Modelowanie procesów ewolucyjnych za pomocą automatów komórkowych

Gabriel Kaszewski

1. Wprowadzenie

Celem pracy jest zbadanie możliwości modelowania procesów ewolucyjnych przy wykorzystaniu automatów komórkowych i **Entity Component System**, który w dalszej części pracy będzie przedstawiany jako **ECS**. Praca ma na celu nie tylko pogłębienie teorytycznych podstaw modelowania ewolucji przy użyciu dyskretnych metod obliczeniowych, ale także popularyzację nowoczesnych technik programistycznych.

Symulację napisałem w języku Rust z wykorzystaniem biblioteki **Bevy**, która jest oparta na **ECS** i dostarcza wszystkie potrzebne komponenty do stworzenia projektu multimedialnego. Do wizualizacji danych użyłem języka **Python** z całym pakietem bibliotek do analizy danych takich jak **NumPy**, **Matplotlib**, **Pandas**.

2. Entity Component System

2.1. Wstęp

ECS to współczesna architektura oprogramowania stosowana głównie w grach komputerowych oraz symulacjach. Umożliwia ona elastyczne i wydajne zarządzanie złożonymi systemami. Główną ideą **ECS** jest oddzielenie danych (komponentów) od logiki (systemów) oraz traktowanie encji jako jedynie identyfikatorów, co pozwala na łatwiejsze skalowanie, modyfikację oraz optymalizację (np. w kontekście wielowątkowości). W tym rozdziale postaram się przybliżyć podstawowe pojęcia związane z **ECS** jego strukturę oraz zalety.

2.2. Encje

Encje (ang. *entities*) stanowią podstawowy element **ECS**. Są one reprezentowane najczęściej jako unikalne identyfikatory (np. liczby całkowite) i same w sobie nie zawierają żadnych danych, ani elementów logicznych. Encje są "nosicielami" komponentów, które definiują ich właściwości. Dzięki temu, encje zajmują mało miejsca w pamięci, a zarządzanie nimi - np. tworzenie, usuwanie czy modyfikacja - odbywa się w sposób efektywny, ponieważ nie wymaga to przeszukiwania złożonych struktur danych.

2.3. Komponenty

Komponenty (ang. *components*) to struktury danych, które zawierają właściwości lub stany encji. Każdy komponent jest dedykowany określonej właściwości obiektu, np. pozycji, prędkości czy "zdrowiu". W **ECS** komponenty nie zawierają żadnej logiki, są one jedynie kontenerami na dane. Z racji tego, że ich główną rolą jest przechowywanie informacji, są one zazwyczaj proste i niezależne od siebie. Co więcej, komponenty powinny być rozdzielane na mniejsze, wyspecjalizowane jednostki. Dzięki temu systemy mogą operować na dokładnie tych danych, które są im potrzebne, co sprzyja modularności i łatwości wprowadzaniu zmian, a przez to wydajności.

2.4. Systemy

Systemy (ang. *systems*) to moduły odpowiedzialne za logikę działania symulacji albo gry. Operują one na zbiorach encji, które posiadają określone komponenty. Przykładowo, system odpowiedzialny za ruch będzie aktualizował pozycje encji, które posiadają komponent *Pozycja* oraz *Prędkość*. Systemy wykonują swoje operacje cyklicznie (np. w każdej klatce gry) i mogą być projektowane tak, aby działały niezależnie od siebie.

2.5. Świat

Świat w kontekście **ECS** to kontener, który przechowuje wszystkie encje, komponenty i systemy danego projektu. Stanowi centralny punkt, za pomocą którego systemy mogą uzyskać dostęp do danych i komunikować się między sobą.

2.6. Zalety Entity Component System

Architektura **ECS** posiada szereg korzyści:

- **Modularność**: Oddzielenie danych od logiki umożliwia łatwe modyfikowanie i rozszerzanie funkcjonalności bez wpływu na całą strukturę aplikacji.
- **Wydajność**: Komponenty są przechowywane w pamięci w sposób ciągły, co sprzyja lokalności danych i pozwala na efektywne operacje na nich, ponieważ procesor o wiele szybciej uzyska dostęp do danych, które znajdują się w tzw. *cache'u* niż z pamięci RAM.
- **Elastyczność**: Można łatwo dodawać lub usuwać funkcjonalności przez modyfikację lub dodanie nowych komponentów i systemów.
- **Skalowalność**: **ECS** doskonale nadaje się do obsługi dużej liczby encji, co jest szczególnie ważne w symulacjach oraz grach z wieloma interaktywnymi obiektami.
- **Łatwość debugowania i testowania**: Dzięki modułowej budowie, testowanie i debugowanie poszczególnych komponentów i systemów jest znacznie prostsze.

