Równoległe algorytmy genetyczne - przegląd problematyki.

Mariusz Podsiadło, Politechnika Rzeszowska, Katedra Automatyki i Informatyki
ul. W. Pola 2, 35–959 Rzeszów, e–mail: podsiadl@prz.rzeszow.pl

1. Wstęp.

Jedną z najpowszechniej dostrzeganych cech algorytmów genetycznych (AG) jest ich naturalna równoległość. Mimo tego przez wiele lat większość prac nt. AG koncentrowało się na ich szeregowych realizacjach. Za jedną z pierwszych prób zrównoleglenia AG uważa się (Goldberg 1989) pracę Grefenstette z 1981r. Rozważał on cztery wersje równoległego algorytmu genetycznego. Trzy z nich operowały na jednej wspólnej populacji wykorzystując różne warianty architektury typu master-slave do zrównoleglenia obliczeń funkcji celu. Czwarta była algorytmem z rozproszoną populacją i autonomicznymi procesami wykonującymi algorytm genetyczny i komunikującymi się między sobą w celu wymiany najlepszych znalezionych osobników. W 1985 roku Grosso jako jeden z pierwszych (Cantú–Paz & Goldberg 1996) zaobserwował bardzo istotne zalety algorytmu równoległego. Stwierdził on mianowicie, że algorytm z kilkoma subpopulacjami jest szybszy niż algorytm z jedną dużą populacją, pod warunkiem, że nie są one od siebie całkowicie odizolowane. Ta obserwacja została potwierdzona wynikami prac wielu innych badaczy (Hoffmeister 1991, Whitley et. al. 1991, Gordon & Whitley 1993).

Ostatnio notuje się duże zainteresowanie i rosnącą liczbę publikacji dotyczących równoległych algorytmów genetycznych (RAG). Większość z nich opisuje przykłady konkretnych implementacji lub zastosowań RAG do złożonych problemów optymalizacji, w których ewaluacja funkcji jakości wymaga dużych nakładów obliczeniowych. Znacznie mniej istnieje opracowań systematyzujących, czy też dokonujących analizy wpływu równoległości algorytmu na jego zachowanie.

W niniejszym artykule dokonano krótkiego przeglądu równoległych algorytmów genetycznych i podjęto próbę ich klasyfikacji. Dokonano również analizy wpływu wybranych technik zrównoleglania AG na jakość i skuteczność algorytmu.

2. Ogólna klasyfikacja równoległych GA.

Próbę usystematyzowania równoległych AG podjął Hoffmeister (Hoffmeister 1991). Sklasyfikował on równoległe AG pod względem równoległości algorytmicznej w powiązaniu z archikekturą maszyn, na których są wykonywane. Wyróżnił trzy typy algorytmów:

I. Algorytmy scentralizowane - równolegle wykonywane są ewaluacje funkcji celu, podział zadań pomiędzy procesami odbywa się w oparciu o schemat master-slave. Taki model zrównoleglenia AG zwykle stosowany bywa w sytuacji gdy platformą sprzętową realizacji algorytmu jest lokalna sieć komputerowa (LAN).

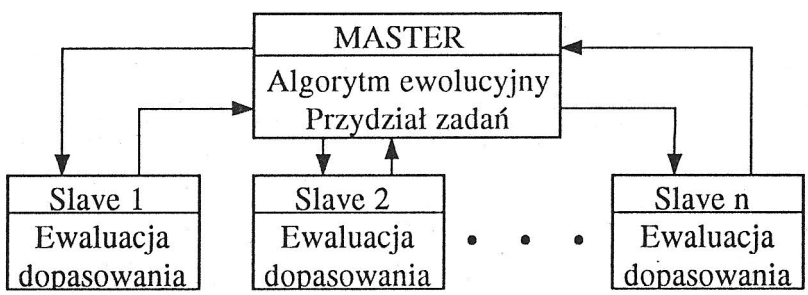
II. Algorytmy o gruboziarnistej równoległości, które dalej nazywane będą subpopulacyjnymi - równoległe procesy realizujące autonomiczne algorytmy ewolucyjne przetwarzają lokalne populacje (subpopulacje), które stanowią część populacji globalnej, wymieniając między sobą najlepsze dotychczas znalezione rozwiązania. Taki model najbardziej nadaje się do stosowania na komputerach charakteryzujących się małymi możliwościami komunikacyjnymi w stosunku do mocy obliczeniowej.

