Designing and Implementing Kaiju Combat and Intelligence Models Using Finite State Machines

Ryan Austin Fernandez
De La Salle University
Manila, Philippines
ryan.fernandez@dlsu.edu.ph

ABSTRACT

Finite state machines are a basic strategy for modeling artificial intelligence in games. This project allows students to explore modeling combat scenarios between two *kaiju*. Students must implement a singular intelligence model. Then they must implement a combat engine between two kaiju. Finally, they must implement an optimizer for a kaiju's intelligence model. Students are expected to properly document their design and implementation details as a technical report at the end of the project period.

CCS CONCEPTS

• **Theory of computation** → Regular languages; *Automata extensions*; *Abstract machines*; **Interactive computation**; **Algorithmic game theory**; **Representations of games and their complexity**.

KEYWORDS

artificial intelligence, theory of computation, finite state machines

1 PROJECT DESCRIPTION AND OVERVIEW

Finite state machines (FSMs) are a simple model for artificial intelligence in games [1]. Whether a non-playable character (NPC) is an ally or not, modeling the character's behavior as several states, adapting as they receive stimulus from other characters or their environment is a viable strategy for intelligence modeling.

In this project, you will be using the idea of modeling kaiju intelligence as finite state machines. This project is designed for groups of 2 – 4 students — select groupmates within your enrolled sections.

This specifications document first explains the FSM model for kaiju intelligence, followed by a prescribed methodology for carrying out the project. The documentation format is then explained, followed by the grading system and the list of final deliverables.

2 THE FSM MODEL FOR KAIJU INTELLIGENCE

Kaiju are complicated creatures. For the purposes of this project, we are going to abstract their mechanics by defining the following mathematical objects:

- Let $\mathbb{M}$ be the finite set of *kaiju skills*. This is the *universe* of all skills a kaiju can use.
- $dmg : \mathbb{M} \mapsto \mathbb{Z}^+$ is a function that maps a skill to its damage value;
- $hp\_cost : \mathbb{M} \mapsto \mathbb{Z}$ is a function that maps a skill to its *hp* cost. A move heals the user if and only if $hp\_cost(m) < 0$. **If using a skill reduces a kaiju to 0 hit points, it deals damage anyway**.
- Let $S = \{hurt, ok\}$ be the set of *simple statuses*.
- Let $\mathbb{K}$ to be the universe of all kaiju (a *monsterverse* if you will).
- Let $hp : \mathbb{K} \mapsto \mathbb{Z}_{\geq 0}$, be defined as the function which maps a kaiju to their current hit points. We assert that $0 \leq hp(k) \leq N$, where N is the maximum hit points of the kaiju k. If a kaiju uses a move that heals it, and the hp total would exceed its maximum, the hp is instead capped at the maximum value.
- Let $status : \mathbb{K} \mapsto S$ be defined as the function that maps a kaiju to its status s. We define the behavior of this function as follows:
  - $(status(k) = ok) \Leftrightarrow (2 \cdot hp(k) > k_{hp_{max}})$
  - $(status(k) = hurt) \Leftrightarrow (2 \cdot hp(k) \leq k_{hp_{max}})$
- Let a kaiju be represented by the 7-tuple
  $K = (Q, K_x, M_K, f, g, q_I, hp_{max}, k_I)$
  where
  - Q - finite set of states
  - K_x - the enemy kaiju, $K_x \in \mathbb{K}$
  - M_K - finite set of skills, $M_K \subseteq \mathbb{M}$
  - $f : Q \times \mathbb{M} \times S \mapsto Q$ - transition function. If K is in state q_i and a move m is used and $status(K_x) = s$, K moves into state $f(q_i, m, s)$.
  - $g : Q \times \mathbb{M} \times S \mapsto M_K$ - move function. If K is in state q_i and a move m is used and $status(K_x) = s$, K uses its skill $g(q_i, m, s)$.
  - q_I - initial state. $q_I \in Q$
  - $hp_{max} \in \mathbb{Z}_{>0}$ - max hit points of K
  - m_I is the initial move of the kaiju, assuming they go first; $m_I \in M_K$.

2.1 Kaiju Combat

Kaiju combat can be described as follows.

