55 lat WNT

zastosowanie narzędzi sztucznej inteligencji do rozwiązywania zadań inżynierskich i ekonomicznych

Henryk Piech
Aleksandra Ptak

Wydawnictwa Naukowo-Techniczne

Henryk Piech
Aleksandra Ptak

Zastosowanie narzędzi sztucznej inteligencji do rozwiązywania zadań inżynierskich i ekonomicznych

Wydawnictwa Naukowo-Techniczne
Warszawa 2004

Recenzent:
Prof. dr hab. inż. Zbigniew Czech

Copyright © 2004 Polskie Towarzystwo Informatyczne

ISBN 83-204-3015-1

Redakcja techniczna:
mgr inż. Tomasz Lis, mgr inż. Jarosław Łapeta
Projekt okładki Marek J. Piwko

Utwór w całości ani we fragmentach nie może być powielany ani rozpowszechniany za pomocą urządzeń elektronicznych, mechanicznych, kopiujących, nagrywających i innych, w tym również nie może być umieszczany ani rozpowszechniany w postaci cyfrowej zarówno w Internecie, jak i w sieciach lokalnych bez pisemnej zgody posiadacza praw autorskich.

Wydawnictwa Naukowo-Techniczne
00-048 Warszawa, ul. Mazowiecka 2/4
tel. (022) 826 72 71, e-mail: wnt@pol.pl
www.wnt.com.pl

Wydrukowano w Zakładzie Graficznym Politechniki Śląskiej w Gliwicach

Zastosowanie narzędzi sztucznej inteligencji do rozwiązywania zadań inżynierskich i ekonomicznych

Spis treści

Wstęp ... 4
1. Genetyczna koncepcja zagospodarowania zasobów rozproszonych ... 4
1.1. Opis zasobów procesowych i konwersyjnych środowiska rozproszonego ... 5
1.2. Koncepcja funkcji przystosowania ... 7
1.3. Koncepcja kodowania struktury chromosomu ... 9
1.4. Procedury genetyczne i ocena przystosowania ... 11
2. Wykorzystanie algorytmów genetycznych do prognozowania gospodarczego .. 12
2.1. Genetyczne formowanie funkcji prognostycznej ... 13
2.2. Genetyczne algorytmy tworzenia prognozy ... 17
3. Wykorzystanie narzędzi genetycznych do sterowania środowiskiem rozproszonym ... 22
3.1. Koncepcja równomiernego obciążenia węzłów sieci ... 22
3.2. Koncepcja genetycznego opisu transferu w sieci ... 26
3.3. Operacje genetyczne wspomagające transfer w sieci ... 27
3.4. Algorytmy genetyczne dystrybucji procesów w sieci ... 29
4.Genetyczna interpretacja zagadnienia najliczniejszych skojarzeń ... 32
4.1. Projekt struktury chromosomu ... 32
4.2. Parametry i algorytm generacji drogi naprzemiennej ... 34
4.3. Algorytmy operacji genetycznych ... 35
4.4. Określenie wartości funkcji przystosowania ... 42
5. Zastosowanie struktury neuronowej do wspomagania operacji genetycznych ... 43
5.1 Składniki środowiska genetycznego ... 43
5.2 Dostosowanie charakteru operacji genetycznych do specyfiki funkcjonowania sieci neuronowej ... 44
6. Wielo-asortymentowe zadanie transportowe w ujęciu genetycznym ... 52
6.1. Założenia wielo-asortymentowego zadania transportowego ... 52
6.2. Struktura danych i dynamika ich zmian ... 53
6.3. Genetyczny zapis dostaw ... 56
6.4. Tworzenie populacji bazowej ... 58
6.5. Algorytm operacji genetycznych realizujący zadanie transportowe ... 59
7. Koncepcja zagnieżdżonych struktur neuronowych ... 61
7.1 Opis zagnieżdżonej struktury neuronowej ... 62
7.2 Warianty zagnieżdżenia ... 65
7.3 Teoretyczna koncepcja sieci zagnieżdżonej ... 69
8. Koncepcja neuronowej kryptografii ... 72

8.1 Opis funkcjonowania kryptograficznych narzędzi i realizowanych przez nie procedur ... 75
8.2 Ocena jakości zabezpieczenia kryptograficznego ... 77
8.3.Złożoność (czasowa) algorytmu neuronowej szyfracji ... 79
9. Neuronowe prognozowanie scenariuszy wspomagające zarządzanie ... 81
9.1 Strategia tworzenia prognozy neuronowej ... 81
9.2 Realizacja neuronowych klasyfikatorów obrazów i scenariuszy ... 86
9.3 Uzyskane rezultaty, ich postać i proces formowania ... 89
10. Analiza efektywności systemów zarządzania przy wykorzystaniu sieci neuronowych ... 91
10.1. Wybór układu sieci neuronowej dostosowanego do struktury systemu zarządzania ... 91
10.2.Algorytm uczenia sieci neuronowej analizującej efektywność systemu zarządzania ... 96
11. Zrównoleglenie procesów w sieciach neuronowych ... 102
11.1 Możliwości podziału procesorów i towarzyszące temu wątki zrównoleglenia ... 102
11.2. Strategia minimalizacji sekwencyjności przy dystrybucji neuronów na jednostki przetwarzające ... 107
Literatura ... 114

Wstęp

Opracowanie wybranych koncepcji zastosowań narzędzi sztucznej inteligencji do zadań optymalizacji, dystrybucji, transportu, szyfrowania etc. i przedłożenie uzyskanych rezultatów studentom, adeptom nauki i pragmatykom, to cel jaki przyświeca autorom monografii. Mamy nadzieję, że zaprezentowane rozwiązania zainspirują czytelników do śmiałych poszukiwań w różnych dziedzinach życia zjawisk, które będąc nieopisanymi lub opisanymi w sposób nie dość prosty mogłyby zaistnieć w modelach genetycznych lub neuronowych. Badanie efektywności wykorzystania narzędzi sztucznej inteligencji to zadanie, którego rozwiązanie da wykładnie ich przydatności. Ponadto stanowić może krok zarówno w dziedzinie penetracji różnych zadań i zjawisk jak i w poszukiwaniu konkretnych mechanizmów odnoszących się do szczegółów realizacji operacji genetycznych i neuronowych rozwiązań. Mamy nadzieję, że wybrana tematyka zaprezentowana jest przystępnym językiem i w sposób klarowny i zrozumiały, nawet dla osób nie wtajemniczonych w detale budowy i algorytmicznej funkcjonalności narzędzi sztucznej inteligencji, ujawnia główne koncepcje i idee ich powstania. Książka nie jest przewodnikiem ani opracowaniem wprowadzającym do zagadnień sztucznej inteligencji, mamy natomiast tutaj do czynienia z próbą zainspirowana wyobraźni czytelników, którzy posiedli podstawową wiedzę o algorytmach genetycznych i sieciach neuronowych. Zabawa w pomysły prowadzi do pogłębiania wiedzy, a także praktycznego doświadczenia w zakresie tworzenia algorytmów i implementacji programowych. Jest to jednocześnie odwołanie się do praktycznie nieograniczonych narzędzi w różnych sferach działalności ludzkiej. Można na przykład tworzyć neuronowy model intuicji, przewidywania i strategii optymalnego zachowania wraz z elementami samodoskonalenia. Praktycznie rzecz biorąc nie znajdziemy dziedziny gdzie nie można by sięgnąć do narzędzi sztucznej inteligencji. Życzymy więc czytelnikom przyjemnej zabawy i zachęcamy do zasilenia grona twórców i kreatorów rozwiązań wykorzystujących algorytmy genetyczne i sieci neuronowe.