III. Algorytmy o drobnoziarnistej równoległości, które dalej nazywane będą komórkowymi - każdy osobnik przetwarzany jest przez oddzielny proces połączony z procesami sąsiednimi, operacje genetyczne wykonywane są we współpracy z procesami sąsiednimi. Ten model narzuca wymaganie bardzo dużych możliwości komunikacyjnych kompuetra.

Wszystkie powyższe typy algorytmów mogą być realizowane jako synchroniczne (Goldberg 1989, Hoffmeister 1991, Whitley et. al. 1991) - komunikacja między procesami odbywa się w ściśle określonych momentach - bądź asynchroniczne (Goldberg 1989, Zeigler & Kim 1993) - nie istnieje żaden mechanizm synchronizacji.

3. Model scentralizowany.

Jest to najłatwiejszy do realizacji model RAG, ponieważ sam algorytm ewolucyjny realizowany jest przez jeden proces (master) i nie różni się zbytnio od zwykłego szeregowego algorytmu. Ze względu na duże rozpowszechnienie LAN model ten może być szeroko stosowany. Bioroąc pod uwagę stosunek kosztów do mocy obliczeniowej różnych typów maszyn równoległych można stwierdzić, że jest to również bardzo ekonomiczny sposób zrównoleglenia AG (Taudes & Netousek 1991).



Rys. 1. Topologia połączeń komunikacyjnych w modelu scentralizowanym.

W modelu master-slave bez względu na fizyczną topologię sieci komputerów, funkcjonalnie realizowana jest topologia gwiazdy (rys. 1). Proces master posiada całą populację osobników, na których wykonuje wszystkie operacje genetyczne, zlecając obliczenie funkcji celu procesom slave. Aby osiągnąć duże przyspieszenie obliczeń, nakłady na ewaluacje muszą być duże w stosunku do nakładów na komunikację. Tak się dzieje np. w przypadku, gdy wartość dopasowania wyznaczana jest na drodze symulacji złożonych systemów (Grefenstette 1991, Grefenstette 1995, Harp & Samad 1991). W tym wypadku jednak czas wartościowania funkcji celu w poszczególnych procesach typu slave może podlegać znaczącym wahaniom. Dlatego znaczenia nabiera sposób rozdziału obciążenia przez proces master oraz wybór pomiędzy modelem synchronicznym, w którym mogą wystąpić duże czasy oczekiwania procesów slave na zadania, lub asynchronicznym, który zapewnia pełniejsze wykorzystanie dostępnych mocy obliczeniowych, jednak wymaga pewnych modyfikacji algorytmu procesu master.

Zaletą modelu scentralizowanego jest łatwość jego implementacji w sieciach heterogenicznych. Przykładem może być system złożony z maszyny symbolicznej realizującej proces master oraz stacji roboczych obliczających wartość funkcji celu (Montana 1991).

4. Model subpopulacyjny.

Mimo, iż Hoffmeister ściśle powiązał model scentralizowany z obliczeniami w sieci LAN, nie jest to jedyny model jaki może być w nich stosowany. Model subpopulacyjny, przez

Hoffmeistera kojarzony z gruboziarnistymi obliczeniami równoległymi, również nadaje się do tego celu. W modelu tym populacja podzielona jest na subpopulacje przydzielone do autonomicznych równoległych procesów. Każdy proces realizuje niezależny algorytm ewolucyjny, wymieniając co pewien czas informację z pozostałymi.

Istnieje dużo przykładów implementacji tego typu RAG (Mühlenbein et. al. 1991, Voigt & Born 1991, Whitley & Starweather 1991, Kröger et. al. 1992, Gordon & Whitley 1993). Różnią się one często od siebie w szczegółach dotyczących komunikacji i/lub równoległości danych. Ze względu na owe różnice można dokonać dalszego podziału algorytmów równoległych opartych na modelu subpopulacyjnym.

Równoległość danych w modelu subpopulacyjnym można rozpatrywać na dwóch poziomach. Pierwszy związany jest z równoległym wykonywaniem tego samego algorytmu na różnych subpopulacjach. Drugi poziom dotyczy odwzorowania dziedziny problemu na dane algorytmu. Dla tak rozumianej równoległości danych spotykane algorytmy subpopulacyjne możemy podzielić na:

- algorytmy bez podziału dziedziny,
- algorytmy z podziałem dziedziny.

