Politechnika
Wrocławska

Wydział Elektroniki, Fotoniki i Mikrosystemów

Sterowanie Procesami Dyskretnymi

Problemy jednomaszynowe

Członkowie grupy:	Eryk Sikora, 263453 Aleksander Biskup, 264346
Prowadzący zajęcia:	dr inż. Agnieszka Wielgus
Grupa zajęciowa:	Poniedziałek $11^{15} - 13^{00}$

2 kwietnia 2024

Spis treści

1 Opis problemu

kilka zdań, typu: dana jest jedna maszyna oraz zbiór J=1,2,. . . , j,. . . , N zadań. Zadania charakteryzują się
W ramach wykonywanych ćwiczeń przerabiane były podstawowe problemy jednomaszynowe. Jak sama nazwa wskazuje dostępna była jedna maszyna oraz zbiory zadań o różnej ilości zadań. Każdy zbiór wyróżniał się unikalnymi zależnościami wartości r_j, q_j oraz p_j, umożliwiając tym samym sprawdzenie poprawności implementowanych algorytmów szeregowania zadań.

2 Wybrane algorytmy

1. Algorytmy heurystyczne oparte o sortowanie (po r_j i q_j)
2. Przegląd zupełny
3. Algorytm Schrage (z i bez wywłaszczeń)
4. Algorytm Carliera
5. Algorytm własny

3 Algorytm własny

3.1 Opis

Zamysłem algorytmu własnego jest nadawanie priorytetów na podstawie czasu stygnięcia, a następnie sprawdzanie pomiędzy czasami rozpoczęcia pracy nad zadaniami nie powstają luki. Jeśli luka zostaje wykryta, algorytm podejmuje próbę wykonania zadania o mniejszym priorytecie, o czasie wykonania równym lub mniejszym niż czas przerwy pomiędzy zadaniami wykonywanymi według priorytetu. Algorytm rozważa wzięcie tylko jednego z zadań o mniejszym priorytecie natomiast wybiera takie, które będzie wykonywać się najdłużej aby zapełnić lukę w możliwie jak największym stopniu.

3.2 Pseudokod

Algorithm 1 Algorytm Bisora

```
1:  ogrinal ← main_list
2:  current_item_occur_time
3:  orginal_size ← list_size
4:  temporary, top_item
5:  idleQueue ← priority queue ordered by compareByIdleTime
6:  helpQueue ← priority queue ordered by compareByIdleTime
7:  workQueue ← priority queue ordered by compareByWorkAndOccurTime
8:  for each item in main_list do
9:      idleQueue.push(item)
10: main_list.clear()
11: list_size ← 0
12: while list_size < orginal_size do
13:     current_item_occur_time ← idleQueue.top().getOccurTime()
14:     temporary ← idleQueue.top()
15:     idleQueue.pop()
16:     while not idleQueue.isEmpty() do
17:         top_item ← idleQueue.top()
18:         item_total_time ← top_item.getOccurTime() + top_item.getWorkTime()
19:         if current_item_occur_time < item_total_time then
20:             helpQueue.push(top_item)
21:         else
22:             workQueue.push(top_item)
23:         idleQueue.pop()
24:     if not workQueue.isEmpty() then
25:         top_item ← workQueue.top()
26:         workQueue.pop()
27:         main_list.push_back(top_item)
28:     main_list.push_back(temporary)
29:     while not workQueue.isEmpty() do
30:         top_item ← workQueue.top()
31:         workQueue.pop()
32:         idleQueue.push(top_item)
33:     while not helpQueue.isEmpty() do
34:         top_item ← helpQueue.top()
35:         helpQueue.pop()
36:         idleQueue.push(top_item)
37:     list_size ← list_size + 1
```

4 Wyniki eksperymentu numerycznego

Link do sheetsa z wynikami w tabeli

Rozmiar problemu	**5**	**6**	**7**	**9**	**10**	**11**
Wartość kryterium Sort r_j	2313	34	3212	2364	746	13959
Czas dział. alg. Sort r_j [ms]	41	6	9	7	7	8
Błąd względny wykonania kryterium	0,04%	6,25%	0,00%	0,00%	16,38%	0,70%
Wartość kryterium Sort q_j	2822	43	3414	3235	886	20300
Czas dział. alg. Sort q_j [ms]	37	7	8	6	8	8
Błąd względny wykonania kryterium	22,06%	34,38%	6,29%	36,84%	38,22%	46,44%
Wartość przeglądu zupełnego	2312	32	3212	2364	641	13862
Czas działanai przeglądu zupełnego [ms]	3234	21239	41278	2170404	22127281	244114441
Błąd względny wykonania kryterium	0,00%	0,00%	0,00%	0,00%	0,00%	0,00%
Wartość kryterium Konstr.	2321	32	3212	2364	806	14801
Czas dział. alg. Konstr. [ms]	56	22	30	31	33	43
Błąd względny wykonania kryterium	0,39%	0,00%	0,00%	0,00%	25,74%	6,77%
Wartość kryterium Schrage V1	2313	32	3212	2364	687	13959
Czas dział. alg. Schrage V1 [ms]	66	12	18	17	34	17
Błąd względny wykonania kryterium	0,04%	0,00%	0,00%	0,00%	7,18%	0,70%
Wartość kryterium Schrage V2	2313	32	3212	2364	687	13959
Czas dział. alg. Schrage V2 [ms]	134	18	44	38	46	205
Błąd względny wykonania kryterium	0,04%	0,00%	0,00%	0,00%	7,18%	0,70%
Wartość kryterium Schrage z wywłaszczeniem	52273	32	3212	2251	641	13826
Czas dział. alg. Schrage z wywłaszczeniem [ms]	228	18	91	64	61	469
Błąd względny wykonania kryterium	2160,94%	0,00%	0,00%	4,78%	0,00%	0,26%

Tabela 1: Tabela wyników działania poszczególnych algorytmów dla różnych zbiorów zadań

5 Wnioski

- Dla większych zbiorów danych algorytm Schrage zwraca nieprawidłowy wynik. Może być to spowodowane tzw. corner case, którego nie udało się odnaleźć podczas debugowania.
- Algorytm Schrage został zaimplementowany na dwa różne sposoby, pierwszy z nich opiera się na wektorach, drugi na kolejkach priorytetowych. Zazwyczaj dają one jednakowe wyniki, natomiast czasami występują różnice. Z obserwacji wynika, że ten dający mniejszy wynik był tym poprawnie działającym.