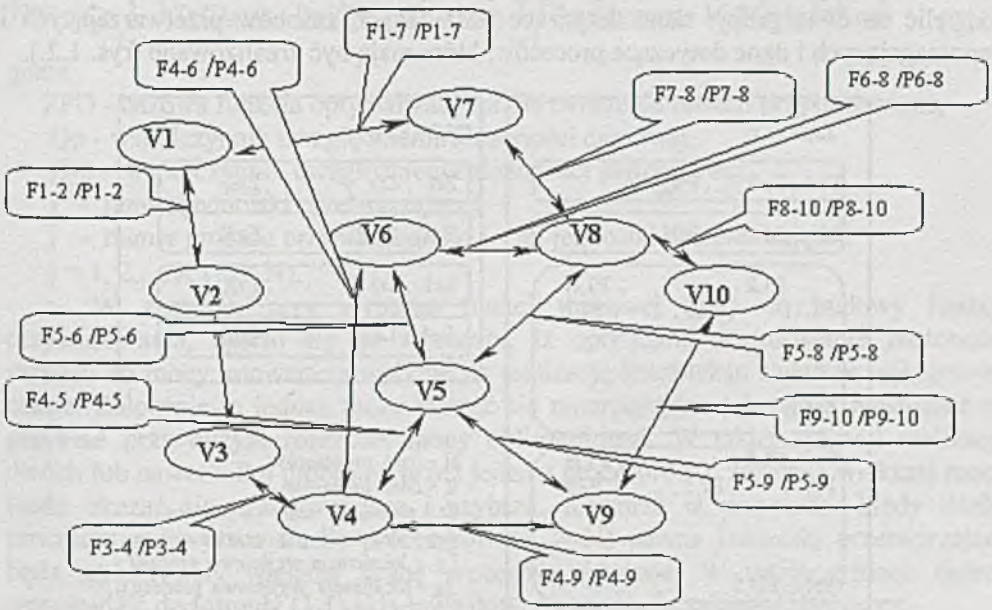
1.Genetyczna koncepcja wykorzystania rozproszonych zasobów

W rozdziale podjęto próbę wykorzystania genetycznych algorytmów do usprawnienia wykorzystania zasobów konwersyjno – pamięciowych sieciowego środowiska rozproszonego do zarządzania dystrybucją procesów. Algorytmizacja dystrybucji ma na celu zbliżenie się do rozwiązania optymalnego, zaś wykorzystanie takiego algorytmu będziemy traktowali jako wspomaganie decyzji o miejscach realizacji procesów w danej konfiguracji sieciowej. W części opisowej sieć komputerową będziemy traktowali jako graf skierowany G = (V,E) o nieujemnych

przepustowościach krawędzi c(u,v), gdzie (u,v) to dowolna krawędź grafu G. Ponadto graf ten, który mógłby być traktowany jako sieć przepływowa z wieloma wejściami i wyjściami [10] posiada pewne specyficzne własności. Problematyka poruszona w pracy dotyczy zdefiniowania funkcji przystosowania oraz organizacji procedur genetycznych, które będą dostosowane do specyfiki środowiska rozproszonego. Celem działań dystrybucyjnych jest także minimalizacja kolizji tj. sytuacji jednoczesnego dostępu do łączy i elementów przetwarzających.

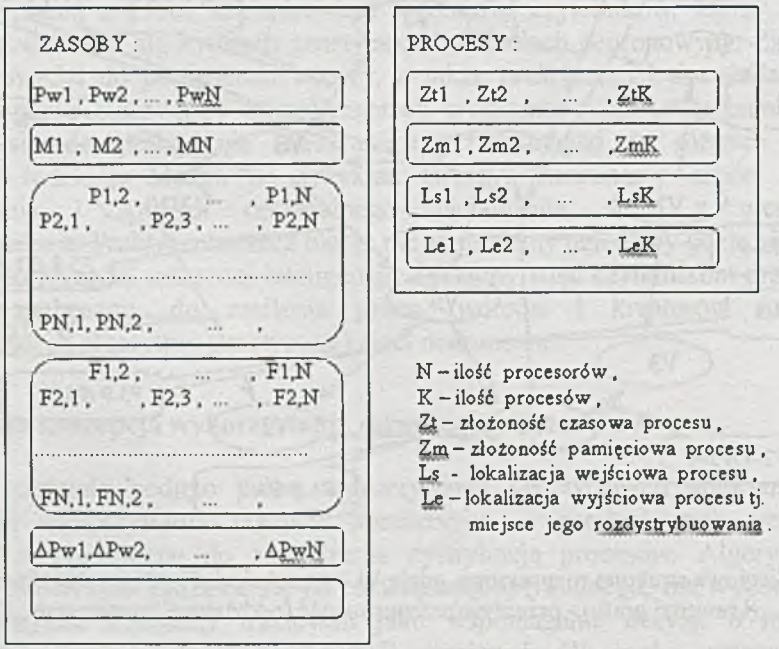
1.1 Opis zasobów procesowych i konwersyjnych środowiska rozproszonego

Środowisko rozproszone można opisać jako sieć przepływową [12] o wielu wejściach i wyjściach G(V,E), w której każda krawędź ma nieujemną przepustowość c(u,v)>0, którą możemy uznać za wielkość stałą i dwukierunkową (jednak wykluczającą jednoczesny dwukierunkowy, a nawet jednokierunkowy tranzyt). Na rys. 1.1 jest przedstawiona przykładowa struktura rozproszona.



Rys.1.1 Sieciowa struktura rozproszona, gdzie Vi – węzeł sieci, Fi-j/Pi-j - charakterystyka łączy (krawędzi grafu) - przepływ/przepustowość (oddzielone separatorem „/")

Każdy węzeł sieci charakteryzuje się następującymi parametrami: moc obliczeniowa (Pw), zasoby dostępnej, lokalnej pamięci (M) oraz stopień zajętości jednostki obliczeniowej (Sz). Stopień zajętości można, z pewnym przybliżeniem, przedstawić zarówno jako zmniejszenie mocy obliczeniowej oraz zmniejszenie zasobów pamięciowych Sz = ΔPw & ΔM, gdzie & oznacza łącznik składowych złożonego parametru. Elementami wejściowymi w strukturze sieciowej są procesy o określonej złożoności (Z), która składa się z komponentów czasowych i pamięciowych. Decyzje dystrybucyjne polegają na skierowaniu procesów do jednostek przetwarzających. Decyzje, w uproszczonej interpretacji, mogą mieć charakter strategiczny tj. uwzględniający optymalną dystrybucję w pewnym przedziale czasowym lub bieżący uwzględniający aktualne potrzeby w zakresie przetwarzania. Również pojęcie optymalności dystrybucji może być zdefiniowane na różne sposoby, można na przykład dążyć do minimalizacji czasu realizacji wszystkich zadań lub minimalizacji zajętości jednostek przetwarzających, minimalizacji zajętości zasobów pamięciowych albo też minimalizacji przesyłów komunikacyjnych. W strukturach rozproszonych o charakterze rozległym należy uwzględnić przepustowości (P) oraz wielkości przesyłanej informacji (przepływy - F). Dane środowiskowe można więc podzielić na dwie grupy: dane dotyczące posiadanych zasobów przetwarzających i wspomagających i dane dotyczące procesów, które mają być zrealizowane (rys. 1.2.).



Rys.1.2. Koncepcja struktury parametrów środowiska rozproszonego

Celem dalszych rozważań będzie tworzenie modeli optymalizujących procedury dystrybucji procesów w środowisku rozproszonym.

1.2 Koncepcje funkcji przystosowania

Funkcję przystosowania należy odnieść do funkcji celu oraz dostosować ją do postaci parametrów dystrybucji. Wybór funkcji celu zależy od wielu towarzyszących okoliczności związanych między innymi z

- rozrzutem złożoności czasowej i pamięciowej zadań,
- ilością przesyłanych danych,
- rozproszeniem mocy obliczeniowej poszczególnych węzłów sieci,
- zróżnicowaniem przepustowości poszczególnych łączy.

Jedną z koncepcji wykorzystywanych przy kreowaniu funkcji przystosowania jest dystrybucja zadań stosownie do ich złożoności i z uwzględnieniem mocy obliczeniowej oraz zasobów pamięciowych jednostek (węzłów) przetwarzających:

$$FPO = Qp \sum_{i=1}^{N} \left| Zti/Ztmax - Pwi/Pwmax \right| + Qm \sum_{i=1}^{N} \left| Zmj/Zmmax - Mi/Mmax \right| \rightarrow \min \qquad (1.1)$$

gdzie

FPO - bazowa funkcja optymalizacyjna dla tworzenia funkcji przystosowania,
Qp - współczynnik uwzględnienia złożoności czasowej,
Qm - współczynnik uwzględnienia złożoności pamięciowej,
i - numer jednostki przetwarzającej,
j - numer procesu przypisanego do i – tej jednostki przetwarzającej
j = 1, 2,..., K (K ≤ N).