W przypadku algorytmów bez podziału (Mühlenbein et. al. 1991) początkowe subpopulacje losowane są z równomiernym rozkładem w całej dziedzinie. Natomiast w algorytmach z podziałem dziedziny początkowe subpopulacje losowane są w pewnych podzbiorach dziedziny (Voigt & Born 1991, Shonkwiler 1993). Te podzbiory są nazywane lokalnym środowiskiem (Voigt & Born 1991).

Komunikacja między procesami ma na celu wymianę między subpopulacjami genotypów osobników i nazywana jest migracją. W skrajnym wypadku może ona w ogóle nie występować (Shonkwiler 1993). Shonkwiler na drodze analizy teoretycznej pokazuje, że algorytm bez migracji oparty na podziale dziedziny problemu osiąga znaczące (ponad liniowe) przyspieszenie. Nie jest to zgodne ze wspomnianymi wyżej wynikami obserwacji doświadczalnych. Może to wynikać z faktu, że Shonkwiler, ze względu na złożoność obliczeniową, rozpatrywał bardzo proste zagadnienia, które w wyniku podziału dziedziny (który można postrzegać jako uproszczenie problemu) stawały się trywialne. W wypadku praktycznych problemów, związanych z dużymi i wielowymiarowymi przestrzeniami poszukiwań, zaobserwowanie tego zjawiska wymagałoby użycia bardzo dużej liczby procesorów. Dlatego też model bez migracji należy uznać za mniej interesujący.

Z migracją wiążą się następujące zagadnienia, wg których można dokonać podziału algorytmów subpopulacyjnych:

a) model migracji,

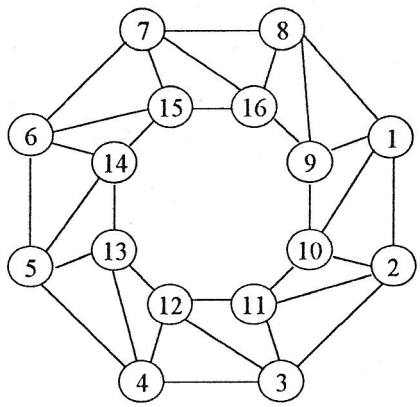
b) topologia sieci połączeń komunikacyjnych,

c) zasięg migracji.

Jeśli idzie o model migracji, to można wyróżnić dwa jego rodzaje: emigrację i imigrację (Kröger et. al. 1992). W przypadku emigracji osobnik migrujący jest usuwany z rdzennej subpopulacji, podczas gdy w przypadku imigracji pozostaje w niej, a do subpopulacji sąsiednich wysyłana jest jedynie jego kopia. (Kröger et. al. 1992) stwierdzają, że mimo iż większość znanych implementacji wykorzystuje model imigracyjny, to lepsze rezultaty można uzyskać stosując emigrację. W szczególności stwierdzili oni, że stosując emigrację uzyskuje się poprawę działania algorytmu wraz ze wzrostem liczby równoległych procesów (subpopulacji). Przyczyną tego zjawiska jest zachowanie większego zróżnicowania

globalnej populacji i uniknięcie przedwczesnej zbieżności. Inaczej dzieje się w przypadku algorytmów wykorzystujących model imigracji, w których subpopulacje mogą gromadzić najlepszych osobników z całej populacji, co prowadzi do jej przedwczesnego ujednolicenia.

Topologie procesów obliczeniowych i sieci połączeń komunikacyjnych stosowane w RAG opartych na modelu subpopulacyjnym nie odbiegają od typowych topologii używanych w obliczeniach równoległych(Kozielski & Szczerbiński 1993). Można tu wymienić pierścien, torus, kratę, czy hipersześcian (Voigt & Born 1991), oraz rzadziej spotykaną drabinę (Mühlenbein et. al. 1991) (rys. 2). Wybór odpowiedniej topologii najczęściej podyktowany jest fizyczną architekturą sprzętu, na którym implementowany jest algorytm.

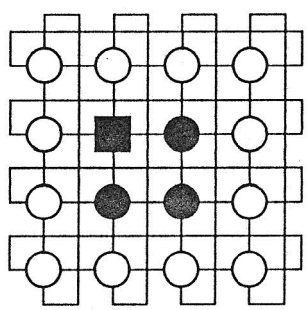


Rys. 2. Topologia typu drabina.

