Stochastic Optimization for Adapting Behaviors of Exploration Agents

B. Engelhardt, S. Chien
Jet Propulsion Laboratory
California Institute of Technology

Keywords: stochastic optimization, planning, adaptation, autonomy, exploration agents

Abstract

Proposed missions to explore comets and moons will encounter environments that are hostile and unpredictable. A successful explorer must be able to adapt to a wide range of possible operating solutions to survive. Constructing special-purpose behaviors requires information about the environment, which is not available *a priori* for these missions. Instead, we propose an explorer that uses a flexible problem-solver with a significant capacity to adapt its behavior. More specifically, the explorer uses stochastic optimization techniques to continually adapt its behavior while limiting the cost of behavior exploration. With Adaptive Problem Solving, we use search techniques to enable a spacecraft to continually adapt its environment-specific behavior in-situ.

1 Introduction

Optimizing spacecraft behavior is a difficult problem; the problem only becomes more difficult when the utility feedback function is stochastic (e.g. a distribution instead of a single value). Stochastic optimization is the problem of optimizing the *expected utility* of a hypothesis behavior for a certain domain. Our adaptive problem solver optimizes the control strategy used by a planning system (for this work, the Automated Scheduling and Planning Environment, or ASPEN [Chien 2000]), which is autonomously controlling a spacecraft. The planner searches through the space of all possible plans to find a single plan that accomplishes the mission goals while satisfying all of the resource and temporal constraints of the model. The control strategy to be optimized is expressed as a vector of weights on heuristic functions used to control the search. Adapting the control strategy will impact the generated plans and thus the decision-making strategy (or behavior) of the spacecraft.

There are two parts to adaptation by stochastic optimization: evaluating hypothesis decision-making strategies, and generating the set of candidate decision strategies. The evaluation component generates a sufficient number of samples for each of the hypothesis behaviors to satisfy a given decision criterion in order to rank the hypothesis behaviors within given statistical bounds [Chien 1995]. The generation method uses the ranking from the evaluation step to generate the subsequent set of hypothesis behaviors to be evaluated. The cycle stops when the highest estimated utility has reached quiescence or a time limit has been reached. The two parts of the stochastic optimization problem address different classes of problems, the first addressing the problem of selection and the second addressing the problem of search, but algorithms for each part can be chosen such that their characteristics can be used in a synertistic way to facilitate the overall problem of adapation.

Strategy evaluation is the process in stochastic optimization of determining which of the possible control strategy hypotheses outperforms the others for a specific set of problem instances in the domain. Because the utility *U(h)* cannot be found directly, sampling is used to estimate the expected performance for each control strategy hypothesis. A *statistical decision criterion*, which may take several forms, determines when enough samples have been taken to be reasonably sure of the relative standing of strategy hypothesis *h*, with some user-defined confidence or error bounds. The purpose of the decision criteria is to enable intelligent sampling to allocate more samples to hypotheses with less certain utility estimates, thus reaching a confident decision while reducing sampling cost.

There are many possible decision criteria for use in the evaluation function. Some simplify the problem by making the assumption that the stochastic data is normally distributed, while others do not. Our current work has focused on three different requirements: the Probably Approximately Correct (PAC) requirement [Valiant 1984], Chernoff bounds (e.g., due to Chernoff [Hagerup & Rub, 1990]), Hoeffding's inequality

[Hoeffding, 1963], and Bernstein's inequality [Bernstein, 1946]. The PAC requirement assumes normally distributed data, whereas Chernoff bounds, Hoeffding's inequality, and Bernstein's inequality are distribution-free, or non-parametric, and do not make any assumptions about the distribution of the data. With the PAC decision criterion, an algorithm makes decisions with a given confidence (probability 1-d, for small d) to select a good hypothesis (within error e of the best hypothesis). Because any specific decision either satisfies or does not satisfy this requirement, the PAC criterion holds that over a large number of decisions that the accuracy rate must meet 1-? . For data that is not normally distributed, distribution-free exponential bounds due to Chernoff, Hoeffding, and Bernstein may be substituted for the exact normal-theory probabilities. We can use such exponential inequalities to get accurate bounds on the probability of incorrect selection, retaining a probabilistic guarantee, while avoiding the normality assumption.

