Tomasz Kuliński 94329

Wojciech Mioduszewski 94344

Sprawozdanie

Temat: Pakowanie palet

## Opis

Podobnie jak we wcześniejszej wersji, również i teraz pierwszą czynnością wykonywaną po wczytaniu danych jest posortowanie paczek zgodnie z malejącą objętością. W ten sposób otrzymywana jest początkowa kolejność paczek. Następnie paczki przydzielane są do kolejnych palet zgodnie z zasadą, że jeżeli zbiór palet, na których da się umieścić paczkę (nawet zmieniając typ palety) jest niepusty, to paczka jest dołączana do najwcześniej utworzonej palety z tego zbioru, w przeciwnym wypadku tworzona jest nowa paleta. Dalsze działanie algorytmu polega na modyfikacji kolejności paczek, ponownym przydzieleniu ich do palet i zaakceptowaniu nowej kolejności, jeżeli wynik okaże się satysfakcjonujący, lub powrocie do poprzedniej kolejności.

Zadanie to zostało przydzielone do czterech wątków, które przez 20 iteracji (nazywanych dalej epokami) zajmują się szukaniem jak najlepszego wyniku. Każda epoka trwa 49 ms., po czym każdy wątek „chwali się” najlepszym uzyskanym wynikiem. Epoka kończy się zapisaniem najlepszego wyniku i przyjęciem go jako kolejność startowa dla każdego wątku w kolejnej epoce.

Całe przetwarzanie opiera się o zasadę zbliżoną do symulowanego wyżarzania, jednak z liniowo malejącą temperaturą. Od temperatury zależy ilość zmian wprowadzanych w kolejności paczek, a także poziom akceptowalności otrzymywanego wyniku (jeżeli okaże się gorszy od poprzednio otrzymanego). Co 6 epok, począwszy od 4. epoki, wykonywane są „losowe” epoki, w których wątki nie zaczynają przetwarzania z najlepszą kolejnością, tylko w pełni losową, co ma pozwolić na znalezienie lepszych, ale bardzo odległych od obecnego optimum, rozwiązań (uzasadnienie obecności takich epok znajduje się na końcu sprawozdania).

## Wyniki (11 uruchomień programu dla każdej instancji):

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Instancja | suma pól | objętość | min(iteracje) | mediana(iteracje) | max(iteracje) |
| pp101.in | 576 | 720 | 268314 | 306862 | 316029 |
| pp102.in | 864 | 1200 | 268314 | 306862 | 316029 |
| pp103.in | 384 | 480 | 3008634 | 3652952 | 3839115 |
| pp104.in | 432 | 1440 | 3359585 | 3488989 | 3658890 |
| pp105.in | 192 | 480 | 736341 | 1039389 | 1078289 |
| pp106.in | 192 | 720 | 1265612 | 1760972 | 1833212 |
| pp107.in | 240 | 960 | 840324 | 1138337 | 1167167 |
| pp108.in | 288 | 240 | 2088176 | 2837216 | 2957259 |
| pp109.in | 432 | 1200 | 309493 | 414884 | 461967 |
| pp110.in | 384 | 1200 | 861025 | 948973 | 995081 |

Ze względu na identyczne wyniki dla każdej instancji w każdym uruchomieniu, zdecydowaliśmy się dodatkowo zaprezentować statystyki dotyczące liczby przejrzanych (niekoniecznie unikalnych) kolejności paczek.

Instancja z pliku *pp109.in* podczas rozwoju algorytmu stwarzała najwięcej problemów, gdyż od czasu do czasu dawała różne wyniki w drugim kryterium optymalizacji (wartość 1200 była zastępowana wartością 1440). Dało to podstawy do oceny wprowadzanych rozwiązań poprzez sprawdzanie odsetka wystąpień pożądanej wartości (1200), przy ograniczeniu czasu trwania epoki, bądź liczby epok. Poniżej prezentujemy zależność między obecnością „losowych” epok, a otrzymywanymi wynikami:

Pomijając ekstremalnie małe wartości współczynników, algorytm z "losowymi" epokami uzyskuje wyniki nie tylko nie gorsze, a nawet lepsze od algorytmu bez takich epok, co uzasadnia jego stosowanie.

## Specyfikacja środowiska:

Procesor: Intel Xeon 1230v3

Pamięć RAM: 16 GB

Java: 1.7.0\_40-b43

Wykorzystywane rdzenie: 4 rdzenie