2.7. Wielowątkowość w Entity Component System

Jednym z kluczowych atutów **ECS** jest możliwość łatwego wykorzystania wielowątkowości. Dzięki wyraźnemu oddzieleniu systemów, które operują na niepowiązanych zestawach komponentów, można równolegle przetwarzać dane w różnych wątkach. To podejście nie tylko zwiększa wydajność symulacji, ale także pozwala na lepsze wykorzystanie współczesnych procesorów wielordzeniowych. W praktyce oznacza to, że systemy nie muszą blokować siebie nawzajem, co znacznie poprawia skalowalność i responsywność aplikacji.

2.8. Podsumowanie

ECS to nowoczesne podejście do projektowania systemów, które wyróżnia się modularnością, wydajnością i elastycznością. Dzięki oddzieleniu encji, komponentów i systemów możliwe jest tworzenie skomplikowanych symulacji oraz gier w sposób przejrzysty i łatwy do skalowania. Dodatkową zaletą jest możliwość równoległego przetwarzania, co jest kluczowe w aplikacjach wymagających wysokiej wydajności. **ECS** stanowi doskonałą bazę do implementacji symulatorów oraz innych systemów, w których liczy się szybkie przetwarzanie dużej liczby obiektów, co czyni go idealnym narzędziem w kontekście bioinformatyki i modelowania ewolucyjnego.

3. Automaty komórkowe

3.1. Wstęp

Automaty komórkowe (ang. *cellular automata*) to dyskretne modele obliczeniowe, w których przestrzeń symulacji dzielona jest na regularną siatkę komórek. Każda komórka może znajdować się w jednym, ze stanów, które należa do zbioru dyskretnego i ograniczonego, a jej ewolucja odbywa się według ustalonych reguł, zależnych od stanów sąsiadujących komórek. Metoda ta umożliwia badanie złożonych układów i procesów dynamicznych przy wykorzystaniu prostych reguł lokalnych, co czyni automaty komórkowe atrakcyjnym narzędziem w symulacjach biologicznych, fizycznych oraz społecznych.

Pierwsze idee dotyącze automatów komórkowych pojawiły się już w latach 40. XX wieku, gdy John von Neumann i Stanisław Ulam badali samoreplikujące się układy. Jednak prawdziwy rozwój tej dziedziny nastąpił w kolejnych dekadach, kiedy to naukowcy zaczęli systematycznie wykorzystywać te modele do badania dynamiki złożonych układów. Jednym z przełomowych momentów była publikacja *gry w życie* (ang. *Game of Life*) autorstwa Johna Conwaya w 1970 roku, która stała się symbolem możliwości generowania złożonych struktur przy użyciu bardzo prostych zasad [1]. W latach 80. i 90. automaty komórkowe znalazły szerokie zastosowanie w badaniach nad zjawiskami samoorganizacji

oraz samoorganizującej się krytyczności (ang. *Self-Organized Criticality*) [2], czyli zdolności układów dynamicznych do spontanicznego przechodzenia w stan krytyczny. Stały się również inspiracją dla badań nad sztucznym życiem, modelując emergentne zachowania przypominające procesy biologiczne [3].

3.2. Definicja

Automat komórkowy definiuje się jako system składający się z trzech podstawowych elementów:

- **Siatka komórek**: Przestrzeń, w której każda komórka ma określoną pozycję (np. w układzie regularnym, takim jak kwadratowa lub heksagonalna siatka).
- **Zbiór stanów**: Dyskretny zbiór wartości, które mogą przyjmować poszczególne komórki (np. 0 lub 1, kolor, liczba, itp.).
- **Reguły przejścia**: Zbiór zasad, według których stan każdej komórki jest aktualizowany w kolejnych krokach czasowych, zależnie od stanów sąsiadów. Aktualizacja zwykle odbywa się synchronicznie dla wszystkich komórek.

