Prace Naukowe Wyższej Szkoły Pedagogicznej w Częstochowie
Matematyka VI, Częstochowa 1999

Algorytmy zachłanne

Bożena Woźna

Algorytmy zachłanne są stosowane do rozwiązywania problemów optymalizacyjnych, w których osiągnięcie optymalnego rozwiązania wymaga podejmowania wielu decyzji. Algorytm zachłanny wykonuje zawsze działanie, które wydaje się w danej chwili najkorzystniejsze. Wybiera zatem lokalnie optymalną możliwość w nadziei, że doprowadzi ona do globalnie optymalnego rozwiązania.

Jest wiele problemów, w których metoda zachłanna prowadzi do optymalnego rozwiązania. Nie zawsze jest jednak łatwo stwierdzić, czy zastosowanie algorytmu zachłannego jest rozwiązaniem poprawnym, tzn. czy otrzymane rozwiązanie jest zawsze optymalne.

Aby zilustrować działanie metody zachłannej, przedstawię "*problem wyboru zajęć*", czyli problem przydzielenia dostępu do zasobu wykorzystywanego podczas wykonywania pewnych zajęć. Pokażę, że algorytm zachłanny stanowi prostą a zarazem elegancką metodę znajdowania największego zbioru wzajemnie zgodnych czynności. Przedstawię również zachłanne rozwiązanie dla następującego problemu optymalizacyjnego:

Dany jest zbiór zajęć, które mają się odbywać
w pewnej liczbie sal wykładowych.
Należy wyznaczyć taki przydział zajęć do sal,
aby liczba użytych sal była najmniejsza.

Pokażę, że problem wydawania reszty za pomocą jak najmniejszej liczby monet, przy odpowiednich założeniach o nominałach, można również rozwiązać stosując metodę zachłanną.

Następnie omówię podstawowe własności metody zachłannej: **własność wyboru zachłannego** oraz **własność optymalnej podstruktury** i na podstawie *ciągłego problemu plecakowego* wyjaśnię na czym one polegają, po czym przedstawię podstawy teorii obiektów kombinatorycznych zwanych MATROIDAMI, dla których algorytm zachłanny zawsze generuje optymalne rozwiązanie.

Udowodnię, że:

1. Uporządkowana para (S, Φ_k), gdzie S jest dowolnym skończonym niepustym zbiorem, a Φ_k jest zbiorem wszystkich podzbiorow S o rozmiarze co najwyżej k, dla $k \leq |S|$, jest matroidem.
2. Uporządkowana para (S, Φ), gdzie S jest zbiorem kolumn danej macierzy T o wartościach rzeczywistych i wymiarze $n \times n$, a $A \in \Phi$ wtedy i tylko wtedy, gdy kolumny należące do A są liniowo niezależne, jest matroidem.
3. Uporządkowana para (S, Φ), gdzie S jest skończonym niepustym zbiorem, a $\Phi = \{A : | A \cap S_i | \leq 1 \; dla \; i = 1, 2, \ldots, k$, gdzie $S_1, S_2, \ldots, S_k$ jest podziałem S na niepuste rozłączne podzbiory $\}$, jest matroidem.

Na koniec, zilustruję zastosowanie matroidów do problemu szeregowania zadań o jednostkowym czasię trwania z karami za przekroczenie terminu wykonania.

Problem wyboru zajęć

Niech będzie dany zbiór $S = \{1, 2, \ldots, n\}$ składający się z n proponowanych zajęć, do których mają być przydzielone zasoby, takie jak np. sala wykładowa, w której może się odbywać w danej chwili tylko jedno z tych zajęć.
Każde zajęcie ma swój **czas rozpoczęcia** s_i oraz **czas zakończenia** f_i, takie że $s_i \leq f_i$. Jeśli zajęcie o numerze i zostanie wytypowane, to zajmuje zasób przez prawostronnie otwarty przedział czasu $< s_i, f_i)$.
Zajęcia o numerach i oraz j są zgodne (nie zakłócają się), jeśli przedziały $< s_i, f_i)$ oraz $< s_j, f_j)$ nie zachodzą na siebie tzn. jeśli $s_j \geq f_i$.
Problem wyboru zajęć polega na wyborze największego podzbioru parami zgodnych zajęć.

Poniżej znajduje się `Algorytm zachłanny dla problemu wyboru zajęć`. Bez utraty ogólności zakładam, że zajęcia są uporządkowane ze względu na czas zakończenia: $f_1 \leq f_2 \leq \ldots \leq f_n$. W przeciwnym razie można najpierw uporządkować je w takiej kolejności, używając do tego dowolnej metody sortowania zbióru n elementowego. W poniższej procedurze zakładam, że s - wektor czasów rozpoczęcia oraz f - wektor czasów zakończenia są reprezentowane jako tablice.

```
GREEDY - ACTIVITY - SELECTOR(s, f)
1. n := length[s];
2. A := {1};
3. j := 1;
4. for i := 2 to n do
5.     if (s_i ≥ f_j) then
6.         A := A ∪ {i};
7.         j := i;
8.     endif;
9. endfor;
10. return A;
```

Działanie tego algorytmu jest następujące:
W zbiorze A są gromadzone wybrane zajęcia. Zmienna j zawiera numer ostatnio dodanego do A zajęcia. Ponieważ zajęcia są rozpatrywane w porządku rosnącego czasu zakończenia, więc f_j jest zawsze największym czasem zakończenia zajęcia należącego do A.
W wierszach 2-3 jest wybrane zajęcie 1, jednoelementowy zbiór $\{1\}$ staje się wartością zmiennej A, a zmiennej j zostaje przypisany numer tego zajęcia.
W wierszach 4-9 są kolejno rozpatrywane wszystkie zajęcia i, każde z nich zostaje dołączone, jeżeli jest zgodne ze wszystkimi dołączonymi dotychczas zajęciami. Aby stwierdzić, czy zajęcie i jest zgodne z każdym zajęciem ze zbióru A, wystarczy sprawdzić czy jego czas rozpoczęcia s_i nie jest wcześniejszy niż czas zakończenia f_j zajęcia ostatnio dodanego do A. Jeśli zajęcie i jest zgodne, to w wierszach 6-7 zostaje ono dodane do zbioru A oraz jest aktualizowana wartość j.

Zajęcie wybrane przez GREEDY-ACTIVITY-SELECTOR ma zawsze najwcześniejszy czas zakończenia wśród zajęć, które mogą być dołączone bez zakłócenia zgodności zbióru A.
Ten wybór jest "zachłanny" w tym sensie, że pozostawia możliwie najwięcej swobody przy wyborze pozostałych zajęć.

Twierdzenie 1 *Algorytm GREEDY - ACTIVITY - SELECTOR generuje rozwiązanie problemu wyboru zajęć o największym rozmiarze.*