W zamyśle, przy wyborze funkcji bazowej (1.1) do budowy funkcji przystosowania, oparto się na założeniu, iż optymalne dostosowanie złożoności procesu do mocy konwertera wpłynie na realizację kompleksu zadań w najkrótszym czasie. Założenie to jednak może okazać się nieprecyzyjne lub nawet niesłuszne na przykład przy dużym rozrzucie mocy obliczeniowej. W takiej sytuacji realizacja dwóch lub nawet kilku procesów przez jeden z procesorów o znacznie większej mocy może okazać się efektywniejsza i szybsza. Również w przypadku kiedy liczba procesów przewyższa liczbę procesorów (K > N) pewne jednostki przetwarzające będą wykonywały większą liczbę procesów niż inne. W takiej sytuacji należy wprowadzić do formuły (1.1) składowe dotyczące stopnia zajętości procesora.

$$FPO = Qp \sum_{i=1}^{N} \{ | Ztj/Ztmax - [Pwi - \sum_{\substack{l=1 \\ l \neq j}}^{qi} \Delta Pwl] /Pwmax | \} +$$

$$+ Qm \sum_{i=1}^{N} \{ | Zmj/Zmmax - [Mi - \sum_{\substack{l=1 \\ l \neq j}}^{qi} \Delta Ml] /Mmax | \} \rightarrow \min \qquad (1.2)$$

gdzie qi – długość kolejki do i –tego węzła przetwarzającego,

ΔPw, ΔM – składowe stopni zajętości przez poszczególne procesy.

Oszacowanie parametrów ΔPw, ΔM można przeprowadzić na różne sposoby. Na przykład wykorzystanie technik normalizacyjnych można odnieść do średniej mocy obliczeniowej, sumarycznej mocy obliczeniowej albo też maksymalnej mocy jednego z węzłów sieci.

$$\Delta Pwi = Zti / \sum_{j=1}^{K} Ztj \; * \sum_{l=1}^{N} Pwl /N \qquad \text{(dla małych K)} \qquad (1.3)$$

$$\Delta Pwi = Zti / \sum_{j=1}^{K} Ztj \; * \sum_{l=1}^{N} Pwl \qquad \text{(dla dowolnych K)} \qquad (1.4)$$

$$\Delta Pwi = Zti / \sum_{j=1}^{K} Ztj \; * \sum_{l=1}^{N} Pwmax \qquad \text{(dla średnich K i małego rozrzutu mocy obliczeniowej)} \qquad (1.5)$$

Parametr K szacowany jest odnośnie liczby węzłów sieci N (uwaga dotyczy rekomendacji w nawiasach w formułach 1.3, 1.4, 1.5).

Ostatnia koncepcja zakłada uwzględnienie przepustowości oraz przepływów informacji.

Odniesienie przepływającej informacji do przepustowości danego łącza stanowi parametr rozproszenia danych. Aby opisać stan rozproszenia informacji w skali całej struktury sieciowej można wykorzystać wartość wariancji znormalizowanych przepływów co przedstawić można następująco:

$$V = 1/N^2 \sum_{i=1}^{N} \sum_{\substack{j=1 \\ j \neq i}}^{N} (Fi\text{-}j/Pi\text{-}j - Av(F/P))^2 \quad , \tag{1.6}$$

gdzie V – współczynnik rozproszenia przesyłanej informacji w sieci,

Av(*) – średnia wartości „*" obliczona tylko dla łączy aktywnych,

Fi-j/Pi-j - znormalizowany przepływ od węzła „i" do węzła „j"; wartość ta jest równa zeru jeżeli między punktami nie istnieje łącze lub jeśli jest ono nie wykorzystane (łącze nieaktywne).

We wzorze (1.6) miast średniej możemy używać wartości absolutnych. Tak czy inaczej współczynnik V w sytuacji optymalnej dystrybucji winien osiągnąć wartość minimalną. Można więc bez skrupułów dołączyć ją do struktury funkcji przystosowania, a ostateczna jej postać przybierze następującą formę:

$$FPO = Qp \sum_{i=1}^{N} \{ | Ztj/Ztmax - [Pwi - \sum_{\substack{l=1 \\ l \neq j}}^{qi} \Delta Pwl] /Pwmax | \} +$$

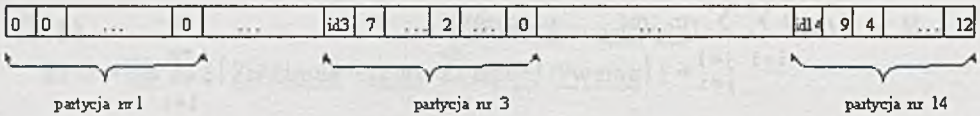
$$+ Qm \sum_{i=1}^{N} \{ | Zmj/Zmmax - [Mi - \sum_{\substack{l=1 \\ l \neq j}}^{qi} \Delta Ml] /Mmax | \} + \rightarrow \min \tag{1.7}$$

$$+ 1/N^2 \sum_{i=1}^{N} \sum_{\substack{j=1 \\ j \neq i}}^{N} (Fi\text{-}j/Pi\text{-}j - Av(F/P))^2 \rightarrow \min ,$$

W wyrażeniu (1.7) uwzględniono zarówno złożoność obliczeniową wejściowych procesów, zajętość procesorów jak i przepływ informacji po łączach o różnych przepustowościach.

1.3 Koncepcja kodowania struktury chromosomu

W proponowanym podejściu zakładamy, iż w chromosomie będzie odzwierciedlone wejście procesu do sieci i droga do procesora realizującego proces. Pozostałe pola mogą być wyłączone z przetwarzania genetycznego bądź mogą stanowić uzupełniające weryfikowanie wymiany genetycznych informacji.



Rys. 1.3 Struktura chromosomu w wersji uproszczonej tj. bez weryfikacji stanów przejściowych

Na powyższym rysunku aktywnymi partycjami są wyłącznie te, które powiązać można z wejściowymi procesami. Przykładowo partycja nr 1 jest wyłączona, ponieważ żaden proces nie został wygenerowany (wprowadzony, wyindukowany) w węźle nr 1. Partycja nr 3 jest aktywna, ponieważ w 3-cim węźle sieci pojawił się proces o identyfikatorze id3, który został skierowany do procesora w węźle 2 – gim, a przesłanie jest realizowane poprzez węzeł o numerze 7. W węźle 14 wygenerowano proces o identyfikatorze id14, który poprzez węzły 9, 4 dociera do procesora w węźle 12. Jeśli w określonych węzłach pojawia się tylko po jednym procesie to jako identyfikatorów procesów można użyć, numerów węzłów. Długość partycji określamy w następujący sposób LP = Lid + max {Lpath}, gdzie Lid – długość kodu identyfikatora procesu, Lpath - długość drogi od węzła wejściowego do węzła realizującego dany proces (wyrażona w węzłach). Całkowita długość chromosomu będzie więc równa N*LP.

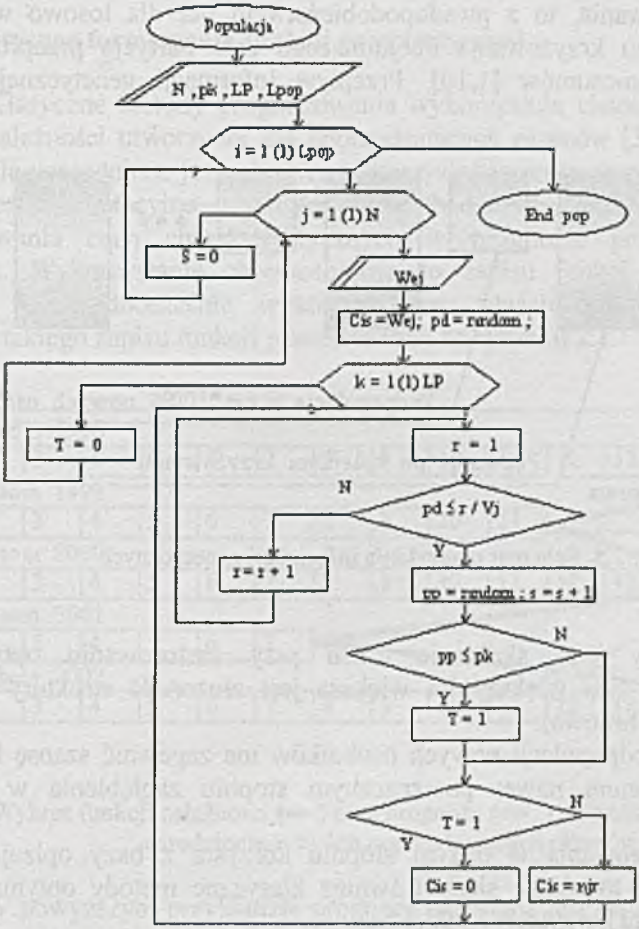
Pierwsze wykorzystanie kodowania systemu będzie miało miejsce przy generacji populacji początkowej. Algorytm takiej generacji przedstawiono na rys.1.4. Działanie algorytmu rozpoczyna się od określenia węzłów, w których pojawiają się procesy wejściowe. Następna faza generacji to tworzenie trajektorii przesyłu procesu do węzła docelowego. Po każdorazowym przejściu między kolejnymi węzłami sprawdzamy czy zgodnie z zadanym prawdopodobieństwem pk aktualny węzeł będzie węzłem docelowym. Procedura przesyłu procesów będzie się powtarzała do momentu kiedy ostatni z procesów nie dotrze do węzła realizacji. W taki sposób zostanie utworzony pojedynczy chromosom.