Zasięg migracji określa odległość (mierzoną w sposób odpowiedni dla danej topologii sieci) na jaką mogą migrować osobnicy subpopulacji. Ze względu na zasięg można wyróżnić dwa modele migracji (Kröger et. al. 1992):

- osobnicy mogą migrować do dowolnej subpopulacji – jest to tzw. *island model*, taki model nazywany będzie globalnym,
- osobnicy mogą migrować do określonych subpopulacji sąsiednich – jest to tzw. *stepping stone model*, taki model nazywany będzie lokalnym,

Z lokalną migracją wiąże się pojęcie sąsiedztwa subpopulacji (Mühlenbein et. al. 1991, Voigt & Born 1991), które definiuje dla niej zbiór subpopulacji sąsiednich. Osobnik podlegający migracji przesyłany jest do wszyskich subpopulacji sąsiednich.



Rys. 3 Przykład sąsiedztwa w sieci o topologii typu torus.

Na rysunku 3 pokazano przykład sąsiedztwa zdefiniowanego w sieci o topologii typu torus zastosowany w (Voigt & Born 1991). Zaczernione kółka oznaczją subpopulacje sąsiednie dla subpopulacji oznaczonej zaczernionym kwadratem.

5. Model komórkowy.

Hoffmeister przytacza dwa przykłady implementacji równoległego AG na maszynach o masowej równoległości: ASPARAGOSS przedstawiony przez Gorges-Schleuter (1989) i FG przedstawiony przez Spissensa i Mandericka (1989). W obu przypadkach zamiast wykonywania równolegle kilku algorytmów szeregowych, dokonano zrównoleglenia operacji genetycznych. Takie podejście jakościowo różni się znacznie od poprzednich.

W modelu komórkowym każdy osobnik populacji przetwarzany jest przez oddzielny proces połączony z procesami sąsiednimi. Każdy proces dokonuje ewaluacji funkcji celu, selekcji i rekombinacji. Tylko ewaluacja może być realizowana niezależnie od obliczeń wykonywanych przez pozostałe procesy, pozostałe operacje wymagają wymiany danych między nimi. Powoduje to znaczny wzrost nakładów na komunikację w porównaniu z modelami przedstawionymi wcześniej. Dlatego też zwykle rekombinacja i selekcja ograniczone są do pewnego otoczenia osobnika. Powstają w ten sposób pokrywające się częściowo subpopulacje (nie należy ich utożsamiać z subpopulacjami w modelu subpopulacyjnym).

Ponieważ subpopulacje składają się z oddzielnych procesów, zatem na dynamikę i zbieżność algorytmu zasadniczy wpływ ma topologia sieci. Często jest ona zdeterminowana rzeczywistą architekturą sprzętu, na którym algorytm jest implementowany. Spotykane są następujące topologie: krata, torus, hiperześcian, drabina.

6. Inne techniki kojarzone z równoległymi AG.

W niniejszym rozdziale zostaną zasygnalizowane techniki, które nie są wyłącznie związane z RAG (tzn. ze specyfiki algorytmu równoległego nie wynika konieczność ich stosowania i/lub mogą być wykorzystywane również w algorytmach szeregowych) które jednak bardzo często są z nimi kojarzone. Pierwsza z nich, to hybrydyzacja (Mühlenbein et. al. 1991, Voigt & Born 1991). Polega ona na połączeniu algorytmu ewolucyjnego z metodami optymalizacji lokalnej - np. metody wspinaczki lokanej. W ściśle określonej sytuacji uruchamiana jest metoda optymalizacji lokalnej dla wybranego osobnika.

Inną ważną techniką jest adaptacja parametrów AG. W pracy (Lis & Lis 1996) opisano algorytm subpopulacyjny, w kórym każda subpopulacja przetwarzana jest przez sekwencyjny AG z różnymi wartościami parametrów sterujących. Wyróżniony proces dokonuje analizy potępów poszczególnych procesów i podejmuje decyzje o zmianie ich parametrów. Np. jeśli najlepszy do danej chwili resultat (w sensie wartości funkcji celu) uzyskał proces o największym prawdopodobieństwie mutacji, to prawdopodobieństwo to jest zwiększane we wszystkich procesach.

