Algorytmy i Struktury Danych

Wykład 13

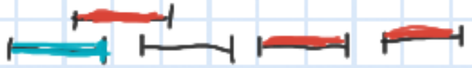
Algorytmy zachłanne — podejmuj "najlepszą lokalną decyzję" i miej nadzieję, że to zadziała

Uwagi

- często nie dają optymalnego rozwiązania
  ↳ ale czasem dają!
- często bardzo szybkie
- czasem dają dobre rozwiązanie przybliżone

Dowód poprawności

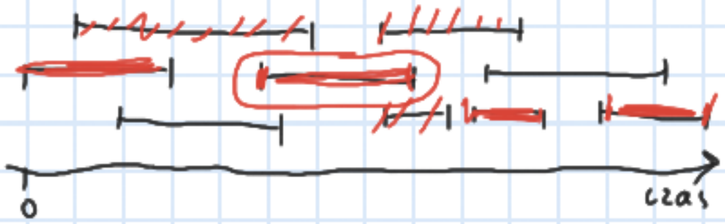
- weźmy pewne rozwiązanie optymalne



- dołóżmy do tego rozwiązania przedział kończący się najwcześniej
- usuńmy powstałe konflikty → jest dokładnie jeden

Problem wyboru zadań

Dane:

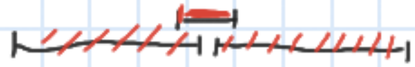


4

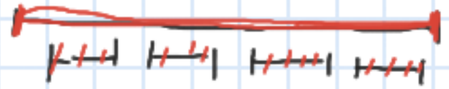
Zadanie: Wybrać jak najwięcej niepokrywających się przedziałów

Pomysły

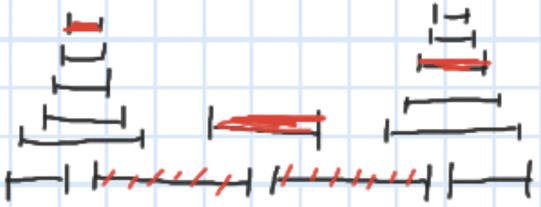
- wybieramy najkrótsze przedziały



- wybieramy przedział kończący się najwcześniej ← rozwiązanie poprawne
- wybieramy przedział zaczynający się najwcześniej



- wybieramy krótszy spośród "kończący się najwcześniej", "startujący najpóźniej"



- wybieramy przedział generujący najmniej konfliktów

Kody Huffmana

Kod binarny z symbolami innej długości nie musi być optymalny

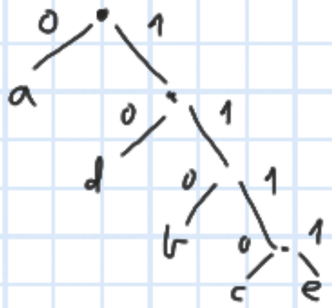
symbole:	a	b	c	d	e
częstości	700	200	120	300	10
kod	000	001	010	011	100

$1330 \times 3 = 3990$

inny kod	0	110	1110	10	1111

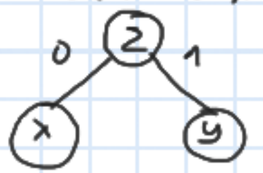
$\underbrace{700 + 600}_{1300} + \underbrace{480 + 600 + 40}_{1120} = 2420$

Kod prefiksowy – kod żadnego symbolu nie jest prefiksem innego kodu



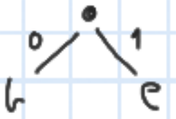
Naturalny algorytm zachłanny tworzenia kodu

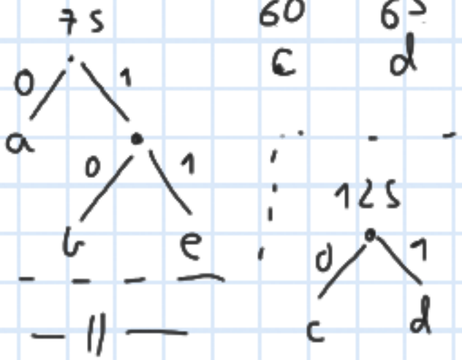
- weź dwa symbole x i y o najmniejszej częstości ---> (x) (y)
- połącz je w jeden nowy symbol z o sumie częstości x i y ---> $f(z) = f(x) + f(y)$



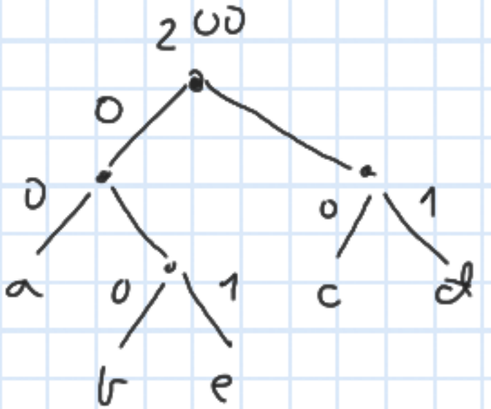
Przykład

	50	5	60	65	20
krok 0:	a	b	c	d	e
krok 1	a		c	d	
krok 2			c	d	
krok 3					





Krok 4:



Dlaczego nasz algorytm jest poprawny?