Po wygenerowaniu populacji bazowej relizujemy klasyczne operacje genetyczne. Do dalszej analizy wybieramy w drodze selekcji osobniki najlepiej przystosowane (1.7), tworząc jednocześnie podpopulację, której cechą będzie obecność dodatkowych osobników wygenerowanych według algorytmu (rys. 1.4.).

Objaśnienia do rys. 1.4:

Lpop	- wielkość populacji
i	- numer osobnika w populacji
Wej	- numer węzła z wejściowym procesem
Cis	- s-ty gen i-tego chromosomu
Vj	- liczba dołączonych węzłów do węzła o numerze: j

T - wskaźnik zakończenia przesyłu procesu
pd, pp - pomocnicze zmienne losowe
njr - numer „mijanego" węzła (przez j – ty proces)

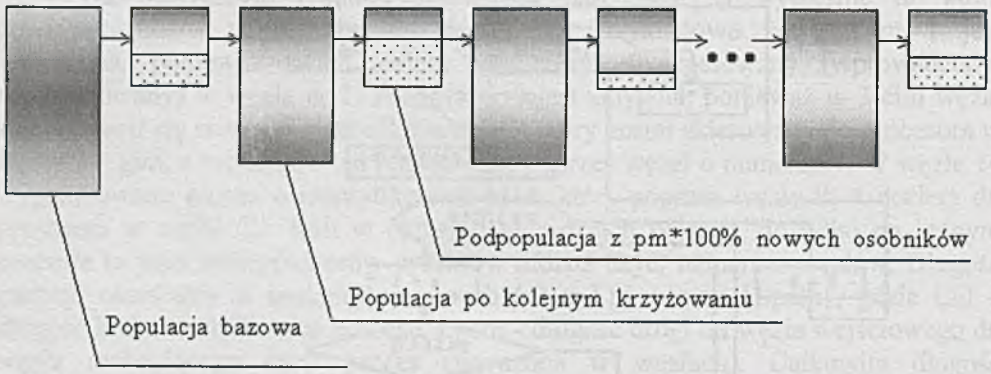


Rys.1.4. Algorym generowania populacji

1.4 Procedury genetyczne i ocena przystosowania

W zestawie opracji genetycznych wykorzystano jedynie krzyżowanie, jako że inwersja genu tutaj reprezentowanego jako partycję opisującą przemieszczenie procesu

z pewnością mogłaby naruszyć topologiczne powiązania danej sieci. Mutację zaś przeprowadzimy tak by nowe geny nie zmieniały topologii sieci co zrealizujemy poprzez wprowadzenie do nowej populacji nowych osobników w skali pm*100%. Nie będzie to mutacja w klasycznym rozumieniu lecz jedynie „przypływ świerzej krwi". Wracając do krzyżowania, to z pwadopodobieństwem pcr dla losowo wybranego punktu (lub punktów) krzyżowania obejmującego całe partycje przeprowadzamy zamianę części chromosomów [1,10]. Przepływ informacji genetycznej ilustruje schemat na rys.1.5.



Rys. 1.5. Schemat przesyłania informacji genetycznych

Wnioski:

1. Zysk rozumiany jako skrócenie czasu przy zastosowaniu optymalizacji genetycznej jest tym większy im większa jest złożoność struktury sieciowej (zależność quasi-liniowa).
2. Włączenie do podpopulacji nowych osobników ma zapewnić szansę lokalizacji globalnego optimum nawet po znacznym stopniu zagłębienia w optimum lokalnym.
3. Funkcja przystosowania w dużym stopniu korzysta z bazy opisującej stan, charakterystyki i topologię sieci. Również klasyczne metody optymalizacji w podobnym stopniu korzystają z takiej bazy.

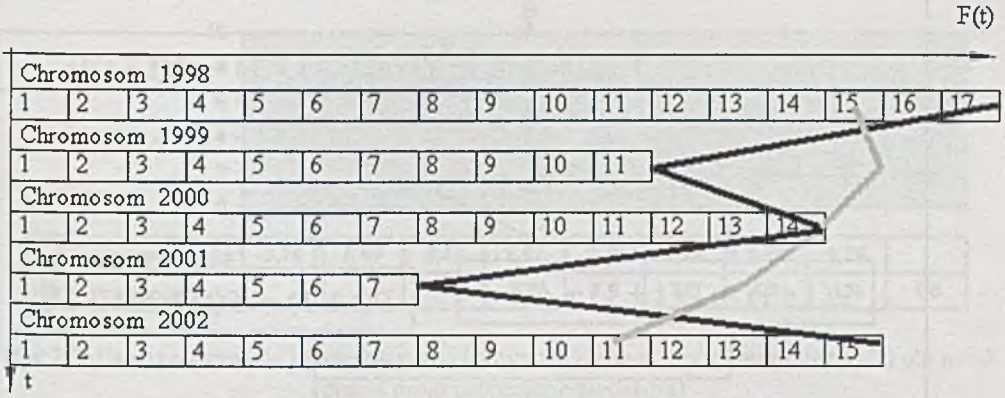
2. Wykorzystanie operacji genetycznych do prognozowania gospodarczego

Prognozowanie polega na określaniu wartości parametru przy jednoczesnym zachowaniu stałości lub tendencji zmian wybranych charakterystyk. Jednocześnie istnieją kryteria oceny jakości prognozy. Fakt ten sugeruje celowość poszukiwania nie tyle innych metod prognozowania (charakteryzujących się lepszymi cechami jakościowymi) ile w ramach danej metody innych parametrów funkcji prognostycznej.

Taką możliwość zapewni nam iteracyjność procedur genetycznych. Inną zaletą metod genetycznych jest ich elastyczność jeżeli chodzi o postać funkcji przystosowania oraz mnogość form funkcji prognostycznych, z których można wyłonić najbardziej adekwatnie opisującą badane zjawisko [10].

2.1 Genetyczne formowanie funkcji prognostycznej

Klasyczne metody prognozowania wykorzystują charakterystyki i parametry funkcji zależności utworzonej dla poprzedzających okresów [3]. Podejście takie jest oczywiście uzasadnione jeśli chcemy wykorzystać w prognozowaniu charakter, trendy i własności korelacyjne występujące w badanym zjawisku. Jednakże sposób odwzorowania cech charakterystycznych w prognozie pozostaje zagadnieniem otwartym. Wykorzystanie chromosomów do zapisu funkcji zależności badanego zjawiska jest jednocześnie wykorzystaniem właściwości zapisu genetycznego. Przykład takiego zapisu funkcji przedstawiono na rysunku 2.1.