1. The kaijus' initial values for $hp(k)$ are equal to their max HP.
2. One of the kaiju is declared to go first. Let K_1 be this kaiju. K_1 uses the move marked as m_I in its formal definition.
3. The *hp_cost* is deducted from the kaiju, and the *dmg* is applied to the opponent.
4. If the opponent, henceforth referred to as K_2, has at least 1 hit point ($hp(K_2) \geq 1$), it uses the last move used by K_1, as well as the value of $status(K_1)$, to determine its new state and the new move, based on its transition function f, and move function g, respectively.
5. The *hp_cost* and *dmg* for the move used by K_2 is applied to K_2 and K_1 respectively. If K_1 has at least 1 hit point, it then

responds using the last move used by K_2, and $status(K_2)$ to determine its next move, similar to how K_2 did.

(6) Repeat steps $4-5$ until one of the kaiju is reduced to ≤ 0 hit points. The winner is the kaiju that has at least 1 hit point remaining.

Note that while using kaiju skills, a kaiju may reduce their own HP to 0 via the hp_cost. If this happens, there are two possibilities:

- The enemy kaiju is also reduced to 0 HP. In this case, the battle is a draw.
- The enemy kaiju has at least 1 HP. In this case, the enemy kaiju wins.

To illustrate, see Tables 1 and 2 for definitions of kaiju and skills.

Table 1: Table Summarizing the Damage and HP Costs of Kaiju Skills

Skill	HP Cost	Damage
Atomic Breath	5	15
Axe Swing	1	10

If we set Godzilla's m_I = Atomic Breath, Kong's m_I = Axe Swing, and Godzilla goes first; the combat goes as follows.

(1) Godzilla goes first and uses Atomic Breath. Godzilla spends 5 hit points, reducing him to 25 HP, and fires atomic breath at Kong, reducing him to 15 HP.

(2) Kong checks his transition table. He is in state C, Atomic Breath, and $status(Godzilla) = ok$ was used. This means he transitions to state C and uses Axe Swing, spending 10 HP. This reduces Kong to 9 HP, but it deals 10 damage to Godzilla, reducing him to 15 HP.

(3) Godzilla just took an Axe Swing and $status(Kong) = hurt$. Godzilla transitions to state B, and uses Atomic Breath, dealing himself 5 damage, but dealing Kong 15 damage, reducing Kong to ≤ 0 HP. Godzilla wins.

3 METHODOLOGY

This project will be done in three major phases: implementing a single kaiju's intelligence model, implementing a combat simulation between two kaiju, and implementing an optimizer for a kaiju's intelligence model.

3.1 Milestone 1: Kaiju Intelligence Model Implementation

In this phase, you will be given the formal definition for the kaiju's intelligence model K. The task is to implement this using either C++14, Java 8, or Python 3. This module will be evaluated via HackerRank. A series of inputs in the format $(move, status) \in \mathbb{M} \times S$ will be provided to your model. The correctness of your program will be determined by the sequence of outputs produced by your model. Please see the HackerRank problem for more information.

3.2 Milestone 2: Kaiju Combat Implementation

For the second phase, you will be given two definitions of kaiju intelligence models. You have to simulate combat between them, as explained in Section 2.1. A detailed log of the battle must be printed. The correctness of your program will be judged based on this log. Judging will be done via HackerRank. Please see the HackerRank problem for more information.

3.3 Milestone 3: Kaiju Intelligence Optimization

For the final phase of the project, you will be provided with the definition of one kaiju intelligence model, available kaiju skills $\mathbb{M}$, and a state count $|Q|$. Your task is to compute the *optimal* intelligence model given full knowledge of the opponent's intelligence model, as well as your state limit $|Q|$. We define the *optimal* intelligence model as the model that either:

- Can always defeat the opposing kaiju, and does so in as few rounds as possible; or
- Cannot beat the opposing kaiju but lasts as many rounds as possible.

We define a round as the time when two kaiju, K_1 and K_2 as defined in Section 2.1, each use one move each in succession and are ***both still standing***.

- If the first kaiju taking their turn in a round results in the second kaiju dropping to ≤ 0 HP, then this last turn is **NOT** counted as a complete round.
- If the second kaiju taking their turn in a round results in the first kaiju dropping to ≤ 0 HP, then this last turn is **NOT** counted as a complete round.
- If a draw occurs when both kaiju drop to ≤ 0 HP, then the last round is **NOT** counted.

Judging of this phase will be done on HackerRank. The output will be two integers A, B, where A is 1 if an intelligence model that can beat the opposing kaiju can be constructed and 0 otherwise. B is the minimum rounds required to beat the opposing kaiju, if possible, the maximum rounds you can survive if it is not possible to defeat the opposing kaiju. Please see the HackerRank problem for more information.

4 DOCUMENTATION

The project documentation must follow the formatting of this document. To make this easier, it is highly recommended that you use Overleaf or LaTeX to typeset your document. You may find the read-only source code for this specifications document at https://tinyurl.com/kaijuspecs.

