing large sized problems that are NP-hard in nature. Anytime algorithms are further classified into two categories, namely interruptible algorithms and contract algorithms. Interruptible algorithms are expected to regularly produce solutions of improved quality and when suddenly terminated, return the best solution produced so far as the result. For a contract algorithm, time is allocated *a priori* and the algorithm is required to provide a *high-quality* solution at the end of the contract.

The development of an effective heuristic search algorithm that works under a node expansion contract requires attention both theoretically as well as from an algorithmic point of view. While most interruptible anytime algorithms can be used for contract purposes, it is important to consider using the time effectively to produce a good solution as compared to several inferior ones. In general, the interruptible heuristic search algorithms follow two basic principles. They use depth guided non-admissible pruning to discover solutions quickly, and work in multiple iterations with gradual relaxation of the constraints. In the past few years, a number of interruptible heuristic search algorithms have been developed. These algorithms can be broadly classified into two categories, weighted A* [6, 8] approaches and structural restriction based approaches, such as beam search algorithms [2, 3, 10], AWA* [1], etc.

One of the major disadvantages of using the interruptible techniques for time constrained purposes is that the contract information can only be used to control the termination, whereas the algorithm's execution remains independent of the constraint specification. Also, in most of the cases the interruptible techniques use non-admissible pruning. As the basic characteristics of A* change with non-admissible heuristics [7], it becomes very difficult to establish a relation between node expansions and the probability of convergence. The beam search technique [2, 9, 10] seems to be a promising choice for contract search purposes if the depth of the goal node is known. However, in spite of encouraging experimental results (with beam search), a strong reason for using a constant beam width at all levels has been lacking. Another approach was proposed [3] where a variant of best first beam search was used with dynamic adjustment of beam width (guided by solution cost a bound obtained from depth first search), but the relation between the chosen beam width, the node (or time) contract and the probability of obtaining an optimal cost solution remains unexplored.

This work is based on the observation that for each level of a search space if we expand only up to the *optimal-path node* (a predecessor of the optimal-cost goal node) we can attain optimality without expanding all nodes having cost less than (or equal to) the optimal cost. We introduce the concept of the *rank* of a node at a particular level, and analytically show how the rank based restriction may be applied to obtain optimal (or near-optimal) solutions with fewer node expansions compared to A*. We develop an algorithm (Contract Search) that considers local competition amongst nodes at the same level and explores only a limited number of nodes at each level depending on the contract following a policy similar to beam search. However, we do not use a constant beam width like beam search. In our approach, the local node expansion limits are optimally selected based on the probability of the optimal cost path lying within this bound. For this, we develop the concept of probabilistic rank profiles. We analyze the characteristics of the profile function and present its properties with respect to the search space structure and heuristic accuracy. We also present an approximation model to estimate the profile function. Using this rank profile, we present an optimal selection strategy that computes the node limits $k(l)$ at every level given an overall contract C such that the probability of obtaining the optimal solution is maximized. We perform experiments on various search problems and compare Contract Search with two anytime algorithms, namely ARA* and beam search, obtaining very encouraging results.

2 Rank of the Optimal-Path Node versus Nodes Expanded by A*

In this section, we formally define the concept of the *rank* of an *optimal-path node* and establish the relation between A* node expansions and the rank limited node expansions. We examine the extra amount of expansions performed by A* even after it has obtained the

[1] Mentor Graphics Pvt. Ltd., USA, email: sandip_aine@mentor.com
[2] Department of Computer Science & Engineering, Indian Institute of Technology Kharagpur, India, email: ppchak,rkumar@cse.iitkgp.ernet.in

'optimal-path' node at a level. For our analysis, we assume that our search space is a tree with a unique optimal cost solution. We also assume that the heuristic used is admissible and consistent [7]. We consider two sets of nodes for each level of the search tree, defined as follows,

Definition 1 *$V(l)$: For a given level l, $0 \leq l \leq h$ (h height of the tree), $V(l)$ denotes the set of nodes belonging to that level that are surely expanded by A*, thus,*

$$V(l) = \begin{cases} \{s\} \quad \textit{if } (l = 0) \\ \{n | Parent(n) \in V(l-1) \textit{ and } f(n) < f^*\} \\ \cup \{n_{g,l}\} \ \textit{otherwise} \end{cases} \tag{1}$$

where f^* is the cost of an optimal solution, s is the start node and $n_{g,l}$ is the 'optimal-path' node at level l.

