# Analiza wyników testów grafu intree15.

Przeprowadzając testy uruchomiono 1000 razy algorytm GEO do szeregowania na grafie intree15 w następujących konfiguracjach parametru prawdopodobieństwa τ: 0.1, 0.2, 0.5, 0.8, 1.0, 1.5, 2.0, 5.0 i 8.0. Wszystkie uruchomienia wykonały 100 iteracji.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **0,1** | **0,2** | **0,5** | **0,8** | **1** | **1,5** | **2** | **5** | **8** |
| **1** | 1,0000 | 1,0000 | 1,0000 | 1,0000 | 1,0000 | 1,0000 | 1,0000 | 1,0000 | 1,0000 |
| **2** | 0,9330 | 0,8706 | 0,7071 | 0,5743 | 0,5000 | 0,3536 | 0,2500 | 0,0313 | 0,0039 |
| **3** | 0,8960 | 0,8027 | 0,5774 | 0,4152 | 0,3333 | 0,1925 | 0,1111 | 0,0041 | 0,0002 |
| **4** | 0,8706 | 0,7579 | 0,5000 | 0,3299 | 0,2500 | 0,1250 | 0,0625 | 0,0010 | 0,0000 |
| **5** | 0,8513 | 0,7248 | 0,4472 | 0,2759 | 0,2000 | 0,0894 | 0,0400 | 0,0003 | 0,0000 |
| **6** | 0,8360 | 0,6988 | 0,4082 | 0,2385 | 0,1667 | 0,0680 | 0,0278 | 0,0001 | 0,0000 |
| **7** | 0,8232 | 0,6776 | 0,3780 | 0,2108 | 0,1429 | 0,0540 | 0,0204 | 0,0001 | 0,0000 |
| **8** | 0,8123 | 0,6598 | 0,3536 | 0,1895 | 0,1250 | 0,0442 | 0,0156 | 0,0000 | 0,0000 |
| **9** | 0,8027 | 0,6444 | 0,3333 | 0,1724 | 0,1111 | 0,0370 | 0,0123 | 0,0000 | 0,0000 |
| **10** | 0,7943 | 0,6310 | 0,3162 | 0,1585 | 0,1000 | 0,0316 | 0,0100 | 0,0000 | 0,0000 |
| **11** | 0,7868 | 0,6190 | 0,3015 | 0,1469 | 0,0909 | 0,0274 | 0,0083 | 0,0000 | 0,0000 |
| **12** | 0,7800 | 0,6084 | 0,2887 | 0,1370 | 0,0833 | 0,0241 | 0,0069 | 0,0000 | 0,0000 |
| **13** | 0,7738 | 0,5987 | 0,2774 | 0,1285 | 0,0769 | 0,0213 | 0,0059 | 0,0000 | 0,0000 |
| **14** | 0,7680 | 0,5899 | 0,2673 | 0,1211 | 0,0714 | 0,0191 | 0,0051 | 0,0000 | 0,0000 |
| **15** | 0,7628 | 0,5818 | 0,2582 | 0,1146 | 0,0667 | 0,0172 | 0,0044 | 0,0000 | 0,0000 |

Tabela : Prawdopodobieństwo wyboru konfiguracji do dalszej analizy w zależności od parametru prawdopodobieństwa

W tabeli Tabela 1 przedstawiono prawdopodobieństwo z jakim może zostać wybrana konfiguracja podziału zadań na procesorach do następnej iteracji jako wartość bazowa. Oznacza to również szansę, że wybrana konfiguracja zostanie wyselekcjonowana do porównania z aktualnie najlepszym ustawieniem i może zastąpić je w przypadku, gdy czas szeregowania będzie lepszy.

Istotne jest, że szansa wybrania danej konfiguracji pojawia się dopiero w sytuacji, gdy wcześniej podana konfiguracja zostanie wylosowana. Rozkład prawdopodobieństwa jest w tym wypadku stały i wynosi 1/k, gdzie k jest ilością konfiguracji. Oznacza to, że przykładowo, gdy wartość parametru τ=1, a ilość konfiguracji wynosi 15, to szansa, że do następnej konfiguracji przejdzie konfiguracja numer 1 wynosi 1/15 \* 1, z kolei szansa wyboru konfiguracji numer 10 wynosi 1/15 \* 1/10 = 1/150. Istnieje także niezerowa szansa, że w losowaniu nie zostanie wybrana konfiguracji i będzie należało ponowić wybór.

