Simulated Annealing Monte Carlo Tree Search for large POMDPs

Kai Xiong
Southwest University of Science and Technology
Mian Yang, Si Chuan 621010, China
xiongkaipai@163.com

Hong Jiang
Southwest University of Science and Technology
Mian Yang, Si Chuan 621010, China
228662865@qq.com

***Abstract*—Many planning and control problems can be modeled as large POMDPs, but very few can be solved scalably because of their computational complexity. This paper proposes a Simulated Annealing based on the Monte Carlo Tree Search for large POMDPs. The proposed algorithm determines an acceptance probability of sampling a back-propagation's outcome in the simulated annealing process. The experiments show that the proposed SAMCTS (Simulated Annealing Monte Carlo Tree Search) outperforms the original Simulated Annealing algorithm when applied to a large POMDP benchmark problem.**

Keywords- Simulated Annealing, MCTS, POMDPs.

I. INTRODUCTION

The Partially Observable Markov Decision Process (POMDP) model is a general description regarding sequential decision problems in partially observable environments, and provides a framework for agents to search for rational solutions [1]. In this framework, agents cannot directly observe an underlying state in unknown environment. Instead, they must maintain a probability distribution over the set of possible states, based on a set of observations and observation probabilities. When agents receive the observation, they take corresponding actions, which change the state and lead to rewards. Agents' goal is to devise a policy for action selection that maximizes the total rewards.

However, solving large POMDPs is very difficult due to the computational complexity. The solutions to POMDPs are generally divided into two categories: online planning and offline planning. In this paper, we focus on online planning. The classic online planning [2] is not suitable for large problems due to the curse of dimensionality and history.

Simulated annealing is a popular meta-heuristic search, in which a crystalline solid is heated and then allowed to cool slowly until it achieves its most regular possible crystal lattice structure. If the cooling schedule is sufficiently slow, the final configuration results in a solid with structural integrity [3]. Simulated annealing establishes a connection between this type of thermodynamic behavior and the search for global minima for a discrete optimization problem. However, when the problem become large and mess, the original Simulated Annealing is helpless [4].

Recently, the Monte Carlo method provides a feasible solution for overcoming the curses [5]. An outperforming Monte Carlo method—Monte Carlo Tree Search (MCTS) shows a great potential to deal with the high dimensional POMDPs [6]. To date, several approaches have been proposed to search policies by using MCTS. Specifically, the POMCP algorithm combines the MCTS with the current belief state [7], which achieves a high performance with no prior knowledge. The Rapid Action Value Estimation (RAVE) MCTS has successfully been used for the MoGo [8]. The algorithm builds a search tree in which each node estimates the value combined AMAF (All Move As Frist).

The MCTS's selection is done by UCT formula, which determines a constant term in the exploration process to capture a tradeoff between the exploitation and exploration [6]. However, the value of the parameter is fixed in the existing MCTS selection, which results in poor adaptability for some problems.

This paper proposes a simulated annealing algorithm based on MCTS, which is referred to as SAMCTS (Simulated Annealing MCTS). A tree search framework can help Simulated Annealing to deal with large POMDPs.

The reminder of the paper is organized as follows. Section II introduces the background about the simulated annealing and the MCTS. In Section III, we propose our main SAMCTS, which samples the back-propagation outcome by simulated annealing criterion. Section IV presents experimental results on a benchmark problem. A summary of the contributions can be shown in Section V.

II. BACKGROUND

A. The POMDPs

A POMDP consists of a tuple $\{S, A, O, T, \Omega, R, \gamma\}$. S, A, and O are set of states, actions, and observations, respectively. T defines a transition probability $P(s_t \mid s_{t-1}, a_{t-1})$, which gives the probability that a agent lies in s_t, after taking action a_{t-1} in state s_{t-1}; R denotes the reward agents gotten by taking action A in S. Ω is an observation probability $P(o_t \mid s_t, a_{t-1})$; γ is a discount factor [9].

The goal of POMDP planning is to compute an optimal policy π^* that maximizes the agent's expected total reward. A policy π maps a belief to an action $\alpha \in A$, which induces a value function $V^\pi : \beta \rightarrow R$. The value of b is the agent's expected total reward of executing π with belief $b: V^\pi(b) = \mathrm{E}(\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t) \mid \pi, b)$. An optimal policy π^* is one that maximizes V^π to get V^*

$$V^*(b_t) = \max\left\{ r(b_t, a_t) + \gamma \sum_{o_{t+1}} P(o_{t+1} \mid b_t, a_t) V^*(b_{t+1}) \right\} \tag{1}$$

This formula is the optimal equation for POMDP.

