Universitat Rovira i Virgili

Master's Thesis

Masters of Artificial Intelligence and Computer Security

# Behavior approximation using fuzzy-genetic systems

Mark Safronov

June 6, 2025

# Contents



# 1 Introduction

## 1.1 Contextualization

Historically, this work started as an attempt to write a formal solver for the Japanese computer game *Princess Maker 2* (1993) by the studio Gainax. The game is a life simulation where the player takes the role of a guardian of a young girl, making decisions that affect her upbringing and future.

The full solver for that particular game is an overly complex task, as it involves understanding the narrative and solving multiple interdependent gameplay mechanics.

Instead, we will extract a simplified model of the core gameplay loop, which we can formalize as follows.

We will define "behavior" as a classical Reinforcement Learning[3] problem.

Assuming we have a character described as a set of numeric characteristics

$$\mathbf{x} \in \mathbb{Z}^n \tag{1}$$

we have a set of possible actions

$$A = \{a_1, a_2, \ldots, a_m\} \tag{2}$$

which collectively form a transfer function

$$f(\mathbf{x}, a) = \mathbf{x}' \tag{3}$$

Now, we have a goal state

$$\mathbf{x}^* \in \mathbb{Z}^n \tag{4}$$

and a planning horizon

$$T \in \mathbb{Z} \tag{5}$$

## 1.2 Problem statement

We want to get an ordered actions sequence of length $T$ which will lead $\mathbf{x}$ to $\mathbf{x}^*$:

$$\mathbf{a} \in A^T, x_o = \mathbf{x} : \bigodot_{i=1}^{T} f(x, a_i) = \mathbf{x}^* \tag{6}$$

where $\bigodot$ is a fold operator. The transfer function $f$ is assumed to be completely determined, and the whole process being non-stochastic.

Normally, there is a number of classical approaches to this problem, including dynamic programming, reinforcement learning and state space searches using graph theory. However, in this work we'll focus specifically on the cases which lead to combinatorial explosion for classical solutions, that is, when we have sufficiently large amount of characteristics, actions to choose from and most importantly, very large planning horizon:

$$n > 50 \tag{7}$$
$$m > 20 \tag{8}$$
$$T > 2000 \tag{9}$$

As an additional restriction, we'll state that it's not enough to reach the desired state $\mathbf{x}^*$ sometime before $T$, instead, it is a must that we perform $T$ actions and only then evaluate the final state $\mathbf{x}'$ of the character. Having this distinction is important, as actions result in *arbitrary* changes in the character state, both positive and negative. So, it is possible, but unacceptable, to reach the desired state $\mathbf{x}^*$ before $T$ and after that perform a number of actions which will lead to a final state $\mathbf{x}'$ which is not equal to $\mathbf{x}^*$.

With these restrictions in place, a need in an heuristic arises to perform efficient search in the state space, as its size becomes unrealistically large.

## 1.3 Proposed approach

In this work we evaluate one possible approach which is defined as follows.

Let's assume that we can segment the set of possible actions to clusters with the following particularities:

1. actions in the same cluster lead to "similar" changes in the character state $\mathbf{x}$.
2. the cluster as a whole can be described symbolically

In this case we can synthesize a set of numeric characteristics which we'll call "inclinations":

$$\mathbf{I} \in \mathbb{Z}^q \tag{10}$$
$$q << n \tag{11}$$

From this, we can define a set of fuzzy rules[2] mapping the inclinations to action choices:

1. if an inclination $I_i$ has a fuzzy value $V_I$,

2. then $P_a$, the priority of an action $a$, is a fuzzy set $V_A$.

After the defuzzification of all the inferred fuzzy values $P_a$ we select an action with the highest priority.

The selection and design of fuzzy rules is a critical aspect of this approach. In this thesis, the fuzzy rule base is constructed manually, leveraging domain knowledge to define the mapping from inclinations to action priorities. Future research may investigate automated methods for generating fuzzy rules, such as clustering or machine learning techniques, to further improve scalability and reduce manual effort.