Dowód Niech $S = \{1, 2, .., n\}$ będzie zbiórem zajęć. Ponieważ zakładałam, że zajęcia są uporządkowane według czasu zakończenia, to zajęcie 1 ma najwcześniejszy czas zakończenia. Wykażę, że istnieje optymalne rozwiązanie, którego konstrukcję można rozpocząć od wyboru zachłannego, tj. od wyboru zajęcia 1.
Załóżmy, że $A \subseteq S$ jest optymalnym rozwiązaniem. Niech elementy w A

będą uporządkowane ze względu na ich czas zakończenia. Przyjmijmy też, że pierwszym zajęciem należącym do A jest zajęcie o numerze k.
Jeśli $k = 1$, to tworzenie rozwiązania A można rozpocząć od podjęcia decyzji zachłannej. Jeżeli natomiast $k \neq 1$, to wykażę, że istnieje inne optymalne rozwiązanie $B \subseteq S$, którego konstrukcję można rozpocząć od zachłannego wyboru, tj. od wyboru zajęcia nr 1.
Niech $B = A - \{k\} \cup \{1\}$. Ponieważ $f_1 \leq f_k$, to zajęcia w B są zgodne. Ponieważ B ma tyle samo zajęć co A, jest więc także rozwiązaniem optymalnym. Stąd wnioskuję, że zawsze istnieje optymalne rozwiązanie, którego konstrukcję można rozpocząć od wyboru zachłannego.
Gdy już został dokonany zachłanny wybór zajęcia nr 1, to problem sprowadza się do znalezienia optymalnego rozwiązania problemu wyboru zajęć dla zbioru tych zajęć z S, które są zgodne z zajęciem nr 1.
Jeśli więc A jest optymalnym rozwiązaniem całego problemu S, to $A' = A - \{1\}$ jest optymalnym rozwiązaniem problemu optymalnego wyboru zajęć dla zbioru $S' = \{ i \in S \ : \ s_i \geq f_1\}$.
Przypuśćmy, że tak nie jest. Istnieje wtedy rozwiązanie B' problemu S' o większej liczbie zajęć niż A'. Dodając do B' zajęcie 1 otrzymujemy rozwiązanie B problemu S o większej liczbie niż A, co przeczy optymalności A.
Widać stąd, że po każdym wyborze zachłannym pozostaje do rozwiązania problem optymalizacyjny tego samego typu co początkowo. Korzystając z indukcji względem liczby podjętych decyzji i dokonując zachłannego wyboru w każdym kroku, otrzymamy optymalne rozwiązanie. ∎

Problem przydziału zajęć do minimalnej liczby sal

Dany jest zbiór zajęć, które mają się odbywać
w pewnej liczbie sal wykładowych.
Należy wyznaczyć taki przydział zajęć do sal,
aby liczba użytych sal była najmniejsza.

Metoda zachłanna rozwiązujaca ten problem

Niech będzie dany zbiór $S = \{1, 2, \ldots, n\}$ składający się z n zajęć, do których mają być przydzielone sale wykładowe. W określonej sali może się odbywać w danej chwili tylko jedno zajęcie.
Każde zajęcie ma swój czas rozpoczęcia s_i oraz czas zakończenia f_i, taki że $s_i \leq f_i$. Zajęcia o numerach i oraz j są zgodne, jeżeli $s_j \geq f_i$.
Niech zajęcia w zbiorze S będą uporządkowane niemalejąco ze względu na ich czas rozpoczęcia. Jeżeli znajdą się dwa zajęcia i oraz j, takie że $s_i = s_j$, to wówczas w uporządkowaniu pierwsze będzie to zajęcie, które ma dłuższy czas trwania.
W poniższej procedurze zakładam, ze s - wektor czasów rozpoczęcia oraz

f - wektor czasów zakończenia są reprezentowane jako tablice.
Zadaniem procedury pomocniczej GREEDY - SELECTOR jest wybranie ze zboru S podzbioru zawierającego wszystkie parami zgodne zajęcia, wśród których jest zajęcie nr 1.
Działanie tej procedury jest analogiczne jak działanie procedury GREEDY - ACTIVITY - SELECTOR. Różnica polega jedynie na tym, że GREEDY - SELECTOR pobiera zajęcia uporządkowane w sposób opisany wyżej, a GREEDY -ACTIVITY - SELECTOR pobierała zajęcia uporządkowane zgodnie z czasem ich zakończenia.

```
GREEDY − SELECTOR(s, f)
n := length[s];
A := {1};
j := 1;
for i := 2 to n do
    if (s_i ≥ f_j) then
        A := A ∪ {i};
        j := i;
    endif;
endfor;
return A;
```

```
GREEDY − ROOM − SELECTOR(s, f)
1. k := 1;
2. A := {1, 2, ..., length[s]};
3. while (A ≠ ∅) do
4.     p[k] := GREEDY − SELECTOR(s, f);
5.     A := A − p[k];
6.     if (A ≠ ∅) then
7.         k := k + 1;
8.     endif;
9. endwhile;
10. return k;
```

Działanie algorytmu GREEDY - ROOM - SELECTOR jest następujące:

Zmienna k pamięta liczbę sal potrzebną do rozdzielenia wszystkich zajęć ze zbioru S. W drugim kroku do zbioru A zostają dodane wszystkie zajęcia ze zbioru S. W wierszach 3-8 są wybierane kolejno ze zbioru A, za pomocą procedury GREEDY - SELECTOR, wszystkie zajęcia zgodne,

wśród których jest zajęcie nr 1 i umieszczone zostają w sali $p[k]$. Następnie zbiór A zostaje pomniejszony o zajęcia wybrane oraz zwiększa się liczba sal. Czynności w wierszach 4 - 8 wykonywane są dopóty, dopóki wszystkie zajęcia nie zostaną rozdzielone.
Liczba sal wyznaczona przez GREEDY-ROOM-SELECTOR jest zawsze najmniejsza. Ten wybór jest zachłanny w tym sensie, że pozostawia w danej sali minimalną ilość niewykorzystanego czasu.

Twierdzenie 2 *Algorytm GREEDY - ROOM - SELECTOR generuje optymalne rozwiązanie problemu przydziału zajęć do jak najmniejszej liczby sal.*

Dowód Niech S będzie zbiorem n zajęć. Niech ponadto zajęcia w zbiorze S będą uporządkowane niemalejąco ze wzgledu na czas ich rozpoczęcia. Z dwóch różnych zajęć o tym samym czasie rozpoczęcia, pierwsze w uporządkowaniu będzie to zajęcie, którego czas trwania jest dłuższy, tzn. ma późniejszy czas zakończenia.
Załóżmy, że k jest minimalną liczbą sal oraz $A_1, A_2, \ldots, A_k$ jest dowolnym optymalnym rozwiązaniem problemu przydziału zajęć do minimalnej liczby sal, takim że $A_i \cap A_j = \emptyset$ dla każdego $i \neq j$ oraz $i \leq k,\ j \leq k$.
Pokażę, że istnieje optymalne rozwiązanie, które można wygenerować przy pomocy algorytmu GREEDY - ROOM - SELECTOR.
Uporządkujmy zajęcia w każdym ze zbiorów A_i dla $i \leq k$ niemalejąco ze względu na ich czas rozpoczęcia. Następnie uporządkujmy zbiory A_i w następujący sposób:

1. Sala A_i będzie w uporządkowaniu przed salą A_j $(A_i < A_j)$ tylko wówczas, gdy pierwsze zajęcie (i_1) należące do sali A_i ma wcześniejszy czas rozpoczęcia niż zajęcie pierwsze (j_1) należące do sali A_j.

2. Jeżeli czas rozpoczęcia zajęcia i_1 (s_{i_1}) jest taki sam jak czas rozpoczęcia zajęcia j_1 (s_{j_1}), to sala A_i będzie w uporządkowaniu przed sala A_j tylko wtedy, gdy czas trwania zajęcia i_1 jest dłuższy od czasu trwania zajęcia j_1.

3. Jeżeli zajęcia i_1 oraz j_1 mają taki sam czas rozpoczęcia i taki sam czas zakończenia, to bierzemy pod uwagę kolejno zajęcia i_2 oraz j_2. Jeżeli i_2 oraz j_2 mają taki sam czas rozpoczęcia i taki sam czas zakończenia, to bierzemy pod uwagę kolejno zajęcia i_3 oraz j_3 itd.

Niech $A_1 = \{1, 2, \ldots, t\}$, $A_i = \{1, 2, \ldots, r\} i = 2, \ldots, k$. Weźmy zajęcie nr 2 należące do zbióru A_1 i popatrzmy na jego czas zakończenia f_{1_2}. Następnie w salach od A_2 do A_k szukajmy zajęcia o czasie rozpoczęcia s_{1_2} i dłuższym czasie trwania tzn. $f_{1_2} < f_{i_j}$.

