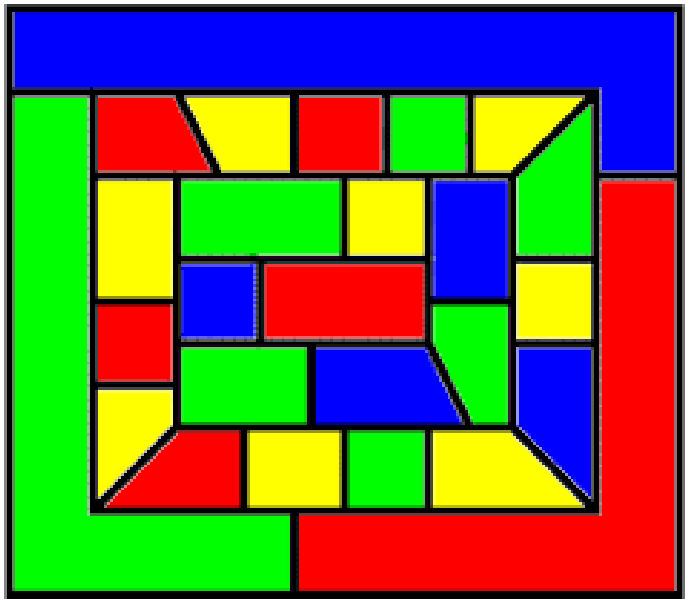
Teoria czterech kolorów
Projekt semestralny



Justyna Jarosz
Marcin Krzanowski
Karolina Niemiec

Symulacje komputerowe

8 lutego 2016

Plan prezentacji

- Rys historyczny
- Twierdzenie
- O projekcie
- Działanie programu
- Code Walkthrough

Rys historyczny

- Pierwszy 'dowód' podał w 1879 roku Alfred Bray Kempe i od tej chwili świat nauki uznał problem za rozwiązany
- W 1890 roku Percy John Heawood znalazl w dowodzie Kempego błąd na tyle poważny, że nie dało się go usunąć.
- Twierdzenie z powrotem stało się hipotezą.