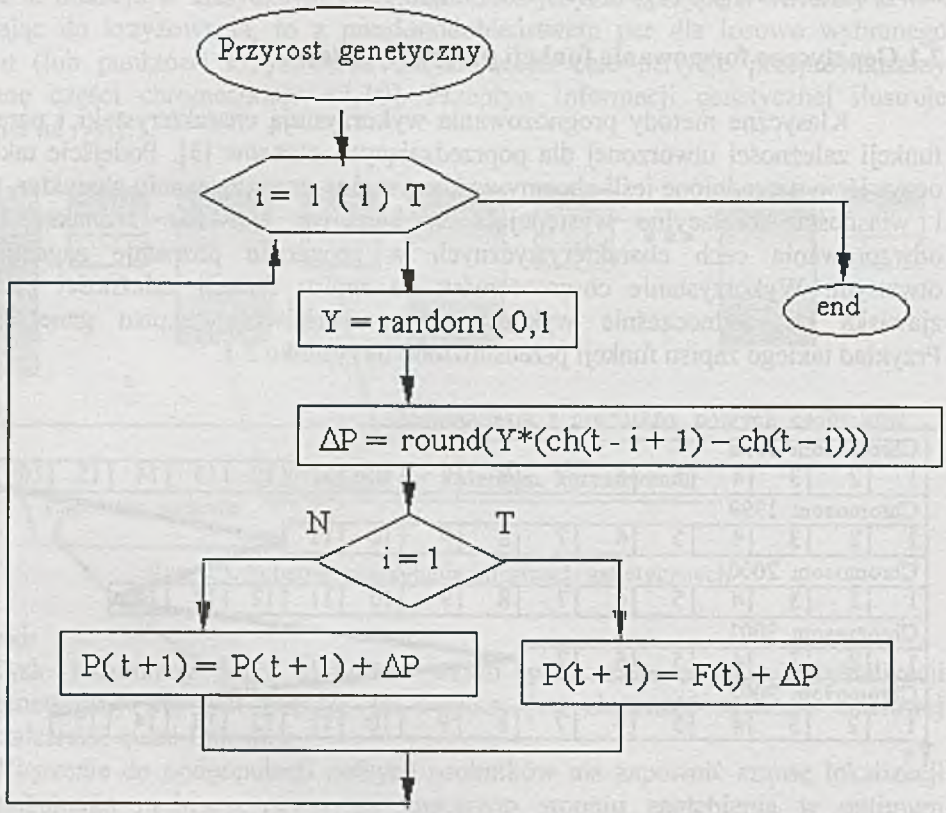


Rys. 2.1. Wykres funkcji zależności (—) oraz prognozy (—) na bazie chromosomów (metodą uśrednienia z dwóch poprzedzających okresów)

W powyższym przykładzie prognoza określana jest jako średnia arytmetyczna wartości uzyskanych w dwu (lub kilku) poprzedzających okresach (P(t+1) = (P(t) + P(t-1))/2). Pozostaje kwestia organizacji przebiegu operacji genetycznych w zależności od ilości uwzględnianych w uśrednieniu znaczeń (wartości) z poprzedzających okresów. Znalezienie algorytmu winno uwzględniać specyfikę i środowisko genetyczne oraz techniki wykorzystywane w realizacji operacji genetycznych.

W pierwszej wersji proponowana jest seria operacji genetycznych o charakterze przyrostowym. Nie będą to operacje przypominające algorytm krzyżowania, mutacji czy inwersji. W tej wersji kolejne wartości prognoz nie zależą od

prognoz dotyczących poprzednich okresów lecz tylko od rzeczywistych danych z tychże okresów. Opis przyrostowej operacji genetycznej przedstawiono na rys. 2.2.



Rys. 2.2. Algorytm przyrostowej operacji genetycznej, gdzie T – ilość okresów poprzedzających uwzględnianych w prognozie, ΔP – przyrost wartości prognostycznej, ch(tr) - długość chromosomu odpowiadającego wartości rzeczywistej z okresu tr, P(t+1) – wartość prognozy dla okresu (t + 1)

Przebieg realizacji serii przyrostów można opisać w następujący sposób:

(1-sza iteracja)

$P = \xi_1 (F(t-1)-F(t)) + F(t) = (1- \xi_1)F(t) + \xi_1 F(t-1)$

(2 – ga iteracja)

$P = \xi_2 (F(t-2) – F(t-1)) + (1- \xi_1)F(t) + \xi_1 F(t-1) = (1- \xi_1) F(t) + (\xi_1 - \xi_2)F(t-1) + \xi_2 F(t-2)$

(3 – cia iteracja)

$P = \xi_3(F(t-3) - F(t-2)) + (1-\xi_1) F(t) + (\xi_1 - \xi_2)F(t-1) + \xi_2F(t-2) = (1-\xi_1) F(t) + (\xi_1 - \xi_2)F(t-1) + (\xi_2 - \xi_3)F(t-2) + \xi_3F(t-3)$

• •

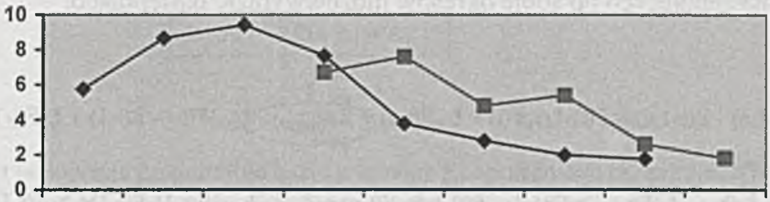
(n-ta iteracja)

$$P = \xi_n(F(t-n) - F(t-n+1)) + (1-\xi_1)F(t) + (\xi_1 - \xi_2)F(t-1) + \ldots + (\xi_{n-2} - \xi_{n-1})F(t-n+2) + \xi_n F(t-n+1) = (1-\xi_1)F(t) + (\xi_1 - \xi_2)F(t-1) + (\xi_2 - \xi_3)F(t-2) + \ldots + (\xi_{n-1} - \xi_n)F(t-n+1) + \xi_n F(t-n) \quad (2.1)$$

Stąd

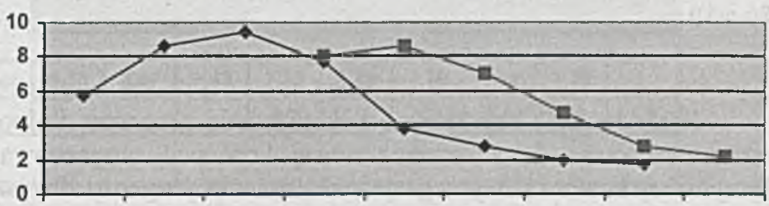
$$P(n) = F(t) - \xi_1F(t) + \xi_nF(t-n) + \sum_{i=1}^{n-1}(\xi_{i+1} - \xi_i)F(t-i) \quad (2.2)$$

gdzie n- głębia prognostycznego śledzenia czyli liczba okresów poprzedzających uwzględnianych w prognozowaniu.



rzeczywistość	5,76	8,67	9,43	7,67	3,78	2,78	1,98	1,76	
prognoza1				6,7	7,6	4,8	5,4	2,6	1,8

Rys.2.3. Dane rzeczywiste i prognostyczne (prognoza opisana algorytmem na rys. 2) dla n = 3 (głębia prognostycznego śledzenia)



rzeczywistość	5,76	8,67	9,43	7,67	3,78	2,78	1,98	1,76	
prognoza2				8	8,6	7	4,7	2,8	2,2

Rys. 2.4. Dane rzeczywiste i prognostyczne (uśrednienie z poprzednich okresów) dla n = 3

Porównanie jakości przedstawionych prognoz (opartych na metodzie przyrostowej i metodzie uśrednienia) wskazuje, iż nie ma między nimi istotnych pod tym względem różnic. W poniższej tabeli przedstawiono ilościowe kryteria jakości prognozowania obiema metodami [3].

Tablica 2.1. Ilościowe kryteria jakości prognozowania

okresy śledzenia	1	2	3	4	5		jakość pr.
rzeczywistość	7.67	3.78	2.78	1.98	1.76		
prognoza1	6.7	7.6	4.8	5.4	2.6		
kwadrat różnicy	0.94	14.59	4.08	11.70	0.71	std.odch.	2.53
rzeczywistość	7.67	3.78	2.78	1.98	1.76		
prognoza2	8	8.6	7	4.7	2.8		
kwadrat różnicy	0.11	23.23	17.81	7.40	1.08	std.odch.	3.15

Zależność między dwoma kolejnymi prognozami czyli prognozami odnoszącymi się do następujących po sobie okresów można wyrazić następująco:

$$P(t,n) - P(t-1,n) = F(t)-\xi_{1+n}F(t) + \xi_{2n}F(t-n) + \sum_{i=1}^{n-1}(\xi_{i+n+1} - \xi_{i+n})F(t-i)- F(t-1) + \xi_1F(t-1) - \xi_nF(t-n-1)$$

$$- \sum_{i=1}^{n-1}(\xi_{i+1} - \xi_i)F(t-i-1)=F(t)-\xi_{1+n}F(t) + \xi_{2n}F(t-n) +(\xi_{n+2} - \xi_{n+1})F(t-1)-F(t-1)+ \xi_1F(t-1)- \xi_nF(t-n-1) -$$

$$-(\xi_n - \xi_{n-1})F(t-n) = F(t)-\xi_{n+1}F(t) +(\xi_1 + \xi_{n+2} - \xi_{n+1} - 1)F(t-1) +(\xi_{2n} -\xi_n+ \xi_{n+1})F(t-n) - \xi_nF(t-n-1)$$