Jeśli takie zajęcie zostanie znalezione, to dokonujemy następującej wymiany pomiędzy zbiorami: wszystkie zajęcia ze zbioru A_1 począwszy od zajęcia nr 2, aż do zajęcia nr t wymieniamy na zajęcia ze zbioru A_i począwszy od zajęcia nr j aż do zajęcia nr r i na odwrót.

Po takiej wymianie zajęcia w zbiorach A_1 i A_i pozostają nadal zgodne. Dlaczego? Zajęcia od numeru 2 do t i od numeru j do r są zgodne z założenia. Ponieważ $s_{1_2} = s_{i_j}$, to $f_{1_1} \leq s_{i_j}$, $f_{i_j-1} \leq s_{1_2}$.

Analogicznie jak z zajęciem nr 2 postępujemy z każdym kolejnym zajęciem należącym do zbioru A_1. W ten sposób w zbiorze (sali) A_1 znajdą się wszystkie zajęcia zgodne, wśród których będzie zajęcie nr 1 ze zbioru S.

W taki sam sposób postępujemy z salami $A_2, \ldots, A_k$, przy czym sala A_2 w ten sposób skonstruowana będzie zawierać wszystkie zajęcia ze sobą zgodne, wśród których będzie zajęcie nr 1 ze zbioru

$$S^1 = S - \{i \in S : i \in A_1\}$$

Sala A_3 będzie zawierać wszystkie zajęcia ze sobą zgodne, wśród których będzie zajęcie nr 1 ze zbioru

$$S^2 = S^1 - \{i \in S^1 : i \in A_2\}$$

Sala A_k będzie zawierać wszystkie zajęcia ze sobą zgodne, wśród których będzie zajęcie nr 1 ze zbioru

$$S^{k-1} = S^{k-2} - \{i \in S^{k-2} : i \in A_{k-1}\}$$

Wszystkie te wymiany zajęć pomiędzy zbiorami (salami) nie spowodują zmniejszenia liczby sal. Gdyby bowiem tak sie stało, to przeczyłoby to założeniu, że k jest minimalną liczbą sal.

Widać z konstrukcji zbiorów $A_1, A_2, \ldots, A_k$, że są to kolejne sale generowane przez algorytm GREEDY - ROOM - SELECTOR. Stąd wnioskuję, że zawsze istnieje optymalne rozwiązanie *problemu przydziału zajęć do minimalnej liczby sal*, którego konstrukcję można rozpocząć od wyboru zachłannego, czyli wyboru, który daje nam algorytm GREEDY - ROOM - SELECTOR. ■

Problem wydawania reszty, za pomocą jak najmniejszej liczby monet

Przyjmijmy założenie, że monety mają nominały $c^0, c^1, \ldots, c^k$ dla pewnych liczb całkowitych $c > 1$ oraz $k \geq 1$. Pokażę, że algorytm zachłanny dla takiego zestawu monet daje zawsze optymalne rozwiązanie.

Poniższy pseudokod jest zapisem algorytmu zachłannego dla *problemu wydawania reszty*. Bez utraty ogólności zakładam, że nominały są uporządkowane malejąco ze względu na ich wartość, tzn. $c^0 \leq c^1 \leq \ldots \leq c^k$. W poniższej procedurze zakładam, że c jest uporządkowaną malejąco tablicą nominałów, natomiast *sum* jest resztą, którą należy wydać.

```
GREEDY - GIVE - CHANGE(sum, c, k)
1. for i := 0 to k do
2.     if (sum div c[i]) ≠ 0 then
3.         change[i] := sum div c[i];
4.         sum := sum mod c[i];
5.     else
6.         change[i] := 0;
7.     endif;
8. endfor;
9. return change;
```

Działanie tej procedury jest następujące:

W krokach od 1 do 8 kolejno do tablicy *change* wpisujemy największą możliwą liczbę nominału $c^k, c^{k-1}, \ldots, c^0$. Jeżeli jakiegoś nominału nie można użyć do wydania określonej reszty, tzn. nie jest spełniony warunek $(sum \operatorname{div} c[i]) \neq 0$, to do tablicy *change* wpisywana jest wartość zero - wiersz 6 algorytmu.

Procedura GREEDY - GIVE - CHANGE zwraca tablice *change*, zawierającą liczby poszczególnych nominałów użytych do wydania danej kwoty. Jeżeli zsumujemy te liczby, to otrzymamy minimalną liczbę monet, jaką należy użyć, aby wydać kwotę *sum*.

Własność

Niech $sum = n_k \cdot c^k + n_{k-1} \cdot c^{k-1} + \ldots + n_1 \cdot c^1 + n_0 \cdot c^0$, dla pewnych całkowitych $k \geq 1$, $c > 1$. n_k - liczba monet nominału c^k.

Jeżeli liczbę n_k zmniejszymy o wartość t ($t > 0$), to uzyskamy o $t \cdot c$ więcej monet nominału c^{k-1}.

Dowód własności

Niech $k = 1$. Wtedy $sum = n_1 \cdot c^1 + n_0 \cdot c^0$. Weźmy o t monet nominału c^1 mniej. Wówczas

$$sum = (n_1 - t) \cdot c^1 + (n_0 + t \cdot c) \cdot c^0$$

Stąd widać, że przybyło $t \cdot c$ monet nominału c^0.
Niech własność będzie prawdziwa dla $k = p-1$. Pokażę, że jest prawdziwa dla $k = p$.
Niech zatem $k = p$, $t > 0$ oraz

$$sum = n_p \cdot c^p + n_{p-1} \cdot c^{p-1} + \ldots + n_1 \cdot c^1 + n_0 \cdot c^0$$

Ponieważ własność jest prawdziwa dla $k = p-1$, to

$$sum = n_p \cdot c^p + (n_{p-1} - t) \cdot c^{p-1} + (n_{p-2} + t \cdot c) \cdot c^{p-2} + \ldots + n_1 \cdot c^1 + n_0 \cdot c^0$$

Weźmy o t monet nominału c^p mniej. Wówczas

$$sum = (n_p - t) \cdot c^p + t \cdot c^p + (n_{p-1} - t) \cdot c^{p-1} + (n_{p-2} + t \cdot c) \cdot c^{p-2} + \ldots + n_1 \cdot c^1 + n_0 \cdot c^0$$

$$sum = (n_p - t) \cdot c^p + (n_{p-1} - t + t \cdot c) \cdot c^{p-1} + (n_{p-2} + t \cdot c) \cdot c^{p-2} + \ldots + n_1 \cdot c^1 + n_0 \cdot c^0$$

$$sum = (n_p - t) \cdot c^p + (n_{p-1} + t \cdot c) \cdot c^{p-1} - t \cdot c^{p-1} + (n_{p-2} + t \cdot c) \cdot c^{p-2} + \ldots + n_1 \cdot c^1 + n_0 \cdot c^0$$

$$sum = (n_p - t) \cdot c^p + (n_{p-1} + t \cdot c) \cdot c^{p-1} - t \cdot c^{p-1} + n_{p-2} \cdot c^{p-2} + t \cdot c^{p-1} + \ldots + n_1 \cdot c^1 + n_0 \cdot c^0$$

$$sum = (n_p - t) \cdot c^p + (n_{p-1} + t \cdot c) \cdot c^{p-1} + n_{p-2} \cdot c^{p-2} + \ldots + n_1 \cdot c^1 + n_0 \cdot c^0$$

Z ostatniej równości wynika, że po zabraniu t nominałów c^p, uzyskaliśmy o $t \cdot c$ nominałów c^{p-1} więcej.