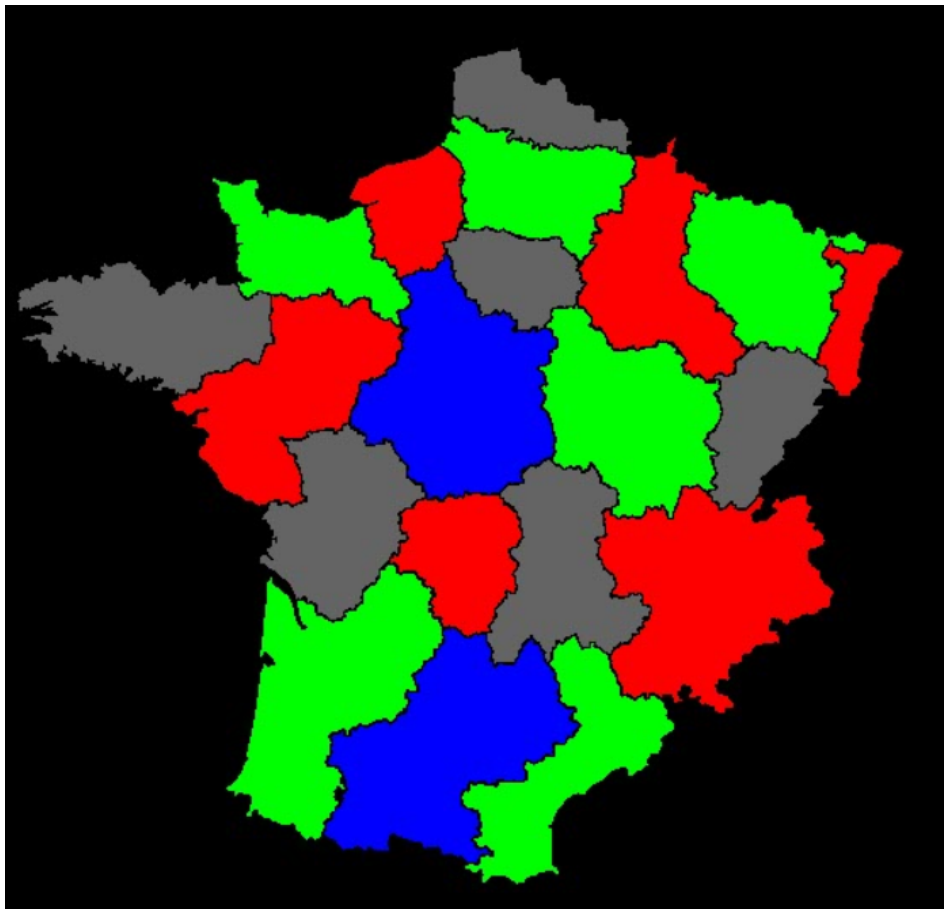
Rys historyczny

- Dopiero w 1976 roku Kenneth Appel i Wolfgang Haken przedstawili wielostronicowy dowód polegający na redukcji problemu do wielogodzinnej komputerowej analizy około dwóch tysięcy pojedyńczych przypadków.

Twierdzenie

- Twierdzenie głosi, że dowolną mapę polityczną, gdzie każdy kraj składa się tylko z jednego kawałka można zabarwić używając tylko 4 kolorów, tak aby żadne dwa kraje mające wspólną granicę (dłuższą niż punkt) nie miały tego samego koloru