Definition 2 *$W(l)$: For a given level l, $0 \leq l \leq h$ (h height of the tree), $W(l)$ denotes the set of nodes that have f-value less than the 'optimal-path' node at that level (l) and whose parents belong to $W(l-1)$, plus the 'optimal-path' node at that level, i.e.,*

$$W(l) = \begin{cases} \{s\} \quad \textit{if } (l = 0) \\ \{n | Parent(n) \in W(l-1) \textit{ and} \\ f(n) < f(n_{g,l})\} \cup \{n_{g,l}\} \ \textit{otherwise} \end{cases} \tag{2}$$

where $n_{g,l}$ is the 'optimal-path' node at level l.

From the definitions, we observe that for each level A* expands at least $V(l)$ nodes. On the other hand, $W(l)$ is the set of nodes which must be expanded at each level to reach the optimal cost goal through a best-first mechanism. Next, we define two rank terminologies related to the cardinality of these two sets.

Definition 3 *$max\_rank\_A^*(l)$: It is the maximum rank (relative position in terms of $f(n)$ value) of a node at level l (among nodes in the same level), which is surely expanded by A*, i.e.,*

$$max\_rank\_A^*(l) = |V(l)| \tag{3}$$

Definition 4 *$opt\_node\_rank(l)$: It is the relative position of the 'optimal-path' node at a level l among the nodes in the open list that are also at level l (when for all levels $l' < l$, only the nodes in $W(l')$ have been expanded), i.e.,*

$$opt\_node\_rank(l) = |W(l)| \tag{4}$$

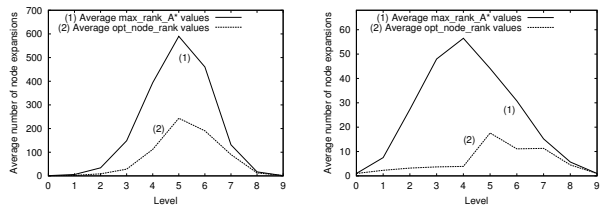
From the definitions of $max\_rank\_A^*(l)$ and $opt\_node\_rank(l)$, we deduce the following relation.

Theorem 1

$$\forall l, 0 \leq l \leq h \quad opt\_node\_rank(l) \leq max\_rank\_A^*(l). \tag{5}$$

Theorem 1 shows that we can attain optimality by expanding fewer nodes than A*. Next, we present some experimental and analytical observations which highlight the gap between the $max\_rank\_A^*(l)$ and $opt\_node\_rank(l)$ values.

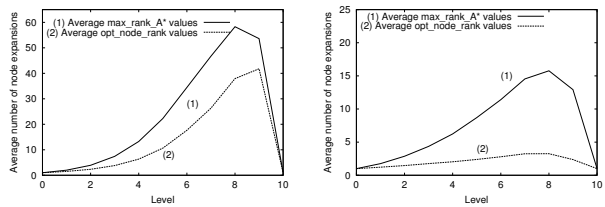
Experimental Observations We perform experiments on two optimization problems, Euclidean TSP and 0/1 Knapsack problem. For TSP, we use two heuristics, minimum remaining edge (h_1) and minimum spanning tree (h_2). For Knapsack, we use two overestimating heuristics, best cost density (h_1) and fitting based (h_2).



(a) Using the minimum remaining edge heuristic (h1)

(b) Using the minimum spanning tree (MST) heuristic (h2)

Figure 1. Comparison of average $max\_rank\_A^*(l)$ and $opt\_node\_rank(l)$ values at different levels (l) of a search tree for Euclidean Traveling Salesman Problem



(a) Using the best density heuristic (h1)

(b) Using the fitting based heuristic (h2)

Figure 2. Comparison of average $max\_rank\_A^*(l)$ and $opt\_node\_rank(l)$ values at different levels (l) of a search tree for 0/1 Knapsack problem

In both the cases, h_2 dominates h_1. Comparisons of the average $max\_rank\_A^*(l)$ and $opt\_node\_rank(l)$ values (for each level), for h_1 and h_2, are shown in Figure 1 (10-city TSP) and 2 (10 object 0/1 Knapsack), respectively. These results are obtained for a set of $10,000$ randomly generated problems.

From the above experiments, we observe that for most levels, there is a significant difference between $max\_rank\_A^*(l)$ and the $opt\_node\_rank(l)$ value. Also, we observe that the stronger the heuristic estimation, the larger is the gap between nodes expanded by A* and *opt_node_rank* at a given level. Next, we try to explain the observed behavior in terms of a search tree model.