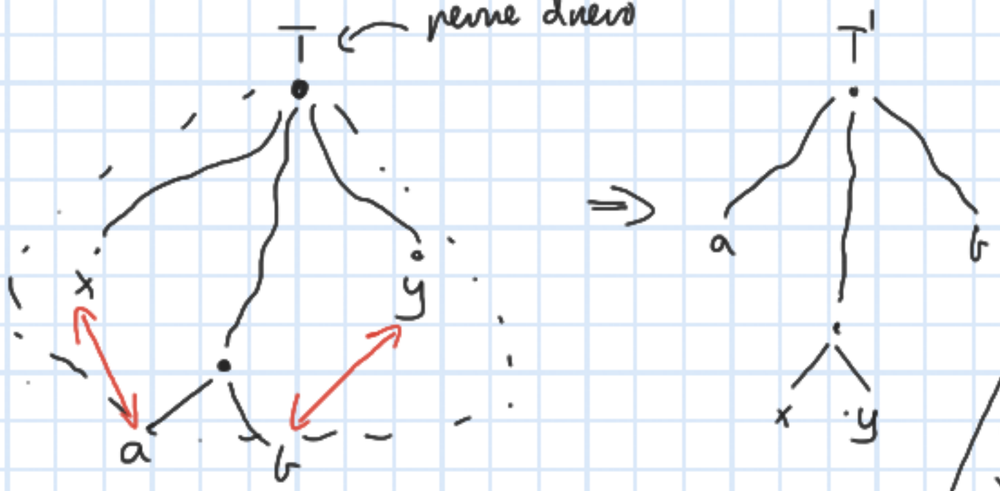
T - drzewo kodujące

$B(T)$ - koszt drzewa

$B(T) = \sum_s f(s) \cdot d_T(s)$

s - symbole; $f(s)$ - częstości s; $d_T(s)$ - długość kodu s w drzewie T

Krok 1: Dwa najrzadsze symbole można umieścić we wspólnym węźle

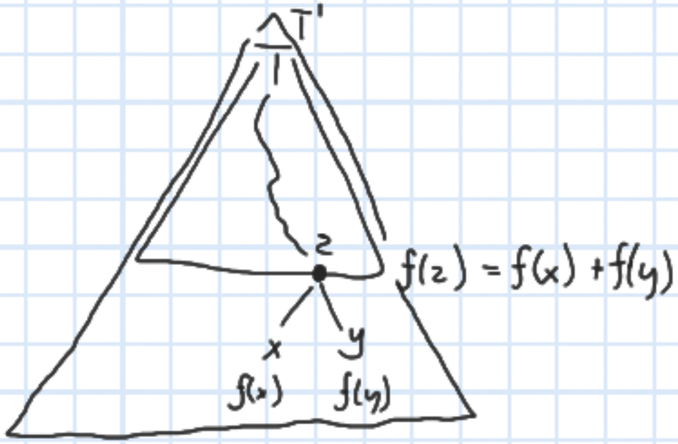


a, b - symbole na najdłuższej ścieżce

x, y - najrzadsze symbole

Krok 2: Optymalna podstruktura

$$B(T') = B(T) + f(x) + f(y)$$



$$\begin{aligned} B(T') = B(T) &- f(a)\,d_T(a) + f(a)\,d_T(x) \\ &- f(b)\,d_T(b) + f(b)\,d_T(y) \\ &- f(x)\,d_T(x) + f(x)\,d_T(a) \\ &- f(y)\,d_T(y) + f(y)\,d_T(b) \end{aligned}$$

$B(T) +$

$$= d_T(a)\big(-f(a) + f(x)\big) + d_T(b)\big(-f(b) + f(y)\big) + d_T(x)\big(-f(x) + f(a)\big) + d_T(y)\big(-f(y) + f(b)\big)$$

$B(T) +$

$$= \underbrace{\big(f(x) - f(a)\big)}_{\leq 0}\underbrace{\big[d_T(a) - d_T(x)\big]}_{\geq 0} + \underbrace{\big(d_T(b) - d_T(y)\big)}_{\geq 0}\underbrace{\big(f(y) - f(b)\big)}_{\leq 0}$$

$$\leq B(T)$$

Problem plecakowy (dyskretny / ciągły)