Strategy generation is the second component in adaptive problem solving. The search landscape is a large, multi-dimensional space. Local search techniques could be used to perform optimization within this search space effectively, as they are generic, adaptable, and easy to understand. But local search techniques will not work well in the search space unless gradient information is helpful, the space is smooth, or at a very minimum the space is continuous. In addition, the landscape of the search space is defined by the neighborhood of the selected step function. This means two vector hypotheses that are one neighborhood step away from each other using a single search function could be many steps from each other using another. To search this space effectively, we must find a search technique that creates a smooth, easily searchable space.

We evaluate three different search techniques: random search, local beam search, and genetics-inspired search. The difficulty of searching a landscape is tightly coupled to the landscape properties of ruggedness, distribution of local optima, number of local optima, and topology of attraction basins. These properties will be quantified for the three search techniques [Stadler 2000]. Furthermore, we compare the actual results of applying the three different search algorithms, and illustrate how effective local search techniques are on the chosen domains.

In this paper we will describe (1) alternative decision criteria to use for evaluating the strategies, (2) alternative local search methods and enhancements used to generate candidate strategies, and (3) performance of the different search algorithms in two different planning domains.

2 Europa Submersible Scenario

Consider NASA's planned mission to Jupiter's moon, Europa, which will send a robot to the surface of the moon Europa in order to analyze the ice surface and hypothesized ocean underneath. Once the submersible has dug through the ice, it might have the scientific goals of moving to the next target 40% of the time allotted to achieving goals, and imaging or performing scientific experiments 60% of that time. All of the images must be uplinked to the orbiter, and an uplink should happen once every orbit in order to ensure that the image buffer capacity will not be exceeded.

Many scenarios would force these percentages and plans to be adapted. One scenario might be that movement in the ocean is much more resource-intensive than expected, in which case moving 40% of the time would not allow the submersible to make as much progress as previously expected, so perhaps it might choose to increase this amount. Another scenario might be that the ice layer is much thicker than previously imagined and uplinking the on-board data to the orbiter requires special positioning of the submersible. In this case, it might be better to uplink once every few orbits to let the submersible have more time to collect experiments before it gets into the uplink position. This creates the need to increase the percentage of time devoted to moving, since the uplink position will require movement to and from the current position of the submersible. The increase in the percentage of time devoted to movement will cause less science to be taken, which is synergistic with the adaptation for fewer uplinks.

Failure to adapt to these situations could cost the submersible the mission, by depleting resources too rapidly, not accomplishing mission objectives, or wearing out equipment. Not all possible situations can be enumerated before the mission; instead an adaptive problem solver checks the current control strategy's performance in the given environment and responds to changes by adapting the decision-making strategy, independent of the cause of the change. An adaptive problem solver would continually adapt the control strategy if it found the current strategy sub-optimal.

In actual operations, the submersible would have a planning system controlling its decision-making processes using a chosen behavior. The preliminary behavior would be responsible for preserving the safety of the submersible while it learned about its new environment. In a separate process, an alternate planner and simulator would be adapting the current behavior to include movement and science goals, while the simulator receives constant updates from the submersible as to the current state of the environment. When a decision-making strategy is learned in the separate process that preserved the safety of the submersible and accomplished mission goals, it could be swapped into the primary planning system and used operationally. If this decision-making strategy began to degrade in performance, the primary planning system would request a new strategy, and update the simulator with possible environmental changes or spacecraft degradations. Thus adaptation relies on environmental feedback to improve the current decision-making strategy on-board a spacecraft, while maintaining the safety of the spacecraft.

3 Planning Domain

We investigate stochastic optimization in the context of learning control strategies for the ASPEN planner [Chien 2000]. ASPEN uses heuristics to facilitate the iterative search for a feasible plan. During each search step, a planner confronts a series of decisions such as which schedule conflict to repair or which action to take to repair a chosen conflict. The planner resolves these choices by stochastically applying the heuristics, based on weights for each choice point heuristic, during iterative repair [Zweben 1994]. Thus the weights define the control strategy of the planner, which impacts the expected utility of the resulting plans.

Specifically, in our experiments, a strategy hypothesis is a vector with a weight for each heuristic function and a weight of 0 for a heuristic not in use. The utility of a hypothesis can be determined by running the planner using the control strategy hypothesis on a certain problem instance and scoring the resulting plan. A problem generator for each domain provides a stochastic set of problem instances to enhance the robustness of the expected solution for the entire planning domain.