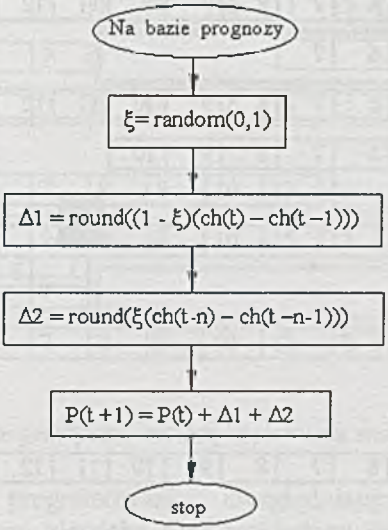
Dokonano tutaj uproszczenia typu $(\xi_{i+n+1} - \xi_{i+n}) = (\xi_{i+1} - \xi_i)$.
I kolejny krok w kierunku schematyzacji $\xi_1 = \xi_2 = \xi_3$ ..., co prowadzi do

$$P(t,n) - P(t-1,n) = (1 - \xi)F(t) -(1- \xi)F(t-1) + \xi F(t-n) - \xi F(t-n-1) = (1- \xi)(F(t)-F(t-1))+ \xi(F(t-n) - F(t-n-1))$$

2.3

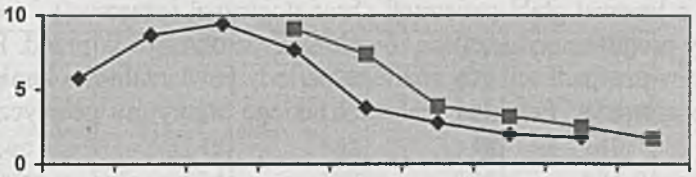
Formułę (2.3) bazującą na wykorzystaniu prognozy dla poprzedniego okresu można zastosować do stworzenia nowej koncepcji prognozowania genetycznego. Algorytm tej koncepcji wymaga jednak istnienia prognozy dla poprzedniego okresu śledzenia

2.2 Genetyczne algorytmy tworzenia prognozy



Rys.2.5 Algorytm budowania prognozy na bazie prognozy z poprzedniego okresu śledzenia

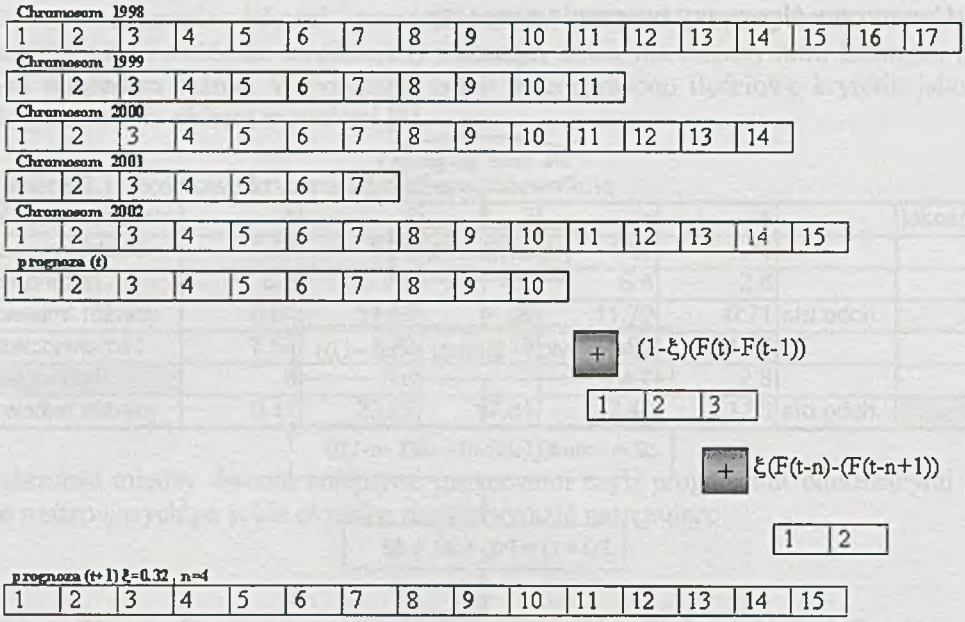
Efekt działania powyższego algorytmu można przedstawić graficznie (rys. 2.6).



rzeczywistość	5,76	8,67	9,43	7,67	3,78	2,78	1,98	1,76	
prognoza3				9,1	7,4	3,9	3,2	2,5	1,7

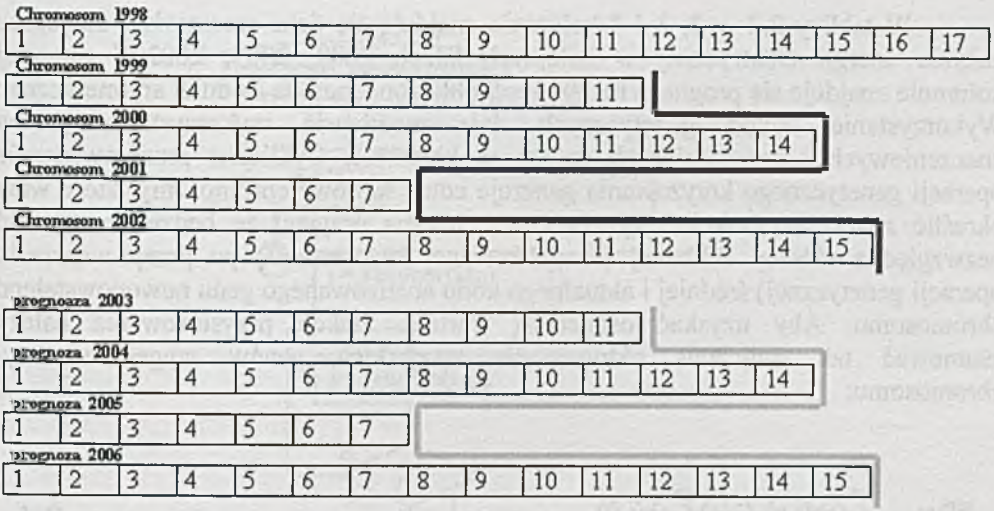
Rys. 2.6.Dane rzeczywiste i prognostyczne (prognoza opisana algorytmem na rys. 5) dla n=3

Interpretacja genetyczna algorytmu opisanego na rys. 2.5. daje również możliwość zadania zakresu i rozkładu prawdopodobieństwa generacji wartości losowej ξ. Opis graficzny realizacji operacji genetycznych przedstawia rys. 2.7.



Rys. 2.7.Realizacja operacji genetycznego złożenia na bazie prognozy z poprzedniego okresu śledzenia

Prognozy mogą odzwierciedlać w dosłownym sensie charakterystyki poprzednich okresów śledzenia jeśli skopiować znaczenia rzeczywistych danych na przykład o losowej głębi penetracji chronologicznej (czasowej) - tpc. Tym sposobem uzyskamy prognozę wybiegającą również na tpc okresów w przód. Byłoby to bardziej przydatne w przypadkach gdy znane są periody powtarzalności koniunktury przebiegu badanego zjawiska. Przykład realizacji takiego algorytmu genetycznego przedstawia rys. 2.8.



Rys. 2.8. Prognozowanie genetyczne metodą kopiowania struktury chromosomów

W koncepcjach programowania uwzględniających sezonowość (lub okresowość) zmian przebiegu zjawiska czas rozkładany jest na dwie współrzędne. Jedną z nich jest numer sezonu (okresu) drugą zaś przebieg wewnątrz danego okresu. Przykładem takiej prognozy może być badanie zapotrzebowania na energię elektryczną w poszczególnych sezonach danego roku.

