Modeling Local Search: A First Step Toward Understanding Hill-climbing Search in Oversubscribed Scheduling*

Mark Roberts and **Adele Howe** and **L. Darrell Whitley**
Computer Science Dept., Colorado State University
Fort Collins, Colorado 80523
mroberts,howe,whitley@cs.colostate.edu
http://www.cs.colostate.edu/sched

Abstract

Previous results for a real-world domain, scheduling communications requests for the Air Force Satellite Control Network (AFSCN), suggested that the best search methods perform a greedy random walk. We examine the degree to which this observation is accurate for AFSCN by modeling a next-descent hill-climbing search as a Markov process. We find that the model is somewhat accurate for AFSCN. To generalize, we apply this model to another oversubscribed scheduling problem: scheduling image requests for a set of Earth Observing Satellites (EOS). We find that the hill-climber follows a trajectory that can be modeled as a Markov process.

Hill-Climbing Search as a Markovian Process

A Genetic Algorithm (GA), Squeaky Wheel Optimization (SWO), and a simple Hill-Climber (HC) have been shown to perform well for AFSCN scheduling (Barbulescu, Whitley, & Howe 2004). Simulated Annealing (SA) and HC perform best for EOS scheduling (Globus *et al.* 2004). All these methods use a permutation representation with some combination of stochastic motion and simultaneous moves. The GA selects approximately half the tasks for movement and is at the far end of multi-move, disruptive motion. The HC is at the other end of single-move, step-wise motion. SA follows a similar path as the HC, but relaxes the greedy bias by allowing uphill moves; its behavior approaches that of the HC as search progresses. SWO selects a subset of 'troublesome' tasks to move and is between the extreme moves of the GA and the small steps of the HC.

Researchers for AFSCN and EOS identify the underperformance of heuristic search in these domains, but it isn't clear exactly why this is the case. A simple hypothesis is that these methods lack a suitable ordering heuristic. Underlying this hypothesis is an implicit belief that the better performing algorithms are finding and exploiting rich, complex problem structure. Previous studies have shown that simple patterns do not account for good AFSCN schedules (Barbulescu, Whitley, & Howe 2004).

Recently for AFSCN, we have also found that an uninformed, completely random neighborhood performs better than systematic or problem specific neighborhoods (Roberts *et al.* 2005). AFSCN problems are dominated by neighborhoods with equivalent moves (plateaus and regions with many equal and only a few non-equal moves) (Barbulescu, Whitley, & Howe 2004). Based on the neighborhood result and evidence of plateaus, it seems reasonable to consider the possibility that the best algorithms may be performing a greedy random walk. Recent work in understanding the behavior of Tabu search for the Job-Shop Scheduling problem links search space features with the cost of locating an optimal solution; search can be modeled with 95% accuracy as a random walk (Watson, Whitley, & Howe 2004).

Our main focus in this work is to construct a Markov model for the simple, but reasonably successful, stochastic HC in two domains and to assess its accuracy. This is a first step toward bringing together a unifying model explaining the behavior of all of these search techniques, and possibly yielding more clues linking problem structure and run-time behavior.

Oversubscribed Scheduling

In oversubscribed scheduling, more requests need to be scheduled than can be feasibly accommodated given the available resources. This necessitates discarding some requests. We examine two oversubscribed scheduling applications: AFSCN and EOS.

For AFSCN, communication requests for earth orbiting satellites are scheduled on a set of 16 antennas at nine ground-based tracking stations. Satellites are grouped according to two orbits. Low-altitude orbits have a short visibility window and few scheduling alternatives. Typically, only one contact request can be scheduled during one 15 minute visibility window. In contrast, requests for high-altitude orbits have longer durations (20 minutes or more) and have much larger visibility windows. The scheduling alternatives can include multiple ground tracking stations. The requests to be scheduled include information about which ground stations and times are possible alternatives.

*This research was sponsored by the Air Force Office of Scientific Research, Air Force Materiel Command, USAF, under grant number F49620-03-1-0233. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon.