In our ASPEN planning system, there are twelve choice points in the repair search space. Higher level choice points include choosing the conflict to resolve and choosing the resolution method, such as preferring open constraints before violated constraints, or preferring to add activities over moving them. Once a resolution method is selected, further choice points influence applications of the choice point such as where to place a newly created activity and how to instantiate its parameters. For each choice point, a set of heuristics specific to that choice point may be used. The hypothesis vector is the list of relative weight that is given to each heuristic for that choice point. Since the planner is stochastic, the choice of heuristics that are used at each step is randomized, so multiple runs even for the same problem instance may yield a range of solutions (plans) and hence a distribution of utilities.

3.1 Domains

The repair heuristics were developed for individual domain search requirements from ASPEN [Chien 2000] applications and subsequently abstracted for application in a generic ASPEN domain. There are also domain-specific heuristics, which reference particular features of a domain in order to affect the search. For each domain, the human expert strategy hypotheses were derived independently from (and prior to) our study by manual experimentation and domain analysis.
We examine two different spacecraft domains, which satisfy the normality assumption of the evaluation method. The first domain, Earth Orbiter-1 (EO-1), is an earth imaging satellite. The domain consists of managing spacecraft operations constraints (power, thermal, pointing, buffers, telecommunications, etc.) and science goals (imaging targets and calibrating instruments with observation parameters). Each problem instance is used to create a two-day operations plan: a typical weather and instrument pattern, observation goals (between 3 and 16), and a number of satellite passes (between 50 and 175). EO-1 plans prefer more calibrations and observations, earlier start times for the observations, fewer solar array and aperture manipulations, lower maximum value over the entire schedule for solar array usage, and higher levels of propellant [Sherwood 1998].

The Comet Nucleus Sample Return (CNSR) domain models landed operations of a spacecraft designed to land on a comet and return a sample to earth. Resources include power, battery, communications, RAM, communications relay in-view, drill, and ovens. Science includes mining and analyzing a sample from the comet, and imaging. The problem generator includes between 1 and 11 mining activities and between 1 and 24 imaging activities at random start times. The scoring functions for the CNSR domain includes preferences for more imaging activities, more mining activities, more battery charge over the entire horizon, fewer drill movements, and fewer uplink activities.

4 Adaptive Problem Solving

We adopt an iterative approach to statistical optimization, as defined by the algorithm in figure 1.
There are two algorithmic parts of this optimization method: ranking the best hypotheses from a given set of hypotheses using hypothesis evaluation, and generating a new set of hypotheses using a

neighborhood function and the current rankings using strategy optimization. In the next sections, we will discuss both elements of stochastic optimization, including describing a range algorithms used for each part, and possibilities for selecting algorithms to aid adaptation synergistically.

4.1 Hypothesis Evaluation

Hypothesis Evaluation is the key process in stochastic optimization of determining which of the possible decision-making strategy hypotheses outperforms the others for a specific set of problem instances in the domain. This is done by estimating the expected utility based on the minimal number of samples as determined by a statistical decision criterion.

```
Iterative Stochastic Optimization
For a set of $n$ hypotheses $H$
While (time not exceeded || quiescence not reached)
  Step 1: Using a decision criterion, select
        the best $h_i$ from $H$ by sampling in the domain
  Step 2: Using $h_i$, generate $H$'' using a chosen
        neighborhood function; $H := H$'
Return current $h_i$
```

In the actual evaluation, initial samples are taken to generate a starting expected utility and estimated variance and select the top control strategy hypothesis thus far. Then the algorithm checks to see whether each pairwise comparison between h_{sel} and the other hypotheses is statistically correct using a decision criterion. If the comparison satisfies the criterion, no more samples are taken to compare the two; if the comparison fails to satisfy the criterion, additional samples are taken until the criterion is satisfied for all of the comparisons. The idea of selection by pairwise comparisons is similar to a common implementation of tournament ranking schemes in genetic programming.

Decision criteria enable a choice among a set of possible hypotheses (e.g., actions, parameters) when the consequence of the choice depends on the interactions of the hypothesis in some stochastic environment. We can determine the value of each choice from sample interactions at a cost to our system. Decision criteria can be applied to many problems, including learning statistical decision models [Maron & Moore, 199**Error! Reference source not found.**3], optimizing function parameters [Dubrawski & Schneider, 1997], design optimization [Fukunaga et al, 1997], algorithm selection [Chien et al, 1999], or optimizing planner control strategies [Engelhardt & Chien, 2000, Gratch & DeJong 1994].

