**Ćwiczenie nr 5**

**„Programowanie dynamiczne” (20.05.2010)**

**Sprawozdanie wykonali:**

**Jakub Kliszkowiak (94315)**

**Jarosław Jankun (94304)**

**I 3**

**godziny zajęć: czwartek, godz. 11:45**

**Poniższe wykresy oraz tabele przedstawiają zależności czasu obliczeń od ilości paczek dostarczanych przez kuriera. W przypadku strategii pełnego przeglądu (BF), ilość paczek (n) deklarowana była z przedziału między 10 a 30, ze skokiem co 2, zaś w przypadku pozostałych strategii (PD, GH), przedział ten wynosił 1000 do 2000, ze skokiem co 100 (z wyjątkiem podpunktu a, ze względu na treść polecenia drugiego).**

1. ***zależność czasu obliczeń od ilości paczek dla metod programowania dynamicznego (PD), pełnego przeglądu (BF1), pełnego przeglądu z eliminacją rozwiązań niedopuszczalnych (BF2) oraz heurystycznej z regułą %!#$%^!@$% przy ładowności b=50%.***

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **b=50%** | **10** | **12** | **14** | **16** | **18** | **20** | **22** | **24** | **26** | **28** | **30** |
| **PD** | 0,028 | 0,046 | 0,055 | 0,066 | 0,096 | 0,083 | 0,142 | 0,171 | 0,166 | 0,2346 | 0,275 |
| **BF1** | 0,034 | 0,111 | 0,435 | 1,748 | 6,703 | 26,558 | 95,974 | 311,974 | 1221,132 | 4926,681 | 19189,123 |
| **BF2** | 0,022 | 0,071 | 0,291 | 1,059 | 4,608 | 16,796 | 48,669 | 200,342 | 770,933 | 3093,054 | 12958,058 |
| **GH4** | 0,003 | 0,003 | 0,004 | 0,004 | 0,004 | 0,004 | 0,004 | 0,004 | 0,005 | 0,005 | 0,005 |

Problem plecakowy jest problemem optymalizacyjnym, NP-trudnym. Celem każdego problemu optymalizacyjnego jest ekstremalizacja, czyli wyznaczenie maksimum funkcji kryterialnej. W przypadku tego konkretnego problemu, jeśli mamy do czynienia ze skończonym zbiorem elementów A={a1,a2,a3,…an} o rozmiarze s(ai) i wartości (wadze) w(ai), wyznaczyć należy taki zbiór A’≤A, że: , a z kolei jest maksymalna, gdzie b oznacza rozmiar „plecaka”.

Dla każdego problemu optymalizacyjnego jednak, można podać odpowiednik decyzyjny. Takim odpowiednikiem dla problemu plecakowego jest pytanie o istnienie rozwiązania takiego, aby wartość co najmniej Y była osiągnięta bez przekraczania wagi w(ai). Tak postąpić można z każdym problemem optymalizacyjnym, jednak problem optymalizacyjny jest nie łatwiejszy – co najmniej tak samo trudny jak jego odpowiednik decyzyjny. Reasumując więc: rozwiązanie problemu optymalizacyjnego polega na znalezieniu rozwiązania, zaś problemu decyzyjnego – na udzieleniu odpowiedzi tak/nie, co może być znacznie łatwiejsze. Wersja decyzyjna owego problemu jest NP-zupełna, a więc problem ten, należy do problemów trudnych – nie może on być rozwiązany przez algorytm wielomianowy.

Można go jednak rozwiązać na przykład metodą programowania dynamicznego, która to jest strategią rozwiązywania problemów zarówno łatwych, jak i trudnych obliczeniowo, ale posiadających pewne własności (nie wszystkich). Opiera się ona na strategii „dziel i rządź”, zaś rozwiązywany za jej pomocą problem, musi dać się podzielić na podproblemy zależne od siebie.

