“Atak z trzeciego wymiaru”
Dokumentacja Projektu

Zespół IV

Skład zespołu:
Miłosz Piórkowski
Filip Stefański
Kamil Ruszkowski
Mateusz Żbikowski

Spis Treści

1. Wstęp

Rozwiązania problemów zostały napisane w języku C++ w wersji 17. Dodatkowo wykorzystana została biblioteka graficzna SDL2 do prezentacji wyników w problemie 1. Ponadto, stworzyliśmy generator przykładowych danych testowych. Pliki źródłowe kompilują się do jednego pliku wykonywalnego. Do pomocy z budowaniem programu wykorzystaliśmy program CMake.

2. Generator

Pierwszym elementem wymaganym do rozwiązania któregokolwiek z problemów są dane wejściowe. Do tego został napisany osobny program Generator (aktualnie jest uwzględniony w głównym programie). Jego działanie jest proste, wykorzystując podstawowe funkcje losujące (nie potrzebujemy dużej losowości) generuje on zestawy danych dla dwóch z trzech problemów. Trzeci problem wykorzystuje dane wynikowe problemu pierwszego. Finalnie otrzymujemy łącznie 6 plików tekstowych. Dla problemu 1 mamy:

- pts.txt który zawiera losowe punkty naszej Krainy
- drogi.txt w którym zawarte są drogi między punktami
- tragarze.txt gdzie mamy wszystkich tragarzy, łącznie z ich preferencjami oraz układem rąk.

Dane do problemu 2 są już częściowo przetworzone. Aby je wygenerować potrzebny jest plik loremipsum.txt który zawarty jest w katalogu src. Na jego podstawie generowany jest tekst naszej pieśni. Następnie generowana jest pieśń z losowo obróconymi literami. Finalnie tworzony jest słownik ze wszystkimi słowami występującymi w pieśni. Czyli mamy:

- text.txt - nasza pieśń
- tekstzpoli.txt - pieśń popsuta przez międzywymiarowe istoty
- słownik.txt - słowa występujące w pieśni

3. Problem Pierwszy

3.1. Opis problemu

Płaszczaki czując zagrożenie z innego wymiaru potrzebują ochrony w postaci płotu. Należy wyznaczyć wymiary płotu wokół krainy oraz ustalić dostawy odcinków z fabryki. Do tego należy jeszcze odpowiednio sparować ze sobą tragarzy, ponieważ nie wszyscy chcą i mogą ze sobą współpracować.

3.2. Opis rozwiązania

Problem pierwszy możemy rozdzielić na trzy podproblemy:

1. Ustalenie projektu płotu
2. Ustalenie ścieżek dostaw z fabryki do punktów na płocie
3. Odpowiednie sparowanie płaszczaków.

Podproblem 1:
Kraina płaszczaków jest zbiorem punktów. Najlepszym sposobem na zaprojektowanie płotu używając minimalną liczbę materiałów jest wyznaczenie otoczki wypukłej tego zbioru. Do tego wykorzystaliśmy algorytm Grahama. Gwarantuje to efektywne wyznaczenie otoczki.

Podproblem 2:
Ścieżki z fabryki do punktów na otoczce wyznaczamy algorytmem Dijsktry, który pozwala na optymalne rozplanowanie pracy płaszczaków.

Podproblem 3:
Dobranie się płaszczaków w pary rozwiązaliśmy naiwnym algorytmem sprawdzającym kompatybilność każdego płaszczaka z każdym oraz uwzględniającym ich preferencje i predyspozycje do transportowania materiału do budowy płotu.

Podsumowując, na podstawie danych obliczonych w wyżej określony sposób, generujemy informacje o czasie i koszcie wybudowania całego płotu.

Pliki wejściowe:
- drogi.txt
- pts.txt
- tragarze.txt

Pliki wyjściowe:
- hull.txt - punkty otoczki, ułożone w kierunku odwrotnym do wskazówek zegara.
- koszt.txt - koszty kursów między fabryką a miejscem budowy płotu, które są zmienne.
- pary_tragarzy.txt - płaszczaki dobrane w pary, pierwsza linijka to ilość par a kolejne linijki to numery płaszczaków, które są dobrane w pary.
- result.txt - całkowity koszt i czas wybudowania otoczki.

4. Problem Drugi

4.1. Opis problemu

Pieśń zapisaną w krainie atakują istoty z innego wymiaru. Robią lustrzane odbicia liter oraz przekręcają je wokół własnej osi, psując tekst melodii. Musimy naprawić tekst melodii oraz znaleźć sposób na jej optymalniejsze przechowywanie.

4.2. Opis rozwiązania

W pierwszym kroku szukamy przekręconych liter i obracamy je aby zgadzały się ze słowami w słowniku.
W drugim kroku kompresujemy tekst melodii. Robimy to na dwa sposoby. Zmodyfikowanym algorytmem LZ78 oraz algorytmem Huffmana.
Algorytm LZ78 to algorytm kompresji słownikowej. Zmodyfikowaliśmy go tak, aby długość bitowa numerów słów ze słownika była dynamiczna. Użyliśmy drzewa binarnego, które jest jednym z dwóch możliwych podejść do tego algorytmu; drugim jest drzewo TRIE.

5. Problem Trzeci

5.1. Opis problemu

Płaszczaki, w celu zwiększenia bezpieczeństwa, postanowiły raz dziennie obchodzić płot, upewniając się,

że nie grozi im niebezpieczeństwo. Potrzebne jest wyznaczenie strażników oraz podanie drogi, którą będzie musiał przejść strażnik. Strażnik może przejść maksymalnie odległość równą ilości swojej energii, jednak gdy zakończy swoją drogę na punkcie, który był jaśniejszy od poprzedniego, musi odpocząć i posłuchać melodii. Płaszczaki nudzą się słuchając melodii wielokrotnie, dlatego jest potrzeba zminimalizowania ilości zatrzymań.

5.2. Opis rozwiązania

Każdy płaszczak posiada losową ilość punktów energii.
Wybieramy 7 płaszczaków, którzy zostaną mianowani na strażników krainy, tj. sortujemy strażników względem ilości punktów energii i wybieramy 7 najsilniejszych, tj. takich którzy mają przynajmniej 2 punkty energii.
Jeśli nie znajdziemy takich strażników, kraina jest w niebezpieczeństwie.
Ilość energii odpowiada za maksymalną odległość, jaką strażnik może przejść przez punkty płotu bez zatrzymania na odpoczynek.
Używamy algorytmu zachłannego, który będzie wybierał najdalszy ciemniejszy punkt na swojej drodze, aby zminimalizować ilość zatrzymań. Punkt zatrzymania musi być ciemniejszy od punktu w którym strażnik poprzednio się zatrzymał.