Stopping rule	**Assumption**
Nádas (PAC)	Random variable distributed normally
Chernoff	Random variable has upper bound B
Hoeffding	Random variable is bounded by $[a, b]$
Bernstein	Maximum difference between random variable mean and single instance bounded by M

The selection of an appropriate decision criterion to use for a selection problem set depends heavily on the distribution of the utility values of the hypotheses. Our work has examined four decision criteria: a probability approximately correct (PAC) criteria [Valiant, 1984], Chernoff Bounds [Hagerup & Rub, 1990], Hoeffding's inequality [Hoeffding, 1963], and Bernstein's inequality [Bernstein, 1946]. Each of these criteria uses information about the data in order to converge faster than the costly brute-force alternatives.

Table 1: List of the assumption made for each decision criteria

Parametric decision criteria assume that the random variable distribution is based on a distribution function given some parameter, and the goal is to estimate the parameter. Our parametric PAC algorithm, based on a stopping rule introduced in [Nádas, 1969], assumes that the random variable distribution is normal, and requires fewer samples. In a specific domain, if the distribution of the data cannot be estimated by a common distribution function, non-parametric (distribution-free) bounds (e.g., due to Chernoff, Hoeffding, and Bernstein) may be substituted for the normal-theory probabilities above. The complexity of these bounds are described by their rate of convergence: the probability that the estimated expected utility is not within error ε of the actual expected utility goes to 0 exponentially fast as the number of samples m increases [Hoeffding, 1963]. Because of the special nature of the utility estimates (i.e., sample means), these bounds typically give accurate results due to the concentration of measure phenomenon [Talagrand, 1991]. Chernoff bounds require the upper bound of the data ("one-sided" bound) be specified. Hoeffding's inequality makes the assumption that the data fall within a given $[a, b]$, and will converge at least as fast as Chernoff bounds because of the additional information ("two-sided" bound). Bernstein's inequality makes the assumption that the maximum variance of a single sample utility from the expected utility is bounded by a value M ("two-sided" bound), which further improves convergence properties of the criterion. Convergence does not guarantee accuracy when the random variable does not satisfy the imposed assumptions, and this discrepancy is illustrated in the accuracy measure.

Given an understanding of how the utility is distributed for a specific domain, the choice of decision criteria should attempt to maximize the amount of encoded information about the distribution. For example, if the distribution is parametric, then the PAC decision criteria should be used, as convergence will require much fewer samples while maintaining accuracy than with the non-parametric decision criteria. Our experiments were conducted using the PAC criterion with the Nádas stopping rule.

4.2 Strategy Optimization

For strategy optimization, we use Memetic Algorithms [Memetic 2000, Merz & Freisleben 1999], which incorporate both local search methods and evolutionary computation approaches. The search methods that we are currently using include random samp ling, a local beam search algorithm, and a genetic algorithm. Search is conducted in the following manner: we begin with an initial control strategy h_0, or in the case of genetic search a set of ranked control strategies, and a candidate generation function for each of the search algorithms. The candidate generation function generates the next set of hypotheses given the current ranked generation. The hypothesis ranking system ranks the new set of hypotheses and choses the best one to become h_1, or in the case of the genetic algorithm, chooses a subset of highest ranking hypotheses for the subsequent generation step. The search algorithm generates another set of hypotheses based on using the step function with h_1, and this process continues until quiessence or a time limit is exceeded.

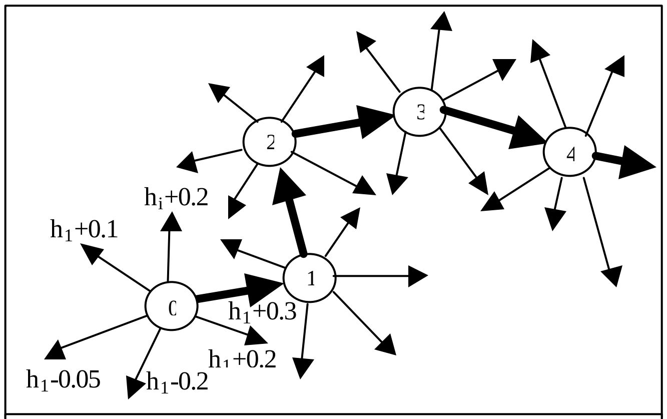


Figure 1: Strategy optimization with a neighborhood search step

The three current search functions used are biologically inspired search, local beam search, and random search. Random search does not rely on the previous generation of hypothesis vectors; instead it creates a new generation from random, independent points in the search space.