Jeśli chodzi o złożoności obliczeniowe metod, branych pod uwagę w tym badaniu, zestawienie wygląda następująco:

* programowanie dynamiczne: O(n∙b)+O(n) = O(n∙b) – jest to złożoność pseudowielomianowa (wykładnicza).
* metoda pełnego przeglądu: O(2n), ponieważ jest tyle możliwych ciągów zero jedynkowych na n polach.
* metoda pełnego przeglądu z eliminacją rozwiązań niedopuszczalnych: @#$%^#$%@$%^#
* metoda heurystyczna: @#$%^#$%@$%^#

Metoda programowania dynamicznego, ze względu na swoją złożoność obliczeniową i jakość generowanych rozwiązań zalicza się do algorytmów @#$%^#$%@$%^#.

Wadą tej metody jest to, że programowanie dynamiczne daje się zastosować jedynie do problemów, które mają tzw. własność optymalnej podstruktury. Chodzi tu o to, że rozwiązując problem "dochodzimy" do pewnego "miejsca" i nasza dalsza droga Z tego miejsca jest całkowicie niezależna od drogi DO tego miejsca.

Metoda pełnego przeglądu, kolokwialnie nazywana metodą Brute Force, oparta jest na algorytmie z powracaniem. Jest to zwykle nieoptymalna, ale najprostsza do zaimplementowania metoda postępowania. Można więc powiedzieć, że jest to algorytm bardzo prosty i wiadomo, że działa, zwykle ma jednak nie najlepszą wydajność i jeszcze gorszą złożoność obliczeniową. Polega on na rozpatrzeniu wszystkich możliwych przypadków po kolei. Jest to algorytm wykładniczy (ze względu na złożoność obliczeniową) dający najlepsze jakościowo rozwiązanie na które nie ma wpływu zakres danych wejściowych. Złożoność czasowa tego algorytmu jest jednak największa (ze względu na w/w idee działania). Brute force ma wiele zastosowań, jest stosowany jako opcja awaryjna, używana jeśli "sprytny" algorytm nie radzi sobie z danym przypadkiem lub w sytuacji, w której prostota jest ważniejsza od wydajności - np. w bardzo rzadko wykonywanym kodzie, który i tak jest już za bardzo skomplikowany.

Metoda pełnego przeglądu z eliminacją rozwiązań niedopuszczalnych dodatkowo sprawdza @#$%^#$%@$%^# co nieco komplikuje nam kod źródłowy programu, jednak w niektórych przypadkach pozwala zaoszczędzić sporo czasu podczas rozwiązywania problemu.

Metody heurystyczne mogą być użyte do wstępnego zawężenia przestrzeni rozwiązań, tak aby na końcu zastosować algorytm klasyczny, który wyliczy wiarygodny rezultat końcowy. W przypadku metody klasycznej, przeszukiwania dziedziny rozwiązań ścieżki poszukiwań często prowadzą na „przysłowiowe manowce” i oznaczają tylko stratę czasu. Człowiek może zatem pomóc komputerowi, aby ten nie tracił czasu na wyszukiwanie rozwiązań na „ścieżkach” prowadzących do porażki lub niezbyt interesujących z punktu widzenia spodziewanego wyniku. Wadą heurystyk jest ich aspekt praktyczny: często zamiast klarownego modelu matematycznego (dającego przełożyć się na kod) wbudowujemy w algorytm pewne reguły postępowania opierające się na intuicji i doświadczeniu. Jakość heurystyk badamy doświadczalnie, analizując wyniki działania algorytmów.