Customers submit requests that are scheduled by humans in a complex arbitration process. Although AFSCN starts as an oversubscribed scheduling problem, all jobs are eventually scheduled through negotiating relaxed task requirements. Our automated scheduler reduces human effort by minimizing the total number of tasks in conflict. When minimizing the number of conflicts, if the request cannot be scheduled on any of the alternative resources, it is dropped from the schedule (i.e., bumped).

Our AFSCN dataset consists of 12 days of real data identified by their dates[1]. Table 1 (left) shows characteristics for these problem instances. The seven older days of data are smaller problems that are easily solved by most of the approaches we have tried. The five newer days are substantially larger problems that are more difficult.

For EOS, as in (Globus *et al.* 2004), image capture requests are scheduled on a set of 1 to 3 identical satellites that are spaced ten minutes apart in a sun-synchronous, earth orbit at 800 km. The imaging sensor is mounted with a total possible cross-track slew of 24 degrees off nadir (straight down). A Solid State Recorder (SSR) stores on-board data and is dumped as the satellite passes over a remote tracking station located in Anchorage, Alaska. For each satellite, 2100 land-based image targets are randomly selected from the Geographic Nameserver Database (NGIA 2004). Image exposure duration ranges from 24 to 48 seconds depending on the problem. We use the free version of STK (AGI 2004) to model one week of orbits and target visibilities[2]. Not all tasks are visible from the satellites in every problem, but the number of requests ensure an oversubscribed problem.

The multi-objective evaluation function is:

$$F = w_p \Sigma_{u \in U} P_u + w_s \mu_{slew} + w_a \mu_{angle}$$

where w_p, w_s, w_a are the weights assigned in the problem definition and where μ_{angle}, μ_{slew}, and $\Sigma_{u \in U} P_u$ are the average image angle, the average slew per task, and the sum of the unscheduled image request priorities, respectively. For this paper, we round the value of F to the nearest integer.

Our EOS dataset consists of ten problem instances using problem parameters outlined in (Globus *et al.* 2004). These problems are specifically designed to mimic realistic satellite scheduling problems. Table 1 (right) shows characteristics for these problem instances. The last three columns show the weights for the evaluation function.

A Simple Hill-climbing Algorithm

We encode potential solutions using a permutation π of the n task IDs, $[1..n]$. A *schedule builder* is used to generate solutions from the permutation. In effect, the permutation π acts as a priority queue, and the schedule builder places task requests in the schedule based on the order that they appear in π. Each task request is assigned to the first available resource from its list of alternatives and at the earliest possible starting time. This assignment treats the list of alternatives as a rank order, although the actual ordering is arbitrary.

We implemented a next-descent HC that employs the *shift* operator; we accept new solutions that are better or equally good. From a current solution π, a neighborhood is defined by considering all $(n-1)^2$ pairs (x, y) of positions in π, subject to the restriction that $y \neq x - 1$. The neighbor π' corresponding to the position pair (x, y) is produced by *shifting* the job at position x into position y, while leaving all other relative job orders unchanged. If $x < y$, then $\pi' = (\pi(1), ..., \pi(x-1), \pi(x+1), ..., \pi(y), \pi(x), \pi(y+1), ..., \pi(n))$. If $x > y$, then $\pi' = (\pi(1), ..., \pi(y-1), \pi(x), \pi(y), ..., \pi(x-1), \pi(x+1), ..., \pi(n))$.

Building a Model

Our goal is to model the expected cost of moving from a random solution to a solution with the best known value. By expected cost, we mean the average number of shifts made in the shift neighborhood. To estimate the search cost, we construct a Markov model of search progress from a set of independent runs and calculate the waiting time from each state. In using a Markov model we make two assumptions: 1) that movement to the next state is entirely dependent on the current state and 2) that we can make a suitable estimate of the transition probabilities between states.

We adapt the methodology from (Frank, Cheeseman, & Stutz 1997) to name states and 'probabilistically paint' the schedule space. For a given problem instance $\mathcal{I}$ with n tasks, let G be an undirected graph of the $n!$ permutations. Two vertices of the graph, g_1, g_2 are adjacent, that is, have an edge between them, if they correspond to two permutations differing by a single shift. G is the search space induced by the shift operator. We associate G with the evaluation (schedule) space by assigning each vertex the objective function value from the schedule builder $\mathcal{B}$ for $\mathcal{I}$.

