Control of Non Player Characters in a Medical Learning Game with Monte Carlo Tree Search

Maxime Sanselone, Stéphane Sanchez, Cédric Sanza, David Panzoli, Yves Duthen
University of Toulouse, IRIT - CNRS UMR 5505
2 rue du Doyen Gabriel Marty, 31042 Toulouse, France
{maxime.sanselone, stephane.sanchez, cedric.sanza, david.panzoli, yves.duthen}@irit.fr

ABSTRACT

In this paper, we apply the Monte Carlo Tree Search (MCTS) method for controlling at once several virtual characters in a 3D multi-player learning game. The MCTS is used as a search algorithm to explore a search space where every potential solution reflects a specific state of the game environment. Functions representing the interaction abilities of each character are provided to the algorithm to leap from one state to another. We show that the MCTS algorithm successfully manages to plan the actions for several virtual characters in a synchronized fashion, from the initial state to one or more desirable end states. Besides, we demonstrate the ability of this algorithm to fulfill two specific requirements of a learning game AI : guiding the non player characters to follow a predefined plan while coping with the unpredictability of the human players actions.

Categories: I.2.8 [ARTIFICIAL INTELLIGENCE]: Problem Solving, Control Methods, and search

Keywords: Planning; Monte Carlo Tree Search; Serious gaming; Artificial intelligence

1. INTRODUCTION

The 3D Virtual Operating Room (3DVOR) project is a learning game aimed at training the operating room staff to communicate in the operating room. Through an immersive 3D operating room, the learners control their avatar and collaborate in a simulated operation. When the pedagogical interest of some roles is limited, an automated controller has to handle the behavior of Non-Player Characters (NPCs). The challenge here is to manage NPCs that have to react as close as possible to what a human would do, as described as fully equal partner by Thomas and al. [6].

We have chosen to focus on Monte Carlo Tree Search [1] to create the AI of the NPCs. This method combines generalization skills of evolutionary methods and the accuracy of decision trees. The algorithm only uses the authorized actions and the evaluation of final states. It creates coherent plans while being a *any-time* algorithm, capable of giving the best option in real-time [3, 2].

We show in this article how MCTS can be used to create an AI capable of controlling many NPC playing a predefined scenario the same way human players would do.

GECCO'14, July 12–16, 2014, Vancouver, BC, Canada.
ACM 978-1-4503-2881-4/14/07
http://dx.doi.org/10.1145/2598394.2598473.

2. MCTS IN 3DVOR

The method known as Monte Carlo Tree Search (MCTS) is a search method aiming to take optimal decisions in a discrete finite environment, according to a search tree generated across multiple stochastic simulations. Inspired by both the Monte Carlo and the tree search methods, it rests on the postulate that a large enough number of simulations, chosen randomly and accordingly to the previous ones, lead a trustful picture of the search space. More details about the MCTS method and its variants can be found in [1].

The resolution method we present in this article is largely inspired by the MCTS method. It makes use of random simulations to evaluate best decisions in a particular situation.

The nodes of our search space are associated to states of the environment. The different actions that any character can perform are the transitions between nodes in the MCTS.

We use a Directed Acyclic Graph topology for the search space, using transpositions (see [5]), and the Upper Confidence Bounds for Trees heuristic (UCT).

In simulation phase, because nodes are registered and updated in previous simulation phases, we can use the tree policy as default policy for each node already in the hash map instead of an uniformly random default policy.

To avoid behavioral loops and useless actions, we add two more enhancements. Any selection of a previously selected node in the same exploration cycle leads to a dead end, with a back-propagated reward value of zero. As in [4], we use a decay mechanism to favors shortest plan of actions.

The scenarii scenarios are represented with BPMN diagrams. These diagrams are used to constraint our MCTS-based approach as follows: The diagrams are interpreted to

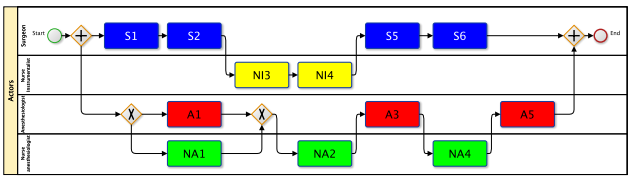

Figure 1: The learning scenario used in the experimental evaluation.

automatically generate mandatory nodes from the scenario, which are tagged as coming from the scenario and registered in the hash table. Any mandatory node has a fixed reward value, depending on the distance between the node and the initial node in the scenario. The reward of the nodes that represent the final states of the scenario has a maximal value of 1.0. Any other mandatory node receives a reward value equals to 0.99^d, with d the minimal number of needed actions to end the scenario from this node. During exploration, he rewards from simulation steps are not back-propagated to the mandatory nodes. However, their presence in the search graph from the start of the game session, and their fixed reward value, ensure that these nodes are favored by our tree policies in exploration phase and selected as best option in exploitation phase.