1. ***zależność czasu obliczeń od ilości paczek dla metod programowania dynamicznego (PD), pełnego przeglądu (BF1), pełnego przeglądu z eliminacją rozwiązań niedopuszczalnych (BF2) oraz heurystycznej z czterema regułami wyboru paczki: losową (GH1), min{s(ai)} (GH2), max {w(ai)} (GH3) i max{w(ai)/s(ai)} (GH4). W tym przypadku dla metod Brute Force wybrano mniejsze wartości ilości paczek, ze względu na złożoność czasową tej metody.***

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | **b=25%** | **PD** | **GH4** | **n(BF)** | **BF1** | **BF2** | | **1000** | 97,364 | 0,172 | **10** | 0,033 | 0,008 | | **1100** | 117,102 | 0,194 | **12** | 0,118 | 0,015 | | **1200** | 140,241 | 0,195 | **14** | 0,454 | 0,049 | | **1300** | 161,597 | 0,213 | **16** | 1,779 | 0,146 | | **1400** | 191,259 | 0,247 | **18** | 4,895 | 0,227 | | **1500** | 219,545 | 0,263 | **20** | 20,517 | 0,886 | | **1600** | 250,591 | 0,273 | **22** | 86,043 | 3,312 | | **1700** | 276,511 | 0,281 | **24** | 349,699 | 7,854 | | **1800** | 307,716 | 0,299 | **26** | 1374,351 | 21,309 | | **1900** | 351,845 | 0,315 | **28** | 5253,321 | 127,043 | | **2000** | 386,921 | 0,357 | **30** | 21107,597 | 493,929 | | | |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | **b=50%** | **PD** | **GH4** | **n(BF)** | **BF1** | **BF2** | | **1000** | 189,944 | 0,161 | **10** | 0,034 | 0,022 | | **1100** | 235,696 | 0,191 | **12** | 0,111 | 0,071 | | **1200** | 270,549 | 0,195 | **14** | 0,435 | 0,291 | | **1300** | 327,099 | 0,214 | **16** | 1,748 | 1,059 | | **1400** | 385,376 | 0,248 | **18** | 6,703 | 4,608 | | **1500** | 427,391 | 0,271 | **20** | 26,558 | 16,796 | | **1600** | 500,973 | 0,267 | **22** | 95,974 | 48,669 | | **1700** | 560,143 | 0,302 | **24** | 311,974 | 200,342 | | **1800** | 634,402 | 0,321 | **26** | 1221,132 | 770,933 | | **1900** | 707,087 | 0,343 | **28** | 4926,681 | 3093,05 | | **2000** | 768,776 | 0,361 | **30** | 19189,123 | 12958,1 | | |
| |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | **b=75%** | **PD** | **GH4** | **n(BF)** | **BF1** | **BF2** | | **1000** | 290,783 | 0,163 | **10** | 0,022 | 0,022 | | **1100** | 360,833 | 0,202 | **12** | 0,081 | 0,081 | | **1200** | 428,994 | 0,209 | **14** | 0,293 | 0,314 | | **1300** | 479,021 | 0,229 | **16** | 1,203 | 1,361 | | **1400** | 567,729 | 0,232 | **18** | 5,028 | 5,065 | | **1500** | 636,209 | 0,268 | **20** | 17,986 | 20,884 | | **1600** | 739,809 | 0,287 | **22** | 75,267 | 84,475 | | **1700** | 852,743 | 0,305 | **24** | 301,605 | 338,936 | | **1800** | 950,979 | 0,323 | **26** | 1214,914 | 1347,24 | | **1900** | 1029,07 | 0,334 | **28** | 4864,525 | 5420,22 | | **2000** | 1138,46 | 0,361 | **30** | 19928,028 | 22155,2 | | | | |
|  | |  | |
|  | |  | |

1. ***rozwiązanie problemu za pomocą metod programowania dynamicznego oraz heurystycznej z czterema regułami wyboru paczki: losową (GH1), min{s(ai)} (GH2), max {w(ai)} (GH3) i max{w(ai)/s(ai)} (GH4).***