Bardzo ważną zależnością widoczną w tabeli jest fakt zmniejszania szansy na wybór gorszych konfiguracji (im wyższy numer tym gorsze ustawienie) wraz ze wzrostem wartości parametru τ. W przypadku τ=8 szansa, że zostanie wybrane ustawienie inne od najlepszego wynosi < 0.1% i w przypadku wykonania 100 iteracji prawdopodobnie nie zostanie wybrana inna konfiguracja niż najlepsza do dalszego przetwarzania*. [We wnioskach ogólnych napisać, że to zły wybór parametru, ze względu na możliwość utknięcia w minimum lokalnym i niewielka szansa na wyjście. Bardzo widoczne w grafach złożonych jak g40 ]*

Wykres : Przebieg algorytmu dla grafu intree15 i τ=8.0

Wykres : Przebieg algorytmu dla grafu intree15 i τ=0.1

Na wykresach Wykres 1 oraz Wykres 2 zauważalna jest różnica przebiegu algorytmu w zależności od parametry prawdopodobieństwa. W przypadku, gdy jego wartość jest mała (np. 0.1) można zauważyć, że nie zawsze wybierana jest do następnej iteracji najlepsza konfiguracja procesów. Jest to różnica między linią niebieską obrazującą najlepszy czas szeregowania w danej iteracji, a czerwoną – wartość czasu szeregowania dla wybranej (wylosowanej) konfiguracji. Druga zauważalna różnica, to odejście od najlepszej konfiguracji do innej, skrajnie najgorszej. Widoczne jest to przy odchyleniach linii czerwonej od zielonej. Taki przebieg pozwala na wyjście z minimum lokalnego, co może prowadzić do znalezienia najlepszej konfiguracji w następnych iteracjach. Warto też zauważyć, że nie dochodzi w przypadku tego grafu do nagłych zmian wybranej konfiguracji. Przy przejściu od lepszego czasu do gorszego czasu szeregowania i odwrotnie nie następuje w skokach o kilka wartości lecz o 1 lub 2 jednostki czasu. Ma to związek z typem grafu. Procesy są tutaj powiązane w linii prostej, nie ma skomplikowanych zależności, a koszty zmiany procesora są niewielkie. To skutkuje tym, że generowanie nowych konfiguracji nie powoduje powstania dużo słabszych, a jedynie powoduje niewielkie odchylenia. Z kolei na wykresie dla dużego prawdopodobieństwa (np. 8.0) brak jest jakichkolwiek odchyleń, wszystkie linie nakładają się na siebie. Przyczyny są tutaj dwie:

* Nie wybrano konfiguracji o rankingu niższym niż 2 (wykorzystano dwie najlepsze w kolejności)
* Graf procesów jest prosty, koszty zmiany procesu są niskie (wynoszą 1) i istnieje wiele konfiguracji, które dają najlepszy czas szeregowania. Przez to zmiana procesora na którym wykonane zostanie jedno zadanie, a pozostawienie pozostałych bez zmian często nie spowoduje zmiany czasu szeregowania.

## Dane statyczne

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **τ** | **średnia** | **wariancja** | **min** | **Q1** | **mediana** | **Q3** | **max** | **moda** | **moda (%)** | **skuteczność (%)** |
| 0.1 | 4,322 | 27,097 | 0 | 1 | 3 | 6 | 42 | 0 | 21,1 | 100 |
| 0.2 | 3,502 | 14,579 | 0 | 1 | 2 | 5 | 32 | 1 | 19,4 | 100 |
| 0.5 | 2,03 | 3,669 | 0 | 1 | 2 | 3 | 11 | 1 | 26,3 | 100 |
| 0.8 | 1,749 | 2,498 | 0 | 1 | 1 | 2 | 10 | 1 | 30,3 | 100 |
| 1.0 | 1,723 | 2,056 | 0 | 1 | 1 | 2 | 12 | 1 | 33,4 | 100 |
| 1.5 | 1,519 | 1,381 | 0 | 1 | 1 | 2 | 9 | 1 | 37 | 100 |
| 2.0 | 1,497 | 1,279 | 0 | 1 | 1 | 2 | 6 | 1 | 33,3 | 100 |
| 5.0 | 1,474 | 1,183 | 0 | 1 | 1 | 2 | 6 | 1 | 38,1 | 100 |
| 8.0 | 1,483 | 1,247 | 0 | 1 | 1 | 2 | 6 | 1 | 40,5 | 100 |