Stąd, na podstawie indukcji wnioskuję, że własność jest prawdziwa dla dowolnego całkowitego $k > 0$. ∎

Wniosek 1 *Algorytm GREEDY - GIVE - CHANGE generuje optymalne rozwiązanie dla problemu wydawania reszty za pomoca minimalnej liczby monet o nominałach postaci:* $c^k, c^{k-1}, \ldots, c^1, c^0$*, gdzie* $c > 1$ *i* $k \geq 1$.

Dowód Rozważmy nominały $c^k, c^{k-1}, \ldots, c^1, c^0$, dla pewnych liczb całkowitych $c > 1$ oraz $k \geq 1$. Niech te nominały będą uporządkowane w kolejności malejącej wg wartości.
Niech *sum* jest kwotą jaką należy wydać, a n_k jest liczbą nominałów c^k. Pokażę, że rozwiązanie rozważanego problemu wydawania reszty, uzyskane przy pomocy algorytmu GREEDY - GIVE - CHANGE jest optymalne.

Weźmy dowolne optymalne rozwiązanie problemu wydawania reszty. Niech ma ono postać:

$$sum = n_k \cdot c^k + n_{k-1} \cdot c^{k-1} + \ldots + n_1 \cdot c^1 + n_0 \cdot c^0$$

gdzie $n_k + n_{k-1} + \ldots + n_1 + n_0$ jest minimalna.
Z wyżej udowodnionej własności wynika, że jeśli tylko zmniejszymy ilość nominału c^s, gdzie $1 \leq s \leq k$, o pewną liczbę $t > 0$, to ilość użytych monet do wydania danej kwoty zwiększy się co najmniej o $(t \cdot c - t)$.

Stąd, aby wydać określoną kwotę - *sum* przy pomocy wyżej ustalonego zestawu nominałów, to najpierw należy wydać największą możliwą ilość najwyższego nominału (c^k), mniejszego od danej kwoty $(sum \geq c^k)$, dla pewnego $k > 0$. Jeśli tak nie uczynimy, to na podstawie powyższej własności uzyskamy o co najmniej $(t \cdot c - t)$ monet więcej i wówczas $n_k + n_{k-1} + \ldots + n_1 + n_0$ nie będzie już minimalna. Następnie należy wydać największą możliwą ilość najwyższego nominału mniejszego od c^k i od kwoty, która jeszcze została do wydania itd. Stąd $n_k = sum\ div\ c^k$, $n_{k-s} = (sum - n_k \cdot c^k)\ div\ c^{k-s}$, dla pewnego $s < k$ itd.

Taki właśnie sposób postępowania proponuje nam algorytm GREEDY - GIVE - CHANGE.
Zatem rozwiązanie optymalne - to rozwiązanie zachłanne. ∎

Istnieją zbiory nominałów dla których algorytm zachłanny nie daje optymalnego rozwiązania.
Rozważmy np. zbiór nominałów: $\{50, 20, 1\}$ i kwotę 66.
Algorytm zachłanny dla powyższego zestawu monet dałby następujące rozwiązanie:

$$66 = 1 \cdot 50 + 16 \cdot 1,$$

czyli wydałby kwotę 66 przy pomocy 17 monet. Tymczasem optymalne rozwiązanie -(9 monet) jest postaci:

$$66 = 3 \cdot 20 + 6 \cdot 1$$

Własności Algorytmów zachłannych

Algorytm zachłanny dochodzi do optymalnego rozwiązania poprzez podejmowanie ciągu decyzji. Za każdym razem podejmowana jest decyzja, która wydaje się w danej chwili najlepsza. Stosując taką strategię, nie zawsze otrzymuje się optymalne rozwiązanie, lecz jak widać było na przykładzie *problemu wyboru czynności* i *problemu przydziału sal*, dla niektórych problemów stosowanie tej strategii jest w pełni uzasadnione.

Nie ma żadnej ogólnej metody, aby przekonać się czy dany algorytm zachłanny rozwiązuje dany problem optymalizacyjny, lecz istnieją dwie cechy charakterystyczne dla większości problemów poddających się strategii zachłannej: **własność wyboru zachłannego** oraz **optymalna podstruktura**.

Pierwsza podstawowa własność to **własność wyboru zachłannego**. W algorytmie zachłannym dokonujemy wyboru, który wydaje się być najlepszy w danej chwili, a następnie rozwiązujemy podproblemy, które wynikają z podjętej decyzji. Wybory podejmowane w algorytmie zachłannym mogą zależeć od dotychczasowych decyzji, lecz nie mogą być uzależnione od przyszłych wyborów lub od rozwiązań podproblemów. Algorytmy zachłanne znajdują rozwiązanie w sposób zstępujący, podejmując kolejno decyzje zachłanne, stopniowo redukując podproblemy do coraz mniejszych.

Należy udowodnić, że wybór zachłanny w każdym kroku prowadzi do globalnie optymalnego rozwiązania.
Dowód polega na sprawdzeniu pewnego globalnie optymalnego rozwiązania. Wykazuje się, że można je sprowadzić do rozwiązania, którego początkiem jest podjęcie decyzji zachłannej w pierwszym kroku, oraz wykazuje się, że ten wybór redukuje problem do podobnego, ale o mniejszym rozmiarze. Aby następnie uzasadnić, że strategię zachłanna można stosować w każdym następnym kroku, wystarczy użyć indukcji. Problem wykazuje optymalną podstrukturę, jeśli optymalne rozwiązanie jest funkcją optymalnych rozwiązań podproblemów.
Jako przykład optymalnej podstruktury przypomnę twierdzenie 1. Wynika z niego, że jeśli optymalne rozwiązanie A problemu wyboru zajęć rozpoczyna się od zajęcia 1, to $A' = A - \{1\}$ jest optymalnym rozwiązaniem problemu wyboru zajęć dla zbioru $S' = \{ i \in S \ : \ s_i \geq f_1 \}$.

Ciągły problem plecakowy

Złodziej rabujący sklep znalazł n substancji (przedmiotów dających się dzielić); i-ta substancja jest warta c_i złotych i waży w_i kilogramów, gdzie c_i oraz w_i są nieujemnymi liczbami całkowitymi. Złodziej dąży do zabrania ze sobą jak najwartościowszego łupu, lecz nie może wziąć do swego plecaka więcej niż W kilogramów. Jakie substancje powinien ze sobą zabrać złodziej? (złodziej może zabierać ułamkowe części substancji).

Aby rozwiązać ciągły problem plecakowy, należy najpierw obliczyć wartość masy jednostkowej (c_i/w_i) każdej substancji. Kierując się strategią zachłanną, złodziej wybiera najpierw największą możliwą ilość najbardziej wartościowej substancji. Jeśli zapas tej substancji się wyczerpał, a w plecaku jest ciągle jeszcze wolne miejsce, złodziej wybiera następną pod względem ceny za jednostkę masy substancję i wypełnia nią plecak, i postępuje tak dopóty, dopóki plecak nie zostanie wypełniony całkowicie.

Podejście to wymaga posortowania "substancji" według nierosnącej wartości jednostki masy.

Wniosek 2 *Ciągły problem plecakowy wykazuje własność wyboru zachłannego.*

Dowód Niech będzie dany zbiór $S = \{1, \ldots, n\}$, w skład którego wchodzi n substancji. i-ta substancja jest warta c_i złotych i waży w_i kilogramów. c_i/w_i jest wartością masy jednostkowej oraz W maksymalna pojemność plecaka.
Substancje w S są uporządkowane nierosnąco ze względu na wartość jednostkową masy. Wykażę, że istnieje optymalne rozwiązanie, którego konstrukcję można rozpocząć od wyboru zachłannego, tzn. od wyboru najwartościowszej substancji.

Załóżmy, że $A \subseteq S$ jest optymalnym rozwiązaniem ciągłego problemu plecakowego (czyli łączna waga substancji należących do A jest równa W oraz jest to zbiór najwartościowszych substancji).