7. Właściwości subpopulacyjnego RAG.

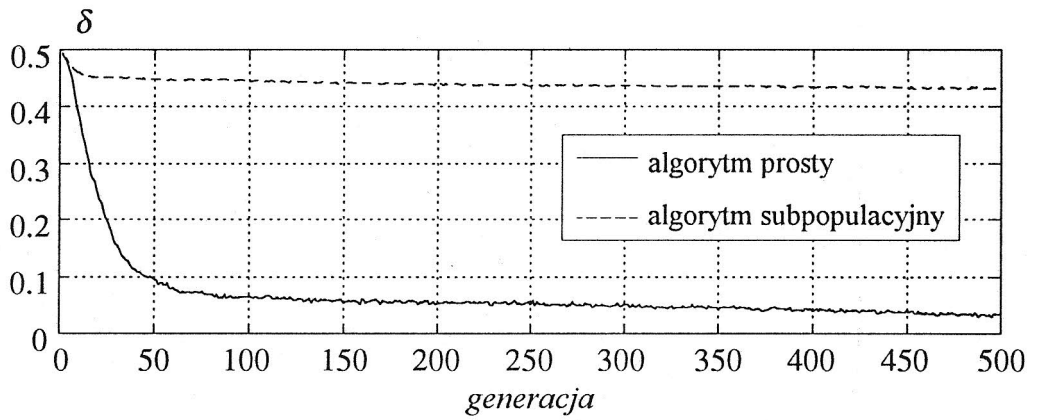
Wielu autorów wskazuje na fakt, że algorytm subpopulacyjny daje lepszej jakości wyniki niż algorytm szeregowy z jedną dużą populacją. Przyczyną tego zjawiska jest utrzynamie większego zróżnicowania globalnej populacji w algorytmie subpopulacyjnym. Aby to zilustrować, poniżej przedstawiono porównanie prostego algorytmu genetycznego SGA (Goldberg 1989) i przykładowego synchronicznego algorytmu subpopulacyjnego z lokalną emigracją w sieci o topologii typu pierścień. W obu algorytmach zastosowano selekcję turniejową, proste krzyżowanie jednopunktowe i mutację, oraz identyczne prawdopodobieństwa krzyżowania i mutacji. Populacja algorytmu SGA liczyła dwustu osobników, natomiast w algorytmie subpopulacyjnym składała się z dziesięciu subpopulacji, z których każda liczyła dwudziestu osobników. Emigracja odbywała się po każdych dziesięciu

generacjach. Ze względu na stochastyczny charakter AG, aby uzyskać miarodajne porównanie, oba algorytmy zrealizowano pięćdziesiąt razy i uśredniono wyniki.

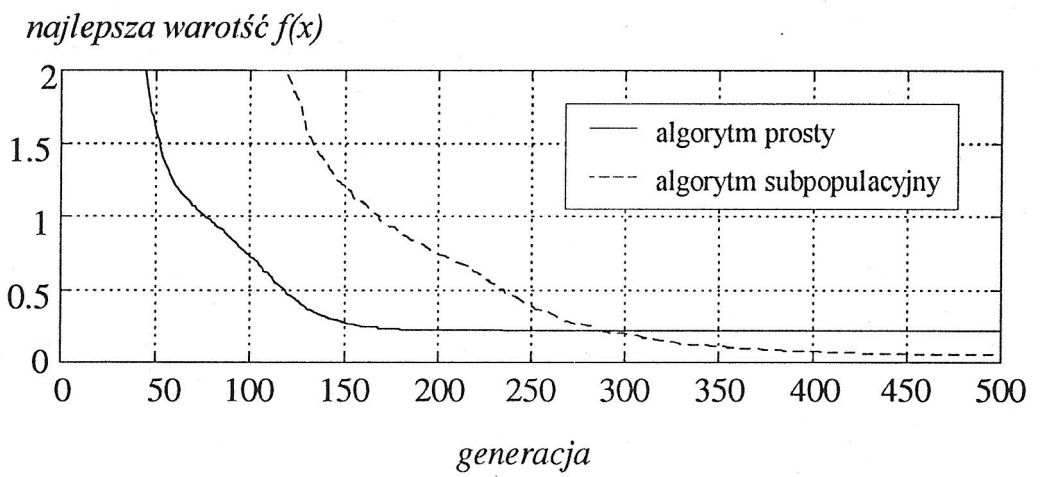
Jako zadanie testowe zastsowano funkcję Griewangka:

$$f(x) = 1 + \sum_{i=1}^{10} x_i^2 / 4000 - \prod_{i=1}^{10} \cos(x_i / \sqrt{i});\ \text{gdzie}\ -600 \le x_i \le 600$$

Nadaje się ona szczególnie do tego testu, ponieważ w otoczeniu minimum globalnego posiada wiele "płytkich" minimów lokalnych. Stanowi to szczególną trudność dla algorytmów genetycznych. Znalezienie minimum globalnego wymaga zachowania dużej różnorodności materiału genetycznego w populacji.