Local beam search takes the top *b* (where *b* is the size of the beam) ranked strategy hypotheses, and builds the next candidate generation using neighborhood steps applied to those hypotheses. A neighborhood step is defined as a mutation of each element of the control strategy hypothesis vector up to a certain total percentage. This is equivalent to reproduction and mutation in biologically inspired search, although the mutation in local search bounds the amount that the vector element can be mutated.

Biologially inspired search takes the top *n* ranking hypotheses, and uses them as the parents to the offspring of the next generation. The neighborhood functions are one of the three basic genetic operators: reproduction, crossover, and mutation. For reproduction, a single parent is selected based on its ranking and duplicated exactly in the next generation based. For crossover, two parent strategy hypotheses are chosen based on their ranking to split and recombine at some point *k* in their vector to form two offspring. For mutation, each single element of the offspring vector is mutated with a low probability.

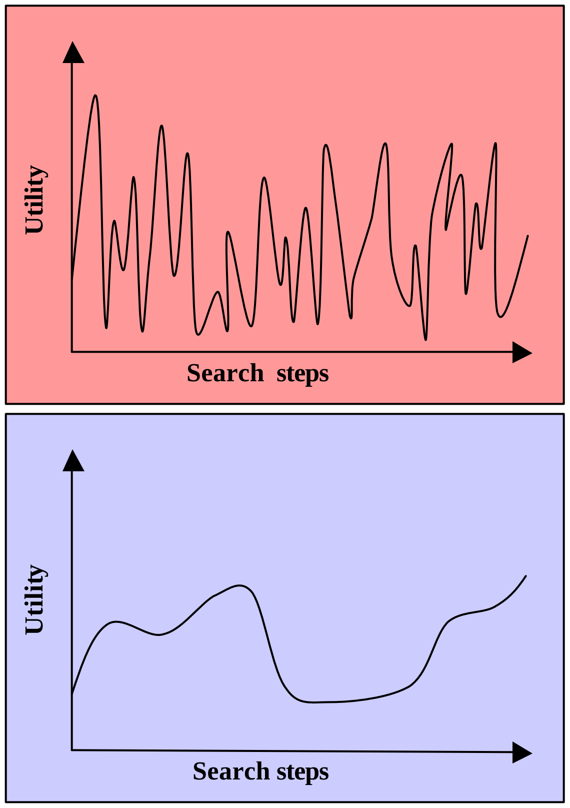
5 Results

We investigated two different aspects of the performance of each candidate generation method: ruggedness and directed search. Each performance element helps to quantify the performance of neighborhood functions in an unsearchably big domain with a large number of local maxima.

5.1 Ruggedness

Ruggedness is a measure of the correlation between adjacent (or neighborhood) points for a given neighborhood function. In order for local search to be effective, adjacent points should be correlated in terms of their expected utility. Without correlation between neighborhood points, random search would be just as effective as local search on that particular domain (see figure 2a). But if neighborhood points are correlated, then local search can use collected samples to learn and search gradients, or positive trends, in the search space in order to reach maxima.

One simple way to test the ruggedness of a particular neighborhood algorithm in a search space is to measure the average change in expected utility of a single step in the search space. For random search, adjacent points can be any two points in the search space. For genetic search, the difference in utility between two adjacent points can be measured by calculating the expected utility of the parents relative to a single offspring. For local search, two adjacent



Figures 2a,b: Graphs show a rough, less continuous space versus a smooth, more continuous space. The efficacy of local search algorithms is correlated with characteristics (including smoothness) of the search landscape they produce.

points can be a starting vector and its adapted vector.

Ruggedness does not entirely determine the smoothness of the search space. In the example of a rugged graph above, if the height was reduced by two-thirds, it would appear that the measure of smoothness for the rugged, unsearchable graph would be similar to the smoothness of the example of the smooth, easily searchable graph by the definition above. The rugged graph is still unsearchable, in spite of the proximity of neighborhood points, because of the large number of local maxima. The number of local maxima will describe the approximate order of how many peaks the landscape has, and give us a more accurate estimate of how easily the landscape can be searched.

