**WOJSKOWA AKADEMIA TECHNICZNA**

**IM. JAROSŁAWA DĄBROWSKIEGO W WARSZAWIE**

wydział cybernetyki

******

***Techniki algorytmiczne***

*Binarny problem plecakowy*

*Prowadzący: mgr inż. Krzysztof Panufnik*

Autorzy :

Małgorzata Filipek

Magdalena Grochowska

Grupa: I7B3S4

1. **Teoretyczny opis problemu**

Problem plecakowy – zagadnienie załadunkowe, polega na załadowaniu do plecaka odpowiednich przedmiotów posiadających swoją wagę oraz wartość. Celem jest, aby wartość zawartości plecaka była jak największa. Wagi określają ograniczenia, a wartości są współczynnikami funkcji celu. Istnieją różne odmiany zagadnienia załadunku:

* Binarne zagadnienie załadunku
* Uogólnione zagadnienie plecakowe
* Nieliniowe zagadnienie plecakowe

Binarny problem plecakowy wyróżnia się, że rozważamy jedynie decyzję czy określony przedmiot powinien być zapakowany czy też nie, bez uwzględnienia ilości sztuk.

Rozwiązaniem problemu plecakowego jest optymalizacja załadunku z uwagi na jego wartość przy nieprzekroczeniu ładowności plecaka.

Model matematyczny:

*Dane problemu plecakowego:*

b – ładowność plecaka

n – liczba przedmiotów

aj – waga j-tego przedmiotu (współczynnik funkcji ograniczającej, liczba całkowita dodatnia),  
j = 1, 2, …, n

cj – zysk z zabrania j-tego przedmiotu (współczynnik funkcji celu, liczba całkowita dodatnia),

j = 1, 2, …, n

*Zmienne:*

xj – zmienne decyzyjne, określające czy j-ty przedmiot ma być zapakowany

*Ograniczenia:*

xj ∈ {0, 1} – ograniczenie wartości zmiennej (przyjmuje wartość 1, gdy decydujemy o zabraniu j-tego przedmiotu do plecaka oraz wartość 0 w przeciwnym wypadku)

*cjxj → max*

funkcja celu (zysk)

*ajxj ≤ b*

ograniczenie ładowności

xj ∈ {0, 1, 0/1}, gdzie

0 – podejmujemy decyzję o niezabraniu przedmiotu do plecaka

1 – podejmujemy decyzję o zabraniu przedmiotu do plecaka

0/1 - czyli na poprzednim etapie wystąpił przedmiot, po zabraniu którego zyski są takie same jak dla rozpatrywanego przedmiotu. Rozpatrujemy zatem dwie sytuacje – tę, w której podejmujemy decyzję o zabraniu rozpatrywanego przedmiotu do plecaka oraz tę, w której decydujemy o niezabieraniu przedmiotu do plecaka.

Praktyczne zastosowanie problemu plecakowego:

* Załadunek palet
* Sprzedaż wysyłkowa – pakowanie zakupów
* Załadunek w odpowiedniej kolejności paczek na pojazd kurierski
* Załadunek kontenerów na statek
* Spakowanie walizki na wakacje

1. **Opis algorytmu dokładnego**

Przegląd zupełny jest określany jako algorytm dokładny dla problemu plecakowego. Metoda ta nie jest efektywna obliczeniowo (złożoność -  {\displaystyle \Theta (2^{n})}θ(2n)), jednak znajduje najlepsze rozwiązanie. Przegląd zupełny polega na systematycznym przeglądzie wszystkich możliwych rozwiązań.

Algorytm sprawdza wszystkie możliwe podzbiory zbioru P (ilość takich zbiorów równa jest 2P), sprawdza podzbiory o liczbie elementów kolejno: 1,2,3,…,n, gdzie n jest liczbą wszystkich przedmiotów.

W i-tej iteracji należy przejrzeć wszystkie podzbiory o długości i. Ilość podzbiorów w i-tej iteracji wyznacza wzór:

Jeżeli dla podzbioru są spełnione ograniczenia zadanie określane jest jako dopuszczalne i zostaje obliczony dla niego zysk, w przeciwnym przypadku podzbiór jest niedopuszczalny.