While it is theoretically possible to define fuzzy rules that map every possible inclination vector $\mathbf{I}$ or even every state $\mathbf{x}$ to action priorities, such exhaustive rule sets would quickly become infeasible due to combinatorial growth. This reinforces the importance of dimensionality reduction and clustering in making the fuzzy-genetic approach tractable for high-dimensional planning problems.

The assumption which we explore among others in this work is the practical possibility to write a coherent set of fuzzy rules which will be clustered around the clusters of actions, and each inclination will tend to map to its own cluster of actions.

Now, using such a fuzzy controller $\xi(I, \mathbf{x})$ we can construct the goal function:

$$g(I, \mathbf{x}) = \bigodot_{i=1}^{T} f(x, \xi(I, x_i)) \tag{12}$$

the above formula being subject to improvements in expressiveness, the main point of which being the fuzzy controller $\xi(I, x_i)$ selecting the action to perform on the step $i$ according to the inclinations and (ideally) the current state $x_i$.

For the $g$ parameter $\mathbf{x}$ is essentially a constant, and given (11), we effectively performed dimensionality reduction on the original problem.

We can find $\arg\max(g)$ now using an appropriate optimization method. For this work, because of a strong biosocial analogies a genetic algorithm[1] was chosen, with the vector of inclinations $\mathbf{I}$ as a chromosome.

## 1.4 Hypothesis

The hypothesis explored in this work is that the combination of assumptions described above constructs an heuristic which allows solving the problem more efficiently than performing the reinforcement learning directly.

## 1.5 Objectives

The objective of this thesis is to investigate whether fuzzy-genetic heuristics can effectively solve high-dimensional deterministic planning problems through dimensionality reduction and symbolic reasoning.

In particular, we aim to:

1. Formalize the problem as an optimization task.

2. Implement a working solver.

3. Evaluate the performance of the solver on a set of test cases of increasing complexity.

4. Analyze the results to draw conclusions about the effectiveness of the approach.

# 2 Methodology

## 2.1 Reinforcement learning

## 2.2 Fuzzy controller

## 2.3 Genetic optimization

# 3 Implementation

# 4 Experiments and Results

## 4.1 Trivial case

2 characteristics, 4 mutually exclusive actions, 3 steps.

This scenario represents a trivial case, with only $4^3$ possible action sequences—a total of 64. The small state space allows for exhaustive enumeration and manual verification of results. This case serves to validate the correctness of the implementation and the fuzzy controller, as the system's behavior can be easily traced and analyzed by hand.

## 4.2 Base control case

4 characteristics, 12 actions, 100 steps.

This case is the base case, as it introduces enough complexity to test the proposed approach and at the same time compare it with classical approaches.

A decision tree of the size $12^{100}$ is already too large to be completely enumerated.

However, with 4 characteristics and 12 actions, the problem is still well within the range where Reinforcement Learning methods—especially those using function approximation—can be applied efficiently. The planning horizon of 100 steps is long enough to be non-trivial, but does not pose significant challenges for standard RL algorithms.

## 4.3 Complex case

The Princess Maker 2 case is a problem with 50 numeric characteristics of a character and 25 actions to choose from, with a planning horizon of 2500 steps.

This case is used to test the performance of the proposed approach on a real-world problem.

# 5 Discussion

# 6 Conclusions and Future Work

Despite the context of the problem being a computer game, the problem itself is a general one, and the proposed approach can be applied to any high-dimensional deterministic planning problem. Which constitutes the core value of this work.

In the span of this work, only three distinct cases were explored, and the more thorough exploration of the parameter space is left for a dissertation-level research.

# References

[1] Melanie Mitchell. *An Introduction to Genetic Algorithms*. Cambridge, MA: MIT Press, 1999. ISBN: 9780262631853.

[2] S. Kumar Ray. *Soft Computing and Its Applications, Volume II*. Boca Raton, FL: CRC Press, 2014. ISBN: 978-1-4822-5793-9.

[3] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. 2nd ed. Available online. MIT Press, 2018. ISBN: 978-0-262-03924-6. URL: http://incompleteideas.net/book/the-book-2nd.html.