Analytical Observations We study the difference between the expected *max_rank* and *opt_node_rank* values with respect to heuristic errors, considering a uniform cost search tree model (Chapter 8 of [7]). Our search space is modeled as a uniform m-ary tree T, with a unique start state S and a unique goal state G, situated at a distance N from S (Figure 3). The solution path is given as $(S \rightarrow n_{g,1} .. \rightarrow n_{g,i} \rightarrow ..n_{g,N-1} \rightarrow G)$ where $n_{g,i}$ denotes the *optimal-path* node at level i. The trees $T_1..T_i..T_N$ are sub-trees of T, one level removed from the optimal path. Each 'off-course' sub-tree T_i is rooted at a direct successor of $n_{g,i-1}$ which is off the solution path (there are $m-1$ such T_i for each i). An 'off-course' node is labeled as $n_{i,j}$, where j denotes the level of that node and i denotes the root level of the 'off-course' sub-tree (T_i) to which the node belongs. Using the search tree model, we compare the expected node

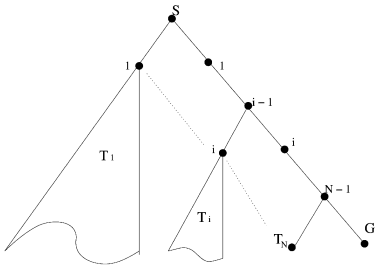


Figure 3. Uniform binary tree (model)

expansions ($max\_rank\_A^*(l)$ versus $opt\_node\_rank(l)$) for each level of the tree. First, we present some definitions.

Definition 5 *$q_{i,j}$: $q_{i,j}$ denotes the probability of node $n_{i,j}$ having f-value less than the optimal cost, i.e.,*

$$q_{i,j} \;=\; P(f(n_{i,j}) < N) \tag{6}$$

Definition 6 *$r_{i,j}$: $r_{i,j}$ denotes the probability of $f(n_{i,j})$ being less than the estimated cost (f-value) of the 'optimal-path' node at that level, i.e.,*

$$r_{i,j} = P(f(n_{i,j}) < f(n_{g,j})) \tag{7}$$

Definition 7 *Expected max_rank_A* : Expected max_rank_A* ($EMR(l)$) denotes the expected value of max_rank_A* value at a given level l of a search tree.*

Definition 8 *Expected opt_node_rank : Expected opt_node_rank ($EOR(l)$) denotes the expected value of opt_node_rank at a given level l of a search tree.*

Next, we use the tree model to compute the expected node expansions.

Lemma 1 *The EMR and EOR values at a given level l of a m-ary uniform cost search tree T (with goal depth = N) are given as,*

$$\begin{aligned} EMR(l) \;&=\; 1 + (m-1)\textstyle\sum_{i=1}^{i=l} m^{l-i} * \Pi_{j=i}^{j=l} q_{i,j} \\ EOR(l) \;&=\; 1 + (m-1)\textstyle\sum_{i=1}^{i=l} m^{l-i} * \Pi_{j=i}^{j=l} r_{i,j} \end{aligned} \tag{8}$$

We now estimate the expansion probability values $(q_{i,j}, r_{i,j})$ from the heuristic information available. We assume that the relative estimation errors, are independent random variables $(e_1 \leq Y(n) \leq e_2)$ with an arbitrary distribution function.

$$Y(n) = [h^*(n) - h(n)]/h^*(n) \tag{9}$$

Lemma 2 *With the chosen error model,*

$$\begin{aligned} q_{i,j} = 1- \quad & P(Y(n) \leq 2(j+1-i)/(N+j+2(1-i))) \\ r_{i,j} = 1- \quad & P(Y(n) - Y(g) * (N-j)/(N+j+2(1-i)) \\ & \leq 2(1+j-i)/(N+j+2(1-i))) \end{aligned} \tag{10}$$

Where $Y(n)$ and $Y(g)$ are two identically distributed random variables within (e_1, e_2) bound.

Theorem 2

$$\forall l, \quad 1 \leq l \leq N, \quad EOR(l) \leq EMR(l) \tag{11}$$

Next, we establish the relation between the heuristic accuracy and the gap between EMR and EOR values. To quantify this gap and to understand its relation with heuristic accuracy, we choose an error distribution function. We assume that $Y(n)$ is a random variable uniformly distributed over $(0, e)$.