The prescribed sections of your document are described below:

4.1 Title

Provide a unique title for your documentation. You may use the name of this document if you cannot think of one.

4.2 Authors

Credit each group member in this section, along with your affiliations.

4.3 Abstract

Provide a single-sentence context for the project, an overview of the implementation, and the results of the tests, i.e., HackerRank results. This should not exceed 150 words.

Table 2: Transition Table for Kong and Godzilla

Kaiju	Godzilla			
Max HP	20			
	Kaiju Skill			
State	Atomic Breath		Axe Swing	
	ok	hurt	ok	hurt
→A	A : Atomic Breath	A : Atomic Breath	B : Atomic Breath	B : Atomic Breath
B	A : Atomic Breath	A : Atomic Breath	B : Atomic Breath	B : Atomic Breath
Kaiju	Kong			
Max HP	30			
	Kaiju Skill			
State	Atomic Breath		Axe Swing	
	ok	hurt	ok	hurt
→C	C : Axe Swing	C : Axe Swing	D : Axe Swing	D : Axe Swing
D	C : Axe Swing	C : Axe Swing	D : Axe Swing	D : Axe Swing

4.4 Introduction

Provide a background for the project. Describe and cite any sources you used to assist you in your implementation of the projects.

4.5 Method

Give a brief overview of the subsections that follow.

4.5.1 Milestone 1: Kaiju Intelligence Model Design Implementation. Describe the high-level design of your kaiju model in your programming language of choice, as well as any specific implementation details concerning how you implemented the intelligence model.

4.5.2 Milestone 2: Kaiju Combat Design and Implementattion. Describe the high-level design of your combat engine and how you modeled the interaction between two kaiju intelligence models, as well as any specific implementation details concerning how you implemented the combat engine.

4.5.3 Milestone 3: Kaiju Intelligence Optimization. Describe the high-level algorithm for your optimization of the intelligence model and any specific implementation details concerning how you implemented the optimization algorithm. What problem-solving paradigms did you employ, if any, to optimize the kaiju's intelligence model?

4.6 Results and Analysis

Describe how you tested each of the milestones. Do not only cite the HackerRank portal in this section. You are expected to come up with your own tests apart from the test cases on HackerRank. Analyze the correctness, time complexity, and space complexity of your implementations. Do your intelligence models, combat engines, and optimization algorithms work properly? How long do they take to execute? How much memory do they consume?

Provide an interpretation and analysis of the results. If there are any issues concerning correctness, time efficiency, or space efficiency, what are your recommendations for improving the implementation? Is your implementation elegant, or did you resort to unreadable or spaghetti code?

Lastly, provide a brief discussion on how a stack memory or random-access memory component could improve the performance of your intelligence model. Would a kaiju with infinite memory always beat a kaiju using only an FSM intelligence model?

4.7 Conclusion

Provide a summary of the major design and implementation details as well as the results. Synthesize whether you were able to implement the tasks assigned successfully.

4.8 Appendix A: Declaration of Work

For each member of the group, list down their contributions to the project. Any contribution towards any components mentioned in Section 5 can be placed in this appendix.

4.9 References

Properly cite your sources in this section. ***Failure to cite sources will result in a grade of ZERO for this project.***

5 GRADING

The grading system for this machine project is shown in Table 3.

Table 3: Grading Scheme for the STALGCM Machine Project, Term 2, AY 2020 - 2021

Component	**Points**
Milestone 1 HackerRank Score	25
Milestone 2 HackerRank Score	25
Milestone 3 HackerRank Score	25
Project Documentation	50
TOTAL	125

The scoring of the source code will be dependent on the partial score provided by HackerRank. No adjustments will be made to the score provided by HackerRank. Failure to submit your source code to HackerRank will result in a score of **ZERO** for that milestone. ***Any request to view the test cases will be declined.***

Only 100 points will be credited under the Machine Project component. Any excess scores will be added to the inspiration component. The rubrics for the project documentation are in Table 4

6 FINAL DELIVERABLES

To reiterate, this project is designed for groups of 2 – 4 students — select groupmates within your enrolled sections. Remember to cite your sources in the documentation. Failure to cite sources will result in a grade of **ZERO** for this project.