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **b=25%** | **1000** | **1100** | **1200** | **1300** | **1400** | **1500** | **1600** | **1700** | **1800** | **1900** | **2000** |
| **PD** | 14700 | 15849 | 18171 | 18838 | 20041 | 21617 | 23290 | 24624 | 26479 | 27580 | 29520 |
| **GH1** | 6348 | 7149 | 7886 | 8476 | 8871 | 9424 | 10762 | 10207 | 11066 | 12599 | 12941 |
| **GH2** | 12785 | 13755 | 15919 | 16271 | 17257 | 18952 | 20416 | 21461 | 23091 | 23628 | 25820 |
| **GH3** | 11315 | 12021 | 13564 | 14306 | 15830 | 16549 | 17814 | 18568 | 19911 | 20443 | 22821 |
| **GH4** | 14697 | 15849 | 18170 | 18838 | 20039 | 21617 | 23288 | 24623 | 26478 | 27578 | 29520 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **śr. błąd (b=25%)** | **GH1** | **GH2** | **GH3** | **GH4** |
| **1000** | 56,82% | 13,03% | 23,03% | 0,02% |
| **1100** | 54,89% | 13,21% | 24,15% | 0,00% |
| **1200** | 56,60% | 12,39% | 25,35% | 0,01% |
| **1300** | 55,01% | 13,63% | 24,06% | 0,00% |
| **1400** | 55,74% | 13,89% | 21,01% | 0,01% |
| **1500** | 56,40% | 12,33% | 23,44% | 0,00% |
| **1600** | 53,79% | 12,34% | 23,51% | 0,01% |
| **1700** | 58,55% | 12,85% | 24,59% | 0,00% |
| **1800** | 58,21% | 12,80% | 24,80% | 0,00% |
| **1900** | 54,32% | 14,33% | 25,88% | 0,01% |
| **2000** | 56,16% | 12,53% | 22,69% | 0,00% |
| **średnia** | 56,04% | 13,03% | 23,87% | 0,01% |

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **b=50%** | **1000** | **1100** | **1200** | **1300** | **1400** | **1500** | **1600** | **1700** | **1800** | **1900** | **2000** |
| **PD** | 20006 | 22712 | 24619 | 26610 | 28801 | 31318 | 32814 | 34273 | 36512 | 38846 | 41085 |
| **GH1** | 12279 | 14194 | 15659 | 16610 | 17669 | 19237 | 20500 | 21553 | 22375 | 23717 | 25444 |
| **GH2** | 17312 | 20110 | 21531 | 23315 | 25142 | 27309 | 28987 | 29953 | 32014 | 34040 | 36044 |
| **GH3** | 18516 | 20987 | 22641 | 24859 | 26754 | 28938 | 30203 | 31369 | 34005 | 35610 | 37698 |
| **GH4** | 20006 | 22712 | 24617 | 26610 | 28801 | 31316 | 32813 | 34273 | 36511 | 38845 | 41085 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **śr. błąd (b=50%)** | **GH1** | **GH2** | **GH3** | **GH4** |
| **1000** | 38,62% | 13,47% | 7,45% | 0,00% |
| **1100** | 37,50% | 11,46% | 7,60% | 0,00% |
| **1200** | 36,39% | 12,54% | 8,03% | 0,01% |
| **1300** | 37,58% | 12,38% | 6,58% | 0,00% |
| **1400** | 38,65% | 12,70% | 7,11% | 0,00% |
| **1500** | 38,58% | 12,80% | 7,60% | 0,01% |
| **1600** | 37,53% | 11,66% | 7,96% | 0,00% |
| **1700** | 37,11% | 12,60% | 8,47% | 0,00% |
| **1800** | 38,72% | 12,32% | 6,87% | 0,00% |
| **1900** | 38,95% | 12,37% | 8,33% | 0,00% |
| **2000** | 38,07% | 12,27% | 8,24% | 0,00% |
| **średnia** | 37,97% | 12,42% | 7,66% | 0,00% |