Twierdzenie

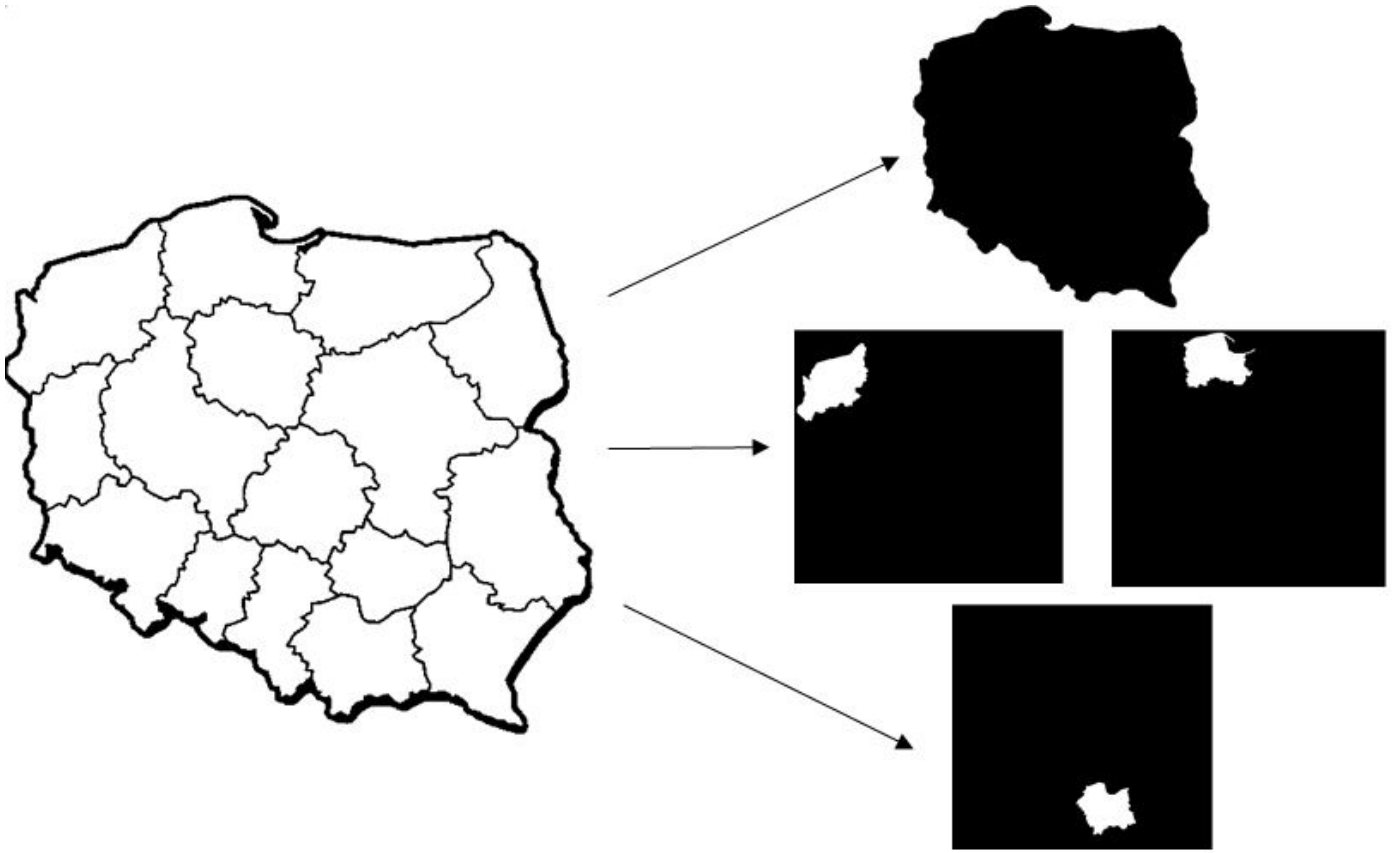


Rysunek: Twierdzenie o 4 barwach na przykładzie mapy Francji

O projekcie

- W celu rozwiązania problemu zastosowania teorii 4 kolorów w praktyce, powstały trzy skrypty napisane w języku Python.
- Pierwszym z nich jest skrypt, który przygotowuje plik ze zdjęciem do rozpoznania.
- Na wejście tego skryptu podaje się plik z obrazkiem, natomiast zwraca on katalog, zawierający poszczególne, ciągłe, nie-czarne obszary tego obrazka, każdy w osobnym pliku.

O projekcie

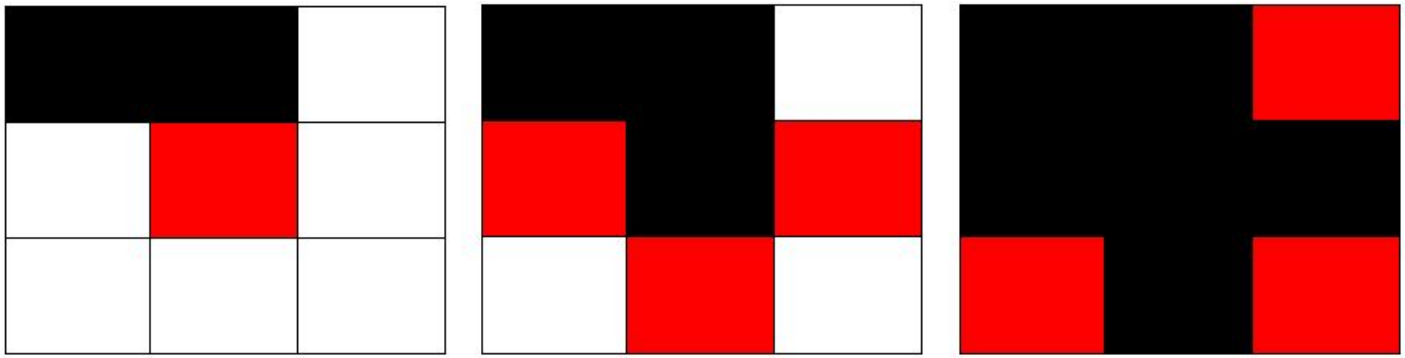


Rysunek: Zasada wyznaczania obszarów

O projekcie

- W skrypcie tym, posługujemy się funkcją rekursywną do mapowania obszarów.
- Funkcja ta zapisuje pierwszy nie-czarny rozpatrywany piksel, a następnie wywołuje się sama na każdym nie-czarnym sąsiedzie rozpatrywanego piksela.