B. Monte Carlo Tree Search

MCTS is generally followed by four steps: Selection, Expansion, Rollout, and Back-propagation [6].

1) Selection: starting from the root, the parents use UCT to select the most expandable child [6]. The policy is recursively applied to descend through the tree until the most urgent expandable child is reached. The expanded child will maximize the q_i in the UCT formula (2)

$$q_i = \overline{x}_i + c \times \sqrt{\frac{\log(n_p + 1)}{n_i}} \tag{2}$$

Where x_i denotes the average score of node i, n_i and n_p , stand for the visited times of the child i and parent p, respectively. The scalar constant c is a tradeoff between the exploration and exploitation [6].

2) Expansion: some nodes are randomly selected from the subset of children which are not added to the tree yet.

3) Rollout: stochastic moves are played, from the position that the newly added node, until the game over or the time expires.

4) Back-propagation: the outcome of the Rollout is propagated back along the previously traversed path up to the root node.

These four phases are repeated until time runs out or the terminal state is reached. When the search is finished, it returns the most robust child with the best tactic.

Furthermore, an improved MCTS algorithm named the RAVE (Rapid Action Value Estimation) is used in the game Go, which gives good results [6]. It combines two values: the original and the AMAF (All Moves As First). The basic idea of AMAF is to update statistics for all actions selected during a simulation as if they were the first action applied.

The RAVE uses a weight coefficient β to determine the tradeoff between the two values, which can be expressed as:

$$\beta = \frac{\overline{n}}{n + \overline{n} + 4bn\overline{n}} \tag{3}$$

Where $\overline{n}$ and n denote the visited counts got by the AMAF and original ones, respectively. b is the RAVE bias.

To estimate the overall value of action a in state s, the weight sum $Q^*(s,a)$ combines the original value $Q(s,a)$ and the AMAF value $\overline{Q}(s,a)$, which is satisfied by:

$$Q^*(s,a) = (1-\beta)Q(s,a) + \beta\overline{Q}(s,a) \tag{4}$$

C. Simulated Annealing

Simulated annealing is a probabilistic method for optimization that may pass through several local optimums [3]. At the core of the simulated annealing algorithm is the Boltzmann distribution. At time n, simulated annealing samples from a distribution given by [3].

$$A_n^f(dx) = \frac{1}{D_n}\exp\left(-\frac{f(x)}{T_n}\right)dx \tag{5}$$

Where f is the fitness function, D_n is a normalizing factor, T_n is a sequence of temperatures with monotone decreasing by a discount cooled parameter β. The sequence T_n is known as the cooled schedule.

The simulated annealing samples from A_f^n repeatedly based on the Metropolis acceptance criterion [10]. The process begins with an initial solution x. At each time step, a proposal distribution Q is used to sample x_n. The proposed solution x is replaced with x_n with probability

$$P(x_n) = min\{\exp(-[f(x) - f(x_n)]/T_n),\ 1\} \tag{6}$$

Acceptance / Rejection probability is given as:

$$x_k = \begin{cases} x_{k+1} & w.p.\ \ p \\ x_k & w.p.\ \ 1-p \end{cases} \tag{7}$$

III. The Simulated Annealing MCTS

A search tree is built up guided by Monte-Carlo simulations. In the tree the nodes represent current states S and the edges represent possible actions A. When the search process is started, the root creates new nodes which stand for the current state. The tree is divided into two parts: policy subtree and rollout subtree [11]. The policy subtree updates nodes to follow the selection strategy faithfully. The rollout tree works by Monte Carlo simulation. We design the policy subtree based on simulated annealing. When back-propagation returns a reward, let the Acceptance/Rejection criterion be applied to the reward acceptance process, for judging whether the agent adopts this action. When new reward meets the accepted criterion, the action will be taken.

Unlike SA, SAMCTS explicitly considers the tree search structure to reduce the cost of Acceptance/Rejection searching. A SAMCTS model can be described as Figure. 1.

Generally, just one annealing process for large POMDP problems cannot promise the optimum outcome. It easily goes into a trap of local optimum. So we propose to adopt the repetitious simulated annealing. Specifically, when the temperature cools down to a threshold T_r, the temperature comebacks the original T_0, as shown in Figure. 2.