Rys. 4. Porównanie zbieżności popualcji w algorytmie szeregowym i subpopulacyjnym.



Rys. 5. Porównanie postępów algorytmu szeregowego i subpopulacyjnego.

Na rysunku 4 przedstawiono porównanie zbieżności populacji agorytmu szeregowego i subpopulacyjnego. Wykres przedstawia zależność zróżnicowania globalnej populacji od iteracji algorytmu (generacji). Zróżnicowanie δ zdefiniowano jako średnią odległość Hamminga $2n$ losowo wybranych par osobników globalnej populacji, gdzie n oznacza liczbę osobników. Tak zdefiniowane zróżnicowanie wystarczająco dobrze przybliża średnią odległość Hamminga osobników populacji lecz jest znaczniej mniej kosztowne obliczeniowo (policzenie średniej dla całej populacji wymaga policzenia $n(n\text{-}1)/2$ odległości Hamminga). Linia ciągła odpowiada algorytmowi prostemu, a przerywana subpopulacyjnemu.

Jak widać, populacja algorytmu prostego szybko ujednolica się - jej zróżnicowanie spada niemal do zera, pozostając na stałym niskim poziomie dzięki istnieniu mutacji. W tej sytuacji algorytm genetyczny sprowadza się do losowego błądzenia i jego efektywność drastycznie maleje. Pokazuje to rysunek 5, na którym przedstawiono zależność najlepszego znalezionego rozwiązania od iteracji algorytmu (generacji). Wykres przeskalowano tak, aby uwidocznić zachowanie algorytmu w końcowej fazie jego działania. Po wyrównaniu się populacji algorytm nie czyni już znaczących postępów.

W przypadku algorytmu subpopulacyjnego zróżnicowanie populacji pozostaje na wysokim poziomie. Algorytm początkowo wolniej się zbiega, jednak osiąga znacznie lepsze rezultaty. Na pięćdziesiąt prób algorytm dwanaście razy znalazł minimum globalne, a średnia najlepszych znalezionych rozwiązań ze wszystkich prób wyniosła 0.0462, podczas gdy algorytm prosty tylko pięć razy znalazł minimum globalne, a średnia wyniosła 0.2171. Podobne rezultaty otrzymali (Gordon & Whitley 1993).

8. Kierunki badań.

W chwili obecnej badania nad równoległymi AG koncentrują się na modelu subpopulacyjnym i komórkowym. Badania subpopulacyjnego RAG mają na celu lepsze zrozumienie i wykorzystanie szczególnych jego właściwości opisanych wyżej. W algorytmach komórkowych decentralizacja operacji genetycznych powoduje osłabienie presji selekcjyjnej i tym samym pogorszenie właściwości AG oraz wymaga dużych nakładów na komunikację. Aktualnie prowadzone prace mają zatem na celu znalezienie metod realizacji rozproszonych operatorów genetycznych nie posiadających tych wad.

8.1. Algorytm subpopulacyjny.

W pracy (Cantú–Paz & Goldberg 1996a) autorzy dokonali uogólnienia modelu opisującego jakość zbieżności prostego AG – tzw. *gambler's ruin model* (Harik et. al. 1996) – na subpopulacyjny RAG. Model ten aproksymuje zależność pomiędzy prawdopodobieństwem, że algorytm znajdzie rozwiązanie z zadaną dokładnością, a rozmiarem populacji. Autorzy uwzględnili dwa przypadki graniczne subpopulacyjnego RAG. Pierwszy, w którym populacje są od siebie całkowicie odizolowane, oraz drugi, w którym subpopulacje są ze sobą kompletnie połączone. Drugi algorytm wymaga dokładniejszego opisu. Migracja zachodzi w sytuacji, gdy stwierdza się całkowitą zbieżność wszystkich subpopulacji. Każda subpopulacja wymienia n/r osobników ze wszystkimi pozostałymi subpopulacjami, gdzie n jest rozmiarem subpopulacji a r jest liczbą subpopulacji. Dla obu algorytmów wyznaczyli rozmiary subpopulacji, które zapewniają określone prawdopodobieństwo znalezienia rozwiązania. Z otrzymanych rezultatów wynika, że rozmiary subpopulacji są znacznie mniejsze dla algorytmu z migracją, niż dla algorytmu z izolowanymi subpopulacjami.

