Game Playing Techniques for Optimization Under Uncertainty

Kiyan Ahmadizadeh Carla P. Gomes Ashish Sabharwal
Dept. of Computer Science, Cornell University, Ithaca NY, USA
{kiyan,gomes,sabhar}@cs.cornell.edu*

Extended Abstract

1 Introduction

This abstract describes ongoing work on the application of Upper Confidence Bounds applied to Trees (UCT) [11], a specific Monte Carlo Tree Search (MCTS) algorithm with many successful applications in game-playing (most famously as the first human expert level player for Go), to more general optimization under uncertainty. The field of *Computational Sustainability* is rich with such problems of pressing concern [9]. Climate change mitigation, infectious disease epidemic control, management of alternative energy policies, and wildlife reserve management are just a few examples from the field that require managers to make decisions using the guidance of complex stochastic simulations [2, 12].

We apply an initial implementation of UCT to an optimization problem with uncertainty where decisions are made in a single stage. This problem involves the conservation of a certain bird species in North Carolina, called the Red-Cockaded Woodpecker (RCW) [7], in collaboration with The Conservation Fund. Relatively recent work on this problem has used *Sample Average Approximation* (SAA) techniques employing Mixed Integer Programs (MIP), with impressive results [1, 13] that provide stochastic upper and lower bounds on the true optimum value [10]. The scalability of this MIP-based optimization technique is, however, limited to only a few dozen stochastic samples, which limits the accuracy to which the true stochastic nature of the dispersal process and survival rate can be captured. UCT offers a promising alternative approach for optimizing over a much larger set of stochastic samples, along with guarantees of convergence to optimality in the limit. In practice UCT produces approximate solutions, and we evaluate these approximations by comparing their solution quality to bounds on the optimum found using SAA techniques in past work [1]. UCT also opens up the possibility of "plugging in" sophisticated simulation models for species movement and survival, essentially as a black box. Finally, UCT is naturally suited to the multi-stage setting as well, where conservation actions are taken at multiple time points in a "reactive" manner based on species response.

2 UCT for Single-Stage Optimization Under Uncertainty

We consider an agent that can take actions from a set $\mathcal{A} = \{a_i...a_M\}$ that influence some stochastic utility, where c_i is the cost of action i and B is the budget available to the agent. An *action set* $a \subseteq \mathcal{A}$ has cost $C(a) = \sum_{a_i \in a} c_i$. We assume a random function $\mathbf{Q}(a, \boldsymbol{\omega})$ that, given an action set a and a random variable $\boldsymbol{\omega}$, gives the utility of action set a. The agent wishes to take actions that maximize its expected utility:

$$\mathbf{max}_{a \subseteq \mathcal{A}} \; \mathbb{E}[\mathbf{Q}(a, \boldsymbol{\omega})] \; \mathbf{s.t.} \; C(a) \leq B$$

In practice, we assume a function $Q(a, \omega)$ is available that, given a pseudo-random number ω, evaluates the utility of a using some simulation. We note that in general our formulation could be extended to a *multi-stage* setting, where multiple rounds of decisions that affect stochastic outcomes are made.

*This work was supported by NSF Expeditions in Computing award for Computational Sustainability (grant 0832782), NSF IIS award (grant 0514429), and IISI, Cornell University (AFOSR grant FA9550-04-1-0151).

Here we give a presentation of UCT as it applies to single-stage optimization under uncertainty [11]. UCT is a specific instance of Monte Carlo Tree Search (MCTS) [4], an algorithm that performs stochastic sampling on its search tree. The tree's root is the node N_θ labeled with the empty action set, and node N_a has the set of children: $\{a' \mid \exists a_j \notin a : a' = a \cup a_j \;\; \textbf{and} \;\; C(a') \leq B\}$ (i.e., node N_a has children labeled with action sets a' extended by one action that do not violate the budget conditions). If a node N_a has no children, we call it *terminal*, and if it has yet to be sampled by the algorithm, the node is a *leaf*. Let $P(N_a)$ be the parent of N_a. Nodes are associated with some value, $V_t(N_a)$, given by some aggregation of the utility values sampled at or below N_a up to iteration t.

