* **algorytm zachłanny** – algorytm, który w celu wyznaczenia rozwiązania na każdym kroku dokonuje zachłannego tj. najlepiej rokującego w danym momencie wyboru rozwiązania częściowego.

Przykład z życia:
np. dysponując monetami
2 zł   5 zł

WYDAĆ 6 ZŁ RESZTY:
~ algorytm zachłanny wybierze błędnie 1 x 5 zł jako pierwszy krok, bo jest bliżej wyniku.

algorytm zachłanny:

* **algorytm Prima** – wyznacza minimalne drzewo rozpinające (MDR-)

wymagania: - graf nieskierowany (bez kierunków)
- graf spójny (dla każdych dwóch wierzchołków istnieje droga pomiędzy nimi)

1) WYBIERZ JEDEN WIERZCHOŁEK, PODAJ GO DO ZBIORU ODWIEDZONYCH
2) DODAJ JEDEN Z SĄSIADUJĄCYCH WIERZCHOŁKÓW DO ZBIORU, ALE WYBIERZ TEN Z NAJTAŃSZĄ DROGĄ. POWTÓRZ.

* sąsiadujące wierzchołki są sąsiadujące względem drzewa, a nie wybranego ostatnio punktu.

---

* **algorytmy typu „dziel i zwyciężaj"**

PROBLEM → 1 NIEZALEŻNY PODPROBLEM, 2 NIEZALEŻNY PODPROBLEM, n NIEZALEŻNY PODPROBLEM → ROZWIĄZANIE

* **programowanie dynamiczne**
  * można używać do zależnych podproblemów tj. rozwiązanie problemu zapisuje i zapamiętuje
  * zwykle stosowane do problemów optymalizacyjnych

---

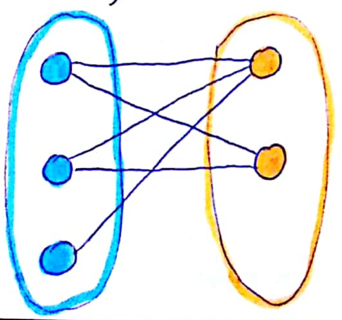
grafy:

graf ważony - każdej krawędzi przyporządkowana jest waga

graf etykietowany - każdy wierzchołek ma etykietę (np. Moskwa, Tel Aviv)

graf spójny - dla każdych dwóch wierzchołków istnieje droga pomiędzy nimi

graf dwudzielny - wierzchołki można podzielić na dwa podzbiory (końce każdej krawędzi mają inne kolory)



cykl Hamiltona - droga złożona z krawędzi grafu
* dokładnie raz przechodzi przez każdy WIERZCHOŁEK
* wraca do wierzchołka początkowego

graf planarny (płaski) - można go narysować na płaszczyźnie tak aby żadne krawędzie nie przecinały się.

Liczba chromatyczna grafu G (oznaczenie : $\chi(G)$)
- to najmniejsza liczba kolorów, którymi można pokolorować wierzchołki grafu tak aby końce każdej krawędzi miały inne kolory

indeks chromatyczny grafu G (oznaczenie : $\chi'(G)$)
- to najmniejsza liczba kolorów, którymi można pokolorować krawędzie grafu tak aby KAŻDE DWIE KRAWĘDZIE PRZY WSPÓLNYM WIERZCHOŁKU MIAŁY INNY KOLOR.

graf r-regularny - każdy wierzchołek ma stopień r

spójność (wierzchołkowa) grafu - to taka liczba, że usunięcie k wierzchołków spowoduje, że graf przestanie być spójny (lub zostanie zredukowany do jednego wierzchołka). Usunięcie k-1 wierzchołków pozostawi graf spójny.
* oznaczenie : $\kappa(G)$

---

cykl Eulera - taki cykl, że przechodzi przez każdą krawędź i wierzchołek dokładnie raz.