If an agent sinks into a local optimum, the agent can get a new stochastic power to win through the mess. This measure helps the agent avoid the local optimum traps.

Algorithm 1 shows the process of SAMCTS. In SAMCTS-Search(S_0), the policyTree is based on the Simulated Annealing to select a good child according to a node's back-propagation outcome. The rolloutTree offers the back-propagation outcome by using Monte Carlo simulation.

IV. Convergence Proof

Assume the sequence of states are $X_0, X_1, \ldots,$ X_k. A potential next state Y_k is chosen from the transition probability distribution $P(Y_k=y|X_k=x)=R(x,y)$. Then, the next state X_{k+1} can be determined as

$$X_{k+1} = \begin{cases} Y_k & w.p.\ min\{\exp(-[f(y) - f(x)]/T_n),\ 1\} \\ X_k & otherwise, \end{cases} \tag{8}$$

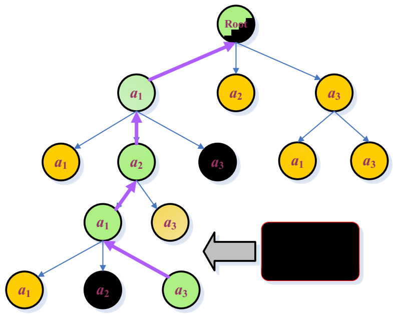

Figure 1. A process of the simulated annealing MCTS. Nodes represent POMDP states. Children are added to expand the tree, according to the available actions. A random simulation is run from the new children. The simulation outcome is backpropagated by the selected nodes to update their statistics. Nodes accept one child when the child's statistics meet the acceptance/rejection criterion.

Algorithm 1: Simulated Annealing MCTS

```
PROCEDURE   SAMCTSSEARCH($S_0$)
    create  root node $v_0$ with state $S_0$
    WHILE $S(v_i)$ is nonterminal DO
    $v_i$ := POLICYTREE ($v_0$);  $r$:= ROLLOUTTREE ($S(v_i)$, depth);
    BACKPROPAGATION($v_i$, $r$)
return  $a$(PSASAMPLE ($S(v_0)$));

PROCEDURE   PSASAMPLE ($S(v)$)
Initialize($T_0$, $T_r$, discount $\beta$, perturbation factors $\alpha$ );
WHILE $S(v)$ is nonterminal DO
    generate $T_{n+1}$ from $T_n$,$T_{n+1} = T_n \cdot \beta$;
    IF $T_n < T_r$
    THEN $T_n = T_0$;
    FOR i:=0 to L DO
        $a \in A(S(v))$;  generate $a_{i+1}$ from $a_i$
        IF $r(S(a_i))-r(S(a_{i+1})) \leqslant 0$  THEN $a_{i+1} \in nerghbor_{ai}$;
        ELSEIF
        $\exp\{-[r(S(a_i))-r(S(a_{i+1}))] / T_i\} > random[0, 1)+ \alpha$
      THEN $a_{i+1} \in nerghbor_{ai}$;
return  $nerghbor_{ai}(random(nerghbor_{ai}\_size))$;

PROCEDURE   POLICYTREE ($v$)
    WHILE $S(v)$ is nonterminal DO
      IF $v$ not expanded
        THEN return  EXPAND($v$);
      ELSE $v$ := PSASAMPLE ($S(v)$);
return $v$;

PROCEDURE   ROLLOUTTREE ($S(v_i)$, depth)
WHILE $S(v_i)$ is nonterminal && non over depth DO
    choose  $a \in A$ in uniform random
    $S := f(s, a)$;
return  reward  $r(S)$;

PROCEDURE   BACKPROPAGATION($v_i$, $r$)
$N(v)$ := $N(v) + 1$;  $Q(v)$ := $Q(v) + r$;
$v$ := parent of $v$;
```

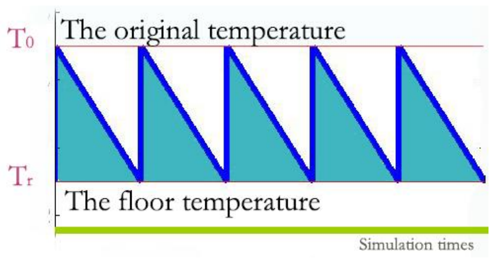

Figure 2. The repetitious simulated annealing steps as follow: Initial temperature T_0, then the temperature decreases progressively with increasing the simulation times. When temperature arrived the floor point, the numerical of temperature returned the original. A state would store the current temperature for Acceptance / Rejection probability.