MCTS is an iterative-deepening style best-first search that, when traversing the tree, selects child nodes with best $V_t(N_a)$ until a leaf or terminal node is reached. A stochastic sampling of the utility of the leaf or terminal is then generated and used to update $V_t(N_a)$ for all nodes N_a selected on the path from the root. The search performs iterations from N_θ until the value at the root meets some convergence criteria. Commonly, $V_t(N_a)$ is the average of samples taken at or below the node N_a up to iteration t. UCT makes the keen insight that a node in the search tree and its children form a *multi-armed bandit*, and uses the UCB1 multi-armed bandit selection strategy [3] as the selection criteria for Monte Carlo Tree Search. The strategy chooses the child node with best value: $V_t(N_a) = \bar{x}_{a,t} + c_{T(P(N_a),t),T(N_a,t)}$ where $c_{t_1,t_2} = 2 * C_p * \sqrt{\frac{\ln t_1}{t_2}}$. Here, $\bar{x}_{a,t}$ is the average of the utilities sampled at or below node N_a up to time t, $T(N_a, t)$ is the number of times node N_a has been sampled up to iteration t, and $C_p > 0$ is an arbitrary constant. The term c_{t_1,t_2} is an upper-confidence bound on the sampled average, based on the Chernoff-Hoeffding bound. The constant C_p controls the exploration-exploitation tradeoff of the search by giving more or less weight to the upper confidence bound.

It is known that for an isolated multi-armed bandit, the UCB1 selection strategy always converges to selecting the arm with optimal distribution exponentially more often than any sub-optimal arm [3]. Kocsis and Szepesvári [11] have shown that this property still holds in a Monte Carlo Search Tree using the UCB1 policy as the selection rule for each node of the search tree. As a result, the convergence of the UCT algorithm guarantees that the selection policy has converged to always choosing the optimal path in the search tree (which can then be traversed to retrieve the optimal solution).

We note that in the worst case, UCT may take exponential time to converge to the true optimum [6] and that in many cases the algorithm will timeout before this occurs. Empirical validation and parameter tuning is thus an essential part of any successful application of UCT.

3 Experimental Results and Ongoing Work

We implemented UCT for single-stage stochastic decision problems in C++ and applied it to the problem of conserving the Red-Cockaded Woodpecker (RCW), an endangered species with populations on the East Coast of the United States [7]. For lack of space, we refer the reader to recent work [1, 13] for details on the RCW problem. In short, the RCW can only survive in land *territories* (grouped into *parcels* for purchase) that The Conservation Fund purchases and maintains as reserves. The goal of The Conservation Fund, operating with a fixed budget, is to maximize RCW population over a large time horizon (∼100 years).

Sheldon et al. [13] used a *patch model* that, given an initial RCW population on a geographic map of territories, predicts which territories become colonized by the RCW over time. The properties of this model allow for the pre-computation of idealized simulations which can then be transformed, given a set of purchased parcels, into a simulation under the real-life reserve constraints (i.e., taking into account which territories are actually bought and preserved). This transformation can be succinctly encoded within a MIP framework, a property that is used by Sheldon et al. to apply the Sample Average Approximation (SAA) method [10] to obtain stochastic upper and lower bounds on the true optimum value of the problem. The SAA method here works by repeatedly solving MIPs formed from a finite (typically 10-20) sample of simulations, which each MIP optimizes for maximizing the expected RCW population over the sampled simulations. The average objective over all MIPs provides a stochastic upper bound on the true optimum, and the independent evaluation of the MIP solution that scores best on a small test set provides a stochastic lower bound [10]. We note that for many other applications, simulations do not have natural succinct encodings as MIPs or larger sample sizes are required, and thus an SAA method employing MIPs would not be practical.

The SAA method employing MIPs (SAA-MIP) as applied in [1] solves 100 MIPs each formed from 10

Table 1: Evaluation and runtime statistics across various budgets.

Budget	SAA-MIP UB	SAA-MIP LB	Mean SAA-MIP RT (s)	Best SAA-UCT Quality	Mean SAA-UCT RT (s)
0.5	51.284	49.387	16.07	47.454	260.11
0.10	67.779	64.998	17.84	64.339	335.28
0.15	76.647	73.031	10.50	73.282	375.27

simulations of RCW distribution over 20 years on a small map representing a region of North Carolina. To compare UCT to this method, we used UCT in an SAA procedure (SAA-UCT) that uses the results of 100 UCT runs, each optimizing over a finite set of 10 simulations (and using a constant $C_p = 1$). Each run for SAA-UCT was given 100,000 iterations before the best solution found was taken. We note SAA-UCT does not provide the same stochastic bounds as SAA-MIP, since UCT provides an approximate solution over each finite set of samples. The 100 solutions found by both methods are evaluated on a test set of 100 simulations; the solution with best test-score is then evaluated on an additional 1000 simulations.