Tablica 2.2 Dane prognostyczne dotyczące zapotrzebowania na energię elektryczną

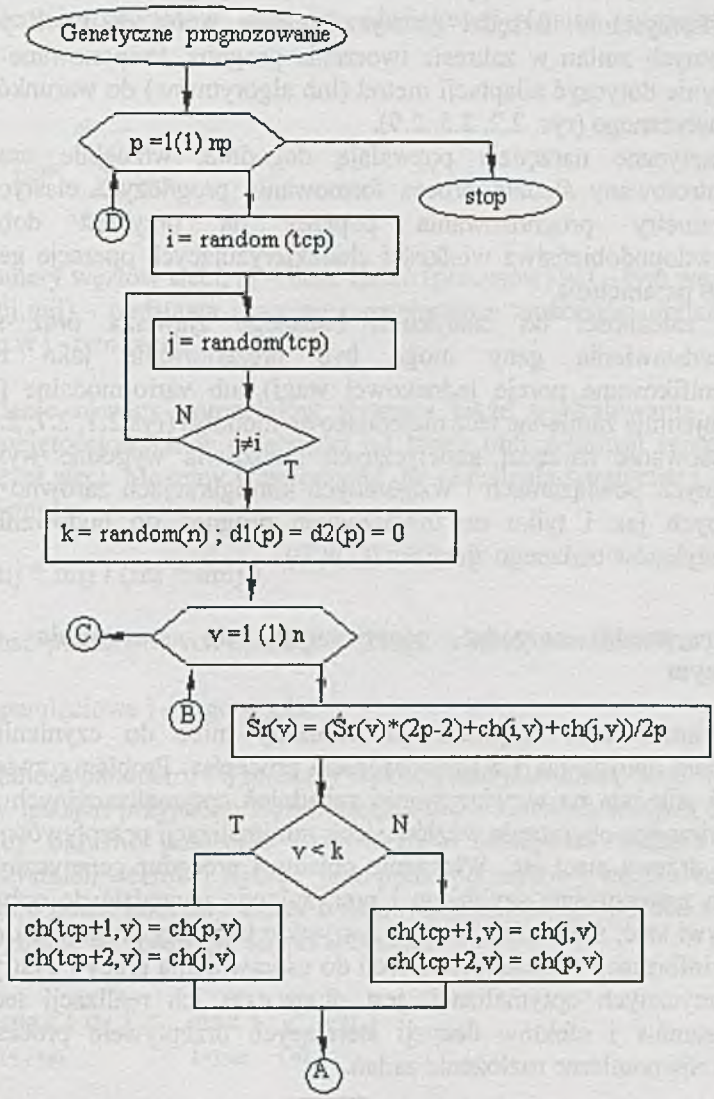
rok >	2000	2001	2002	progn.2003
styczeń	175	177	172	174.7
luty	154	161	158	157.7
marzec	148	152	149	149.7
kwiecień	134	132	136	134.0
maj	121	126	122	123.0
czerwiec	118	125	119	120.7
lipiec	113	112	115	113.3
sierpień	125	119	122	122.0
wrzesień	138	128	134	133.3
pazdziernik	146	149	151	148.7
listopad	165	162	163	163.3
grudzień	181	179	172	172.0

W tablicy 2.2 w 2, 3 i 4 kolumnie znajdują się dane rzeczywiste dotyczące zużycia energii elektrycznej w kolejnych latach 2000, 2001, 2002.W czwartej kolumnie znajduje się prognoza na 2003 rok obliczona metodą średniej arytmetycznej. Wykorzystanie metod genetycznych daje możliwość wykorzystania genów znaczeniowych odpowiadających danym w kolumnie. Cykliczne przeprowadzanie operacji genetycznego krzyżowania generuje coraz to nowe chromosomy, które wraz określić znaczenie funkcji przystosowania. Można przyjąć, że będzie to wartość bezwzględna różnicy aktualnej (modyfikowanej każdorazowo po przeprowadzeniu operacji genetycznej) średniej i aktualnego kodu analizowanego genu nowopowstałego chromosomu. Aby uzyskać ostateczną wartość funkcji przystosowania należy zsumować tak obliczone różnice dla wszystkich genów wygenerowanego chromosomu:

$$FP = \sum_{i=1}^{12} (ch(i,n) - (1/n) \sum_{j=1}^{n} ch(i,j)) \qquad 2.4$$

gdzie ch(i,n) – kod i-tego genu chromosomu powstałego w n-tej generacji.

Operację genetycznego krzyżowania realizować można wybierając losowo punkt krzyżowania i wymianę końcowej lub początkowej części chromosomów (rys. 2.9).



Rys. 2.9. Algorytm genetycznego prognozowania przebiegu zjawiska o charakterze sezonowym lub koniunkturalnym bazujący na dynamicznej średniej (np – liczba generacji genetycznego poszukiwania optimum, i, j – kody rodziców, k – punkt krzyżowania, Śr(v) – aktualna wartość dynamicznej średniej, n – długość chromosomu, ch(i,v) – gen chromosomu, tcp – aktualna wielkość populacji, t – kod optymalnego rozwiązania - chromosomu)

Wnioski

1. Wykorzystanie narzędzi genetycznych nie wnosi do teorii prognozowania istotnych zmian w zakresie tworzenia prognoz. Proponowane zmiany mogą jedynie dotyczyć adaptacji metod (lub algorytmów) do warunków środowiska genetycznego (rys. 2.2, 2.5, 2.9).
2. Genetyczne narzędzia pozwalają dogodnie, wizualnie oraz w sposób kontrolowany śledzić proces formowania prognozy i elastycznie dobierać parametry prognozowania poprzez na przykład dobór rozkładu prawdopodobieństwa wielkości charakteryzujących operacje genetyczne oraz jego parametrów.
3. W zależności od charakteru badanego zjawiska oraz sposobu jego przedstawienia geny mogą być prezentowane jako mono-modalne (zunifikowane porcje jednakowej wagi) lub vario-modalne (wyskalowane, wzajemnie zamienne lecz niejednakowe moduły) (rys.2.1, 2.7, 2.8, 2.9).
4. Stosowanie narzędzi genetycznych umożliwia wygodne wykorzystanie w różnych powiązaniach i wzajemnych konfiguracjach zarówno rzeczywistych danych jak i tylko co utworzonych prognoz do budowania kolejnych predykatów badanego zjawiska ((2)i(3)).

3. Wykorzystanie narzędzi genetycznych do sterowania środowiskiem rozproszonym

W środowisku rozproszonym możemy mieć do czynienia zarówno z rozproszeniem danych jak i z rozproszeniem procesów. Problemy związane z takimi strukturami polegają na rozwiązywaniu zagadnień optymalizacyjnych zmierzających do równomiernego obciążenia węzłów sieci, minimalizacji przepływów, minimalizacji rozpiętości drzewa sieci etc. Włączenie opisów i procedur genetycznych nie będzie przykładem zastosowania szybkiego i precyzyjnego narzędzia do optymalizacji lecz niejako ożywi sieć, śledząc na bieżąco wszystkie transfery procesów i danych i może dostarczać informacji wykorzystywanych do usprawnienia pracy i dystrybucji w sieci. Wadą genetycznych optymalizacji jest długi czas ich realizacji lecz równoległe śledzenie stanów i efektów decyzji sterujących przepływem procesów i danych pozwoli na równomierne rozłożenie zadań.

3.1 Koncepcja równomiernego obciążenia węzłów sieci

Istniejące algorytmy optymalizacji w strukturach sieciowych mimo, że wykorzystują deterministyczne zasady przetwarzania nie prowadzą do uzyskania absolutnie optymalnych rozwiązań [33]. Proponowana metoda angażuje narzędzia probabilistyczne i mimo iż nie prowadzi do czysto optymalnych rozwiązań pozwoli na niwelowanie lokalnych tendencji wzrostowych lub spadkowych rozkładając

równomiernie obciążenie węzłów sieci i komunikację między węzłami. Przyjmując terminologię teorii potencjałów możemy zdefiniować różnicę potencjałów między węzłami sieci:

$$\Delta\varphi = \sum_{i=1}^{nj} Pij - \sum_{i=1}^{nk} Pik, \qquad 3.1$$

gdzie j,k – numery węzłów sieci, nj – ilość zadań (procesów) w j – tym węźle,
Pij = (tij,mij) - podstawa czasowa i pamięciowa złożoności obliczeniowej i . – tego procesu w j –tym węźle.

