

Algoritmy Zachłanne 

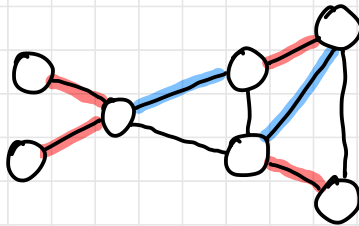
# Bomurka

Jest to algorytm wyznaczania MST, czyli maksymalnie rozpiętego drzewa dla grafu nieskierowanego.

Algorytm:

1. Dla każdego wierzchołka w grafie  $G$  dobierz najmniejszą krawędź łączącą ten wierzchołek z jakimś innym.
2. Teraz graf zawiera nie więcej niż  $\frac{|V|}{2}$  spójnych składowych. Scal wierzchołki utworzonych spójnych składowych do jednego wierzchołka, który reprezentuje tę wspólną składową nie tracąc przy tym krawędzi łączących różne wspólne składowe.
3. Wyrotujemy krawędzie 1 i 2 dopóki nie otrzymamy jedną wspólną składową.

Przykład



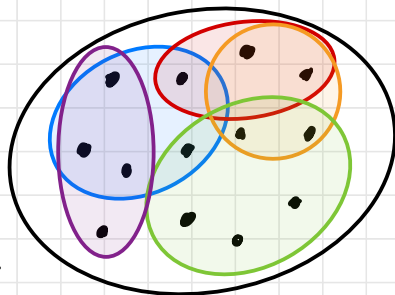
- iteracja 1
- iteracja 2

# Set Cover

Problem pokrycia zbiorów, czyli jakie podzbiory wybrać, by pokryć najwięcej elementów najmniejszym kosztem.

Strategie:

1. Wybierać zawsze taki, gdzie jest najwięcej elementów.
2. Wybierać zawsze najmniejszy podzbiór
3. Obliczać średnią cenę pokrycia elementów - daje najlepsze przybliżenie



Jakie kolory podzbiorów wybrać by koszt był najmniejszy?

## Strategia III

1. Liczymy koszt każdego zbioru
2. Wybieramy zbiór z najmniejszym kosztem i dodajemy do zbioru

Niech  $c_i$  = koszt  $i$ -tego podzbioru

$u_i$  = liczba niepokrytych elementów w tym zbiorze

Nasz algorytm minimalizuje stosunek  $\frac{c_i}{u_i}$ .

Wzimy jakiś optymalny wybór  $k$  pierwszych elementów w podzbiorach wybranych przez  $T_{OPT}$ .

Niech  $OPT$  oznacza cenę optymalną jaką musimy zapłacić za ten wybór. Zatem

$$\min_{T_{OPT}} \frac{c_i}{u_i} \leq \frac{\sum_{i \in T_{OPT}} c_i}{\sum_{i \in T_{OPT}} u_i} \leq \frac{OPT}{k}$$

Zatem na każdy kolejny ( $j$ -ty) element musimy wypłacić co najwyżej  $\frac{OPT}{n-j-1}$ .

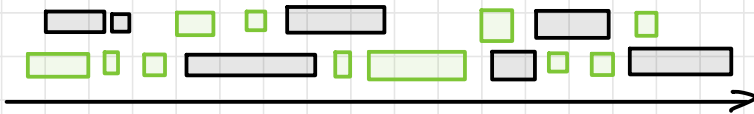
Zatem koszt wynosi



$$\sum_{j=1}^n \frac{OPT}{n-j-1} = OPT \cdot \underbrace{\sum_{j=1}^n \frac{1}{j}}_{\text{suma harmoniczna}} = OPT \cdot \log(n)$$

# Szeregowanie Zadań

Algorytm zadatanny rozwiązujący problem ustawienia zadań w taki sposób by wykonał ich jak najwięcej wygląda tak, że sortuje te zadania po czasie ich zakończenia i wybiera po kolei te które się jeszcze nie rozpoczęły.

Przykład:



-  - Zadania nie wybrane przez algorytm
-  - Zadania wybrane przez algorytm

Dowód poprawności:

Każdy algorytm zadatanny dowodzi się zazwyczaj w ten sposób, że bierze się optymalne rozwiązanie, a następnie pokazuje, że nasz algorytm wybrał niegorsze.

# Wydawanie Reszty

W jaki sposób wydać pieniądze, aby wykorzystać jak najmniej monet?

Algorytm w wersji zachłannej będzie zwracał od największego nominału (o ile się "zmieści") aż do najmniejszego. Czyli np.

13 PLN w nominatach





Wynik:  +  +  +  = 13 PLN

Zachłanna wersja algorytmu rozwiązujący ten problem nie daje poprawnego rozwiązania - tak samo jak Set Cover - daje on jedynie pewne przybliżenie jeśli pozostawimy nominałami nie jesteśmy pokryw n-1 nominał.

Dla przykładu założymy, że mamy nominały:



Założymy, że chcemy wydać 8 PLN. Nasz algorytm zachłannie wybierze ,  i na tym się zatrzyma.

Prawidłowa odpowiedź to oczywiście  $4 \cdot \text{} = 8 \text{ PLN}$ , lecz nasz algorytm zwróci tylko 7 PLN.