Tabela : Dane statystyczne 1000 uruchomień algorytmu dla grafu intree15 i 100 iteracji

Tabela 2 zawiera dane statystyczne opisujące wykonanie 1000 uruchomień algorytmu. Nie zostały odfiltrowane żadne informacje. Oznacza to, że zostały zawarte uruchomienia, w których już wylosowana konfiguracja bazowa miała najlepszy czas szeregowania (ilość iteracji wynosi 0, co może obniżać średnią) oraz nie znaleziono najlepszej konfiguracji przy uruchomieniu 100 iteracji (do takiej sytuacji nie doszło w przypadku tego grafu).

Dane pokazują zależność pomiędzy ilością iteracji potrzebną do znalezienia optymalnego rozwiązania a parametrem prawdopodobieństwa τ. Spoglądając na średnią i wariancję można zauważyć, że najlepsze wyniki otrzymano, gdy wartość τ wynosiła 5. Wtedy średni ilość iteracji potrzebna do znalezienia optimum wyniosła 1,474 i jest znacznie lepsza od średniej dla wartości parametru równej 0.1 i wynoszącej 4,322. Także wariancja potwierdza, że różnica między potrzebnymi iteracjami jest nieznaczna. Dodatkowym potwierdzeniem jest wartość mediany, kwartylu górnego oraz maksymalnej liczby iteracji. Porównując wartości dla τ równego 0.1 i 5.0 mamy odpowiednio:

* medianę równą 3 i 1, co oznacza, że w 50% uruchomień potrzebne były co najmniej 3 iteracje w przypadku τ=0.1 i tylko 1 w przypadku τ=5.0
* Q3 równe 6 i 2, jest to duża różnica jeśli chodzi o ilość potrzebnych iteracji
* maksymalną liczbę iteracji równą 42 i 6.

W przypadku dominanty sytuacja prawie we wszystkich przypadkach wygląda podobnie. Najczęściej potrzeba tylko jednej iteracji do znalezienia optymalnego rozwiązania. Odstępstwem jest uruchomienie algorytmu z parametrem τ=0.1. Tutaj większe prawdopodobieństwo ma znalezienie optimum przy losowaniu bazowej konfiguracji (wynosi 21,1%), niż po pierwszej iteracji (wynosi ono 17%). Dla τ=5.0 prawdopodobieństwo znalezienie najlepszego rozwiązania w 1 iteracji wynosi 38,1 % a łącznie z szansą wylosowania w zerowej iteracji osiąga 56,5%, dla τ=8.0 wartości te to odpowiednio 40,5% oraz 58,2%.

Mimo dużej rozbieżności w wynikach dla wszystkich konfiguracji udało się osiągnąć najlepszy rezultat w 100 iteracjach.

## Podsumowanie

Analizując powyższe dane nasuwa się wniosek, że najlepszą wartością parametru τ dla grafu intree15 jest 5.0 i 8.0. Przyczyny zostały wymienione wcześniej przy opisywaniu różnicy przebiegu algorytmu dla różnych wartości parametru τ, graf intree15:

* posiada proste zależności procesów
* koszt zmiany procesora jest niewielki i wynosi 1

Te czynniki powodują, że istnieje znaczny procent rozwiązań, które dają najlepszy czas szeregowania. Przejście z gorszej do najlepszej konfiguracji wymaga wykonania tylko kilku kroków co oznacza konieczność zmieniania tylko 2-3 zdań na procesorze. Nasuwa się stwierdzenie, że w przypadku algorytmu GEO nie istnieje minimum lokalne wśród zbioru konfiguracji. To sprawia, że łatwiej jest znaleźć najlepsze rozwiązanie zawsze wybierając minimum w aktualnym zbiorze rozwiązań, niż przeszukiwać wybierając losowo kolejny zbiór. Druga ewentualność w przypadku tego grafu prowadzi do zbytniego skupienia się na ominięciu minimum lokalnego, które nie istnieje i próbie przeszukania rozwiązań wśród zbyt wielu konfiguracji.