Kontynuacją opisanych rozważań jest praca (Cantú–Paz & Goldberg 1996b), w której zastosowano wcześniejsze wyniki do predykcji przyspieszeń subpopulacyjnych RAG. Aby uzyskać miarodajne oszacowanie przyspieszenia, autorzy zdefiniowali je jako stosunek czsów wykonania algorytmów o takiej samej oczekiwanej szybkości zbieżności i jakości rozwiązania. Podobnie jak poprzednio rozważano dwa graniczne przypadki migracji. Otrzymane wyniki teoretyczne autorzy zweryfikowali porównując je z rezultatami doświadczalnymi. Z analizy tych wyników można wyciągnąć następujące wnioski. Algorytm bez migracji daje niewiekie przyspieszenia, jednak rosną one monotonicznie wraz ze wzrostem liczby subpopulacji. Algorytm z migracją daje dużo lepsze przyspieszenia. Istneje pewna optymalna liczba subpopulacji, dla której przyspieszenie jest największe.

8.2. Algorytm komórkowy.

W pracy (De Jong & Sarma 1995) przeanalizowano, na podstawie wyników doświadczeń symulacyjnych, przydatność rozporszonych wersji kilku popularnych metod selekcji: proporcjonalnej (ruletkowej), turniejowej i rankingowej, w dwuwymiarowej toroidalnej sieci procesorów. Wynika z niej, że najbardziej odpowiednią metodą selekcji dla algorytmu komórkowego jest lokalna elitarystyczna selekcja turniejowa. Daje ona podobną globalną presję selekcyjną jak selekcja rankingowa jednak wymaga przy tym mniejszych nakładów na komunikację.

Analizę wpływu kształtu i rozmiaru sąsiedztwa osobnika na dynamikę lokalnej selekcji przeprowadzono w pracy (Sarma & De Jong 1996). Z rozważań autorów wynika, że presja selekcyjna zależy przede wszystkim od promienia sąsiedztwa zdefiniowanego następująco:

$$r = \sqrt{\frac{\sum_{i=1}^{n}(x_i - \bar{x})^2 + \sum_{i=1}^{n}(y_i - \bar{y})^2}{n}}\ ;\ \bar{x} = \frac{\sum_{i=1}^{n} x_i}{n},\ \bar{y} = \frac{\sum_{i=1}^{n} x_i}{n},$$

gdzie n jest liczbą osobników należących do sąsiedztwa, a x_i i y_i są indeksami osobników należących do sąsiedztwa.

9. Podsumowanie.

W artykule dokonano przeglądu i klasyfikacji równoległych algorytmów genetycznych. Na podstawie wyników doświadczeń symulacyjnych przeanalizowano właściwości subpopulacyjnego RAG. Z rozważań tych wynika, że algorytm subpopulacyjny daje lepsze rezultaty niż porównywalny algorytm z globalną populacją. Algorytm subpopulacyjny może być w łatwy sposób zaprogramowany na maszynie szeregowej i używany jako "lepszy" algorytm szeregowy.

Podziękowania.

Niniejszy artukuł powstał w ramach projektu badawczego nr 8 T11A 025 12 pt. "Zastosowanie algorytmów genetycznych do optymalizacji w czasie rzeczywistym" finansowanego przez Komitet Badań Naukowych.

Autor dziękuje Panu Profesorowi Marianowi Wysockiemu za cenne uwagi, które wpłynęły na ostateczny kształt artykułu.

Literatura.

1. Andre, D., Koza, J., *A Parallel Implementation of Genetic Programming that Achives Super–Linear Performance*, in *Proc. of the International Conference on Parallel and Distributed Processing Techniques and Applications*, Sunnyvale, 1996.
2. Cantú–Paz, E., Goldberg, D. E., *Predicting Speedups of Ideal Bounding Cases of Parallel Genetic Algorithms*, (IlliGAL Report No. 96008) . Urbana, IL: University of Illinois at Urbana–Champaign, 1996a.
3. Cantú–Paz, E., Goldberg, D. E., *Modeling Idealized Bounding Cases of Parallel Genetic Algorithms*, (IlliGAL Report No. 96007) . Urbana, IL: University of Illinois at Urbana–Champaign, 1996b.
4. De Jong, K., Sarma, J., *On Decentralizing Selection Algorithms*, in Eshelman, L., ed., *Genetic Algorithms: Proc. of the 6th International Conference (ICGA'95)*, Morgan Kaufmann, San Francisco, 1995.
5. Goldberg, D. E., *Genetic algorithms in optimization, search and machine learning.* Addison–Wesley Publishing Company, Inc., 1989 (tłum. polskie WNT, W–wa 1995).