Niech substancje w A będą uporządkowane ze względu na wartość jednostkową masy nierosnąco. Niech ponadto pierwszą substancją należącą do A jest w kilogramów substancji k o wartości jednostkowej masy c_k/w_k. Jeśli $k = 1$, to tworzenie rozwiązania A można rozpocząć od podjęcia decyzji zachłannej. Jeżeli natomiast $k \neq 1$, to wykażę, że istnieje rozwiązanie optymalne $B \subseteq S$, którego konstrukcję można rozpocząć od wyboru zachłannego, czyli od wyboru substancji 1 o wartości c_1/w_1 za jednostkę masy.
Niech $B = A - \{l_k\} \cup \{l_1\}$, gdzie l_k, to w kilogramów substancji k, a l_1, to w kilogramów substancji 1. Ponieważ wymieniamy po w kilogramów poszczególnych substancji, to waga plecaka nie zmienia się. Teraz wystarczy pokazać że wartość plecaka jest taka sama. Wiadomo, że $(c_k/w_k) \leq (c_1/w_1)$ (substancje w zbiorze S są uporządkowane nierosnąco wg wartości za jednostkę masy), więc wartość plecaka spakowanego według rozwiązania B nie jest mniejsza od wartości plecaka spakowanego według rozwiązania A.
Przypuśćmy, że rozwiązanie B jest bardziej wartościowe. Ponieważ wymieniliśmy w kilogramów substancji k, na w kilogramów substancji 1, to stąd wynika, że A nie jest optymalnym rozwiązaniem, co przeczy założeniu o optymalności A.
Zatem wartość plecaka spakowanego według rozwiązania B jest taka sama jak wartość plecaka spakowanego według rozwiązania A. Czyli rozwiązanie B jest również optymalne. Stąd wynika, że zawsze istnieje rozwiązanie,

którego konstrukcję można rozpocząć od wyboru zachłannego.
Poza tym, gdy już dokonaliśmy wyboru zachłannego substancji 1, to problem sprowadza się do znalezienia optymalnego rozwiązania ciągłego problemu plecakowego dla $n-1$ oryginalnych substancji ze zbioru S oraz $w_1 - w$ kilogramów substancji 1.
Widać, że po wyborze zachłannym pozostaje do rozwiązania problem optymalizacyjny tego samego typu, co na początku.
Korzystając z indukcji względem liczby podjętych decyzji zachłannych i dokonując zachłannego wyboru w każdym kroku, otrzymujemy optymalne rozwiązanie.

Wniosek 3 *Ciągły problem plecakowy wykazuje własność optymalnej podstruktury.*

Dowód Rozważmy najwartościowszy ładunek o masie W. Jeśli usuniemy z optymalnego ładunku w kilogramów pewnej substancji j, to pozostający ładunek jest najwartościowszym ładunkiem o wadze co najwyżej $W - w$, który złodziej może skompletować z $n - 1$ oryginalnych substancji, plus $w_j - w$ kilogramów substancji j.

A teraz rozważmy problem plecakowy sformułowany podobnie jak ciągły problem plecakowy, lecz z tą różnicą, że złodziej musi podjąć dramatyczną decyzję, albo zabiera dany przedmiot, albo nie. Złodziej nie może zabrać ułamkowej części przedmiotu, ani wielokrotności żadnego z nich.
Problem ten nosi nazwę *dyskretnego problemu plecakowego.*

Obydwa problemy plecakowe wykazują *własność optymalnej podstruktury*, co łatwo wykazać. Niestety *dyskretny problem plecakowy* nie wykazuje własności wyboru zachłannego i dlatego algorytm zachłanny nie rozwiązuje tego problemu optymalnie.

Aby się o tym przekonać rozważmy następujący przykład: Dany jest plecak o pojemności 25kg i trzy przedmioty.
Przedmiot 1 waży 5kg i wart jest 60zł.
Przedmiot 2 waży 10kg i wart jest 100zł.
Przedmiot 3 waży 15kg i wart jest 120zł.
Cena jednostkowa przedmiotu 1 wynosi 12zł/kg, przedmiotu 2 - 10zł/kg, przedmiotu 3 - 8zł/kg.

Strategia zachłanna doprowadziłaby do wyboru przedmiotu 1 na samym początku, jako najbardziej wartościowego, a następnie przedmiotu 2. W ten sposób spakowany plecak miałby wartość 160zł.

Optymalnym jednak rozwiązaniem jest wybór przedmiotu 2 i 3. Przy takim wyborze plecak spakowany jest maksymalnie - waży 25kg i jego wartość wynosi 220zł.

W *dyskretnym problemie plecakowym*, zanim podejmiemy decyzje, czy należy zabrać przedmiot, musimy porównać rozwiązanie podproblemu, w którym nie jest on brany pod uwagę.

Teoretyczne podstawy strategii zachłannej

W określeniu sytuacji, w których strategia zachłanna prowadzi do optymalnych rozwiązań użyteczna jest teoria MATROIDÓW.

Definicja 1 *Matroid jest uporządkowaną parą $M = (S, \Phi)$ spełniającą następujące warunki:*

1. *S jest skończonym niepustym zbiorem.*

2. *Φ jest niepustą rodziną podzbiorów zbioru S, zwaną rodziną niezależnych podzbiorów S, taką że jeśli $B \in \Phi$ $A \subseteq B$, to $A \in \Phi$. Mówi się, że Φ jest* ***dziedziczne****, jeśli ma tę własność.*

3. *Jeśli $A \in \Phi, B \in \Phi$ i $| A |<| B |$, to istnieje taki element $x \in B - A$, że $A \cup \{x\} \in \Phi$. Mówi się, że M ma własność wymiany.*

Uwaga 1 *Zbiór pusty jest zawsze elementem Φ*

PRZYKŁADY:

1) Uporządkowana para (S, Φ_k), gdzie S jest dowolnym, niepustym skończonym zbiorem, a Φ_k jest zbiorem wszystkich podzbiorów S o rozmiarze co najwyżej k dla $k \leq| S |$, jest matroidem.

   Dowód Oczywiste

2) Przykładem matroidu jest również **matroid macierzowy**, w którym elementami zbioru S są kolumny pewnej macierzy.

Twierdzenie 3 *Uporządkowana para (S, Φ), gdzie S jest zbiorem kolumn danej macierzy T o wartościach rzeczywistych i wymiarze $n \times n$, a $A \in \Phi$ wtedy i tylko wtedy, gdy kolumny należące do A są liniowo niezależne, jest matroidem.*

Dowód Aby wykazać, że (S, Φ) jest matroidem, wystarczy pokazać, że: S jest niepustm, skończonym zbiorem, Φ jest dziedziczne oraz (S, Φ) ma własność wymiany.

Niech $T = \begin{bmatrix} a_{11} & \dots & a_{1n} \\ \dots & \dots & \dots \\ a_{n1} & \dots & a_{nn} \end{bmatrix}$ gdzie $A_k = \begin{bmatrix} a_{1k} \\ \dots \\ a_{nk} \end{bmatrix}$, dla $A = \{A_1, \dots, A_k\}$,

gdzie $k \leq n$, $a_{ij} \in R$ dla $i \leq n$, $j \leq n$.

Warunki 1 i 2 definicji są oczywiste.

Wystarczy tylko pokazać, że (S, Φ) ma własność wymiany.