# Wyniki tree15

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **τ** | **średnia** | **wariancja** | **min** | **Q1** | **mediana** | **Q3** | **max** | **moda** | **moda (%)** | **skuteczność (%)** |
| 0.1 | 4,114 | 21,388 | 0 | 1 | 3 | 6 | 34 | 0 | 21 | 100 |
| 0.2 | 3,073 | 11,363 | 0 | 1 | 2 | 5 | 22 | 0 | 22 | 100 |
| 0.5 | 2,152 | 3,781 | 0 | 1 | 2 | 3 | 16 | 1 | 27,8 | 100 |
| 0.8 | 1,642 | 2,03 | 0 | 1 | 1 | 2 | 8 | 1 | 33,4 | 100 |
| 1.0 | 1,673 | 1,744 | 0 | 1 | 1 | 2 | 8 | 1 | 33,8 | 100 |
| 1.5 | 1,443 | 1,43 | 0 | 1 | 1 | 2 | 6 | 1 | 36,3 | 100 |
| 2.0 | 1,433 | 1,209 | 0 | 1 | 1 | 2 | 5 | 1 | 35,4 | 100 |
| 5.0 | 1,418 | 1,192 | 0 | 1 | 1 | 2 | 6 | 1 | 41,8 | 100 |
| 8.0 | 1,394 | 1,138 | 0 | 1 | 1 | 2 | 6 | 1 | 38 | 100 |

Tabela : Dane statystyczne uruchomienia 1000-krotnego algorytmu na drzewie tree15 - 100 iteracji

Wyniki przeprowadzonych testów na grafie tree15 są podobne do wyników otrzymanych przy wykonaniu na drzewie intree15. Przyczyną podstawową jest podobieństwo grafów, intree jest odwróceniem grafu tree. Tutaj także najlepsze wyniki pojawiają się przy wprowadzeniu parametru prawdopodobieństwa równego 5.0 lub 8.0. Wszystkie uruchomienia zakończyły się sukcesem i zawsze otrzymano najlepszy wynik.

# Wyniki g18

Graf g18 jest znacznie bardziej złożony niż opisane wcześniej intree15 i tree15. Wagi krawędzi są identyczne i wynoszą 1. Oznacza to, że koszt zmiany procesora jest niewielki. Czasy wykonania zadań na poszczególnych poziomach grafu są różne, a zależności są tak skonstruowane, że przejście do następnego poziomu n+1 przeważnie wymaga wykonania wszystkich zadań na poziomie n. Powoduje to, że wybierając najlepszą ścieżkę ważne jest równoległe wykonywanie zadań na poziomie n i podział zadań w taki sposób, żeby przejście do poziomu n+1 powodowało niewielką stratę czasową związaną z przesłaniem danych między procesorami.

Najlepszy czas szeregowanie dla tego grafu wynosi 46 na 2 procesorach.

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **τ** | **średnia** | **wariancja** | **min** | **Q1** | **mediana** | **Q3** | **max** | **moda** | **moda (%)** | **skuteczność (%)** | **średni czas szeregowania** |
| 0.1 | 21,813 | 454,781 | 0 | 7 | 15 | 29 | 100 | 4 | 5,4 | 98,5 | 46,016 |
| 0.2 | 13,55 | 172,434 | 0 | 4 | 9 | 19 | 82 | 3 | 6,2 | 100 | 46 |
| 0.5 | 7,095 | 80,985 | 0 | 3 | 4 | 8 | 80 | 3 | 15,2 | 100 | 46 |
| 0.8 | 8,765 | 268,376 | 0 | 2 | 4 | 7 | 100 | 3 | 18 | 99 | 46,01 |
| 1.0 | 10,234 | 468,444 | 0 | 2 | 3,5 | 5 | 100 | 3 | 20,9 | 96,5 | 46,041 |
| 1.5 | 16,896 | 1067,88 | 0 | 2 | 3 | 5 | 100 | 3 | 23,2 | 87,7 | 46,156 |
| 2.0 | 16,625 | 1116,62 | 0 | 2 | 3 | 4 | 100 | 3 | 25,3 | 86,7 | 46,185 |
| 5.0 | 17,341 | 1193,44 | 0 | 2 | 3 | 4 | 100 | 3 | 24,3 | 85,4 | 46,242 |
| 8.0 | 18,643 | 1292,1 | 0 | 2 | 3 | 4 | 100 | 3 | 26,7 | 83,7 | 46,324 |