Dane: $p_1, \ldots, p_n$ — przedmioty

$v(p_1), \ldots, v(p_n)$ — wartość przedmiotu

$m(p_1), \ldots, m(p_n)$ — waga przedmiotu

M — ile unosi złodziej

Zadanie: Wybrać przedmioty, których masa nie przekracza M i których wartość jest maksymalna

dyskretny — przedmiot bierzemy lub nie

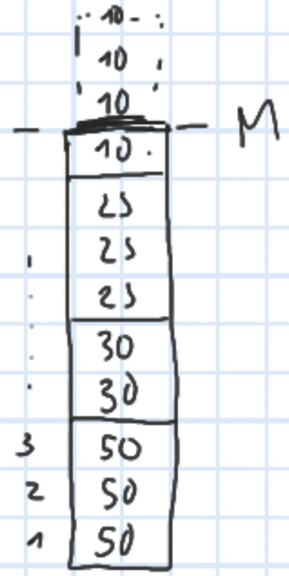
ciągły — można brać fragmenty przedmiotów

Wersja dyskretna → algorytm dynamiczny

Wersja ciągła → algorytm zachłanny

↓

wybieramy przedmioty o najlepszym stosunku ceny do wagi

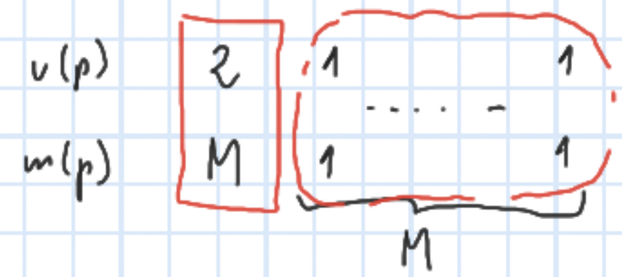


Algorytm zachłanny działa dla prob. ciągłego, ale nie dla dyskretnego

	1	2	3	
$v(p)$:	70	55	35	$M = 100$
$m(p)$:	70	60	40	

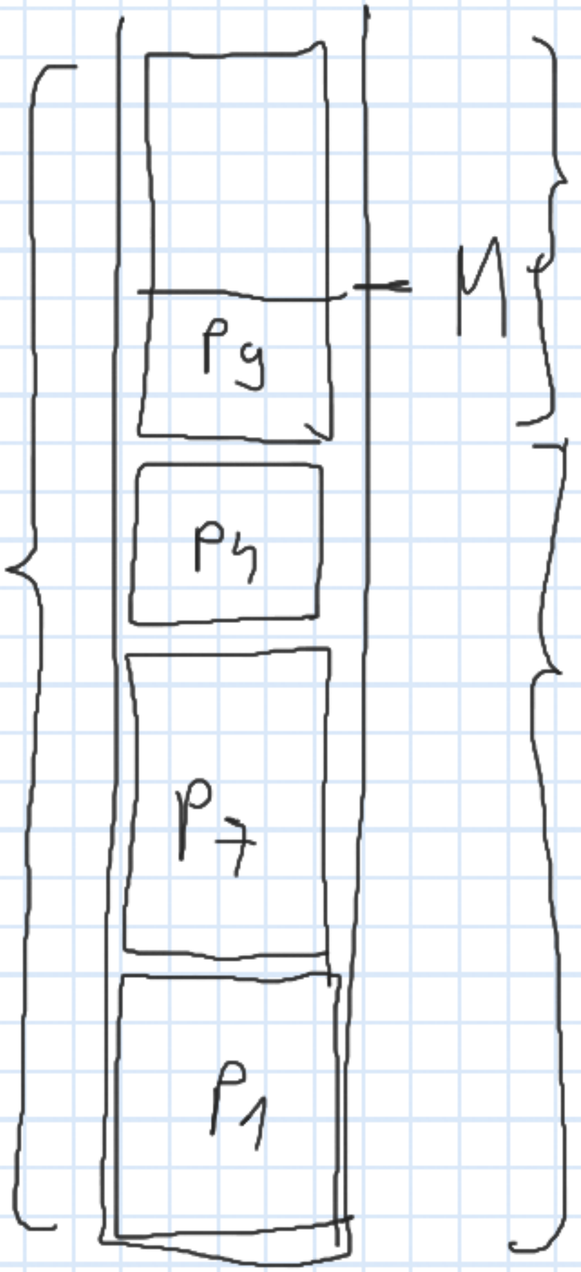
$v(p)$	2	$M-1$	M
$m(p)$	2	M	

Algorytm zachłanny 2: Wybieray najcenniejsze przedmioty



X

suma wartości tych przedmiotów $\geq$ wartości wozu, dyskiet.



$Y_1 + Y_2 = X$

$\Downarrow$

$Y_1 \geq \frac{X}{2}$

lub

$Y_2 \geq \frac{X}{2}$