KIEDY JEST W GRAFIE?
* graf jest spójny
* każdy wierzchołek ma parzysty stopień

dla grafu skierowanego powinno wychodzić tyle samo krawędzi ile wchodzi!

*** algorytm Fleury'ego** - odszukiwanie w grafie eulerowskim cyklu Eulera

1. WYBIERZ WIERZCHOŁEK
2. USUWAJ PRZECHODZONE KRAWĘDZIE I WIERZCHOŁKI IZOLOWANE (POZOSTAŁE)
3. PRZECHODŹ PRZEZ MOST TYLKO GDY NIE MASZ INNEJ MOŻLIWOŚCI

Scanned with CamScanner

2 PODSTAWY INFORMATYKI

Trasa vs ścieżka vs droga

* skończony ciąg krawędzi

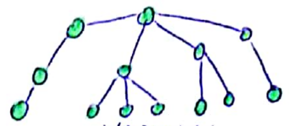
DŁUGOŚĆ TRASY: LICZBA WIERZCHOŁKÓW

* trasa, ALE WSZYSTKIE KRAWĘDZIE SĄ RÓŻNE

* cykl - to zamknięta ścieżka

* ścieżka, ALE WSZYSTKIE WIERZCHOŁKI SĄ TEŻ RÓŻNE (oprócz początkowego)

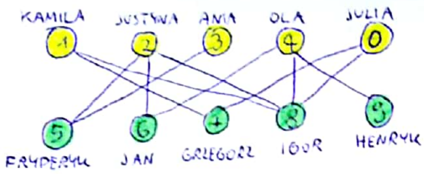
drzewo



WARUNKI:
* graf nieskierowany
* graf acykliczny (nie można chodzić w kółko)
* graf spójny (droga między każdymi wierzchołkami)

---

* Problem kojarzenia małżeństw - twierdzenie Halla



* związki są heteroseksualne → graf dwudzielny

WARUNEK KONIECZNY I WYSTARCZAJĄCY

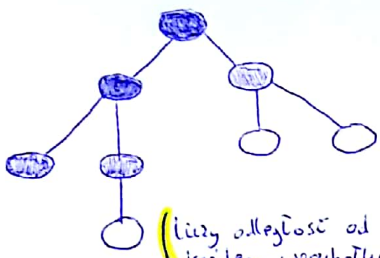
każdy podzbiór k panien akceptuje k kawalerów

---

* Algorytm przeszukiwania wszerz

* rozpoczyna się z danego wierzchołka (korzeń s)
* odwiedza wszystkie wierzchołki sąsiadujące z aktualnym
* gdy zostały odwiedzone powtarza dla sąsiadów

WYNIK: drzewo przeszukiwania wszerz
- o korzeniu s
- zawiera wszystkie wierzchołki osiągalne z s.



(liczy odległość od s do każdego wierzchołka)

- do każdego z wierzchołków prowadzi jedna ścieżka z s, która jest ścieżką najkrótszą.

założenia w 4 PODSTAWY INFORMATYKI ↳

---

* Problem chińskiego listonosza

* listonosz wyrusza z poczty
* musi przejść po każdej ulicy CONAJMNIEJ RAZ
* wraca na pocztę.

DROGA MA BYĆ NAJKRÓTSZA

Rozwiązanie:

* cykl Eulera - jeśli takowy posiada (stopień każdego wierzchołka jest parzysty)
* cykl w którym suma krawędzi wybranych więcej niż raz jest najmniejsza.