Tabela : Dane statystyczne 1000 uruchomień na grafie g18 - 100 iteracji

Dane w Tabeli 4 znacznie odbiegają od wyników otrzymanych przy grafach intree15 i tree15. Uruchomienia dla tego grafu najlepsze wyniki osiągały przy τ=0.5. Ilość iteracji potrzebnych do osiągnięcia optimum była najmniejsza (średnia = 7.095, mediana 5), a wszystkie uruchomienia zakończyły się sukcesem – znaleziono najlepsze rozwiązanie.

Wykres : Przebieg algorytmu t=0.5, najlepszy rezultat osiągnięty w iteracji 4

Typowy przebieg algorytmu został przedstawiony na Wykres 3. Można zauważyć, że wielokrotnie wybierana jest słabsza konfiguracja (dłuższy czas wykonania), jednak w efekcie w większości przypadków w kilku kolejnych krokach algorytm znów osiąga optymalny rezultat.

Wykres : Przebieg algorytmu t=8.0, optimum osiągnięto w 3 iteracji.

Analizując dane wynikowe należy zwrócić uwagę na dominantę ilości potrzebnych iteracji oraz procent wystąpień tej wartości. W przypadku, gdy wartość parametru jest równa 8.0 przebieg algorytmu skupia się na wybieraniu najlepszego wyniku do następnej iteracji co przedstawia Wykres 4. Taki sposób postępowania okazał się sukcesem w 83.7% przypadków. Pozostałe uruchomienia nie zakończyły się sukcesem ze względu na minima lokalne w których algorytm zapętlił się. Zobrazowane jest to na wykresach Wykres 5 i Wykres 6 przedstawiających 2 uruchomienia dla parametru τ=5.0. Pierwszy z nich pokazuje przebieg, w którym znaleziono optimum. Jak można zauważyć algorytm próbuje wybierać konfiguracje zbliżone do optimum (linia czerwona) cały czas wybierając do następnej iteracji najlepszy wynik (linia niebieska). Dopiero w iteracji 58 zostaje wylosowana słabsza konfiguracja i następuje wyjście z minimum lokalnego, w efekcie prowadzi to do znalezienia w kilku następnych iteracjach minimum globalnego.

Wykres : Przebieg algorytmu τ=5.0, ilość iteracji do osiągnięcia minimum = 61

Minimum lokalne w przypadku algorytmu GEO to sytuacja w której wybranie najlepszej konfiguracji spośród możliwych w iteracji n, spowoduje że w iteracji n+1 najlepszą konfiguracją będzie ta z iteracji n. Wybierając zawsze najlepsze rozwiązanie następuje w zapętlenie. Jest to widoczne na drugim wykresie. W trakcie przebiegu trzykrotnie nastąpiła próba wyjścia z minimum lokalnego. W iteracji 37 próba pozwoliła na wyjście z jednego minimum lokalnego do drugiego, co w rezultacie poprawiło wynik algorytmu. Następnie w iteracjach 52 i 72 nastąpiła nieudana próba wyjścia poza minimum lokalne i w rezultacie algorytm zakończył się po 100 iteracjach z wyznaczonym czasem szeregowania 48, nie osiągnięto optymalnego wyniku.

Wykres : Przebieg algorytmu τ=5.0, nie znaleziono minimum globalnego

# Wyniki g40

Graf g40 jest bardziej złożony niż g18. Podobnie jak g18 można zauważyć podział zadań na poziomy, jednak jest ich więcej oraz są bardziej złożone. Minimalny czas szeregowania na dwóch procesorach wynosi 80.