Definition 1 ***(Level)*** *Let $f : G \rightarrow \mathcal{Z}^+$ be a function mapping permutations to integers such that $f(\mathcal{B}(g)) = z$ if and only if the permutation corresponding to g results in an evaluation of z using schedule builder $\mathcal{B}$.*

Definition 2 ***(Plateau)*** *Let P be a connected subgraph of G and let $z \in \mathcal{Z}^+$ be a constant. Then P is a plateau if P is a maximal connected subgraph of G such that $f(\mathcal{B}(p)) = z$ for all $p \in P$. Further, z is defined to be the level of the plateau.*

Definition 3 ***(Aggregate Plateau)*** *Let A be the set of all plateaus in G at level $z \in \mathcal{Z}^+$. Then A is an aggregate plateau if every plateau $a \in A$ is a plateau at level z and every plateau $p \in G$ at level z is in the set A. Further, z is defined to be the level of the aggregate plateau.*

Definition 4 ***(Model State)*** *Let s be a single state in the Markov model with a set of states $\mathcal{S}$ and a transition probability matrix $\mathcal{T}$. Then s is an aggregate plateau in G at level $z \in Z^+$. Further, s_z is defined to be the level of the state. A transition probability, t_{ij}, signifies the probability of moving from state s_i to s_j. Let s_{min} to denote the lowest s_z in the model and s_{max} to denote the highest s_z in the model; $0 < s_{min} < s_{max} < O(n)$.*

[1]We thank Dr. James T. Moore, Associate Professor, Dept. of Operational Sciences, Air Force Institute of Technology and Brian Bayless and William Szary from Schriever Air Force Base for providing the AFSCN data.

[2]We thank Al Globus of CSC at NASA Ames for walking us through retrieving target visibilities from STK.

ID	Date	Size	# Low	# High
Day1	10/12/92	322	153	169
Day2	10/13/92	302	137	165
Day3	10/14/92	311	146	165
Day4	10/15/92	318	142	176
Day5	10/16/92	305	142	163
Day6	10/17/92	299	144	155
Day7	10/18/92	297	142	155
Mar07	03/07/02	483	225	258
Mar20	03/20/02	457	194	263
Mar26	03/26/03	426	183	243
Apr02	04/02/03	431	185	246
May02	05/02/03	419	178	241

ID	Size	SSR	Slew	SSRUse	Priority	w_p	w_s	w_a
1	1 (2100)	75	2	1	1-6(50)	1	0.01	0.02
2	3 (6300)	50	2	1,3,5	1-6(50)	1	0.01	0.50
3	3 (6300)	50	2	1,3,5	1-6(50)	1	0.01	0.00137
4	3 (6300)	75	2	1,3,5	1-6(50)	1	0.01	0.02
5	3 (6300)	50	10	1,3,5	1-51(5)	1	0.50	0.20
6	3 (6300)	75	2	1,5,8	1-16(50)	1	0.10	0.20
7	3 (6300)	75	2	1,5,8	1-16(50)	1	0.10	0.20
8	3 (6300)	75	2	1,3,5	1-6(50)	1	0.01	0.02
9	2 (4200)	75	2	1,3	1-6(50)	1	0.01	0.02
10	3 (6300)	100	1	1,10,25	1-6(50)	1	0.10	0.70

Table 1: The left table shows AFSCN problem characteristics for the 12 days of data used in our experiments. *ID* is used to identify the instance throughout the paper. *Size* is the number of requests in the problem. *# Low* and *# High* are the number of low and high-altitude requests in each problem. The right table shows the problem characteristics for the EOS problem. *ID* is used to identify the instance throughout the paper. *Size* is the number of satellites (image requests) in the problem. *SSR* is the size of the on-board storage. *Slew* is the speed of the slew motor in degrees/second. *SSRUse* is the amount of SSR usage and is uniformly divided among the tasks in the problem. *Priority* is the value range (number of levels) for priority assignments to tasks. The last three columns signify the weights for the evaluation function.