Po zakończeniu algorytmu dostarczane jest rozwiązanie zwracające największy zysk.

Przykład:

Ładowność plecaka*, b = 15*

|  |  |  |
| --- | --- | --- |
| Lp. | Waga | Wartość |
| 1 | 5 | 40 |
| 2 | 4 | 50 |
| 3 | 3 | 30 |
| 4 | 6 | 30 |

Wszystkie możliwe podzbiory z ich wyceną:

|  |  |  |  |
| --- | --- | --- | --- |
| Podzbiór | Waga | Wycena | Dopuszczenie |
| {1} | 5 | 40 | T |
| {2} | 4 | 50 | T |
| {3} | 3 | 30 | T |
| {4} | 6 | 30 | T |
| {1,2} | 5+4=9 | 40+50=90 | T |
| {1,3} | 5+3=9 | 40+30=70 | T |
| {1,4} | 5+6=11 | 40+30=70 | T |
| {2,3} | 4+3=7 | 50+30=80 | T |
| {2,4} | 4+6=10 | 50+30=80 | T |
| {3,4} | 3+6=9 | 30+30=60 | T |
| {1,2,3} | 5+4+3=12 | 40+30+50=120 | T |
| {1,2,4} | 5+4+6=15 | 40+50+30=120 | T |
| {1,3,4} | 5+3+6=14 | 40+30+30=100 | T |
| {2,3,4} | 4+3+6=13 | 50+30+30=110 | T |
| {1,2,3,4} | 5+4+3+6=18 | 40+50+30+30=150 | F |

Najlepsze rozwiązanie dla postawionego zadania stanowi podzbiór elementów: 1,2,3, którego waga wynosi 12, co sprawia, że zbiór jest dopuszczony, ponieważ maksymalna ładowność wynosi 15. (12<15), natomiast zysk jest największy dla zadanych ograniczeń i wynosi 120.

1. **Opis algorytmu aproksymacyjnego – programowanie dynamiczne**

Jednym z rozwiązań aproksymacyjnych problemu załadunku jest zagadnienie programowania dynamicznego. Polega ono na stworzeniu tabeli pokazującej maksymalny zysk dla danego podzbioru elementów o sumarycznym rozmiarze nie większym niż ładowność plecaka.

Wzór pozwalający na obliczenie wartości zysku dla kolejnych pozycji w tabeli V:

*V[0,s]=0 dla 0≤s≤b*

*V[i,s]=max{V[i-1,s],V[i-1,s-ai)]+ci dla 1≤i≤n, 0≤s≤b*

Gdzie:

V[i-1,s] oznacza najlepsze (poprzednio obliczone i zapamiętane w tabeli ) rozwiązanie dla rozważanej aktualnie pojemności plecaka s oraz podzbioru elementów {1, …., i-1}

V[i-1, s-s(ai)] - oznacza najlepsze (poprzednio obliczone i zapamiętane w tabeli) rozwiązanie dla aktualnie rozważanej pojemności plecaka s pomniejszonej o rozmiar elementu s(ai), który próbujemy dołożyć do plecaka, dla podzbioru elementów {1, … i-1}

Przykład:

Ładowność plecaka*, b = 15*

|  |  |  |
| --- | --- | --- |
| Lp. | Waga | Wartość |
| 1 | 5 | 40 |
| 2 | 4 | 50 |
| 3 | 3 | 30 |
| 4 | 6 | 30 |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| V | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| 1 | 0 | 0 | 0 | 0 | 40 | 40 | 40 | 40 | 40 | 40 | 40 | 40 | 40 | 40 | 40 |
| 2 | 0 | 0 | 0 | 50 | 50 | 50 | 50 | 50 | 90 | 90 | 90 | 90 | 90 | 90 | 90 |
| 3 | 0 | 0 | 30 | 50 | 50 | 50 | 80 | 80 | 90 | 90 | 90 | 120 | 120 | 120 | 120 |
| 4 | 0 | 0 | 30 | 50 | 50 | 50 | 80 | 80 | 90 | 90 | 90 | 120 | 120 | 120 | 120 |

V[1,1]=max{V[0,1],V[0,1-5]+40}=max{0,0}=0

V[2,5]=max{V[1,5],V[1,5-4]+50}=max{40,50}=50

V[4,12]=max{V[3,12],V[3,12-6]+30}=max{120,50+30}=max{120,80}=120

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Keep | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | **1** | 1 | 1 | 1 |
| 2 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | **1** | 1 | 1 |
| 3 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | **1** | 1 |
| 4 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | **0** |

Najlepsze rozwiązanie dla postawionego zadania stanowi podzbiór elementów: 1,2,3, którego zysk wynosi 120.