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **τ** | **średnia** | **wariancja** | **min** | **Q1** | **mediana** | **Q3** | **max** | **moda** | **moda (%)** | **skuteczność (%)** | **średni czas szeregowania** |
| 0.1 | 28,964 | 611,51 | 0 | 10 | 21 | 40 | 100 | 4 | 3,2 | 97,2 | 80,04 |
| 0.2 | 15,945 | 175,063 | 0 | 6 | 12 | 22 | 100 | 7 | 5,4 | 99,9 | 80,001 |
| 0.5 | 9,705 | 96,671 | 0 | 4 | 7 | 12 | 100 | 5 | 8,9 | 99,9 | 80,001 |
| 0.8 | 8,948 | 134,232 | 0 | 3 | 6 | 10 | 100 | 3 | 12,8 | 99,8 | 80,002 |
| 1.0 | 8,529 | 123,577 | 0 | 3 | 5 | 9 | 100 | 4 | 12,4 | 99,6 | 80,004 |
| 1.5 | 10,167 | 247,521 | 0 | 3 | 5 | 9 | 100 | 3 | 15,5 | 98,8 | 80,012 |
| 2.0 | 14,134 | 528,551 | 0 | 3 | 5 | 12 | 100 | 3 | 13,9 | 96,5 | 80,035 |
| 5.0 | 31,425 | 1733,47 | 0 | 3 | 5 | 85 | 100 | 100 | 23,9 | 76,1 | 80,262 |
| 8.0 | 38,93 | 2122 | 0 | 3 | 5 | 100 | 100 | 100 | 35,6 | 64,4 | 80,38 |

Tabela : Dane statystyczne 1000 uruchomień algorytmu na grafie g40 - 100 iteracji

W przypadku grafu g40 w żadnej z konfiguracji skuteczność nie wyniosła 100%, zarówno dla τ równego 0.2 jak i 0.5 w jednym uruchomieniu na 1000 nie udało się znaleźć optymalnego rozwiązania. Kwartyle 1, 2 i 3 wskazują, że najlepszą wartością parametru będzie 1.0. Przy zastosowaniu takiej wartości zmiennej byłoby konieczne wykonanie najmniejszej liczby iteracji w celu znalezienia optimum. Potwierdza to także średnia. Jednak wariancja wskazuje najmniejszą rozbieżność wyników dla τ=0.5, skuteczność algorytmu także jest najlepsza dla tej wartości. Celem implementacji algorytmu jest uruchamianie go na grafie o nieznanym czasie szeregowania, dlatego ważne byłoby kilkukrotne uruchomienie programu w różnym zakresie parametru τ w przedziale [0.5, 1.0] z różną ilością iteracji. Dopiero zestawienie wyników otrzymanych w tym procesie pozwoliłoby na określenie najlepszej proponowanej konfiguracji.

Wykres 7 przedstawia typowy przebieg algorytmu na grafie g40. Wykres różni się od wykresów dla grafów intree15, tree15 i g18 większą różnicą między czasem szeregowania w kolejnych iteracjach. Na przykład w iteracji 42 czas szeregowania wyniósł 80, a w następnej 43 czas ten wynosił 84, w kolejnej znów 80. Oznacza to, że konfiguracje są bardzo czułe na zmianę przypisania procesorów. Pamiętając o tym, że w jednej iteracji zmieniany zostaje tylko jeden bit (procesor dla pojedynczego zadania), można stwierdzić, że taka ewolucja będzie skutkować częstym zagłębieniu się w minimum lokalnym. Można zauważyć, że przy iteracjach 13 i 55 wyjście z takiego minimum przez wybór gorszej konfiguracji pozwala w następnych iteracjach naleźć optymalny wynik.

Wykres : Przebieg algorytmu na grafie g40 τ=0.5 - 100 iteracji, najlepszy wynik osiągnięto w 21 iteracji

Godnym zauważenie jest też fakt, że algorytm nie wychodzi poza rozwiązania optymalne (minima lokalne) na więcej niż 2 iteracji. Kontrprzykładem jest Wykres 8 przedstawiający przebieg dla τ=0.1. Tutaj widoczne jest między iteracjami 44-49 i 86-93 odejście od minimum lokalnego i wykonywanie algorytmu „bez celu”. Tracony jest czas na wykonywanie bezcelowych obliczeń.