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **b=75%** | **1000** | **1100** | **1200** | **1300** | **1400** | **1500** | **1600** | **1700** | **1800** | **1900** | **2000** |
| **PD** | 24517 | 26352 | 29402 | 31774 | 32935 | 36745 | 38329 | 41104 | 43499 | 46089 | 47886 |
| **GH1** | 19829 | 20807 | 22889 | 25092 | 26165 | 29617 | 29835 | 32475 | 34584 | 37072 | 38008 |
| **GH2** | 22222 | 24456 | 26599 | 28919 | 29740 | 33325 | 35239 | 37464 | 39196 | 41624 | 43751 |
| **GH3** | 24228 | 26060 | 28916 | 31269 | 32379 | 36261 | 37560 | 40394 | 42639 | 45377 | 47026 |
| **GH4** | 24515 | 26352 | 29401 | 31772 | 32935 | 36745 | 38329 | 41103 | 43499 | 46089 | 47885 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **śr. błąd (b=75%)** | **GH1** | **GH2** | **GH3** | **GH4** |
| **1000** | 19,12% | 9,36% | 1,18% | 0,01% |
| **1100** | 21,04% | 7,19% | 1,11% | 0,00% |
| **1200** | 22,15% | 9,53% | 1,65% | 0,00% |
| **1300** | 21,03% | 8,99% | 1,59% | 0,01% |
| **1400** | 20,56% | 9,70% | 1,69% | 0,00% |
| **1500** | 19,40% | 9,31% | 1,32% | 0,00% |
| **1600** | 22,16% | 8,06% | 2,01% | 0,00% |
| **1700** | 20,99% | 8,86% | 1,73% | 0,00% |
| **1800** | 20,49% | 9,89% | 1,98% | 0,00% |
| **1900** | 19,56% | 9,69% | 1,54% | 0,00% |
| **2000** | 20,63% | 8,64% | 1,80% | 0,00% |
| **średnia** | 20,65% | 9,02% | 1,60% | 0,00% |

Algorytmem heurystycznym nazywamy metodę znajdowania rozwiązań, dla której nie ma gwarancji znalezienia rozwiązania optymalnego, a często nawet prawidłowego. Rozwiązań tych używa się np. wtedy, gdy pełny algorytm jest z przyczyn technicznych zbyt kosztowny, lub gdy jest nieznany (np. przy przewidywaniu). Metody używa się też często do znajdowania rozwiązań przybliżonych, na podstawie których później wylicza się ostateczny rezultat pełnym algorytmem.

Algorytmem zachłannym nazywamy algorytm, który w celu wyznaczenia rozwiązania w każdym kroku dokonuje zachłannego, tj. najlepiej rokującego w danym momencie wyboru rozwiązania częściowego. Innymi słowy algorytm zachłanny nie patrzy czy w kolejnych krokach jest sens wykonywać dane działanie, dokonuje decyzji lokalnie optymalnej, wydającej się w danej chwili najlepszą, kontynuując rozwiązanie podproblemu wynikającego z podjętej decyzji. Typowe zadanie rozwiązywane metodą zachłanną ma charakter optymalizacyjny.

Algorytmem listowym nazywamy zaś algorytm, który układa elementy według pewnej zasady czy też kryterium

Z powyższych danych wnioskujemy, że im większa liczba paczek brana jest pod uwagę, tym większa jest też ilość rozwiązań problemu, jednakże wraz ze wzrostem tej liczby, rośnie też procent średniego błędu popełnianego przez heurystyki – szczególnie zauważalne w przypadku większej ładowności dla GH3. Ogółem jednak wnioskować można, że im większa jest ładowność samochodu, tym rozwiązania są dokładniejsze – mniejszy jest średni procent błędu.

Dla metody GH2 najkorzystniejsza będzie jak największa ładowność samochodu przy jak najmniejszej liczbie paczek. Dla metody GH3 algorytm znajdzie najdokładniejsze rozwiązanie przy podobnym rozkładzie jak dla metody GH2 – duża ładowność, mało paczek, jednak widzimy tutaj dużo większy wzrost tej korzyści, przy większych ładownościach. Ostatnia z metod (GH4) niezależnie od danych wejściowych jest zawsze najdokładniejszą metodą, gdzie procent błędu bliski jest zeru.