Niech $A = \{A_1, \ldots, A_n\} \in \Phi$, $B = \{B_1, \ldots, B_m\} \in \Phi$, zatem A oraz B są zbiorami liniowo niezależnych kolumn i niech $|A| < |B|$, tzn $n < m$. Pokażę, że istnieje $B_i \in B - A$, gdzie $1 \leq i \leq m$, takie że $A \cup \{B_i\} = \{A_1, \ldots, A_n, B_i\}$ jest zbiorem liniowo niezależnych kolumn. Przypuśćmy, że dla każdego $1 \leq i \leq m$ układ $\{A_1, \ldots, A_n, B_i\}$ jest liniowo zależny. Wtedy z defnicji liniowej zależności mamy:

$$\alpha_1 \cdot A_1 + \alpha_2 \cdot A_2 + \ldots + \alpha_n \cdot A_n + \alpha_{n+1} \cdot B_i = \Theta \ \wedge \underset{1 \leq j \leq n+1}{\exists} \alpha_j \neq 0$$

Zauważmy, że musi być $\alpha_{n+1} \neq 0$. Gdyby bowiem $\alpha_{n+1} = 0$ to wśród współczynników $\alpha_1, \alpha_2, \ldots, \alpha_n$, któryś musiałby być różny od zera, a to przeczyłoby liniowej niezależności układu $\{A_1, \ldots, A_n\}$

Zatem mamy, że

$$B_i = \left(-\alpha_1 \cdot \alpha_{n+1}^{-1}\right) \cdot A_1 + \left(-\alpha_2 \cdot \alpha_{n+1}^{-1}\right) \cdot A_2 + \ldots + \left(-\alpha_n \cdot \alpha_{n+1}^{-1}\right) \cdot A_n$$

co oznacza, że B_i jest kombinacją liniową układu $\{A_1, \ldots, A_n\}$. Stąd wynika, że dla każdego $1 \leq i \leq m$ $B_i \in L(A_1, \ldots, A_n)$ (B_i należy do przestrzeni R^n, której bazą jest układ $\{A_1, \ldots, A_n\}$) oraz $L(B_1, \ldots, B_m) \subseteq L(A_1, \ldots, A_n)$. Stąd $dim(B_1, \ldots, B_m) \leq dim(A_1, \ldots, A_n)$. Ponieważ $dim(B_1, \ldots, B_m) = m$, a $dim(A_1, \ldots, A_n) = n$, więc $m \leq n$, co przeczy założeniu, że $n < m$. Zatem $A \cup \{B_i\}$ jest liniowo niezależny, czyli $A \cup \{B_i\} \in \Phi$.

3) Uporządkowana para (S, Φ), gdzie S jest skończonym niepustym zbiorem, a $\Phi = \{A : |A \cap S_i| \leq 1 \, dla \, i = 1, 2, \ldots, k$, gdzie $S_1, S_2, \ldots, S_k$ jest podziałem S na niepuste rozłączne podzbiory $\}$, jest matroidem.

Dowód Aby wykazać, że (S, Φ) jest matroidem, wystarczy pokazać, że S jest niepustm, skończonym zbiorem, Φ jest dziedziczne oraz (S, Φ) ma własnośc wymiany.

Warunek 1 definicji matroidu jest oczywisty.

Warunek 2: Niech $B \in \Phi$ $A \subseteq B$. Pokażę, że $A \in \Phi$. Ponieważ $A \subseteq B$, to $B = A \cup (B \setminus A)$.

Ponieważ $B \in \Phi$, to $|B \cap S_i| \leq 1$ dla $1 \leq i \leq k$.

Ale $|B \cap S_i| = |[A \cup (B \setminus A)] \cap S_i| = |A \cap S_i| + |(B \setminus A) \cap S_i| \leq 1$. Stąd $|A \cap S_i| \leq 1$. Zatem $A \in \Phi$.

Warunek 3: Niech $A \in \Phi$, $B \in \Phi$ oraz $|A| < |B|$. Pokażę, że istnieje takie $x \in B \setminus A$, że $A \cup \{x\} \in \Phi$.

Niech $x \in B \setminus A$. Dla każdego $1 \leq i \leq k$

$$|(A \cup \{x\}) \cap S_i| = |A \cap S_i| + |\{x\} \cap S_i|$$

Jeśli $S_i \supseteq \{x\}$, to $| \{x\} \cap S_i |=| \{x\} |= 1$ $A \cap S_i = 0$, bo $x \in B \setminus A$.
Jeśli $S_i \not\supseteq \{x\}$, to $| \{x\} \cap S_i |= 0$ $A \cap S_i \leq 1$, bo $A \in \Phi$.
Zatem $A \cup \{x\} \in \Phi$. ■

Definicja 2 *Dla danego matroidu $M = (S, \Phi)$ element $x \notin A$ nazywamy* **rozszerzeniem** *$A \in \Phi$, jeśli x można dodać do A, zachowując niezależność; mówiąc inaczej x jest rozszerzeniem A, jeśli $A \cup \{x\} \in \Phi$.*

Definicja 3 *Jeśli A jest niezależnym zbiorem matroidu M, to zbiór A jest* **maksymalny**, *jeśli nie ma rozszerzeń.*

Twierdzenie 4 *Wszystkie maksymalnie niezależne podzbiory matroidu mają ten sam rozmiar.*

Dowód Załóżmy niewprost, że A jest maksymalnym niezależnym podzbiorem matroidu M oraz że istnieje inny, większy, maksymalny niezależny podzbiór B matroidu M. Wtedy z własności wymiany wynika, że A można rozszerzyć do większego zbioru niezależnego $A \cup \{x\}$ dla pewnego $x \in B \setminus A$, co przeczy założeniu o maksymalności A.

Definicja 4 *Matroid $M = (S, \Phi)$ jest* **ważony**, *jeśli istnieje związana z nim funkcja wagi w, która przypisuje dodatnią wagę $w(x)$ każdemu elementowi $x \in S$. Funkcja wagi rozszerza się do podzbiorów S za pomocą sumowania:*

$$w(A) = \sum_{x \in A} w(x)$$

po dowolnym zbiorze $A \subseteq S$.

Algorytmy zachłanne na matroidzie ważonym

Wiele problemów, dla których strategia zachłanna daje optymalne rozwiązanie, można sformułować w kategoriach wyznaczania niezależnego zbioru o największej wadze w ważonym matroidzie.
Taki niezależny podzbiór o maksymalnej wadze nazywamy **optymalnym podzbiorem matroidu**. Waga $w(x)$ każdego elementu $x \in S$ jest dodatnia, optymalny podzbiór jest więc **maksymalnym zbiorem niezależnym.**

A oto algorytm zachłanny, który działa dla dowolnego matroidu ważonego. Dane wejściowe dla algorytmu stanowi matroid ważony $M = (S, \Phi)$ wraz z dodatnią funkcją wagi w. Za pomocą algorytmu jest znajdowany optymalny podzbiór A. W poniższej procedurze składniki matroidu M oznaczane są przez $S[M]$ oraz $\Phi[M]$, a funkcja wagi przez w.

```
GREEDY(M, w)
1. A := ∅;
2. uporządkuj S[M] nierosnąco wg wagi w
3. for x ∈ S[M], brane w porządku nierosnącym wg wagi w(x) do
4.    if (A ∪ {x} ∈ Φ[M]) then
5.       A := A ∪ {x};
6.    endif;
7. endfor;
8. return A;
```

Elementy zbioru S są rozpatrywane kolejno w porządku malejącej wagi. Jeśli właśnie rozważany element x może zostać dodany do A, nie wyprowadzając A poza klasę zbiorów niezależnych, to zostaje dodany. W przeciwnym razie x zostaje odrzucony. Zbior pusty jest niezależny z definicji matroidu. x jest dodawany do A tylko wtedy, kiedy $A \cup \{x\}$ jest niezależny. Korzystając z indukcji wnioskuję, że podzbiór A jest zawsze niezależny.