O projekcie



Rysunek: Powyższy schemat przedstawia działanie funkcji rekursywnej. Piksel na którym wywołujemy funkcję (zaznaczony na czerwono) jest zapisywany do osobnego obrazka, a na pierwotnym zapełniany jest na czarno. W kolejnym kroku funkcja jest wywoływana na każdym sąsiednim nie-czarnym pikselu, aż do momentu w którym żaden nie zostaje.

O projekcie

- Oprócz plików z obrazkami przygotowuje także specjalny plik tekstowy opisujący, które obszary sąsiadują ze sobą, to znaczy stykają się w więcej niż jednym punkcie.
- Plik ten musi być uzupełniony ręcznie, przed przystąpieniem do kolejnego kroku.

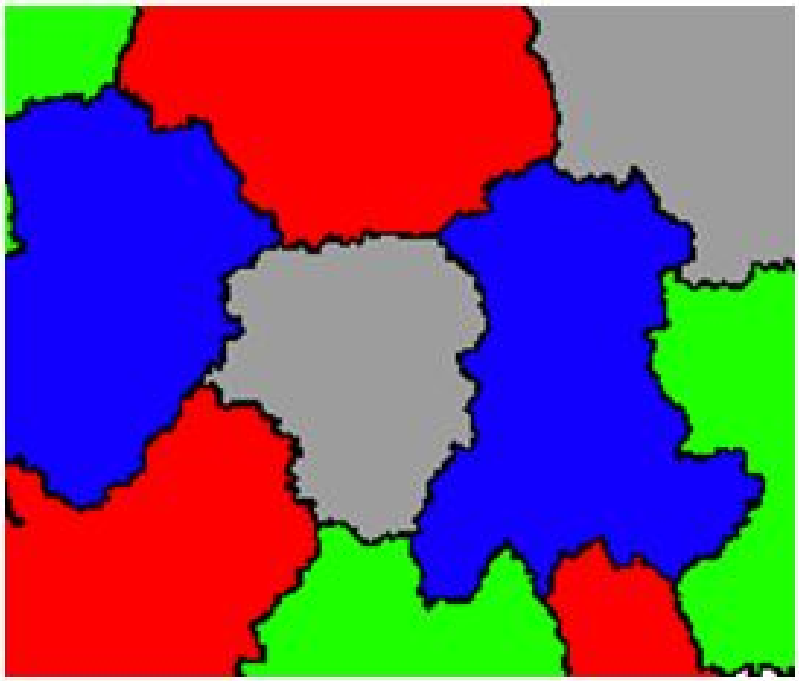
O projekcie

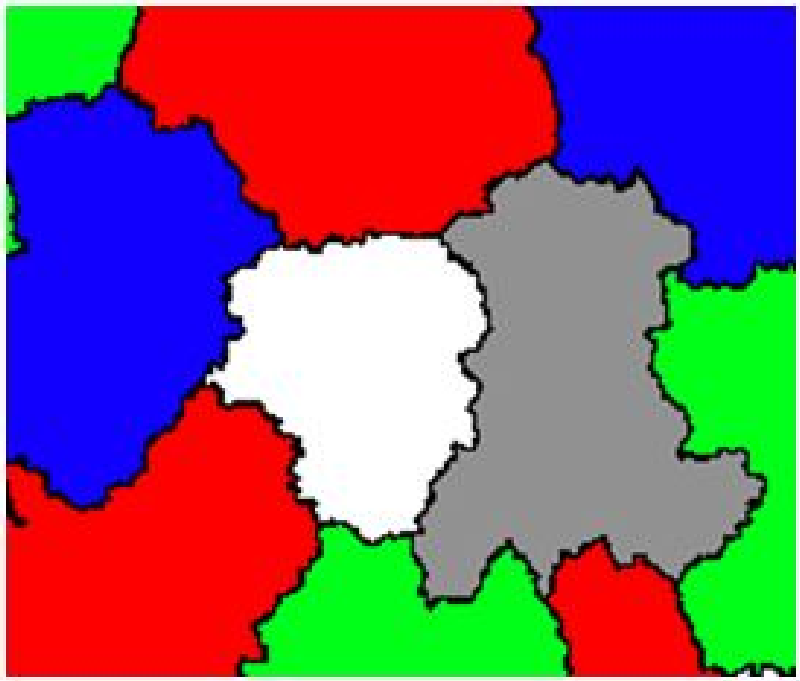
- Następny skrypt otwiera wszystkie pliki z obrazkami stworzonymi w tym kroku oraz stworzony ręcznie plik opisujący sąsiadujące obszary, i zamienia je na plik testowy zawierający jedną strukturę, gotową do kolorowania.
- Struktura ta zawiera wiele informacji potrzebnych nam do pokolorowana obrazka:
  - wielkość obrazka
  - listę pikseli stanowiących każdy obszar
  - listę sąsiadujących ze sobą obszarów
  - kolor na jaki pokolorowany jest każdy obszar (domyślnie czarny)
  - środek obszaru (średnia położenia wszystkich pikseli)
- Na wyjściu tego skryptu powstaje plik tekstowy zawierający wszystkie te informacje, gotowy do pokolorowania i wyświetlenia przez ostatni skrypt.

O projekcie

- Pierwszym założeniem jest to, że twierdzenie o czterech kolorach jest prawdziwe, to znaczy, każdą prawidłowo przygotowaną mapę obszarów ciągłych da się pokolorować 4 kolorami tak, żeby każdy z sąsiadujących obszarów miał inny kolor.
- Stąd nasuwa się pierwsza teza – jeżeli pokolorujemy mapę 4 kolorami ale nie wszystkie obszary zostaną pokolorowane, oznacza to że coś zostało zrobione źle.