To summarize, the final deliverables for this project are as follows:

- Source Code for Milestone 1: Kaiju Intelligence Model Implementation (submitted to HackerRank);
- Source Code for Milestone 2: Kaiju Combat Implementation (submitted to HackerRank);
- Source Code for Milestone 3: Kaiju Intelligence Optimization (submitted to HackerRank); and
- Project Documentation (submitted in Canvas)

All code submissions must be made at this Hackerrank Portal: https://www.hackerrank.com/kaiju-fsm. ***Any request to view the test cases will be declined.***

The project deadline is on May 24, 2021, at 11:59 PM. This is inclusive of all the deliverables previously mentioned.

REFERENCES

[1] David M Bourg and Glenn Seemann. 2004. *AI for game developers.* " O'Reilly Media, Inc.".
[2] Peter J. Denning, Jack B. Dennis, and Joseph E. Qualitz. 1980. Machines, Languages, and Computation. *Journal of Symbolic Logic* 45, 3 (1980), 630–631. https://doi.org/10.2307/2273429
[3] John E. Hopcroft and Jeff D. Ullman. 1979. *Introduction to Automata Theory, Languages, and Computation.* Addison-Wesley Publishing Company.
[4] Michael Sipser. 2013. *Introduction to the Theory of Computation* (third ed.). Course Technology, Boston, MA.
[5] Adam Wingard. 2021. *Godzilla vs. Kong.* Warner Brothers Pictures.

Table 4: Grading Rubrics for the Project Documentation

Component	Outstanding (4.0)	Good (3.0)	Improving (1.5)	Poor (0.0)
Abstract (5 pts)	Clearly summarizes the objective of the project, work done, results of the project testing, and conclusions.	Is missing one of the following: the objective of the project, work done, results of the project testing, and conclusions.	Is missing two of the following: the objective of the project, work done, results of the project testing, and conclusions.	Is missing three or more of the following: the objective of the project, work done, results of the project testing, and conclusions
Introduction (5 pts)	Provides a brief context of the study, with properly cited sources, and gives an overview of the technical paper.	Is missing one of the following: brief context of the study, properly cited sources, and an overview of the technical paper.	Is missing two of the following: brief context of the study, properly cited sources, and an overview of the technical paper	Is missing three or more of the following: brief context of the study, properly cited sources, and an overview of the technical paper
Method (Depth of Discussion, 10 pts)	The design and implementation of each of the milestones are discussed with adequate depth and elaboration. The approach is clearly defined and formulated.	The design and implementation of each of the milestones are discussed with some depth or elaboration. The approach is defined and formulated, albeit with some ambiguities or unclear steps/processes.	The design and implementation of each of the milestones are discussed with little depth or elaboration. The approach is defined and formulated, albeit with several ambiguities or unclear steps/processes.	The design and implementation of each of the milestones are discussed with little to no depth or elaboration. The approach is barely defined and formulated with numerous ambiguities or unclear steps/processes.
Method (Correctness, 10 pts)	Based on the discussion, the design and implementation of the modules are correct, efficient, and elegant.	Based on the discussion, the design and implementation of the modules are mostly correct, efficient, or elegant, with possible problems in one of these metrics.	Based on the discussion, the design and implementation of the modules are mostly correct, efficient, or elegant, with possible problems in two of these metrics.	Based on the discussion, the design and implementation of the modules are incorrect, inefficient, and inelegant.
Method (Writing, 5 pts)	The writing is clear and concise. There are no filler words. The paper and each section are structured very well.	The writing is somewhat clear and concise. There are some filler words. The paper and each section are structured adequately.	The writing is somewhat unclear or bloated. There are several filler words. The paper and each section are structured confusingly.	The writing is somewhat unclear or bloated. There are several filler words. The paper and each section are structured confusingly.
Results and Analysis (10 pts)	Experiments and testing process were clearly defined. Test results were clearly communicated. A complete and thorough analysis of the results was provided.	Experiments and testing process were somewhat defined. Test results were communicated, with some unclear aspects. Some analysis of the results was provided.	Experiments and testing process were loosely defined. Test results were communicated, with several unclear aspects. Barely any analysis of the results was provided.	Experiments and testing process were not defined. Test results were poorly communicated. Little to no analysis of the results was provided.
Conclusion (5 pts)	Significant results from the experiments were summarized. It is clearly stated whether the objective of the project was satisfied or not. Some recommendations for the improvement of the system were provided.	Some results from the experiments were summarized. It is somewhat stated whether the objective of the project was satisfied or not. Few recommendations for the improvement of the system were provided.	Very few results from the experiments were summarized. It is not stated whether the objective of the project was satisfied or not. No recommendations for the improvement of the system were provided.	No results from the experiments were summarized. It is not stated whether the objective of the project was satisfied or not. No recommendations for the improvement of the system were provided.
TOTAL: 50				