Lemma 3 *With the uniform tree model and the uniform ($[0, e]$) heuristic error distribution, the $q_{i,j}$ and $r_{i,j}$ values are given as,*

$$\begin{aligned} q_{i,j} &= e(N+j+2(1-i)) - 2(j+1-i)/e(N+j+2(1-i)) \\ r_{i,j} &= (e(N+j+2(1-i)) - 2(j+1-i))^2/2e^2(N+1-i))^2 \end{aligned} \tag{12}$$

This quantification of $q_{i,j}$ and $r_{i,j}$ values is obtained by using the cumulative properties of uniform distribution. While $q_{i,j}$ contains a single random variable $r_{i,j}$ is expressed as a subtraction of two independent random variables having a triangular distribution.

Theorem 3 *With the uniform ($[0, e]$) heuristic error distribution,*

$$\forall i, j,\ 1 \leq j \leq N, 1 \leq i \leq j,\ r_{i,j}/q_{i,j} \leq 1-(j+1-i)/e(N+1-i) \tag{13}$$

That is, with $e > 0$ the difference between $EMR(l)$ and $EOR(l)$ is magnified with greater heuristic accuracy.

In Theorems 2 and 3, we analytically prove the basic observations obtained from the experiments, i.e., $EOR(l)$ is less than (or equal to) $EMR(l)$ and the gap between $EOR(l)$ and $EMR(l)$ is magnified with increasing heuristic accuracy.

3 Probabilistic Rank Profile

The analysis presented in Section 2 shows that if we restrict A* using the rank information (up to opt_node_rank) we can expect faster convergence. However, the question is, *how do we compute such a rank estimate*. We must note that determining a constant as well as meaningful rank bound is not likely. More importantly, a constant bound will not help us in a constrained scenario as (even with such a bound) either we converge within the contract or we do not get a solution. Therefore instead of a constant bound, we propose to obtain a probabilistic estimation of *opt_node_rank* values, and use that information to guide the search. For this, we put forward the concept of Probabilistic Rank Profile (PRP).

Probabilistic Rank Profile is a model which represents the chance of expanding the 'optimal-path' node at a level if a maximum number of $k(l)$ nodes are expanded at that level.

Definition 9 *Probabilistic Rank Profile (PRP) : Probabilistic Rank Profile $P(S \mid l, k(l))$ (S denotes success) of a search space represents the probability of obtaining the 'optimal-path' node (in case of multiple, at least one of them) within the best (in terms of $f(n)$) $k(l)$-nodes at level l, i.e.,*

$$P(S \mid l, k(l)) = P(opt\_node\_rank(l) \leq k(l)) \tag{14}$$

If we expand the 'optimal-path' node at a level l, we say that we have achieved local success at that level. Thus, the PRP basically represents the chance of achieving local success at a level l with at most $k(l)$ expansions. Next, we investigate the basic characteristics of the PRP and explore the possibilities of computing (or approximating) the PRP values from search space configuration (branching factor, height, heuristic error estimates, etc)s.

For the first phase of analysis, we use the same uniform m-ary tree model [7] and compute the $P(S \mid l,\ k(l))$ values from the heuristic error information. First, we introduce the following notation:

Definition 10 *$P_{i,l}$: $P_{i,l}$ denotes the probability of an 'off-course' node $n_{i,l}$ to be in $W(l)$. Thus,*

$$P_{i,l} = P(n_{i,l} \in W(l)) = \Pi_{j=i}^{j=l} r_{i,j} \tag{15}$$

In the following theorem we present the relation between the PRP and the search space configuration,

Theorem 4 *With the uniform cost m-ary tree model, the PRP function can be computed as,*

$$\begin{array}{ll} & P(S|\ l,\ k(l)) \approx 0.5[1+\phi((k(l)-\mu)/\sigma\sqrt{2})] \\ where & \mu = 1+(m-1)\sum_{i=1}^{i=l} m^{l-i} * P_{i,l}, \\ and & \sigma^2 = (m-1)\sum_{i=1}^{i=l} m^{l-i} * P_{i,l} * (1-P_{i,l}) \end{array} \tag{16}$$

$\phi(x)$ is the standard error function for Normal distribution.

Proof 1 *By definition,*

$$P(S \mid l, k(l)) = P(opt\_node\_rank(l) \leq k(l)) \tag{17}$$

Now, the total number of nodes to be expanded at level l to obtain the 'optimal-path' node at that level, is a summation (over all possible 'off-course' sub-trees at level l) of independent Bernoulli trials of the nodes $n_{i,l}$. Thus, the distribution function for $opt\_node\_rank(l)$ can be obtained as a sum of Binomial Distributions $B((m-1)*m^{l-i}, P_{i,l})$ for $1 \leq i \leq l$ (with the obvious addition of the 'optimal-path node'). Approximating the Binomial distribution function using the corresponding Normal distribution, we get a more compact expression as,

$$\begin{array}{l} P(opt\_node\_rank(l)) \approx N(\mu,\sigma) \\ where\ \mu = 1+(m-1)\sum_{i=1}^{i=l} m^{l-i} * P_{i,l} \\ and\ \sigma^2 = (m-1)\sum_{i=1}^{i=l} m^{l-i} * P_{i,l} * (1-P_{i,l}) \end{array} \tag{18}$$