Takie podejście umożliwia analizę, jak proste reguł lokalne mogą prowadzić do powstawania skomplikowanych, globalnych wzorców i struktur. [4]

3.3. Przykłady

Najbardziej znanym przykładem automatu komórkowego jest gra w życie Johna Conwaya [1], w której proste zasady dotyczące narodzin, przetrwania i śmierci komórek prowadzą do złożonych, często nieprzewidywalnych zachowań. Inne przykłady obejmują:

- **Elementarne automaty komórkowe**: Badane przez Stephena Wolframa, gdzie komórki mają tylko dwa stany, a reguły są określone na podstawie stanu sąsiadów w jednym wymiarze. Przykłady takich reguł to reguła 30 czy reguła 110 [5].
- **Automaty oparte o inne struktury siatek**: Automaty działające na siatkach heksagonalnych lub trójwymiarowych, które mogą lepiej modelować niektóre procesy naturalne.
- **Specjalistyczne modele**: Automaty komórkowe stosowane w modelowaniu wzrostu tkanek, rozprzestrzeniania się epidemii czy dynamiki ruchu tłumu. [6], [7]

Te przykłady pokazują, że automaty komórkowe są niezwykle wszechstronnym narzędziem, które znalazło zastosowanie zarówno w teorii, jak i w praktycznych zastosowaniach.

4. Mój model

4.1. Wstęp

Zainspirowany automatami komórkowymi postanowiłem zrobić symulację ewolucji populacji organizmów. (... do napisania reszta)

4.2. Opis modelu

Plansza, bądź świat w którym odbywa się symulacja jest dwuwymiarową siatką, gdzie każdy kafelek może być jednym z czterech rodzajów: woda, las, pustynia i trawa. Każdy typ terenu ma swoje własne właściwości jak np. dostępność pożywienia, prędkość odnowy pożywienia, jak szybko i chętnie organizmy poruszają się po nim.

W mojej symulacji mam dwa rodzaje organizmów: bierne i drapieżniki. Bierne organizmy dostają energię z jedzenia, które jest na danym kafelku. Drapieżniki z kolei dostają energię z jedzenia biernych organizmów.

(... do napisania reszta)

4.3. Implementacja

4.4. Wyniki

5. Podsumowanie

Bibliografia

[1] M. Gardner, "Mathematical Games," *Scientific American*, vol. 223, no. 4, pp. 120–123, Oct. 1970, doi: 10.1038/scientificamerican1070-120.

[2] P. Bak, C. Tang, and K. Wiesenfeld, "Self-organized criticality: An explanation of the 1/f noise," *Phys. Rev. Lett.*, vol. 59, no. 4, pp. 381–384, Jul. 1987, doi: 10.1103/PhysRevLett.59.381.

[3] Z. Olami, H. J. S. Feder, and K. Christensen, "Self-organized criticality in a continuous, nonconservative cellular automaton modeling earthquakes," *Phys. Rev. Lett.*, vol. 68, no. 8, pp. 1244–1247, Feb. 1992, doi: 10.1103/PhysRevLett.68.1244.

[4] A. Ilachinski, *Cellular Automata*, 0th ed. WORLD SCIENTIFIC, 2001. doi: 10.1142/4702.

[5] C. G. Langton, "Studying artificial life with cellular automata," *Physica D: Nonlinear Phenomena*, vol. 22, no. 1, pp. 120–149, 1986, doi: https://doi.org/10.1016/0167-2789(86)90237-X.

[6] P. Bak and K. Sneppen, "Punctuated equilibrium and criticality in a simple model of evolution," *Phys. Rev. Lett.*, vol. 71, no. 24, pp. 4083–4086, Dec. 1993, doi: 10.1103/PhysRevLett.71.4083.

[7] S. A. Kauffman, *The Origins of Order: Self-Organization and Selection in Evolution*. Oxford University Press, 1993. doi: 10.1093/oso/9780195079517.001.0001.

6. Kod źródłowy

Repozytorium z kodem źródłowym dostępne jest pod adresem: https://github.com/GKaszewski/evolution_cellular_automata