'Guided' Restarts Hill-Climbing

D. Catteeuw, M. Drugan, and B. Manderick

Artificial Intelligence Lab, Vrije Universiteit Brussel, Pleinlaan 2, 1050 Brussels, Belgium
{dcatteeu,mdrugan,bmanderi}@vub.ac.be

1 Introduction

Hill-climbing is a well-known and simple local search method for discrete optimization problems. It starts with a randomly chosen feasible solution, then repeatedly and greedily selects the current solution's best neighbor until no improvement is possible—a local optimum is reached. A drawback of hill-climbing is that the quality of the resulting solution is highly dependent on the initial one. One solution is 'random restarts hill-climbing:' repeatedly restart hill-climbing and return the best solution found so far when no more time remains.

Drawing initial solutions *uniformly* from the search space is inefficient when some local optima have large basins. We propose a solution which selects initial solutions more intelligently: 'guided restarts hill-climbing.' The algorithm divides the solution space in several regions and learns which regions are worth exploring further and which are not.

2 Multiarmed Bandits

The problem of selecting a region to explore is similar to a multiarmed bandit—a sequential decision making problem, where an agent must repeatedly select one out of m actions with unknown reward distribution. The agent's goal is to maximize his total reward. He faces the so-called 'exploration-exploitation dilemma.' Should he exploit what he thinks is the best arm and possibly loose out on an even better arm; or explore further hoping to discover a better arm but risking to get only mediocre rewards?

Many learning algorithms address this problem. Some are simple but effective in practice (ϵ-greedy, softmax, and their variants [1]) and often outcompete more complicated algorithms (UCB1, EXP3, and their variants [2, 3]) even though these have theoretical performance guarantees. As an example, we explain ϵ-greedy Q-learning [4]. It estimates the expected reward of each arm with the exponentially weighted moving average of the observed rewards and stores this in each arm's Q-value. This can be calculated iteratively by incrementing the previous value q_i with $\alpha(r - q_i)$, where α is the learning rate and r is a new reward for arm i. Selecting arms in ϵ-greedy fashion means: selecting the arm with the highest Q-value with high probability $(1 - \epsilon)$, and selecting a random arm with small probability (ϵ, which a parameter of the algorithm). A nice characteristic of Q-learning is that it can quickly track changes in non-stationary reward distributions and its parameters are easy to tune ($\alpha = \epsilon = 0.1$ usually just works).

3 Guided Restarts

We model the problem of selecting regions from the solution space as a multiarmed bandit. Here, we explain the algorithm using ϵ-greedy Q-learning, but the idea of 'guided restarts' is independent of the choice of multiarmed bandit algorithm.

1. Parameters: learning rate α and exploration rate ϵ.
2. The solution space is divided in m regions, each of which corresponds to an arm $i = 1, \ldots, m$. All Q-value are initialized with $Q_0 = 1$.
3. The set of discovered local optima is empty.
4. Repeat the following until no more time remains:
   (a) With probability ϵ select an arm at random, with probability $1 - \epsilon$ select the arm with the highest Q-value (ties are broken at random).
   (b) Select a random solution from the corresponding region.

(c) Perform hill-climbing starting with that solution.
(d) If the resulting local optimum is the best one seen, store it as the best solution.
(e) If the resulting local optimum was never seen before, the selected arm i is rewarded with $r = 1$, otherwise $r = 0$; the arm's Q-value is updated: $q_i \leftarrow q_i + \alpha(r - q_i)$; and the local optimum is added to the set of discovered local optima.

5. Return the best solution.

4 Applied to the Quadratic Assignment Problem

We applied our algorithm to an instance of the quadratic assignment problem ('Random 12' from [5]): $n = 12$ objects must be assigned to a location, the flow between these object are given in a matrix F, and the distance between each two locations in a matrix D. A solution is represented by permutation p of n numbers. The goal is find the assignment p^* that minimizes the sum of the products of distance and flow between each two objects: $p^* = \text{argmin}_p \sum_{i=1}^{n} \sum_{j=1}^{n} F(i,j)D(p(i),p(j))$, where $p(i)$ represents the location of object i.

There are many ways to divide the solution space. We chose an object i at random and each arm $j = 1, \ldots, n$ assigned object i to a location j: $p(i) = j$. We then simply applied the above algorithm with $\alpha = \epsilon = 0.1$.

Fig. 1 shows how the cost of the best solution found (averaged over 1000 simulations) decreases with the number of restarts. Our algorithm requires a short learning phase (30 to 40 restarts) before it can perform better than random restarts. We are convinced that our method can outperform random restarts on more difficult instances where thousands of restarts are needed to find the optimal solution.

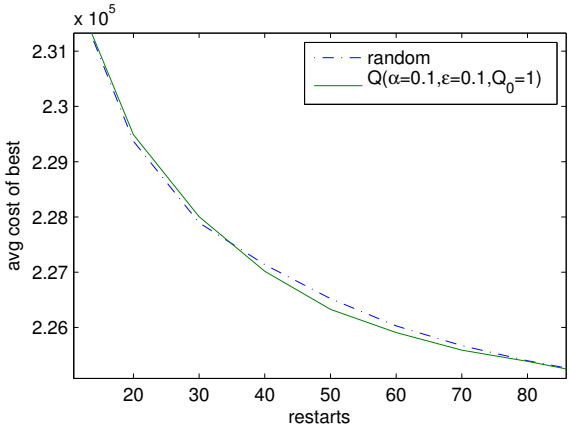

Fig. 1. The cost of the best solution found decreases (averaged over 1000 simulations) with the number of restarts. After some initial learning phase, guided restarts (solid green line) performs slightly better random restarts (dashed blue line).

References

1. J. Vermorel and M. Mohri, "Multi-armed Bandit Algorithms and Empirical Evaluation," in *Proceedings of the 16th European Conference on Machine Learning* (J. a. Gama, R. Camacho, P. Brazdil, A. Jorge, and L. Torgo, eds.), Lecture Notes in Computer Science, (Porto, Portugal), pp. 437–448, Springer-Verlag, 2005.
2. P. Auer, N. Cesa-Bianchi, Y. Freund, and R. E. Schapire, "The Nonstochastic Multiarmed Bandit Problem," *SIAM Journal on Computing*, vol. 32, no. 1, pp. 48–77, 2003.
3. P. Auer, N. Cesa-Bianchi, and P. Fischer, "Finite-time Analysis of the Multiarmed Bandit Problem," *Machine Learning*, vol. 47, no. 2, pp. 235–256, 2002.
4. C. J. C. H. Watkins, *Learning from Delayed Rewards*. Phd thesis, Cambridge University, 1989.
5. E. Taillard, "Robust taboo search for the quadratic assignment problem," *Parallel Computing*, vol. 17, pp. 443–455, July 1991.