From the above presented expression, $P(opt\_node\_rank(l) \leq k(l)))$, $\forall k(l), 1 \leq k(l) \leq m^l$ can be obtained from the cumulative distribution function of $P(opt\_node\_rank(l))$. This probability denotes the probability of success at level l when at most $k(l)$ best nodes are selected for expansion. Thus, following the Normal distribution properties,

$$\begin{array}{l} P(S|\ l,\ k(l)) \approx 0.5[1+\phi((k(l)-\mu)/\sigma\sqrt{2})] \\ where\ \mu = 1+(m-1)\sum_{i=1}^{i=l} m^{l-i} * P_{i,l} \\ and\ \sigma^2 = (m-1)\sum_{i=1}^{i=l} m^{l-i} * P_{i,l} * (1-P_{i,l}) \end{array} \tag{19}$$

From Theorem. 4, we observe how the PRP values can be computed using the search space configuration and heuristic error information (which is captured through the $P_{i,l}$ definition). However, acquiring an accurate estimation of the heuristic error distribution for a given search space is a non-trivial task. Thus, we try to identify the basic properties of the PRP function and use them to suggest a function to approximate the PRP values. First, we present a set of observations about the nature of PRP.

- The success probability of expanding best $k(l)$ nodes of a frontier should be greater than or equal to $k(l)/frontier\_size(l)$, otherwise the path estimation is a *misleading* one, i.e., for a *reasonable* cost estimation function,

$$P(S \mid l, k(l)) \geq k(l)/m^l \tag{20}$$

- The improvement in success probability (with more node expansions) follows a sub-linear gradient (actually the improvement reduces exponentially) after a given number of expansions, i.e., after a given number of expansion $\psi(l)$, for any Δ

$$P(S \mid l, \psi(l)+\Delta) - P(S \mid l, \psi(l)) < \Delta/m^l \tag{21}$$

- The PRP curve obtained for a given level $l+1$ dominates the PRP curve for level l,

$$\forall c,\ \ P(S \mid l, c) \leq P(S \mid l+1, c*m) \tag{22}$$

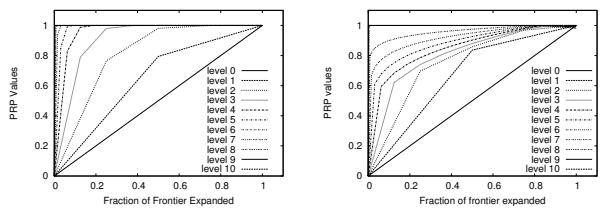
- If we have two heuristics h_1 and h_2, such that h_1 is probabilistically more accurate than h_2, in the average case, the PRP using h_1 dominates the PRP using h_2,

$$\forall\, l,\ P(P(S\ |l,c)_{h_1} \geq P(S\ |l,c)_{h_2}) \geq 0.5 \tag{23}$$

Using these basic properties, we suggest a function to approximate the PRP values for unknown search spaces. We approximate $P(S|l,k(l))$ using the following equation,

$$P(S|l,k(l)) = min((k(l)/m^l) * \alpha)^{(h-l/h)*\beta-\gamma}, 1) \tag{24}$$

where α, β and γ are positive constants chosen depending on the nature of the search space, with $\alpha \geq 1.0$, $\beta \leq 1.0$ and $\gamma \geq 0.0$. It should be noted that the suggested approximation function closely follows the normal distribution error function curve (Theorem 4). In Figure 4, we show two sets of PRP curves obtained for a uniform binary tree of height 10. Figure 4(a) shows the curves obtained using Eqn. 16 with heuristic error uniformly distributed over $[0,1]$, and Figure 4(b) shows the curves obtained using the approximation model with $\alpha = 1.2, \beta = 0.5$, and $\gamma = 0.1$. From the figure, we observe that the approximating function has similar growth characteristics with the computed profile curves.



(a) Using Eqn. 16

(b) Model curve, $\alpha = 1.2$, $\beta = 0.5$, and $\gamma = 0.1$

Figure 4. PRP values for uniform binary search tree

4 Contract Search : Formulation and Methodology

In this section, we present a heuristic search algorithm that maximizes the probability of reaching the optimal goal node(s) under a given node expansion constraint.