Założenie:

* aby rozwiązanie problemu istniało graf musi być spójny (droga między każdymi dwoma wierzchołkami)

Algorytm ogólny

1) SPRAWDŹ CZY GRAF JEST SPÓJNY (MUSI BYĆ)
2) WYSZUKAJ WIERZCHOŁKI O NIEPARZYSTYCH STOPNIACH
3) JEŚLI ICH LICZBA TO ZERO, IDŹ DO PUNKTU 7
4) WYZNACZ DIJKSTRA NAJKRÓTSZE ŚCIEŻKI POMIĘDZY NIEPARZYSTYMI WIERZCHOŁKAMI
5) SKOJARZ TE WIERZCHOŁKI W PARY O NAJMNIEJSZEJ SUMIE WAG.
6) KRAWĘDZIE WCHODZĄCE W SKŁAD WYZNACZONYCH ŚCIEŻEK ZDUBLUJ W GRAFIE
7) WYZNACZ CYKL EULERA
8) KONIEC

Scanned with CamScanner

* Problem komiwojażera

* chce on dotrzeć do każdego z zaplanowanych miast
* następnie wrócić do początkowego

graf pełny – (każda krawędź łączy się z każdą inną krawędzią) → dowolny ~~graf pełny~~ posiada conajmniej jeden cykl Hamiltona

Algorytmy: **dokładność** vs. **szybkość**

(* np. obliczenie w grafie całkowitej długości wszystkich cykli Hamiltona)

(* algorytmy przybliżone)

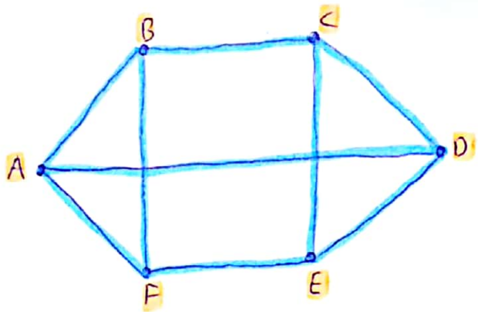
---

* Problem niezawodności sieci

* rozważane sieci: komputerowa lub drogowa

wierzchołki – punkty węzłowe

krawędzie – bezpośrednie połączenie dwóch węzłów



[**spójność wierzchołkowa** – (usunięcie k wierzchołków spowoduje) niespójność

MINIMALNA LICZBA AWARII W WĘZŁACH KTÓRA SPOWODUJE AWARIĘ SIECI (zerwane połączenie między pewnymi węzłami)]

[**spójność krawędziowa** – **MINIMALNA LICZBA AWARII POMIĘDZY WĘZŁAMI KTÓRA SPOWODUJE AWARIĘ SIECI**]

niezawodność sieci – **maksymalna liczba awarii, których wystąpienie nie spowoduje awarii sieci**

* czym większa spójność – tym większa niezawodność – ale też większy koszt
* można więc założyć daną minimalną niezawodność i próbować zbudować jak najtańszą sieć z tą niezawodnością

Techniczne: k – niezawodność sieci, n – liczba węzłów

Szukam grafu o n wierzchołkach, możliwie najmniejszej liczbie krawędzi i spójności wierzchołkowej lub spójności krawędziowej równej k.

---

* Znajdowanie grubości grafu

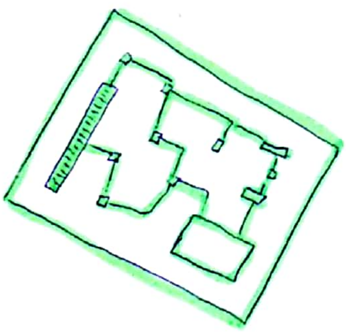
– najmniejsza liczba planarnych podgrafów na które można podzielić dany graf.

* KAŻDY GRAF PLANARNY MA WIĘC GRUBOŚĆ RÓWNĄ 1.