6. Gordon, V. S., Whitley, D., *Serial and Parallel Genetic Algorithms as Function Optimizers*, *Proc. of the 5th Intern. Conf. on Genetic Algorithms*, Morgan Kaufmann Publ., San Mateo, California, 1993.
7. Grefenstette, J. J., *Strategy Acquisition with Genetic Algorithms*, in Davis, L. (ed.) *Handbook of Genetic Algorithms*, Van Nostrand Reinhold, New York, 1991.
8. Grefenstette, J. J., *Robot Learning with Parallel Genetic Algorithms on Networked Computers*, in *Proc. 1995 Summer Computer Simulation Conference (SCSC'95)*, Society of Computer Simulation, Ottawa, 1995.
9. Harik, G., Cantú–Paz, E., Goldberg, D., Miller, B., *The gambler's ruin problem, genetic algorithms, and the sizing of populations*, (IlliGAL Report No. 96004). Urbana, IL: University of Illinois at Urbana–Champaign, 1996.
10. Hoffmeister F., *Scalable Parallelism by Evolutionary Algorithms*, in Grauer M., Pressmar D. B., *Parallel Computing and Mathematical Optimization. Lecture Notes in Economics and Mathematical Systems*. vol. 367, Springer Verlag, 1991.
11. Kozielski S., Szczerbiński Z., *Komputery równoległe; architektura, elementy programowania*. WNT, Warszawa, 1993.
12. Kröger, B., Schwenderling, P., Vornberger, O., *Massive Parallel Genetic Packing*, in Reijns G. L., Luo, J., eds., *Transputing in Numerical and Neural Network Applications*, IOS Press, 1992.
13. Lis, J., Lis, M., *Self adapting Parallel Gentic Algorithm with the Dynamic Mutation Probability, Crossover Rate and Population Size*, *Materiały I Krajowej Konferencji Algorytmy Ewolucyjne*, Oficyna wydawnicza Politechniki Wrszawskiej, Warszawa, 1996.
14. Montana, D. J., *Automated Parameter Tuning for Interpretation of Synthetic Images*, in Davis, L. (ed.) *Handbook of Genetic Algorithms*, Van Nostrand Reinhold, New York, 1991.
15. Mühlenbein H., Schomisch M., Born J., *The Parallel Genetic Algorithm as Function Optimizer*, Parallel Computing 17, North–Holland Elsevier, 1991.
16. Sarma, J., De Jong, K., *An Analysis of the Effects of Neighborhood Size and Shape on Local Selection Algorithms*, in *Proc. of the 4th International Conference on Parallel Problem Solving from Nature (PPSN '96)*, Berlin, 1996.
17. Schonkwiler R., *Parallel Genetic Algorithms*, in *Proc. of the 5th Intern. Conf. on Genetic Algorithms*, Morgan Kaufmann Publ., San Mateo, California, 1993.
18. Taudes, A., Netousek, Th., *Implementing Branch–and–Bound Algorithms on a Cluster of Workstations – A survey, some new Results and open Problems*, in Grauer M., Pressmar D. B., *Parallel Computing and Mathematical Optimization. Lecture Notes in Economics and Mathematical Systems*. vol. 367, Springer Verlag, 1991.
19. Voigt H.–M., Born J., *A Structured Distributed Genetic Algorithm for Function Optimization*, in Grauer M., Pressmar D. B., *Parallel Computing and Mathematical Optimization. Lecture Notes in Economics and Mathematical Systems*. vol. 367, Springer Verlag, 1991.
20. Whitley, D., Starkweather, T., Shaner, D., *The Treveling Salesman and Sequence Scheduling: Quality Solutions Using Genetic Edge Recombination*, in Davis, L. (ed.) *Handbook of Genetic Algorithms*, Van Nostrand Reinhold, New York, 1991.
21. Zeigler B. P., Kim J., *Asynchronous Genetic Algorithms on Parallel Computers*, in *Proc. of the 5th Intern. Conf. on Genetic Algorithms*, Morgan Kaufmann Publ., San Mateo, California, 1993.