To summarize, we aggregate states into sets partitioned by level. An aggregate plateau combines all states of the same level regardless of whether they are actually on the same plateau. It is intractable to explicitly model every solution ($\Theta(2^n)$ for the schedule space or $O(n!)$ for the permutation space). Using the level as a surrogate state makes many assumptions about the independence between schedules and movement between states. These assumptions could lead to model inaccuracy.

The size of the AFSCN model is straightforward: s_{min} denotes the best value seen and s_{max} denotes the highest value seen during the search. We are reasonably sure that s_{min} is in fact the optimal value for each problem.

Memory and time limitations prohibit us from constructing a full EOS model. We are also less confi dent that we've located the best values for EOS. To capture the runtime dynamics for the 'settled' behavior of the HC, we model the lowest and largest portion of search that we can. EOS runs incur a high computational cost. To include as many runs as possible, we set s_{min} to $max(finalEval(r)), r \in R$, where R is the set of runs used to construct the model. We limit the model size such that $s_{max} = s_{min} + 1000$. These model choices assume that all runs are equal, which could lead to a negative result if some runs are stuck on suboptimal plateaus.

We run each problem for as many evaluations as we can given our computing resources. For AFSCN, we stop the search at 50,000 evaluations per run (2 to 10 minutes of CPU time). For EOS, we stop search at 100,000 evaluations per run (almost one day of CPU time).

We estimate the transition matrix by collecting transition counts from a set of independent runs. For AFSCN, we use 25 runs per state; each run is guaranteed to reach s_{min}. For EOS, we use approximately 50 runs for the entire model. From these transition counts, we calculate the transition probabilities for the states. We then use the transition matrix to calculate the expected waiting time from each s to s_{min}. In our analysis of the model, we make two simplifying assumptions: s_{min} is absorbing, and s_{max} is a reflecting barrier.

VALIDATE-ENTRY

```
1  numShifts ← 0
2  while v in S_validate has less than m runs
3      do repeat x ← random solution
4           until LEVEL(x) > v_z
5         while LEVEL(x) > v_z
6             do HC proceeds one shift
7         while LEVEL(x) > s_min
8             do HC proceeds one shift
9                numShifts ← numShifts +1
10 return numShifts
```

Figure 1: Pseudo-code for VALIDATE-ENTRY. LEVEL(x) returns the integer level of the solution x using the appropriate schedule builder. All runs reach s_{min}.

Validating The Model

We are interested in measuring how well the model predicts the search cost from each model state given as large as possible a set of independent runs. For each state in the model, we correlate the predicted cost of the model with the actual cost of arriving at s_{min}. We obtain m validating runs for each state in the set of states we wish to validate, $\mathcal{S}_{validate}$. Figure 1 shows the algorithm for obtaining the actual search cost of each state $v \in \mathcal{S}_{validate}$. We observed that the run length distributions for most states $\mathcal{S}_{validate}$ have heavy tails, so we correlate the model cost with the *median* actual cost.

For AFSCN, we show results for all 12 days of data where $m = 25$ and $\mathcal{S}_{validate} = \mathcal{S} - \{s_{min}\}$. For EOS, we present a random sampling of the ten problems where $m = 7$ and

AFSCN	r^2
Day1	.040
Day2	.855
Day3	.743
Day4	.643
Day5	.052
Day6	.716

AFSCN	r^2
Day7	.844
Mar07	.569
Mar20	.611
Mar26	.287
Apr02	.720
May02	.688

EOS	r^2
1	.805
3	.851
6	.805
7	.839
9	.831
10	.606

Table 2: R-squared values for the hypothesis that the correlation for a given problem is linear.

$\mathcal{S}_{validate} = \{s_{min+1}, s_{min+333}, s_{min+666}, s_{max}\}, s \in \mathcal{S}$. Table 2 shows the r-squared values for linear regression. This preliminary analysis shows that there is a strong linear relationship between the model for most of the problems.