3. EXPERIENCES

We define a sample scenario on which we apply the MCTS. The scenario is composed of 27 actions pertaining to 8 interactive objects in the virtual environment, making in total 12 binary properties defining the world semantic representation. The vector coding a state is represented by a 4 digit integer.

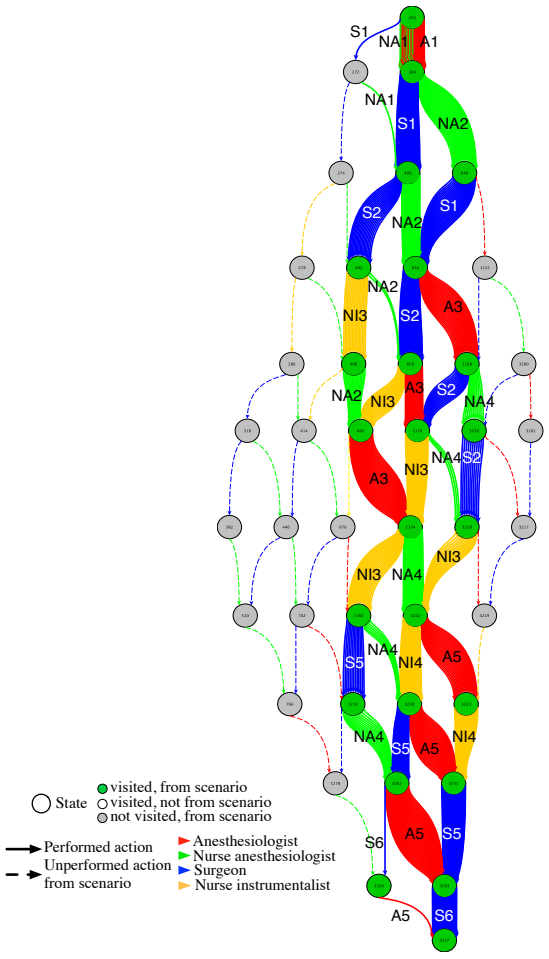

Figure 2: Performed actions in 100 gaming experiences with learning scenario.

In this scenario, the surgical team prepares a neurological operation. Two sub-teams are asynchronous and one part of the job can be completed before the other. However, the members of each sub-team must coordinate their actions to accurately follow the aimed sub-plans.

We added in the research graph precomputed paths, issued from flow sequences from the BPMN model of the learning scenario. Our approach never proposes an action that lead to a state outside of the precomputed paths of the scenario. This ensures that the non-player characters are playing accordingly to the surgical protocol and the aimed learning experience. Constraining the MCTS-based approach even helps to correct the inconsistencies of the environment modeling. The method does not propose one but several slightly different plans.

Our MCTS-based approach provides an action each time an idle NPC requests something to do. During computational time, the method performs as much as possible explorations to compute a best plan of actions from the actual state in the environment to the final state of the scenario. When any character ask for an action, the algorithm answers with the best so far evaluated action.

The last requested specification of the AI in 3DVor is that it must replace a player as a "fully equal partner". This means that the AI must acts in order to keep following the learning scenario as much as possible. In the last experiment, we still use the learning scenario previously described, but a human player plays as the surgeon.

Experiment shows that when the player deviates from the learning scenario (by playing moves out of the schedule), the AI performs as expected: as soon as possible, a NPC guides the player back in the learning scenario.

Thus simple, these last experience is promising because we can expect our MCTS-based AI to replace players and to guide them through the learning scenarios. However, we must validate our approach with larger scenarios and more players as soon as the advances in 3DVOR project allows multi-player gaming.

Acknowledgement: 3DVOR is supported by the 12th innovation cluster French funding scheme "Fonds Unique Interministériel" (FUI).

4. REFERENCES

[1] C. Browne, E. Powley, D. Whitehouse, S. Lucas, P. Cowling, P. Rohlfshagen, S. Tavener, D. Perez, S. Samothrakis, and S. Colton. A survey of monte carlo tree search methods. *Computational Intelligence and AI in Games, IEEE Transactions on*, 2012.

[2] T. Cazenave. Multi-player go. In *Computers and Games*, pages 50–59. Springer, 2008.

[3] J. Kloetzer. Experiments in Monte-Carlo Amazons. *J. Inform. Process. Soc. Japan*, 2010-GI-24(6):1–4, 2010.

[4] L. Kocsis and C. Szepesvári. Bandit based monte-carlo planning. In *Machine Learning: ECML 2006*, pages 282–293. Springer, 2006.

[5] A. Saffidine, T. Cazenave, and J. Méhat. Ucd: Upper confidence bound for rooted directed acyclic graphs. *Knowledge-Based Systems*, 34:26–33, 2012.

[6] D. Thomas and L. Vlacic. Collaborative decision making amongst human and artificial beings. In *Intelligent Decision Making: An AI-Based Approach*, pages 97–133. Springer, 2008.