Integrating Agent Actions with Genetic Action Sequence Method

Man-Je Kim
Gwangju Institute of Science and Technology
Gwangju, South Korea
jaykim0104@gist.ac.kr

Jun Suk Kim
Gwangju Institute of Science and Technology
Gwangju, South Korea
junsuk89@gmail.com

Donghyeon Lee
Gwangju Institute of Science and Technology
Gwangju, South Korea
cheetos@gist.ac.kr

Sungjin James Kim
LG Electronics
Seoul, South Korea
sj88.kim@lge.com

Min-Jung Kim
LG Electronics
Seoul, South Korea
mjung.kim@lge.com

Chang Wook Ahn*
Gwangju Institute of Science and Technology
Gwangju, South Korea
cwan@gist.ac.kr

ABSTRACT

Reinforcement learning in general is suitable for putting actions in a specific order within a short sequence, but in the long run its greedy nature leads to eventual incompetence. This paper presents a brief description and implementative analysis of Action Sequence which was designed to deal with such a "penny-wise and pound-foolish" problem. Based on a combination of genetic operations and Monte-Carlo tree search, our proposed method is expected to show improved computational efficiency especially on problems with high complexity, in which situational difficulties are often troublesome to resolve with naive behaviors. We tested the method on a video game environment to validate its overall performance.

CCS CONCEPTS

• **Theory of computation → Evolutionary algorithms**; **Bio-inspired optimization**;

KEYWORDS

Artificial Intelligence, Evolutionary Computing and Genetic Algorithms, Video Game

ACM Reference Format:
Man-Je Kim, Jun Suk Kim, Donghyeon Lee, Sungjin James Kim, Min-Jung Kim, and Chang Wook Ahn. 2019. Integrating Agent Actions with Genetic Action Sequence Method. In *Proceedings of Genetic and Evolutionary Computation Conference Companion (GECCO '19 Companion).* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3319619.3326772

1 INTRODUCTION

As the latest AI technologies constantly tackle the real world in various aspects, the methodological shift from empirical exploration to probabilistic speculation in machine learning seems rising; empirical methods require time, which does not wait for anyone. This is bad news for reinforcement learning (RL), which is based around the learner's experience, and the risk of it falling into the local maximum trap in a real-world environment is certainly intimidating. Although several studies claim that the distribution technique can handle this mess, its requirement of huge computational cost makes it virtually impractical for all but only few research institutes with substantial support and investments. We suggest that combining genetic operations with MCTS, our genuine **Action Sequence** method can resolve the RL's intrinsic problem.

In order to evaluate our model's validity, we decided to test it on a video game, which is a useful tool to simulate the real world's interactions. Fighting Game AI Competition hosted by The IEEE Conference on Games (CoG) is a great exhibition to witness how much modern game AIs have advanced so far. It is a competition based on *FightingICE*, a real-time, person-to-person combat video game where player characters battle each other in a limited physical space. Under real-time setting, each player is given the updated current game status every 16.67 ms, which is apparently far from sufficient to take every possible action into consideration for the next move. Therefore, potent FightingICE AIs are engineered to efficiently find a large set of optimized actions within each limited time frame. Such high complexity in developing AIs for FightingICE makes it essentially a real-world problem, if a bit simplified. Action Sequence, optimized via genetic operators[1], was attached to MCTS, with which our FightingICE AI was tested. We measured its performance against other AIs under the official competition settings and rules. The result showed that the method indeed improves the action searching efficiency.

2 THEORY

RL's greedy tendency of choosing bigger and physically closer rewards helps achieve proximal goals, but the learner's farseeing ability suffers from castigation. This is a fatal defect in handling real-world problems where even small changes often cause unexpected consequences. Although, in FightingICE, optimizing the agent's action for rather simple, fragmentary short-term goals is prioritized, making sure that it possesses any form of an integrated and compound plan should also be a significant concern. For example, initiating a combination of actions in a certain, complex series could constantly pressure the opponent into changing its combat policy,

*The corresponding author

GECCO '19 Companion, July 13–17, 2019, Prague, Czech Republic
© 2019 Copyright held by the owner/author(s).
ACM ISBN 978-1-4503-6748-6/19/07…$15.00
https://doi.org/10.1145/3319619.3326772

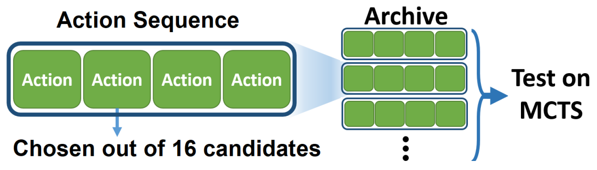

Figure 1: General Action Sequence Structure.

bringing confusion to its intended mechanism and eventually ruining it. In a fundamental sense, our method aims to assist the AI in calculating long-term reward expectancy so that it can ultimately accumulate the higher rewards.