Summary and Future Work

For two oversubscribed scheduling problems, we hypothesize that a random walk model may capture the dynamics of search. We built a Markov model of the search using aggregate plateaus as the states. We found that this model is well-correlated for EOS, though it is less consistently correlated for AFSCN. This is simply the first step toward explaining search in such problems. We now identify some possible sources of model failure that we are currently addressing.

First, we note that states far away from s_{min} are much less correlated with the actual search cost. This effect is most noticeable for the larger problems in AFSCN. Figure 2 shows the correlation plot for Mar07, where this compression of states is seen on the right side. We conjecture that this compression results from a lack of sufficient state for the large plateaus seen during search. So we are currently building a more complex model with additional states representing the length of a plateau walk. This model refinement may also provide a more clear estimate of the plateau size for these problems. Preliminary evidence suggests that the larger model does resolve some of this compression.

Second, the model is clearly lacking in strength for some problem instances. Specifically, Day1, Day5, Mar26, and EOS10 show low correlation in comparison to the other problem instances. There is strong evidence that the compression mentioned above plays a role in the low correlation. The random walk may not be the best model for these problems; in which case, we will seek to capture the differences between EOS and AFSCN that might explain the differences in performance.

Third, the issue of knowing the best-value for EOS is still open, and search could be made more efficient. We will continue our analysis of the model as other problems accumulate more runs.

Finally, we have only taken one step toward modeling the simplest algorithm for these domains. We hope to generalize this model to other algorithms with the goal of finding a unifying model.

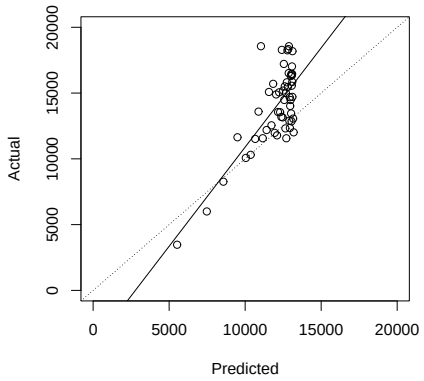

Figure 2: The correlation plot for Mar07, the largest AFSCN instance. The dashed line indicates perfect correlation. This plot reveals evidence of compression in the model for states far away from the optimum.

References

(AGI) Analytical Graphics, Inc. Satellite Toolkit (STK). www.stk.com. Accessed Sept. 9, 2004.

Anderson, E. J. 2002. Markov chain modeling of the solution surface in local search. *Journal of the Operational Research Society* 53:630–636.

Barbulescu, L.; Whitley, L.; and Howe, A. E. 2004. Leap before you look: An effective strategy in an oversubscribed problem. In *Proceedings of the Nineteenth National Artificial Intelligence Conference (AAAI-04)*.

Frank, J.; Cheeseman, P.; and Stutz, J. 1997. When gravity fails: Local search topology. *Journal of Artificial Intelligence Research* 7:249–281.

Globus, A.; Crawford, J.; Lohn, J. D.; and Pryor., A. 2004. A comparison of techniques for scheduling earth observing satellites. In *Proceedings of the Sixteenth Conference on Innovative Applications of Artificial Intelligence*, 836–843.

Kemeny, J. G., and Snell, J. L. 1976. *Finite Markov Chains*. New York: Springer-Verlag. chapter Absorbing Markov Chains, 43–68.

(NGIA) National Geospatial-Intelligence Agency. Geographic Name Server (GNS). http://gnswww.nima.mil/geonames/GNS/index.jsp. Accessed Aug. 31, 2004.

Roberts, M.; Whitley, L. D.; Howe, A. E.; and Barbulescu, L. 2005. Random walks and neighborhood bias in oversubscribed scheduling. In *2nd Multidisiplinary Conference on Scheduling: Theory and Applications (MISTA-05)*, to appear.

Watson, J.; Whitley, L.; and Howe, A. 2004. Linking search structure, run-time dynamics, and problem difficulty: A step toward demystifying tabu search. *Journal of Artificial Intelligence Research* under submission.