5.2 Overall Performance

Given the same starting points, genetic search and local search made significant improvements, and performed very well compared with the best hypothesis found in an equivalent number of random samples. There were two features of the genetic search that enable a more robust search for any given starting hypothesis (see figure 3a). First, the improvements have a constant upwards slope until the plateau. Unlike the local search method, which made most of the improvement in a single search iteration (see figure 3b), genetic search had relatively consistent improvements in the mean and high scores for each generation of hypotheses throughout the entire optimization run. This consistent improvement makes genetic search a more useful anytime algorithm, and when CPU resources are limited, a small number of iterations will most likely be more effective than a few steps of local search optimization.

Second, the mean of each generation is much closer to the high score, meaning that the average of the entire generation as a whole has improved. This illustrates that the landscape created using genetic search is searchable in practice, as siblings from high scoring parents are much more tightly clustered around a similarly high score than the sibling offspring from high scoring parents using local search.

These preliminary results show the promise of using a genetic algorithm for an anytime adaptation algorithm. Further experiments are being produced to confirm and expand on this hypothesis.

Domain	**Random Search**		**Local Beam Search**		**Genetic Search**	
	Mean	**Std Dev**	**Mean**	**Std Dev**	**Mean**	**Std Dev**
Comet Lander	0.0435	0.0293	0.0086	0.0066	0.0134	0.0093
EO-1	0.0442	0.0466	0.0114	0.0331	0.0145	0.0244

Table 2: Smoothness property of each search algorithm for both domains. The mean shows the average change in expected utility between two steps of the search, which is a measure of the smoothness of the step function

6 Related Work

Evaluating decision-making strategies is a growing research topic, although many of the methods employed in the related work do not rely on any evolutionary-based computational strategies. Horvitz originally described a method for evaluating algorithms based on a cost versus quality tradeoff [Horvitz 1988]. Russell et al. used dynamic programming to rationally select among a set of control strategies by estimating utility [Russell et al. 1993]. The MULTI-TAC system considers all k-wise combinations of heuristics for solving a CSP in its evaluation which also avoids problems with local maxima, but at a large expense to the search [Minton 1996]. Previous articles describing adaptive problem solving have discussed general methods for transforming a standard problem solver into an adaptive one [Gratch & DeJong 1992, 1996], illustrated the application of adaptive problem solving to real world scheduling problems [Gratch & DeJong 1996], and showed how adaptive problem solving can be cast as a resource allocation problem [Chien 1999]. We have expanded on these topics by evaluating methods for generating hypotheses from biologically-inspired algorithms, and using adaptive problem solving to evaluate those candidate hypotheses based on an efficient estimate of their utility and cost, considered separately in the ranking process.

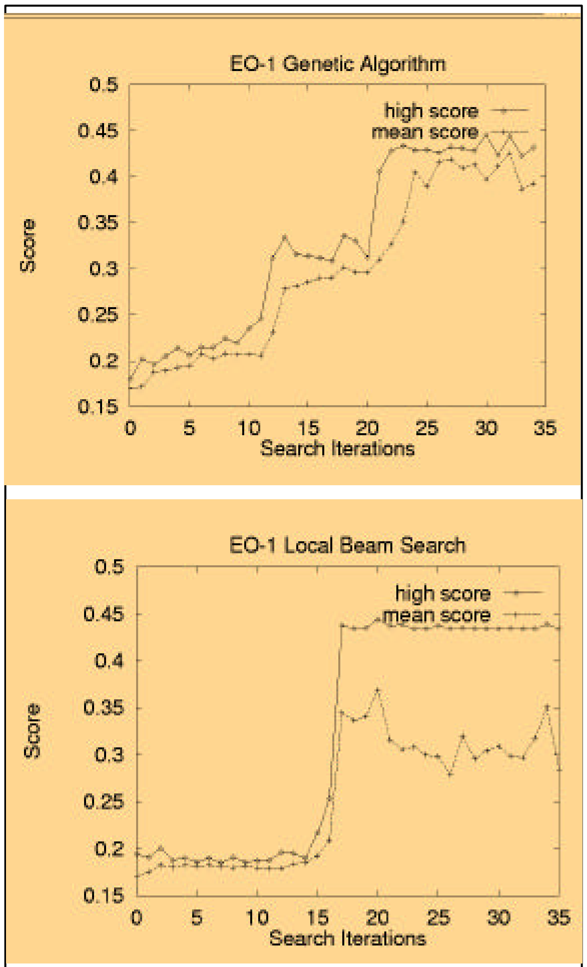


Figure 3a,b: Preliminary results of the improvements made by biologically inspired search on a planning domain. Each point represents the estimated value of the high and mean scores for one generation of hypotheses, which is why the maximum can be seen to decrease occasionally.