Figure 1 shows the overall procedure of the Action Sequence-based method. In FightingICE, there are 56 actions in total that a player character can perform. Among these, we filtered out 40 actions, such as jumping up in the mid-air, which can barely be followed by any subsequent, desirable actions. The rest 16 actions are expected to be capable of yielding higher rewards. Each gene in our structure is an action randomly chosen out of the 16 candidates, and 4 such genes, duplicated or not, form up an Action Sequence (AS) chromosome. An AS chromosome is the basic unit of our method, representing single action combo in a distinct order, which we aimed to optimize. In pursuit of the population diversity, no AS chromosomes share the same first gene, i.e. the same initial action. While identical or similar genes could also be capable of generating diverse action combos, there is no reason to use the second or third best choices with the number one already in our hand.

Through iterative generations, the AS chromosomes are processed with genetic operators, including selection, crossover, and mutation, and then stored in a genetic archive. The archive then selects chromosomes possessing the first genes with the highest fitness values. This architecture was derived from an intuition that choosing an initial action with positive reward would result in a preferable action combo. Finally, the archive is mounted on a MCTS which primarily controls the playing AI. Although the periodic time limit of 16.67ms is a harsh constrict that would impede any attempt to find the optimal solution, MCTS-based AIs have constantly shown marked excellence in the past competitions.[2] In our model, MCTS chooses the agent's action in each time frame. Initially, it activates the first gene in each chromosome from the archive and, instead of making a new choice for the subsequent action, operates the wholes series of genes of the corresponding chromosome in order. Action Sequence is intended to achieve successful combinations of such chosen actions.

3 EXPERIMENT AND RESULT

The fitness value of each AS chromosome, defined as the difference between the health points (HP) of the two playing agents, was obtained via 10 rounds (against a random-acting AI) with genetic operations under the parameter setting as follows: population size: 48 (three times the aforementioned 16 actions); selection K: 0.9; 3-point crossover; chromosome size: 4; mutation probability: 0.01. After 50 such generations, 16 chromosomes with the best first genes formed an archive, which ultimately chose 8 particular chromosomes with positive rewards for MCTS. MCTS' greedy

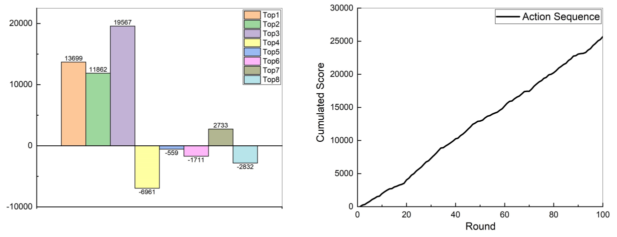

Figure 2: (Left) Scores of AS-based AI with each individual chromosome. (Right) Accumulated score of AS-based AI with integrated top chromosomes.

nature causes shallow depth of action steps under the real-time pressure of 16.67 ms. Our main idea here is to provide it with 3 optimized, subsequent actions evaluated via genetic operations without spending extra time for manually figuring them out.

To precisely measure how much our method advances from the pure MCTS[3], we played 100 rounds of our AS-based AI against the basic MCTS-based AI. We first mounted each of the 8 best chromosomes on MCTS and measured their performances as in score separately, which are presented in the left-hand graph of Figure 2. The graph shows that 4 of them outperforms the MCTS-based AIs. Second, we combined those 4 chromosomes with MCTS at once and recorded its scores against the MCTS-based AI in 100 rounds. As seen in the right-hand graph of Figure 2, our model overwhelms the MCTS AI with 99% winning ratio, proving that integrating top chromosomes reinforces the performance more effectively than mounting them individually. Our approach presents genetic algorithms as a qualified means to accompany RL.

4 CONCLUSION

Our attempt in this paper shows with the superior performance to the original MCTS that genetic algorithms can be exploited to alleviate the RL greediness. Since our method does not merely solve a particular problem but resolves a common obstacle that every RL problem shares, we expect that it can be applied to the popular deep neural network-based RL techniques such as DQN and A3C, which could be covered for future studies. We would also like to note that it's applicable not only to real-time video games but also to the problems of much wider aspects containing complex solutions under continuous time series.

ACKNOWLEDGMENTS

This work was supported by the LG Electronics (LGE) grant and also by GIST Research Institute (GRI) grant funded by the GIST in 2019.

REFERENCES

[1] David E. Goldberg. 1989. *Genetic Algorithms in Search, Optimization and Machine Learning.* Addison-Wesley Publishing Company.
[2] Man-Je Kim and Chang Wook Ahn. 2018. Hybrid fighting game AI using a genetic algorithm and Monte Carlo tree search. In *GECCO '18 Proceedings of the Genetic and Evolutionary Computation Conference Companion.* ACM, 129–130.
[3] Yoshida Shubu, Ishihara Makoto, Miyazaki Taichi, Nakagawa Yuto, Harada Tomohiro, and Thawonmas Ruck. 2016. Application of Monte-Carlo tree search in a fighting game AI. In *2016 IEEE 5th Global Conference on Consumer Electronics.* IEEE, 1–2.