Zastosowanie: * projektowanie obwodów drukowanych

wierzchołki – części układu elektronicznego

krawędzie – kabelki



* przewody nie mogą się przecinać (brak izolacji)

grubość grafu – minimalna liczba ~~[illegible]~~ obwodów drukowanych potrzebnych do złożenia układu

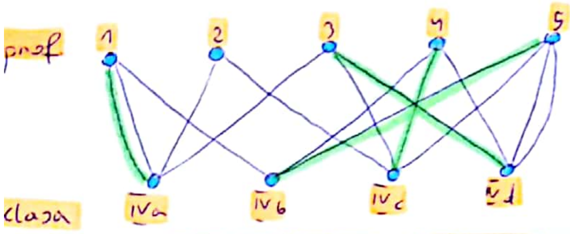
Scanned with CamScanner

* Znajdowanie indeksu chromatycznego

(najmniejsza liczba kolorów, jaką można pokolorować krawędzie TAK ABY KAŻDE DWIE PRZY TYM SAMYM WIERZCHOŁKU MIAŁY INNE KOLORY)

Problem układania planu zajęć

* dany zbiór m nauczycieli i n klas
* nauczyciele muszą odbyć konkretną ilość godzin z każdą klasą
* SZUKANE: NAJMNIEJSZA LICZBA GODZIN ABY WSZYSTKO ZOSTAŁO ODBYTE
* ZAŁOŻENIA
  - nauczyciel może uczyć daną klasę (tylko jedną) w godzinę
  - klasa może być uczona tylko przez jednego nauczyciela



	kolor godzina 1	kolor godzina 2	kolor godzina ...
prof 1	IV a	IV b	...
prof 2		IV c	...
prof ...	...	...	...

Rozwiązanie:

* graf dwudzielny
* tyle krawędzi połączenia ile godzin ma odbyć z daną klasą
* koloruję tak, aby każde dwie krawędzie mające wspólny koniec miały różne kolory
* KOLORY = GODZINY
* minimalna liczba kolorów ? min. liczba godzin

Przeszukiwanie wszerz - założenia algorytmu:

* wierzchołki są na początku białe
* jeśli wierzchołek jest czarny to wszyscy jego sąsiedzi byli już odwiedzeni
* wierzchołki szare mogą mieć białych sąsiadów
* graf jest reprezentowany przez listy sąsiedztwa, każdy wierzchołek u ma przypisaną listę Ls[u].
* zapamiętuję kolor wierzchołka i jego poprzednika
* jeśli nie ma poprzednika to wartość to -1.
* liczę odległość źródła do wierzchołka
* szare wierzchołki zapamiętuję w kolejce FIFO.

---

* Algorytm przeszukiwania w głąb

kolory wierzchołków:

BIAŁY - nieodwiedzony

SZARY - odwiedzony po raz pierwszy

CZARNY - przetworzony (wszyscy sąsiedzi przeszukani)

1) WYBRAĆ WIERZCHOŁEK STARTOWY
2) PRZEJŚĆ DO DOWOLNEGO SĄSIADA NIEODWIEDZONEGO
3) JEŚLI NIE MA JUŻ ŻADNEGO OZNACZAM JAKO PRZETWORZONY, WRACAM DO POPRZEDNIKA I PONAWIAM PKT 2.
4) JEŚLI ZOSTAŁY JESZCZE WIERZCHOŁKI NIEODWIEDZONE ZACZYNAM OD NOWA

* każdemu wierzchołkowi przypisuje się dwie etykiety

d[v] - nr kroku obliczeń w którym v jest odwiedzany po raz pierwszy (kolorowanie na szaro)

f[v] - nr kroku w przeszukiwaniu, w którym kończy się badanie listy sąsiedztwa wierzchołka v. (kolorowanie na czarno) (przetworzenie wierzchołka)

* należy wprowadzić zmienną czas
  - wyzerować na początku
  - zwiększyć zawsze gdy nowy wierzchołek (o 1)
  - zwiększyć zawsze gdy przetworzenie wierzchołka

  } zawsze gdy zmiana koloru czas = czas+1;

---

* GDY KOŃCZĄ SIĘ MOŻLIWOŚCI KOŃCZYMY TO DRZEWO I WYBIERAMY INNY PUNKT

* DRZEW PRZESZUKIWANIA W GŁĄB JEST TYLE ILE WYBRANO WIERZCHOŁKÓW STARTOWYCH

} LAS PRZESZUKIWANIA W GŁĄB

Scanned with CamScanner