The random process $X = (X_k : k \geqslant 0)$ produced by the algorithm is a discrete time Markov chain. After one-step transition, X_k will go into its neighbors $n(X_k)$. The transition probability at step k is

$$p_k(\mathrm{x},\mathrm{y}) = p[X_{k+1} = y \mid X_k = x] = \begin{cases} 0 & \text{if } y \notin n(X_k) \\ R(x,y)\cdot\exp(-[\frac{f(y)-f(x)}{T_k}]) & \text{if } y \in n(X_k) \end{cases} \quad (9)$$

Once a temperature drops to T_r, it will turn back to the original T_0. So, the temperature is confined in a range $(T_r, T_0]$ in the repetitious simulated annealing process.

A T_k is bounded in $(T_r, T_0]$, therefore, we can assume the T_k is a constant c, the X has a stationary one-step transition probability P_k. Obviously, if no annealing algorithm is applied on process, the X_k just chooses a best reward X_{k+1} of the set $n(X_k)$. The Markov chain is convergent. But the outcome is mostly not good.

Lemma 1: A stationary Markov chain is reversible if and only if there exists a collection of positive numbers $\pi(j)$ $, j \in \zeta$, which satisfy the detailed balance conditions

$$\pi(j)\mathrm{p}(j,k) = \pi(k)p(k,j) \qquad j,k \in \zeta$$

According to Markov Chain property, there is a probability distribution α on S, $\alpha(x)\mathrm{R}(x,y) = \alpha(y)\mathrm{R}(y,x)$. Now, we define [12]

$$H(x) = \frac{\alpha(x)\exp\left(-\frac{f(x)}{c}\right)}{\sum_x \alpha(x)\exp\left(-\frac{f(x)}{c}\right)} \quad (10)$$

Then,

$$H(x)\cdot p_k(x,y) = \frac{\alpha(x)\exp\left(-\frac{f(x)}{c}\right)}{\sum_x \alpha(x)\exp\left(-\frac{f(x)}{c}\right)}\cdot R(x,y)\cdot\exp(-[\frac{f(y)-f(x)}{c}])$$
$$= \alpha(x)\cdot R(x,y)\cdot\frac{\exp\left(-\frac{f(y)}{c}\right)}{\sum_x \alpha(x)\exp\left(-\frac{f(x)}{c}\right)} = \alpha(y)\cdot R(y,x)\cdot\frac{\exp\left(-\frac{f(y)}{c}\right)}{\sum_y \alpha(y)\exp\left(-\frac{f(y)}{c}\right)}$$
$$= H(y)\cdot p_k(y,x)$$

Lemma 2: P_k is reversible with equilibrium distribution $H(x)$. The Markov ergodic convergence theorem implies that [12]

$$\lim_{k\to\infty} P[X_k \in S^*] = \sum_{x\in S^*} H_c(x)$$

Obviously, $\sum_{x\in S^*} H_c(x) = 1$, which proofs convergence.

Above analyses are infinitely homogeneous Markov chains. However, our POMDP is a finite sequence of inhomogeneous Markov chain, which converges on the basis of controlling the temperature T_k and the transition probability. The examination of T_r, T_0 implies that the T_r should be small.

V. Experiments and Analyses

We use hide-and-seek game, the Rock Sample, to verify our algorithm. Rock Sample is a classic POMDP benchmark, which simulates rover science exploration. The task is to judge which rocks are valuable. The rocks are divided into three states: "good", "bad", and "none". In the Rock Sample, an agent takes samples of valuable rocks. The agent goal is to maximize the sampled rocks value. We consult the settings in [7] to build the rover map. A number of valuable and invaluable observations are counted for each rock. The agent chooses four basic directions: north, south, west, and east. When rock is the Good, the reward is +10, -10 for the Bad, -100 for the None. Actions that sampled rocks with more valuable observations are preferred. There are 247,809 states. For more information, we refer the interested readers to see [7].

Figure 3 plots the mean discount rewards versus the episode, which shows the performance comparisons between the proposed SAMCTS and the original Simulated Annealing (SA) in the POMDP problem.

