CELAT - Universal Cellular Automaton

Milan Abrahám

Zadání

Cílem práce je vytvořit univerzální celulární automat, kterému lze specifikovat pravidla na změnu stavů. Program bude obsahovat uživatelské rozhraní. Pro demonstraci funkčnosti v něm bude naimplemetovány automaty Game of Life [1] a Wireworld [2].

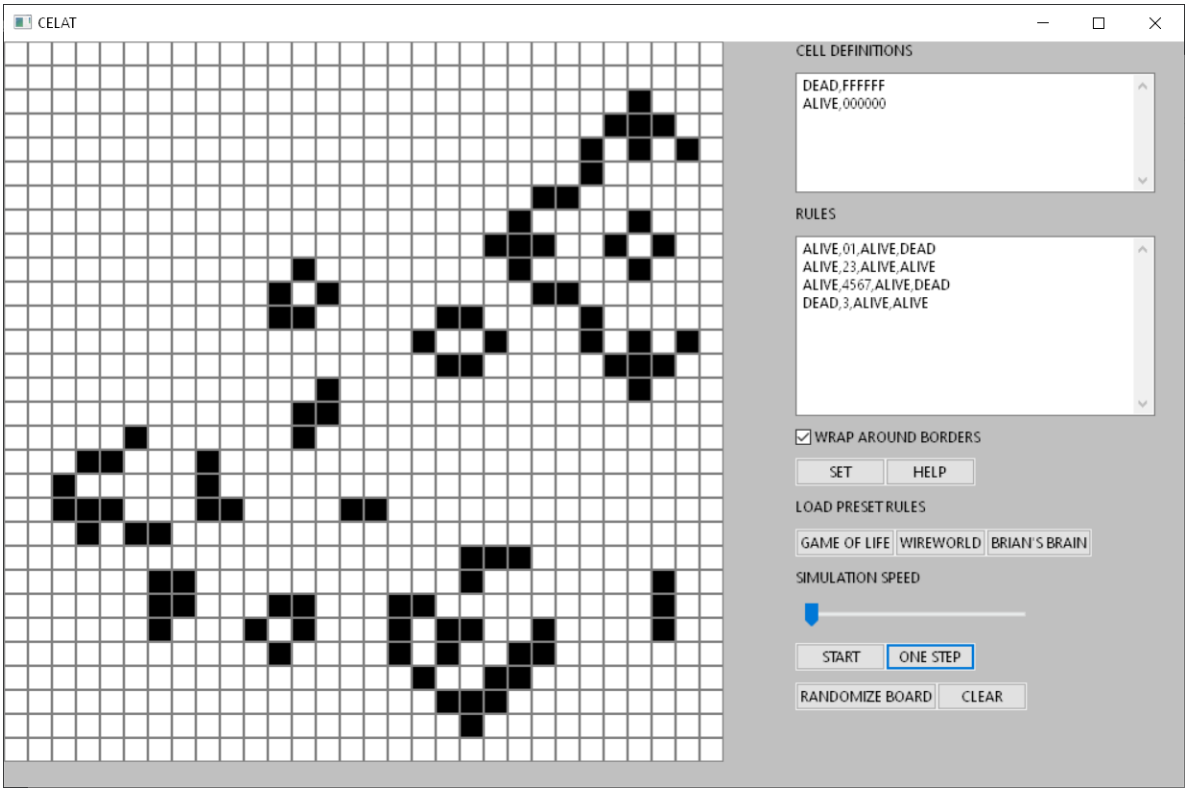
Úvod

Program napsaný v jazyce C++ obsahuje uživatelské rozhraní, jehož součástí je reprezentace automatu a ovládací prvky. Stavy(buňky) jsou reprezentovány v mřížce 30x30, buňky mají rozdílné barvy definované uživatelem. V ovládacím panelu uživatel upravuje chování automatu pomocí definice buněk, pravidel a dalších nastavení.

Součástí projektu je uživatelská dokumentace obsahující návod k instalaci a sestavení, vysvětlení použití automatu a formátu pravidel. Aplikace obsahuje tři předefinované automaty a to Game of Life [1], Wireworld [2] a Brian's Brain [3].

Projekt jsem rozdělil na dvě hlavní části - knihovna implementující univerzální automat a uživatelské rozhraní používající tuto knihovnu.