Optimal Bound Selection From the probabilistic rank profiles (PRP), we obtain the local success probabilities for each level of a search space conditioned by the number of expansions. In this part, we use the PRP to compute the expansion bounds for each level of the search space depending on the contract specification. We consider two different scenarios, one in which the optimal goal depth is known in advance and the other with unknown goal depth. In this first case, let the goal depth be N. We formulate the bound selection (k-selection) problem under a given contract C as follows: For

a problem having PRP $P(S|l,k(l))$ we generate a k-cutoff for each level l ($0 \leq l \leq N$) such that,

$$\sum_l k(l) \leq C \quad and \quad P_S(C) \text{ is maximized} \tag{25}$$

With this formulation, we define the k-selection strategy for a particular level as a Markov Decision Process (MDP).

Strategy 1 *Optimal k-Selection Strategy with Known Goal Depth : The $k(l)$-choice for a given level l, having remaining contract R and maximum available choice c (number of nodes at that level) can be obtained by solving the following equation for success probability (P_S),*

$$P_S(l,c,R) = \underset{n \leq min(c,R)}{max\ n} \begin{cases} P(S \mid l,n)* \\ P_S(l+1, m*n, R-n) \\ \quad if\ (l\ <\ N), \\ P(S \mid h, min(c,R)) \\ \quad otherwise. \end{cases} \tag{26}$$

Where m is the branching factor. Now for a given contract C, solving the equation for $P_S(0,1,C)$ we get the optimal $k(l)$-value selection for each level l.

We use dynamic programming to solve the above equation. The strategy can be computed off-line and then used to guide the actual search under specified node limitations.

Theorem 5 *The k-Selection strategy specified maximizing the above P_S function, represents an optimal guiding strategy for a contract search algorithm.*

It is to be noted, that the k-selection problem itself is NP-hard (reducible to 0/1 knapsack). However, a pseudo-polynomial time algorithm can be used to solve the problem.

If the depth of the goal node is not known, we can use a probabilistic estimate of the goal depth along with the rank profiles. For this we define a term called $GoalProbability(l)$, which represents the probability of the goal node to be at depth l.

Definition 11 *Goal Probability(l) : The Goal Probability ($P_G(l)$) of a search space represents the probability of the goal node to be at depth l of the search graph/tree, i.e.,*

$$P_G(l) = P(goal\ \ depth\ =\ l) \tag{27}$$

Using the Goal Probability values, the optimal k-selection strategy can be obtained using the following equation,

$$P_S(l,c,R) = \underset{n \leq min(c,R)}{\max\ n} \begin{array}{l} P(S \mid l,n) * (P_G(l)+ \\ P_S(l+1, m*n, R-n)) \end{array} \tag{28}$$

Contract Search Algorithm In this part, we present the contract based heuristic search algorithm (Contract Search). For a specified contract we compute the node expansion restrictions for each level using the k-selection routine. Contract Search uses these $k(l)$ values to limit the expansions for a particular level. A naive way of incorporating the restrictions would be to expand the best $k(l)$ nodes at level l.However, this approach will be inefficient, as it does not use the pruning capabilities of A*. Our aim is to restrict the number of competitions A* performs at a given level. If A* does not require to expand more than $k(l)$ nodes at a level l, we run it without restriction, otherwise we restrict. With this idea, we present the algorithm Contract Search(C) (Algorithm 1) using the off-line k-selection. In Contract Search, we logically (not necessarily physically) maintain a different open list for each level. Nodes are expanded in the best first mode across all open lists. With each expansion from a given level we update the node expansion count for that level. If the number of nodes expanded at a given level l equals the $k(l)$ value, the level is suspended. The search terminates when all levels are suspended or there are no more nodes having $f(n)$ less than the obtained solution.

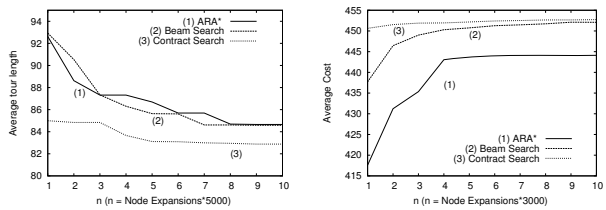
Algorithm 1 Contract Search (C)