1. **Oszacowania**

n – wielkość zadania (ilość obiektów)

b – ładowność plecaka

* 1. **Teoretyczna pesymistyczna złożoność** 
     1. **Pamięciowa**
        1. Przegląd zupełny

Teoretyczna pesymistyczna złożoność algorytmu przeglądu zupełnego wynosi

T(1)= 1

T(2)= 2+1=3

T(3)= 3+3+1=7

T(4)=4+6+4+1=15

T(5)=5+10+10+5+1=31

T(n)= 2T(n-1)+1

b=2, c=1, a=1

* + - 1. Programowanie dynamiczne

Teoretyczna pesymistyczna złożoność pamięciowa algorytmu programowania dynamicznego określona jest wzorem b\*n.

T(1)=b

T(n)=T(n-1)+b

* + 1. **Obliczeniowa**
       1. Przegląd zupełny

Teoretyczna pesymistyczna złożoność wynosi 2n-1.

-1 =>

* + - 1. Programowanie dynamiczne

Teoretyczna pesymistyczna złożoność wynosi n\*b

* 1. **Teoretyczna złożoność oczekiwana** 
     1. **Obliczeniowa**
        1. Przegląd zupełny
        2. Programowanie dynamiczne
     2. **Pamięciowa**
        1. Przegląd zupełny
        2. Programowanie dynamiczne
  2. **Teoretyczna wrażliwość pesymistyczna**

Obydwa z wybranych algorytmów zarówno przegląd zupełny oraz programowanie dynamiczne są niewrażliwe, ponieważ:

* 1. **Teoretyczna wrażliwość oczekiwana** 
     + 1. Przegląd zupełny
       2. Programowanie dynamiczne

1. **Porównanie wyników**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Dane** |  | | | | | |
|  | w={1,2,3}  Val1={4,5,6}  Lad1=4 | | w={4,11,15,13,12,18}  val2={5,3,8,6,1,17,9}  lad2=58 | | w={3,4,7,8,5,6,1,9,10,  11,12,14,16,19}  val3={2,3,4,5,6,8,1,9,13, 14,15,11,15,17}  lad3=80 | |
| **Algorytm** | Przegląd zupełny | Programowanie dynamiczne | Przegląd zupełny | Programowanie dynamiczne | Przegląd zupełny | Programowanie dynamiczne |
| **Złożoność czasowa** | 282 | 430 | 141 | 418 | 16806 | 1120 |
| **Złożoność pamięciowa** | 1220 | 9 | 912 | 67 | 165445 | 96 |

**Przeprowadzone badania:**

1. **Wnioski**

Na podstawie przeprowadzonych badań możemy zauważyć, że programowanie dynamiczne jest algorytmem lepiej sprawdzającym się przy większych zbiorach danych, natomiast przegląd zupełny nadaje się lepiej do bardzo małych zbiorów danych.

Dzięki przeprowadzonym testom potwierdzone zostały wyniki oszacowań przeprowadzone w punkcie 4. Złożoność obliczeniowa wyboru przedmiotów do plecaka pokryła się z oszacowaniami, gdyby uwzględnić czynności przygotowujące w algorytmie dokładnym generowałyby one znaczną ilość operacji. Rząd złożoności pamięciowej zgadza się z oszacowanym rzędem oczekiwanej złożoności pamięciowej.

W zadanym problemie występuje przypadek, dla wybranych algorytmów, w którym algorytm dokładny mimo gorszej złożoności daje lepsze wyniki dla mniejszych zbiorów danych niż algorytm heurystyczny.