Ukázka rozhraní:



Pravidla

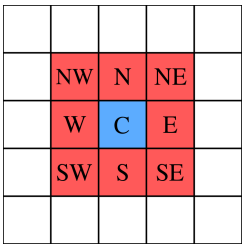
Nejdříve jsem si prošel běžné automaty a analyzoval jejich pravidla. Existuje velké množství automatů, každý obsahující trochu jiná pravidla a funkčnost. Některé automaty jsou velmi rozdílné a nebylo možné nalézt podobnost v jejich pravidlech. Ovšem velkou podmnožinu automatů lze definovat dvěmy typy pravidel:

1. Pokuď buňka se stavem A má X sousedů se stavem B změn její stav na C.
2. Buňku se stavem A vždy přeměň na stav C.

Tyto dva typy pravidel proto automat umí zpracovat. Více informací o formátu pravidel obsahuje uživatelská dokumentace.

Sousedství buňky a přetékaní hran

Většina automatů využívá Moorovo sousedství [4]. To znamená, že buňka má právě 8 sousedů.



Aumatová mřížka by teoreticky měla být nekonečná, což samozřejmě není možné implementovat. Je proto potřeba mřížku omezit. To lze vyřešit dvěma způsoby. Lze nastavit pevný okraj a okrajové buňky necháme s méně sousedy, což ale může způsobit nekonzistentí chování pravidel. Druhou možností je nechat okraje přetékat, všechny buňky poté budou mít 8 sousedů. Aplikace umožnuje obě nastavení.

Automat

Knihovna `automat.hpp` je navržena tak, aby byla nezávislá na zbytku aplikace a může tak být použita v jiných projektech.

Obsahuje hlavní třídu `Automat` a struktury `CellType` a `Rule` pro definice stavů a pravidel.

Konstruktor

Definice stavů a pravidel jsou poskytnuty konstruktoru třídy `Automat`. Ten je zpracuje, zkontroluje formátování, převede do struktury `CellType` nebo `Rule` a uloží do příslušného vektoru. Konstruktor dále požaduje výšku a šířku mřížky automatu a boolean, indikující zda automat na hranách přetéká.

Pravidla mají formát, který je potřeba striktně dodržovat. Je popsán v uživatelské dokumentaci. Pokud je nalezena chyba je vyvolána výjimka `Automat::InvalidFormatException` obsahující chybovou zprávu. Pokud je tato výjimka vyvolána objekt se nevytvoří a je potřeba zavolat konstruktor znovu se spravným formátem pravidel.

Poté co se zpracují pravidla, inicializují se všechny potřebné struktury. Tou hlavní je `std::vector<size_t> cells`, pole obsahující všechny buňky automatu. Buňky mají defaultní hodnotu, tou je první definovaný typ buňky. Pole obsahuje indexy typů ve vektoru `std::vector<CellType> cellTypes`. Používány jsou indexy místo pointerů pro rychlejší měnění stavů, mazání a randomizaci.

Souřadnice

Buňky jsou sice interně ukládány do 1-rozměrného pole, ovšem veřejné funkce poskytují přístup pomocí souřadnicového systému. Osa X je vertikální, Y horizontální. Nejlevější horní

buňka má souřadnice 0, 0. Souřadnice dolů a doprava se zvyšují.

Funkce automatu

Hlavní funkcí automatu je `void doOneEvolution()`, která všechny buňky posune vpřed o jednu evoluci. To znamená, že jsou na každou buňku aplikována pravidla automatu.

Dále obsahuje funkce pro obsluhu:

`std::string getColourAt(const size_t x, const size_t y) const` vrátí string obsahující barvu buňky v hexadecimálním formátu.

Uživatelské rozhraní

Poznámka: Automat v uživatelském rozhraní umožňuje jen pevnou velikost 30x30 buněk a to kvůli obtížnému nastavování velikosti uživatelského rozhraní. Knihovna ovšem umožňuje jakoukoliv výšku a šířku automatu.

Distribuce

Zdroje

[1] Conway's Game of Life - Wikipedia

[2] Wireworld - Wikipedia

[3] Brian's Brain - Wikipedia

[4] Moorovo okolí – Wikipedie

Obrázky

Moorovo okolí