Sprawozdanie PTSZ nr 2

Kamil Kowalczyk
136742, i2, L1
zajęcia środa 11:45

Generator instancji

Generator został zaprojektowany z użyciem maszyny generującej liczby pseudolosowe.

Maszyny:

$$b_1 = 1$$
$$b_2 = random(1,\ sum - 4)$$
$$b_3 = random(1,\ sum - 4 - (b_2 - 1))$$
$$b_4 = random(1,\ sum - 4 - ((b_2 - 1)\ +\ (b_3 - 1)))$$
$$b_5 = random(1,\ sum - 4 - ((b_2 - 1)\ +\ (b_3 - 1) + (b_4 - 1)))$$

gdzie:

$random(a,\ b)$ - funkcja generująca liczbę rzeczywistą z przedziału [a, b],

sum - parametr ustalający oczekiwaną wartość sumy współczynników prędkości

Zadania:

$$p_j = randomInt(1,\ 10)$$
$$r_j = randomInt(1,\ 2size)$$

gdzie

$randomInt(a,\ b)$ - funkcja generująca liczbę naturalną z przedziału [a, b]

$size$ - rozmiar instancji

Parametrowi *sum* nadano wartość 10. Dzięki temu wartości współczynników prędkości poszczególnych maszyn, dla każdej generowanej instancji, sumują się do ok. 10.
Pomysł na sposób generowania zadań został oparty na następującym rozumowaniu:

- czas trwania każdego zadania to losowa liczba całkowita z przedziału [1, 10], zatem średnio zadanie będzie trwało 5

- moment gotowości zadania to losowa liczba całkowita z przedziału [1, *size* * 2], więc dla przykładowego rozmiaru instancji równego 100, momenty gotowości tych stu zadań będą “rozrzucone” na osi [1, 200]. Gdyby posortować gotową listę zadań po wartościach momentów gotowości rosnąco, można by wyliczyć, że średnio wartość bezwzględna z różnicy wartości momentów gotowości zadań stojących obok siebie, dla *size* = 100, wynosi 2. A więc na osi czasu punkty reprezentujące momenty gotowści zadań będą oddalone od siebie o średnio 2.
- wartość sumy współczynników prędkości maszyn jest bliska 10. Maszyn jest 5, więc średnia prędkość jednej maszyny to 2

Podsumowując ten tok rozumowania dla rozmiaru instancji równego na przykład 100, posiadamy 5 maszyn o średniej prędkości 2 i 100 zadań o średnim czasie trwania równym 5. Można, więc wyliczyć, że średnio jedno zadanie na jednej maszynie będzie wykonywane przez czas wynoszący 5 * 2 = 10. Maszyn jest 5, zadania trwają 10, a kolejne zadania pojawiają się co 2 na osi czasu. W ten sposób zadania powinny być ciasno “upchane” na osi czasu, ale jednocześnie niezbyt ciasno, by dać szansę algorytmowi rozwiązującemu problem na optymalizację poprzez minimalizację opóźnień w rozpoczynaniu zadań.

Algorytm

Wpierw program sortuje zadania po wartościach momentów gotowości rosnąco. Następnie dzieli zbiór tych zadań na pięcioelementowe podzbiory, które zawierają kolejne, występujące obok siebie zadania:

[J1, J2, J3, J4, J5, J6, J7, J8, J9, J10] => [J1, J2, J3, J4, J5], [J6, J7, J8, J9, J10]

Zadania z każdej piątki są przypisywane do maszyn, w taki sposób, że zadanie z najkrótszym czasem trwania w danej piątce jest przypisywane do najwolniejszej maszyny, a to z najdłuższym czasem do najszybszej. Pozostałe trzy zadania z podgrupy przydzielane są analogicznie, w zależności od ich czasów trwania. Każda maszyna otrzymuje tylko jedno zadanie z każdej pięcioelementowej grupy. Efektem takiego działania algorytmu jest to, że w rozwiązaniu problemu wszystkie maszyny mają taką samą ilość przydzielonych zadań równą *size*/5. Jest to oczywista wada tego rozwiązania. Jednak pomimo różnych prób rozszerzenia tego rozwiązania i testów nie udało się osiągnąć lepszych rezultatów.

Złożoność

Sortowanie listy po kluczu (python, funkcja sort): $O(n \ log \ n)$
źródło: https://wiki.python.org/moin/TimeComplexity

Przejście po całej liście i przydzielanie zadań do maszyn: $O(n)$

Złożoność: $O(n \ log \ n)$

Wyniki

n	Wartość kryterium dla sztucznego pliku wynikowego	Wartość kryterium uzyskana przez własny algorytm	Względna różnica w [%]	Czas działania algorytmu [s]
50	58,11	8,26	85,79	0,286
100	201,05	49,1	75,58	0,269
150	282,35	20,7	92,67	0,37
200	288,2	25,99	90,98	0,263
250	483,76	18,04	96,27	0,27
300	514,41	23,8	95,37	0,285
350	634,36	31,6	95,02	0,279
400	714,25	23,26	96,74	0,269
450	819,69	12,99	98,42	0,285
500	929,15	34,69	96,27	0,285

Średnia względna różnica dla powyższych danych wynosi: 92,31%
Średnia względna różnica dla całej biblioteki wynosi: 63,20%