7 Future Work

We will continue to perform landscape analysis to measure and characterize the epistatis of the search space, the number of local maxima, and the average distance between local maxima (to test for the "big valley" hypothesis). We are also looking into meta-level learning techniques such as landscape mapping functions and multi-restart techniques in order to better evaluate search strategy techniques.

Alternative directions include determining how to adapt the domain model itself as the domain and operations shift, designing and maintaining safe operations during behavior adaptation, and testing in situ behavior adaptation on an autonomous craft.

8 Acknowledgements

The research described in this paper was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

9 References

Bernstein, S.N. (1946). *The Theory of Probabilities*, Gastehizdat Publishing House.

Chien, S., Rabideau, G., Knight, R., Sherwood, R., Engelhardt, B., Mutz, D., Estlin, T., Smith, B., Fisher, F., Barrett, T., Stebbins, G., Tran, D. 2000. "ASPEN – Automating Space Mission Operations using Automated Planning and Scheduling." *SpaceOps 2000*, Toulouse, France.

Chien, S., Stechert, A., Mutz, D. 1999. "Efficient Heuristic Hypothesis Ranking." *Journal of Artificial Intelligence Research* Vol 10: 375-397.

Chien, S., Gratch, J., Burl, M. 1995. "On the Efficient Allocation of Resources for Hypothesis Evaluation: A Statistical Approach." In *Proceedings of the IEEE Transactions on Pattern Analysis and Machine Intelligence* 17(7), p. 652-665.

Dubrawski, A., Schneider, J., "Memory Based Stochastic Optimiation for Validation and Tuning of Function Approximators" 6th Intl Workshop on Artificial Intelligence and Statistics, Fort Lauderdale, Florida. 1997

Goldberg, D., *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley Publishing Company, USA. January1989.

Gratch, J; DeJong, G. 1996. A Decision-Theoretic Approach to Solving the EBL Utility Problem. In *Artificial Intelligence*.

Gratch, J. DeJong, G. 1992. COMPOSER: A Probabilistic Solution to the Utility Problem in Speed-up Learning. In *Proceedings of the Tenth National Conference on Artificial Intelligence*.

Hagerup, T. and Rub, C. (1990). A Guided Tour of Chernov Bounds, Information Processing Letters 33: 305-308.

Hoeffding, W. (1963). Probability inequalities for sums of bounded random variables, Journal of the American Statistical Association 58(301):13-30

Horvitz, E. 1988. Reasoning under Varying and Uncertain Resource Constraints. *Proceedings of the Seventh National Conference on Artificial Intelligence*, 111-116.

Maron 0., Moore, A., "Hoeffding Races: Accelerating Model Selection Search for Classification and Function Approximation," *Advances in Neural Information Processing Systems 6,* Morgan Kaufmann, 1993.

Minton, S. 1996. Automatically Configuring Constraint Satisfaction Programs: A Case Study. In *Constraints* 1:1(7-43).

A. Nádas (1969), "An extension of a theorem of Chow and Robbins on sequential confidence intervals for the mean," The Annals of Mathematical Statistics 40:2, pp. 667-671, 1969.

Russell, S., Subramanian, D., and Parr, R. 1993. Provably Bounded Optimal Agents. In *Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence*.

Sherwood, R., Govindjee, A. Yan, D., Rabideau, G., Chien, S., Fukunaga, A. 1998. Using ASPEN to automate EO-1 Activity Planning. In *Proceedings of the 1998 IEEE Aerospace Conference*.

Stadler, P., "The Structure of Fitness Landscapes," *Proc. of International Workshop on Biological Evolution and Statistical Physics*, Max Planck Institute for Complex Systems Physics, May 2000.

Valiant, L. "A Theory of the Learnable," *Communications of the ACM*, 1984.

Taalagrand, "A new isoperimetric inequality and the concentration of measure phenomenon," Lecture Notes in Mathematics, 1469: 94–124, Springer, 1991.

Zweben, M. Daun, B., Davis, E., and Deale, M. 1994. Scheduling and Rescheduling with Iterative Repair. In Zweben, M., and Fox, M., eds., *Intelligent Scheduling*. San Francisco, CA: Morgan Kaufmann. 241-256.