Table 1 compares stochastic bounds on the optimum found by SAA-MIP with the quality of the solution with the best test score from SAA-UCT. We also provide the mean runtimes for performing one iteration for each SAA procedure (either one run of UCT or solving one MIP). The solutions are for problems given an increasing *proportion of the total cost of all parcels* as the budget. The results show that the solution with best test score found by SAA-UCT has an evaluated quality close to, or better than, the lower bound found by SAA-MIP. Since the bounds found by SAA-MIP are close, this indicates that SAA-UCT finds approximate solutions close to the true optimum. This shows that UCT can provide good results on optimization problems with uncertainty that lie outside the game-playing context typically associated with UCT.

SAA-UCT is, however, slower than SAA-MIP and we are working on improving the runtime. For example, tuning the exploration-exploitation parameter C_p can have a drastic impact on performance [8]. Observing that our search-space is in fact a directed-acyclic graph, modifications to UCT can be made to take advantage of this structure [5]. The eventual goal is to expand UCT to a multi-stage optimization setting and optimization involving more complex simulations from the sustainability field. The performance of UCT, as demonstrated both here and for game-playing, indicate that UCT could be successful in finding solutions to such extremely challenging problems.

References

[1] K. Ahmadizadeh, B. Dilkina, C. P. Gomes, and A. Sabharwal. An empirical study of optimization for maximizing diffusion in networks. In *CP-2010: 16th Intl. Conf. on Principles and Practice of Constraint Programming*, St Andrews, Scotland, Sept. 2010. To appear.

[2] R. Anderson and R. May. *Infectious diseases of humans: dynamics and control.* Oxford University Press, 1992.

[3] P. Auer, N. Cesa-Bianchi, and P. Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, 47 (2-3):235–256, 2002.

[4] G. Chaslot, S. Bakkes, I. Szita, and P. Spronck. Monte-carlo tree search: A new framework for game ai. In *AIIDE*, 2008.

[5] B. E. Childs, J. H. Brodeur, and L. Kocsis. Transpositions and move groups in monte carlo tree search. In P. Hingston and L. Barone, editors, *IEEE Symposium on Computational Intelligence and Games*, pp. 389–395. IEEE, December 2008. ISBN 978-1-4244-2974-5. URL `http://www.csse.uwa.edu.au/cig08/Proceedings/papers/8057.pdf`.

[6] P.-A. Coquelin and R. Munos. Bandit algorithms for tree search. *CoRR*, abs/cs/0703062, 2007.

[7] U. Fish and W. Service. Red-cockaded woodpecker recovery plan. U.S. Fish and Wildlife Service, Southeast Region, Atlanta, GA, 2003.

[8] S. Gelly and D. Silver. Achieving master level play in 9 x 9 computer go. In *AAAI*, pp. 1537–1540, 2008.

[9] C. P. Gomes. Computational Sustainability: Computational methods for a sustainable environment, economy, and society. *The Bridge, National Academy of Engineering*, 39(4), Winter 2009.

[10] A. Kleywegt, A. Shapiro, and T. Homem-de Mello. The sample average approximation method for stochastic discrete optimization. *SIAM Journal on Optimization*, 12(2):479–502, 2002.

[11] L. Kocsis and C. Szepesvári. Bandit based monte-carlo planning. In *ECML*, pp. 282–293, 2006.

[12] E. McDonald-Madden, P. W. Baxter, and H. P. Possingham. Making robust decisions for conservation with restricted money and knowledge. *Appl. Ecol.*, 45:1630–1638(9), 2008.

[13] D. Sheldon, B. Dilkina, A. Elmachtoub, R. Finseth, A. Sabharwal, J. Conrad, C. P. Gomes, D. Shmoys, W. Allen, O. Amundsen, and B. Vaughan. Maximizing spread of cascades using network design. In *UAI-2010: 26th Conference on Uncertainty in Artificial Intelligence*, pp. 517–526, Catalina Island, Avalon, CA, July 2010.