O projekcie





Rysunek: Z lewej:poprawne kolorowanie, z prawej: błędne kolorowanie

O projekcie

- Wiemy, że kiedy jakiś obszar zostanie pokolorowany na wybrany kolor, to na ten sam kolor nie mogą zostać pokolorowane obszary sąsiadujące z nim.
- Oznacza to, że po pokolorowaniu jakiegoś obszaru, przed pokolorowaniem następnego obszaru tym samym kolorem, należy wykluczyć wszystkie obszary z nim sąsiadujące z tego zbioru.

O projekcie

- Ostatnim problemem, jaki napotkaliśmy podczas implementacji tej metody, jest dobór kolejności obszarów do kolorowania.
- Stosując nawet powyższe założenia, możemy zauważyć, że zły dobór kolejności kolorowania jednym kolorem może nie pozwolić na prawidłowe pokolorowanie obszaru.
- Najpewniejszym rozwiązaniem tego problemu, jest losowe dobieranie kolejności obszarów z kolorowaniem, a następnie, jeśli jakaś kolejność malowania nie doprowadzi do prawidłowego rezultatu, wylosowanie nowej kolejności z wykluczeniem tych, które nie dały prawidłowego wyniku.
- Widzimy więc, że taki algorytm jest bardzo zachłanny i skomplikowany.

O projekcie

- Nowe założenie
- Prawidłowe pokolorowanie map odbędzie się przez dobranie losowo punktu wejściowego, a następnie rozpatrzenie najbliższych obszarów.
- Taka metoda daje prawidłowe wyniki (przynajmniej dla obszarów pośród których przeważają obszary wypukłe).
- Prawdopodobnie podejście to zadziałało by dla każdej mapy, ale należało by wtedy określić odległość między obszarami w inny sposób, niż między ich środkami geometrycznymi.

Działanie programu

- Wizualizacja działania programu

Code Walkthrough

- Prezentacja kodu

Koniec

- Dziękujemy za uwagę