Shown in Figure 3 is that rewards by the proposed algorithm is worse than original SA in the initial stage (before 32 simulations). However, after a short burn-in period, the SAMCTS significantly outperforms SA, which confirms the effectively of employing the MCTS framework for SA selecting the back-propagation rewards. Furthermore, Fig. 3 shows the mean reward of SAMCTS is higher than SA about 32.1%. When the reward curves flat (e.g. when simulations are larger than 512), the SAMCTS is higher than 105.4% over the SA. The reasons are that the SAMCTS can prune some outcomes by SA selecting back-propagation, which not only prunes poor ones, but also cuts good ones, because the next child is randomly selected from the good ones and the other ones which meet acceptance/rejection criterion. Obviously, this trim process is important for agents to save time and avoid local optimum traps.

VI. Conclusion

This paper proposes an improved Simulated Annealing algorithm based on MCTS framework for solving large POMDPs. Without full prior knowledge of the model, it is achieved by following the Monte Carlo trials and fails. Particularly, when the policy space becomes large and messy, the traditional Simulated Annealing may falls into pretty pass. The large POMDP problems are not easily valued by prior knowledge. The experiments exhibit how the Simulated Annealing working on MCTS for searching optimum.

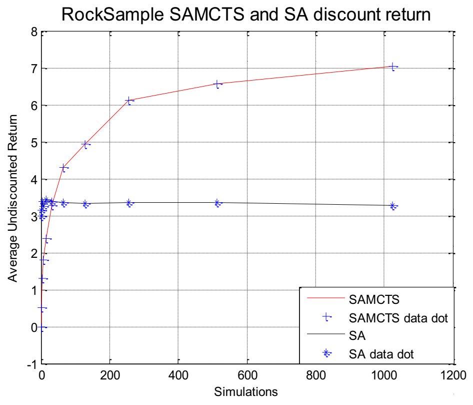

Figure 3. The mean discount rewards of SAMCTS versus SA and MCTS in Rock Sample

Acknowledgment

This work is supported by National Nature Science Foundation of China under Grant of No. 61379005.

References

[1] S. Thrun. "Monte Carlo POMDPs." In S.A. Solla, T.K. Leen, and K.-R. Müller, editors, Advances in Neural Information Processing Systems 12, pages 1064-1070, 2000.

[2] S. Ross, J. Pineau, S. Paquet & B. Chaib-draa. "Online Planning Algorithms for POMDPs." In Journal of Artificial Intelligence Research (JAIR), vol. 32, p. 663-704, 2008

[3] S. Anily, A. Federgruen. "Simulated annealing methods with general acceptance probabilities." Journal of Applied Probability, 24(3):657–667, 1987.

[4] B. Suman, P. Kumar. A survey of simulated annealing as a tool for single and multiobjective optimization. Journal of the Operational Research Society, 57, 1143-1160, 2006

[5] S. M. Ross, "Introduction to Probability Models." San Diego, CA, Elsevier, 2006.

[6] Cameron Browne, Edward J. Powley, Daniel Whitehouse, Simon M. Lucas, Peter I. Cowling, Philipp Rohlfshagen, Stephen Tavener, Diego Perez, Spyridon Samothrakis & Simon Colton. "A Survey of Monte Carlo Tree Search Methods." IEEE Transactions on Computational Intelligence and AI in Games, 4(1):1-43, 2012.

[7] David Silver, Joel Veness. "Monte-Carlo Planning in Large POMDPs." eural Information Processing Systems (NIPS), 2010.

[8] Sylvain Gelly, David Silver. "Monte-Carlo tree search and rapid action value estimation in computer Go." Artificial Intelligence, Vol. 175, No. 11, 2011.

[9] R. M. Gasic, F. Jurcicek, B. Thomson and S. Young. "Optimisation for POMDP-based Spoken Dialogue Systems" in Data-driven Methods for Adaptive Spoken Dialogue Systems, Ed. O. Lemon and O. Pietquin, Springer ISBN 978-1-4614-4803-7, 2012.

[10] Metropolis, N., Rosenbluth, A., Rosenbluth, M., Teller, A. and Teller, E. Equation of state calculations by fast computing machines. Journal of Chemical Physics, 21, 1087–1092, 1953

[11] J. (Pim) A. M. Nijssen, Mark H. M. Winands. "Monte-Carlo Tree Search for the game of Scotland Yard." IEEE Conference on Computational Intelligence and Games: 158-165, 2011

[12] Y. J. Cao, Q. H. Wu, "Asymptotic Convergence Properties of the Annealing Evolution Algorithm" UKACC International Conference on CONTROL 96, 2-5, September 1996.