*W tej chwili przychodzi mi do głowy, że parametr prawdopodobieństwa można byłoby wzbogacić o funkcję kary. Polegałaby to na zwiększaniu szansy wyboru gorszej konfiguracji w przypadku, gdy kilka razy pod rząd wybrano najlepsze konfiguracja (lub o najlepszym czasie, bo np. konfiguracje 1-4 mogą mieć ten sam czas) i zwiększeniu szansy na wybór najlepszego, gdy wybrano poprzednio gorszy.*

Wykres : Przebieg algorytmu na grafie g40 τ=0.1 - 100 iteracji, najlepszy wynik osiągnięto w 18 iteracji

# Wyniki gauss18

Gauss18 jest najbardziej złożonym grafem spośród testowanych. Koszty zmiany procesora i czasy wykonania poszczególnych zadań są bardzo zróżnicowane, powodują, że bardziej optymalne jest wykonanie dwóch zadań na jednym procesorze niż przesłanie na drugi i wykonanie równoległe.

W tabeli Tabela 6 przedstawiono dane statystyczne uruchomienia algorytmu na grafie gauss18. Można zauważyć, że w przeciwieństwie do pozostałych grafów skuteczność algorytmu była w tym wypadku niska, w najlepszym wypadku przy τ=0.5 tylko w 46,4% uruchomień udało się odnaleźć najlepszy czas szeregowania. Dla tej samej wartości parametry także średni czas szeregowania był najlepszy i wyniósł 45,646 przy optymalnym czasie 44.

O złożoności grafu świadczy też fakt, że w żadnym wypadku (8000 uruchomień) nie udało się znaleźć w iteracji 0 (losowe uszeregowanie bazowe) najlepszego rozwiązania, optymalne rozwiązanie zostało znalezione najwcześniej po 2 iteracjach.

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **τ** | **średnia** | **wariancja** | **min** | **Q1** | **mediana** | **Q3** | **max** | **moda** | **moda (%)** | **skuteczność (%)** | **średni czas szeregowania** |
| 0.1 | 98,866 | 71,419 | 2 | 100 | 100 | 100 | 100 | 100 | 97,6 | 2,4 | 49,667 |
| 0.2 | 94,264 | 314,152 | 8 | 100 | 100 | 100 | 100 | 100 | 87,9 | 12,2 | 47,339 |
| 0.5 | 74,294 | 1127,88 | 5 | 41 | 100 | 100 | 100 | 100 | 53,9 | 46,4 | 45,646 |
| 0.8 | 72,011 | 1325,61 | 3 | 31 | 100 | 100 | 100 | 100 | 56,4 | 43,7 | 46,046 |
| 1.0 | 71,972 | 1413,95 | 3 | 28 | 100 | 100 | 100 | 100 | 60,3 | 39,7 | 46,409 |
| 1.5 | 75,241 | 1450,87 | 3 | 31 | 100 | 100 | 100 | 100 | 68,2 | 31,8 | 47,36 |
| 2.0 | 78,849 | 1295,43 | 2 | 58 | 100 | 100 | 100 | 100 | 71,9 | 28,1 | 47,904 |
| 5.0 | 86,383 | 1006,16 | 2 | 100 | 100 | 100 | 100 | 100 | 82,6 | 17,4 | 49,559 |
| 8.0 | 88,933 | 901,31 | 2 | 100 | 100 | 100 | 100 | 100 | 87,8 | 12,2 | 50,07 |

Tabela : Dane statystyczne 1000 uruchomień algorytmu na grafie gauss18 - 100 iteracji