Twierdzenie 5 *Jeśli $M = (S, \Phi)$ jest ważonym matroidem z funkcją wagi w, to w wyniku wywołania procedury GREEDY(M,w) jest wyznaczony optymalny podzbiór.*

Lemat 1. (Matroidy mają własność zachłannego wyboru) Załóżmy, że $M = (S, \Phi)$ jest matroidem ważonym z funkcją wagi w oraz że S jest uporządkowany nierosnąco według wagi. Niech x będzie pierwszym elementem S takim, że $\{x\}$ jest niezależny. Jeśli takie x istnieje, to wtedy istnieje optymalny podzbiór A zbioru S zawierający x.

Dowód lematu Jeśli nie istnieje takie x, to jedynym zbiorem niezależnym jest zbiór pusty i teza jest spełniona. W przeciwnym razie niech B będzie niepustym zbiorem optymalnym. Przypuśćmy, że $x \notin B$ (w przeciwnym wypadku mamy $A = B$ i teza jest spełnina). Żaden element zbioru B nie ma wagi większej niż $w(x)$, bo wystarczy zauważyć, że jeśli $y \in B$, to $\{y\}$ jest niezależny, ponieważ $B \in \Phi$, a Φ jest dziedziczne.
Wybór x gwarantuje to, że $w(x) \geq w(y)$ dla każdego $y \in B$.

Konstrukcja zbioru A przebiega nastepująco. Zaczynamy $A = \{x\}$. Dzięki wyborowi x zbiór A jest zbiorem niezależnym. Korzystając wielokrotnie z własności wymiany, dodajemy do A pewne elementy B, zachowując niezależność A, aż nie osiagniemy $\mid A \mid = \mid B \mid$. Wtedy
$A = B \setminus \{y\} \cup \{x\}$ dla pewnego $y \in B$, więc

$$w(A) = w(B) - w(y) + w(x) \geq w(B)$$

Zbiór A jest zatem optymalny (bo B jest optymalny) oraz $x \in A$. ∎

Wykażę teraz, że jeśli element nie może być początkowo dołączony, to już nigdy nie będzie musiał być rozpatrywany.

Lemat 2. Niech $M = (S, \Phi)$ będzie dowolnym matroidem. Jeśli $x \in S$ nie jest rozszerzeniem zbioru pustego, to x nie jest rozszerzeniem żadnego niezależnego zbioru A.

Dowód lematu Załóżmy, że x jest rozszerzeniem zbioru A, choć nie jest rozszerzeniem zbioru pustego. Ponieważ x jest rozszerzeniem A, to zbiór $A \cup \{x\}$ jest niezależny. Ale z definicji matroidu wiadomo, że Φ jest dziedziczny, zatem $\{x\}$ jest zbiorem niezależnym, co jest sprzeczne z założeniem, że x nie jest rozszerzeniem zbioru pustego. ∎

Lemat 3. (Matroidy wykazują własność optymalnej podstruktury) Niech x będzie pierwszym elementem wybranym, przez procedurę GREEDY w matroidzie ważonym $M = (S, \Phi)$. Pozostający problem znalezienia niezależnego podzbioru o maksymalnej wadze zawierającego x sprowadza się do problemu znalezienia niezależnego podzbioru o maksymalnej wadze w matroidzie ważonym $M' = (S', \Phi')$, gdzie:

$$S' = \{y \in S \ : \ \{x, y\} \in \Phi\}$$
$$\Phi' = \{B \subseteq S - \{x\} \ : \ B \cup \{x\} \in \Phi\}$$

oraz funkcja wagi dla M' jest funkcją wagi dla M ograniczoną do S'. (Taki matroid nazywamy **kontrakcją** M o element x).

Dowód lematu Jeśli A jest niezależnym podzbiorem o maksymalnej wadze zawierającym x, to $A' = A \setminus \{x\}$ jest niezależnym podzbiorem w M'. Odwrotnie, każdy niezależny podzbiór A' matroidu M' odpowiada niezależnemu podzbiorowi $A = A' \cup \{x\}$ matroidu M. W obu przypadkach $w(A) = w(A') + w(x)$, zatem rozwiązanie o maksymalnej wadze w matroidzie M zawierające x odpowiada optymalnemu rozwiązaniu w M' i na odwrót. ∎

Na podstawie wyżej udowodnionych lematów twierdzenie o procedurze GREEDY jest oczywiste.

Problem szeregowania zadań

Jednym z interesujących problemów, które można rozwiązać za pomocą matroidów, jest problem optymalnego szeregownia zadań o jednostkowym czasie wykonania z karami za przekroczenie terminu.

Zadanie o jednostkowym czasie wykonania jest czynnością, taką jak program działający na komputerze, której wykonanie wymaga jednostkowego czasu. Dla danego zbioru zadań S ich **uszeregowanie** jest permutacją S odpowiadajacą kolejności ich wykonywania. Pierwsze zadanie w uszeregowaniu zaczyna się w chwili 0 i kończy w chwili 1, drugie zaczyna się w chwili 1 i kończy w chwili 2 itd. Problem szeregowania zadań o jednostkowym czasie wykonania ma następujące dane wejściowe:

1) Zbiór $S = \{1, 2, \dots, n\}$ zawierający n zadań o jednostkowym czasie wykonania;

2) Zbiór n dopuszczalnych **terminów wykonania** $d_1, d_2, \dots, d_n$, takich że $1 \leq d_i \leq n$, gdzie wykonanie zadania i ma zostać zakończone przed chwilą d_i;

3) Zbiór nieujemnych wag lub **kar** $w_1, w_2, \dots, w_n$, takich że kara w_i zostaje naliczona tylko wtedy, kiedy zadanie i nie jest zakończone przed chwilą d_i;

Zadanie polega na znalezieniu uszeregowania zadań ze zbioru S, dla którego suma naliczonych kar za niedotrzymanie terminu wykonania jest najmniejsza.

Aby rozwiązać ten problem wystarczy udowodnić następujące twierdzenie:

Twierdzenie 6 *Jeśli S jest zbiorem zadań o jednostkowym czasie wykonania, a Φ jest rodziną wszystkich niezależych zbiorów zadań, to (S, Φ) jest matroidem.*

Zanim przejdę do dowodu tego twierdzenia wyjaśnię kilka terminów.
Rozważmy pewne uszeregownie.
Powiemy, że zadanie jest **spóźnione** w tym uszeregowaniu, jeśli jego wykonanie kończy się po terminie, w przeciwnym razie mówimy, że zadanie jest **terminowe**.
Dowolne uszeregownaie można zawsze doprowadzić do **terminowej postaci normalnej**, w której zadania terminowe poprzedzają zadania spóźnione, co więcej, każde uszeregowanie można doprowadzić do **postaci kanonicznej**, w której terminowe zadania poprzedzają zadania spóźnione oraz wszystkie terminowe zadania są uporząkowane niemalejąco według dopuszczalnych terminów wykonania.
Mówimy, że zbiór zadań A jest niezależny, jeśli istnieje uszeregowanie tych zadań, w którym żadne z nich nie jest spóźnione.

Lemat 4. Dla dowolnego zbioru zadań A następujące warunki są równoważne:

1) Zbiór A jest niezależny;

2) Dla każdego $t = 1, 2, \dots, n$ zachodzi nierówność $N_t(A) \leq t$, gdzie $N_t(A)$ oznacza liczbę zadań w A, których dopuszczalny termin wykonania jest wcześniejszy lub równy t.

3) Jeśli zadania w A są uporządkowane niemalejąco według dopuszczalnego terminu wykonania, to żadne zadanie nie jest spóźnione.

Dowód lematu

(1) $\Rightarrow$ (2): Niech A jest zbiorem niezależnym. Przypuśćmy, że dla pewnego t zachodzi warunek $N_t(A) > t$. Wtedy nie jest możliwe uszeregowanie zadań w A bez zadań spóźnionych, ponieważ jest więcej niż t zadań, które należy wykonać przed chwilą t, co przeczy niezależności A.