STEP 0 :: **INPUT**
A graph G and a start node s. A cutoff $k(l)$ for each level of the graph (depending on the contract specification C).
STEP 1 :: **INITIALIZATION**
for i = 0 to l, such that $k(l) > 0$ **do**
$OpenList[i] \leftarrow \phi$; $SuspendFlag[i] \leftarrow 0$; $ExpCount[i] \leftarrow 0$;
end for
$ClosedList \leftarrow \phi$; $BestSol \leftarrow \infty$; Insert s to $OpenList[0]$;
STEP 2 :: **SELECTION**
if ($NodeCount \geq C$) Return $BestSol$;
Select node n having the least $f(n)$-value across all $OpenLists$ with $SuspendFlag = 0$;
if (No such n exists) *or* ($f(n) \geq BestSol$) **then**
Return BestSol;
end if
Insert n into $ClosedList$;
STEP 3 :: **EXPANSION**
$NodeCount \leftarrow NodeCount + 1$;
$ExpCount[Level(n)] \leftarrow ExpCount[Level(n)]\ +\ 1$;
if IsGoal(n) **then**
if $f(n) < BestSol$ $BestSol \leftarrow f(n)$; $Goal \leftarrow n$;
Goto STEP 2 (SELECTION);
end if
for Each successor node n' of n **do**
Calculate $f(n')$;
if n' is not in the $OpenLists$ or $ClosedList$ **then**
$Parent(n') \leftarrow n$; $Level(n') \leftarrow Level(n) + 1$; Insert n' to $OpenList[Level(n')]$;
else if n' is in $OpenLists$ **then**
if $f(n')$ < previously calculated path estimation **then**
$Parent(n') \leftarrow n$; $Level(n') \leftarrow Level(n) + 1$; Move n' from the earlier $OpenList$ to $OpenList[Level(n')]$;
end if
else if n' is in $ClosedList$ **then**
$Parent(n') \leftarrow n$; $Level(n') \leftarrow Level(n) + 1$; Move n' from $ClosedList$ to $OpenList[Level(n')]$;
end if
end for
if $ExpCount[Level(n)] \geq k(Level(n))$ **then**
$SuspendFlag[k] =\leftarrow 1$;
end if
STEP 4 :: **Goto STEP 2 (SELECTION);**

5 Experimental Results

We perform experiments on 4 core optimization problems, namely Euclidean TSP, 0/1 Knapsack problem, 15-puzzle problem and instruction scheduling problem [5]. We compare Contract Search against two *anytime* algorithms, ARA* [6] and best first beam search (which can also be termed as band search [2]).

For Contract Search, we use the PRP curve approximated using Eqn. 24. The approximation Eqn. 24 is used with the basic set of coefficient values ($\alpha = 1.0, \beta = 1.0$ and $\gamma = 0.0$). We assume 100 nodes to be to be a unit, and accordingly adjust the contract specification, i.e., a contract of $50,000$ nodes is equivalent to 500 units. We compute the $k(l)$ values for each level of the search space(for different problem). The maximum computation steps vary (for different problem) from $50,000 - 60,000$ basic operations, and the time required is almost insignificant ($<< 1ms$). Moreover, the $k(l)$ selection is performed off-line and once only (for each problem domain). In ARA*, we start with a weight bound of 2.0 and gradually decrease the bound by 0.1 at each iteration. For beam search, the beam width is calculated depending on the contract specification ($beam\_width = contract/height$). In Figure 5(a), we present the

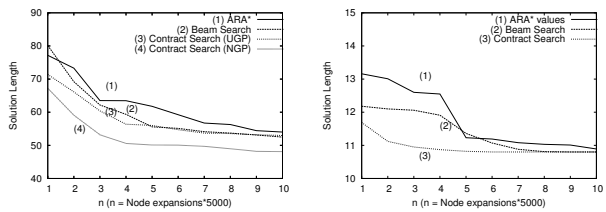


(a) Average Tour Length vs Contract for 100-city Traveling Salesman Problem

(b) Cost versus Contract for 200-object $0/1$ Knapsack Problem

Figure 5. Comparative results for TSP and 0/1 Knapsack: ARA*, beam search, and Contract Search.

tour length values obtained for a random set of 20 100-city TSP instances for 10 contracts ranging from $5,000 - 50,000$ nodes (with the MST heuristic). Figure 5(b) shows the cost vs contract results for 20 200 object $0/1$ Knapsack instances for a constraint range of $3,000 - 30,000$ nodes (with the best fitting heuristic). For Knapsack, the weights and costs of individual objects are generated randomly while the constraint is chosen within $0.4 - 0.6$ of the sum of weights. In Figure 6(a) and Figure 6(b), we include the results obtained for the



(a) Average Solution Length vs Contract for 15-Puzzle Problem

(b) Average Operator Cost versus Contract for Instruction Scheduling Problem (100-node DAGs)

Figure 6. Comparative results for 15-Puzzle and Instruction Scheduling: ARA*, beam search, and Contract Search.