Na wykresie Wykres 9 przedstawiono typowy przebieg algorytmu dla parametru τ=0,5. W porównaniu do pozostałych algorytmów wyraźnie zauważalna jest różnica pomiędzy linią pokazującą najlepszą konfigurację w iteracji oraz wybraną. Dzieje się tak ze względu na złożoność grafu, w którym często zmiana w szeregowaniu przypisania pojedynczego zadania do procesu skutkuje znaczną zmianą czasu szeregowania. W grafach tree15, intree15, g18 i g40 taka zmiana wielokrotnie nie wpływała na czas wykonania lub powodowała, że konfiguracja szeregowania miała czas różniący się o nieznaczną wielkość. Konfiguracje można było pogrupować i wybranie konfiguracji 3 lub 4 oznaczało wybranie jednej z wielu optymalnych konfiguracji. W przypadku tego grafu wybranie konfiguracji 3 i wyższej oznacza wybranie gorszego rozwiązania. Spowodowało to także, że poczynając od iteracji 28 do iteracji 47, gdy nastąpił częsty wybór słabszych konfiguracji znacznie pogorszył się najlepszy czas wykonania algorytmu w danych iteracjach. Podobna sytuacja rozpoczęła się przy 80 iteracji. Jednak tak jak dla poprzednich grafów znalezienie najlepszej konfiguracji wiąże się z wyjściem z minimum lokalnego i próbą zmiany tego minimum na globalne.

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **τ** | **średnia** | **wariancja** | **min** | **Q1** | **mediana** | **Q3** | **max** | **moda** | **moda (%)** | **skuteczność (%)** | **średni czas szeregowania** |
| 0.5 | 124,08 | 12447,1 | 3 | 30 | 92,5 | 189 | 400 | 400 | 5 | 95 | 44,128 |

Tabela : Dane statystyczne 400 uruchomień algorytmu na grafie gauss18 - 400 iteracji

Kolejny przeprowadzony eksperyment miał za zadanie sprawdzić działanie algorytmu na grafie gauss18 przy zwiększeniu liczby iteracji ze 100 do 400. Dzięki takiemu zabiegowi algorytm GEO pozwolił na znalezienie najlepszego czasu szeregowania w 95% uruchomień. Średnia liczba uruchomień potrzebna do znalezienia optimum to 124,08 a mediana 92,5. Średni czas szeregowania wyniósł 44,128. Jest to bardzo dobry wynik, jednak w efekcie przy takiej liczbie iteracji należało sprawdzić 7200 przypadków, co jest 2,75% wszystkich możliwych kombinacji dla tego grafu. W przypadku 100 iteracji konieczne jest sprawdzenie 1800 konfiguracji, co stanowi 0,69% kombinacji. Należy zwrócić uwagę, że w praktyce znaczna liczba kombinacji jest sprawdza wielokrotnie.

Wykres : Przebieg algorytmu na grafie gauss18 τ=0.5 - 100 iteracji, najlepszy wynik osiągnięto w 74 iteracji

## Porównanie wyników algorytmu GEO z algorytmami opartymi na automatach komórkowych.

W rozprawie doktorskiej dr Anny Piwońskiej analizowano działanie algorytmów komórkowych przy szeregowaniu zadań. W Tabela 7 dokonano porównaniu otrzymanych wyników średnich czasów szeregowania algorytmów komórkowych z wynikami otrzymanymi w poniższej pracy.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| graf programu | algorytm szeregowania oparty na automatach komórkowych | | standardowy algorytm genetyczny | algorytm GEO |
| tryb szeregowy | tryb równoległy |
| tree15 | 9 | 9 | 9 | 9 |
| intree15 | 9 | 9 | 9 | 9 |
| gauss18 | 44 | 46 | 44 | 45,646 |
| gauss18 (400 iteracji) | 44 | 46 | 44 | 44,128 |
| g18 | 46 | 46 | 46 | 46 |
| g40 | 80 | 80 | 80 | 80,001 |

Tabela : Średnie czasy szeregowania najlepszych reguł uzyskane dla grafów testowych w fazie normalnego działania

Wyniki są bardzo zbliżone. Odstępstwem są czasy szeregowania otrzymane dla grafu gauss18 przy 100 iteracjach. W przypadku algorytmu GEO są one lepsze od czasów otrzymanych dla algorytmu opartego na automatach komórkowych w trybie równoległym, jednak są zdecydowanie gorsze w porównaniu do czasów otrzymanych dla trybu szeregowego i standardowego algorytmu genetycznego. Zwiększenie liczby iteracji pozwala na osiągnięcie dużo lepszych rezultatów w przypadku grafu gauss18.