(2) $\Rightarrow$ (3): Skoro każdy i–ty w kolejności dopuszczalny termin wykonania jest nie mniejszy od i, to szeregując zadania niemalejąco według tego terminu, żadne zadanie nie jest spóźnione.

(3) $\Rightarrow$ (1) - oczywiste z def. niezależności zbioru A. ∎

Dowód Zbiór S jest niepustym zbiorem skończonym z definicji. Każdy podzbior niezależnego zbioru zadań jest niezależny - oczywiste. Wystarczy pokazać, że zachodzi własność wymiany.
Niech A i B są zbiorami niezależnymi oraz $A < B$. Niech k będzie największym takim t, że zachodzi $N_t(B) \leq N_t(A)$. Wiadomo, że $N_n(A) = A$, $N_n(B) = B$ i $A < B$, więc $k < n$ oraz dla każdego j z przedziału $k+1 \leq j < n$ zachodzi $N_j(B) > N_j(A)$. Stąd B zawiera więcej zadań niż A o dopuszczalnym terminie wykonania nie większym od $k+1$.
Niech x będzie zadaniem w $B \setminus A$ o dopuszczalnym terminie nie większym od $k+1$. Niech $A' = A \cup \{x\}$. Korzystając z własności 2 lematu 4, pokażę że zbiór zadań A' jest niezależny. Dla $1 \leq t \leq k$ mamy $N_t(A') \leq N_t(A) \leq t$, ponieważ A niezależny. Dla $k < t \leq n$ mamy $N_t(A') \leq N_t(A) \leq t$, ponieważ B jest niezależny. Stąd A' jest niezależny, co kończy dowód, że (S, Φ) jest matroidem.

Przykład 1

Rozwiązać problem szeregowania zadań z poniższej tabeli:

Zadania	1	2	3	4	5	6	7
d_i	4	2	4	3	1	4	6
w_i	70	60	50	40	30	20	10

Aby rozwiazać ten problem, zgodnie z wyżej udowodnionym twierdzeniem 5 i 6, należy zastosować algorytm GREEDY.
Algorytm zachłanny GREEDY najpierw nakazuje uporządkowć zadania nierosnąco według wag (kar). Ponieważ powyższe zadania są już uporządkowane, to przechodzimy do wybierania optymalnego, niezależnego zbioru zadań.
Niech tym zbiorem będzie zbiór A. Na poczatku jest to zbiór pusty. Nastepnie wybieramy zadanie 1 i sprawdzamy, czy $A = \{1\}$ jest zbiorem niezależnym. Na podstawie lematu 4, własności drugiej mamy: jeśli dla

każdego $1 \leq 7$ zachodzi $N_t(A) \leq t$, to zbiór A jest niezalżny. Stąd ponieważ:
$N_1(A) = 0 \leq 1, N_2(A) = 0 \leq 2, N_3(A) = 0 \leq 3, N_4(A) = 1 \leq 4$, $N_5(A) = 1 \leq 5, N_6(A) = 1 \leq 6, N_7(A) = 1 \leq 7$, to $A = \{1\}$ jest niezależny.
Wybieramy zadanie 2 i sprawdzamy, czy $A = \{1,2\}$ jest zbiorem niezależnym. $N_1(A) = 0 \leq 1, N_2(A) = 1 \leq 2, N_3(A) = 1 \leq 3, N_4(A) = 2 \leq 4$, $N_5(A) = 2 \leq 5, N_6(A) = 2 \leq 6, N_7(A) = 2 \leq 7$. Stąd $A = \{1,2\}$ jest niezależny.
Wybieramy zadanie 3 i sprawdzamy, czy $A = \{1,2,3\}$ jest zbiorem niezależnym. $N_1(A) = 0 \leq 1, N_2(A) = 1 \leq 2, N_3(A) = 1 \leq 3, N_4(A) = 3 \leq 4$, $N_5(A) = 3 \leq 5, N_6(A) = 3 \leq 6, N_7(A) = 3 \leq 7$. Stąd $A = \{1,2,3\}$ jest niezależny.
Wybieramy zadanie 4 i sprawdzamy, czy $A = \{1,2,3,4\}$ jest zbiorem niezależnym. $N_1(A) = 0 \leq 1, N_2(A) = 1 \leq 2, N_3(A) = 2 \leq 3, N_4(A) = 4 \leq 4$, $N_5(A) = 4 \leq 5, N_6(A) = 4 \leq 6, N_7(A) = 4 \leq 7$. Stąd $A = \{1,2,3,4\}$ jest niezależny.
Wybieramy zadanie 5 i sprawdzamy, czy $A = \{1,2,3,4,5\}$ jest zbiorem niezależnym. $N_1(A) = 1 \leq 1, N_2(A) = 1 \leq 2, N_3(A) = 2 \leq 3, N_4(A) = 5 > 4$, Stąd $A = \{1,2,3,4,5\}$ jest zależny.
Wybieramy zadanie 6 i sprawdzamy, czy $A = \{1,2,3,4,6\}$ jest zbiorem niezależnym. $N_1(A) = 0 \leq 1, N_2(A) = 1 \leq 2, N_3(A) = 2 \leq 3, N_4(A) = 5 > 4$, Stąd $A = \{1,2,3,4,6\}$ jest zależny.
Wybieramy zadanie 7 i sprawdzamy, czy $A = \{1,2,3,4,7\}$ jest zbiorem niezależnym.
$N_1(A) = 0 \leq 1, N_2(A) = 1 \leq 2, N_3(A) = 2 \leq 3, N_4(A) = 4 \leq 4$, $N_5(A) = 4 \leq 5, N_6(A) = 5 \leq 6, N_7(A) = 5 \leq 7$.
Stąd $A = \{1,2,3,4,7\}$ jest niezależny.
Zatem w tym zadaniu algorytm zachłanny wybiera zadania 1,2,3,4, pomija zadanie 5,6, po czym dobiera zadanie 7. W efekcie otrzymujemy optymalne uszeregowanie: $< 2,4,1,3,7,5,6 >$, dla którego suma kar wynosi $w_5 + w_6 = 50$.

Przykład 2

Rozwiązać problem szeregowania zadań z poniższej tabeli:

Zadania	1	2	3	4	5	6	7
d_i	4	2	4	3	1	4	6
w_i	10	20	30	40	50	60	70

Aby rozwiazać ten problem, podobnie jak w przykładzie 1 należy zastosować algorytm GREEDY.

Algorytm zachłanny GREEDY najpierw nakazuje uporządkowć zadania niemalejąco według wag (kar).
Zatem ten problem po uporzadkowaniu przyjmie poniższą postać:

Zadania	7	6	5	4	3	2	1
d_i	6	4	1	3	4	2	4
w_i	70	60	50	40	30	20	10

Stosując algorym zachłanny GREEDY w taki sposób jak w przykładzie 1, wybierzemy kolejno następujace zadania: $7, 6, 5, 4, 3$, a pominiemy zadanie 2 i 1. W efekcie otrzymamy optymalne uszeregowanie: $< 5, 4, 3, 6, 7, 1, 2 >$, dla którego suma kar wynosi $w_1 + w_2 = 30$.

Wspomniany w tym artykule problem wydawania reszty, można formułować na różne sposby. W dalszej mojej pracy chciałabym opracować założenia, przy jakich problem wydawania reszy będzie optymalnie rozwiązywalny przez algorytm zachłanny. W przyszłości zamierzam zająć się teorią greedoidów, jako rozszerzeniem teorii matroidów.

Bibliografia

Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest *Wprowdzenie do algorytmów*. Wydawnictwo Naukowo - Techniczne. Warszawa 1997.

Bożena Woźna
Pedagogical University
Institute of Mathematics
Al.Armii Krajowej 13/15
Częstochowa 42-201