15-puzzle problem (random set of 50 problems) and the instruction scheduling problem (randomly generated set of 50 100 node DAGs). For 15-puzzle, which is a graph problem having unknown solution depth, the framework needs to have an idea about goal depth distribution. For this, we use two simple functions to approximate the goal probabilities, uniform (from level $20 - 60$) and standard normal ($mean = 40$) distribution. In Figure 6(a), the two Contract Search variants are labeled as *Contract Search (UGP)* and *Contract Search (NGP)*, respectively. If no solution is obtained within the expansion limit, the cost is taken as 100 [3] For the time constrained instruction scheduling problem, we run the experiments with random time constraints between $10 - 20$ steps. All operator costs are taken as 1.

[3] This penalty is chosen in a random manner to have an equivalent distribution curve across all algorithms and contracts. For most of the contracts the number of unsolved problems for beam search and contract search (both UGP and NGP) is equivalent, while ARA* seems to perform worst in this regard.

The results show that for most of the contract specifications, solution qualities obtained with Contract Search are are superior than the other algorithms. For 3 of the 4 problems, Contract Search performance is significantly better than both ARA* and beam search. For 15-puzzle, Contract Search with normal goal depth distribution outperforms the other algorithms. However, if uniform goal probability is assumed, the results are equivalent with beam search. Comparing ARA* and best-first beam search (band search), we observe that beam search performs marginally better than ARA*, validating utility of obtaining a single solution over a stream of solutions. Overall, the trend shows that Contract Search with level specific expansions is a robust algorithm which provides quality solutions for different problems over a range of constraint specifications. It may be noted that the results are generated considering the simplest form of the approximation function (without any prior knowledge about the heuristic error distribution). If adequate information about the search space is acquired, the performance can be improved further. However, even with insufficient information, the search strategy can be appropriately adapted to a node constraint specification. using some generic knowledge about the search space characteristics [4].

6 Conclusions

In this paper, we have presented a contract specific heuristic search algorithm (Contract Search), which uses rank based local restrictions to adapt itself according the constraints. We have developed an analytical method to compute the level specific bounds from the search space configuration. Experimental results obtained on 4 important combinatorial optimization problems show the efficacy of the proposed algorithm.

REFERENCES

[1] S. Aine, P. P. Chakrabarti, and Rajeev Kumar, 'AWA* - a window constrained anytime heuristic search algorithm', in *IJCAI*, pp. 2250–2255, (2007).

[2] L. Chu and B. W. Wah, 'Band search: An efficient alternative to guided depth-first search', in *Proceedings of the 4th International Conference on Tools with Artificial Intelligence*, (1992).

[3] L. Chu and B. W. Wah, 'Solution of constrained optimization problems in limited time', in *Proceedings Workshop on Imprecise Computation, IEEE*, (1992).

[4] T. Dean and M. Boddy, 'An analysis of time-dependent planning', in *Proceedings of 6th AAAI 88*, pp. 49–54. AAAI Press, (1988).

[5] A. Kumar, A. Kumar, and M. Balakrishnan, 'Heuristic search based approach to scheduling, allocation and binding in data path synthesis.', in *VLSI Design*, pp. 75–80, (1995).

[6] M. Likhachev, G. J. Gordon, and S. Thrun, 'Ara*: Anytime A* with provable bounds on sub-optimality', in *Advances in Neural Information Processing Systems 16*, MIT Press, Cambridge, MA, (2004).

[7] J. Pearl, *Heuristics: intelligent search strategies for computer problem solving*, Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1984.

[8] I. Pohl, 'Heuristic search viewed as path finding in a graph.', *Artif. Intell.*, **1**(3), 193–204, (1970).

[9] E. Rich and K. Knight, *Artificial Intelligence*, McGraw-Hill Higher Education, 1991.

[10] R. Zhou and E. A. Hansen, 'Beam-stack search: Integrating backtracking with beam search.', in *Proceedings of the ICAPS-05*, pp. 90–98, Monterey, CA, (2005).

[4] It should be noted that the although the results are presented in terms of node expansion bounds, it has direct correlation with the actual run-time constraints. All the algorithms use the same expansion and heuristic computation policy. The only difference is in the node selection and pruning, while ARA* uses a weighing policy, both band search and Contract search maintain a level-wise counter to perform pruning.