Określenie różnicy potencjałów wymaga także wyskalowania parametrów czasowych i objętościowych w zależności od mocy obliczeniowej reprezentowanej przez dany węzeł sieci. Możemy więc przyjąć dla określenia parametru j –tego węzła następującą formułę:

$$Pij = tij * mij / (ntj * nmj), \qquad 3.2$$

gdzie ntj – ilość podstawowych operacji w ciągu sekundy, realizowana przez j – ty węzeł,
nmj – zasoby pamięciowe j – tego węzła,

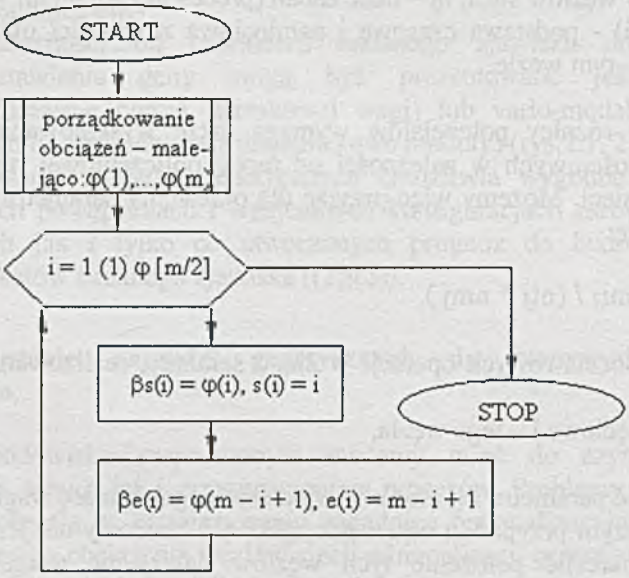
Odwrotność parametru Pij możemy wykorzystać jako miarę wagi w algorytmie Dijkstry [5].W naszym przypadku węzeł początkowy i końcowy nie jest od początku określony. Aby określić położenie tych węzłów należy na wstępie oszacować obciążenie wszystkich węzłów i wybrać jako punkt początkowy węzeł o największym obciążeniu i jako punkt końcowy węzeł o najmniejszym obciążeniu. Do wyznaczenia obciążenia wygodnie jest posłużyć się potencjałem:

$$\beta s = \max_{1<j<m} \{ \varphi j \} = \max_{1<j<m} \{ \sum_{i=1}^{nj} Pij \}$$

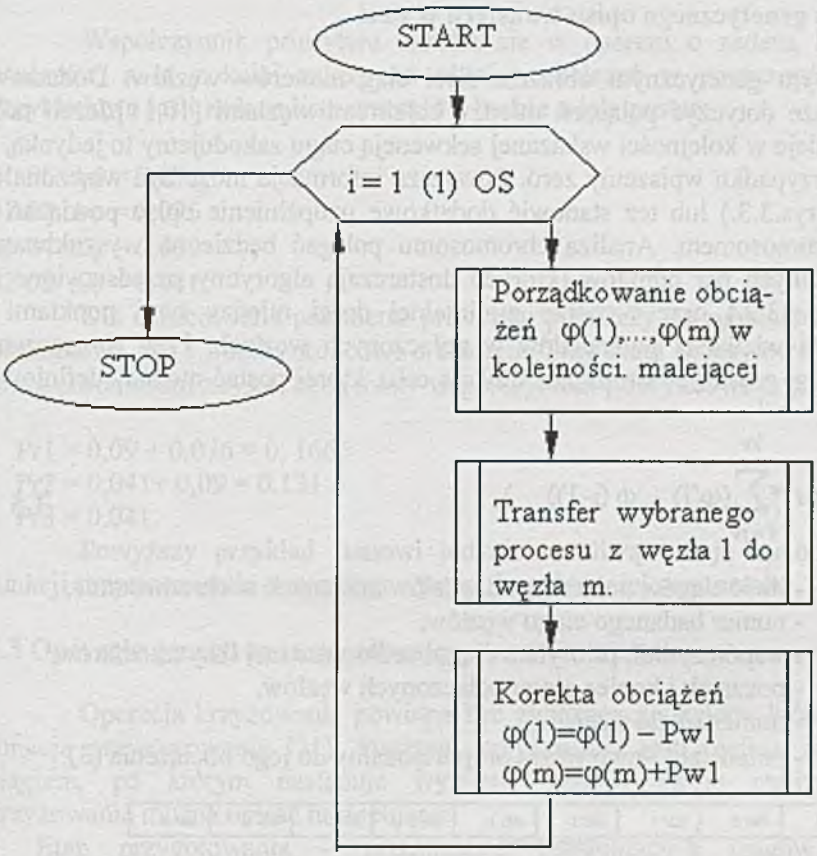
$$\beta e = \min_{1<j<m} \{ \varphi j \} = \min_{1<j<m} \{ \sum_{i=1}^{nj} Pij \} \qquad 3.3$$

gdzie βs,βe - obciążenie punktu startowego i końcowego.

Zestaw w taki sposób uporządkowanych par (βs,βe) tworzy kolejkę, która może być utworzona z pomocą algorytmu stanowego lub śledzącego, których koncepcję ilustrują rysunki 3.1 i 3.2.W algorytmie stanowym o kolejności procedur optymalizacji decyduje aktualny stan obciążenia węzłów. W algorytmie śledzącym po przesłaniu kolejnego procesu przy wyborze kolejnej pary do optymalizacji drogi transferu procesu uwzględniamy zmiany obciążenia wynikające z poprzedniego transferu.



Rys. 3.1. Algorytm stanowy do tworzenia listy transferów procesów między węzłami sieci, i – numer pary przesyłowej (s(i) - punkt początkowy, e(i) - punkt końcowy))



Rys. 3.2 Algorytm śledzący do ustalania kolejności transferów w okresie śledzenia OS, gdzie Pw1- wybrany do transferu proces o numerze w węźle 1

Przed podjęciem ostatecznej decyzji o transferze procesu z oczywistych względów należy sprawdzić warunek:

$$Pwj \leq \varphi(j) - \varphi(k) \qquad 3.4$$

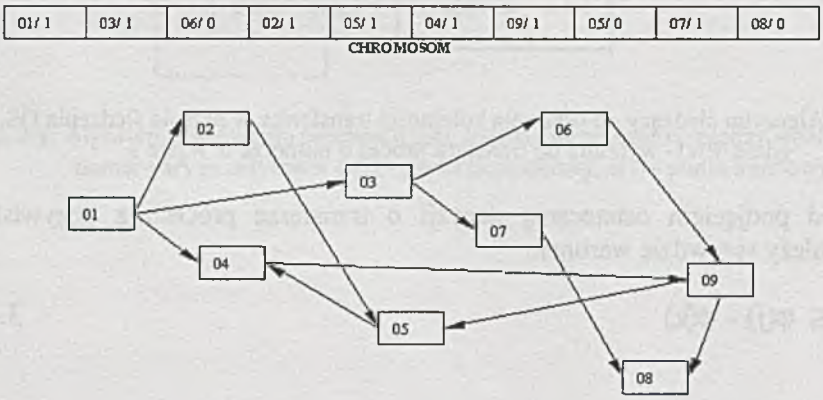
gdzie
j – numer węzła początkowego transferu,
k – numer węzła końcowego.

3.2 Koncepcja genetycznego opisu transferu w sieci

Typowym genetycznym obrazem sieci ciąg numerów węzłów. Dodatkowa informacja może dotyczyć połączeń między kolejnymi węzłami [10,11].Jeżeli takie połączenie istnieje w kolejności wskazanej sekwencją ciągu zakodujemy to jedynką, w przeciwnym przypadku wpiszemy zero. Powyższa informacja może być włączona do chromosomu (rys.3.3.) lub też stanowić dodatkowe uzupełnienie opisu powiązań w sieci poza chromosomem. Analiza chromosomu polegać będzie na wyszukiwaniu powiązań zadanych par punktów (których dostarczają algorytmy przedstawione na rys.3.1., lub rys.3.2.) oraz wyborze minimalnej drogi między tymi punktami z uwzględnieniem wielkości potencjałów w połączonych węzłach. Przy wyznaczaniu minimalnej drogi posłużyć się można funkcją celu, której postać można zdefiniować następująco:

$$F = \max_{1 \le v \le w} \{ pr * \sum_{i=jv}^{kv} (\varphi(i) - \varphi(i-1)) \}, \qquad 3.5$$

gdzie
- w - ilość ciągów połączonych węzłów opisanych w chromosomie,
- v - numer badanego ciągu węzłów,
- pr - współczynnik priorytetu ciągu według zadanej listy transferów,
- jv,kv - początek i koniec ciągu połączonych węzłów,
- i - numer węzła sieci,
- φ(i) - potencjał i –tego węzła proporcjonalny do jego obciążenia (3).



Rys. 3.4. Struktura chromosomu i sieci posiadającej połączenia opisane przez niego
i / 1 – oznacza, iż i-ty węzeł nie jest węzłem końcowym w sekwencji opisanej ciągiem genów,
i / 0 - i-ty węzeł jest węzłem końcowym w danej sekwencji.

Współczynnik priorytetu określa się w oparciu o zadaną listę transferów procesów, a ta z kolei zależy od różnicy obciążeń w poszczególnych węzłach. Przykładowo jeśli zadana lista przesyłów będzie miała postać:

01-05 $\Delta\varphi = 0,09$
01-07 $\Delta\varphi = 0,076$
06-04 $\Delta\varphi = 0,069$
02-08 $\Delta\varphi = 0,041$

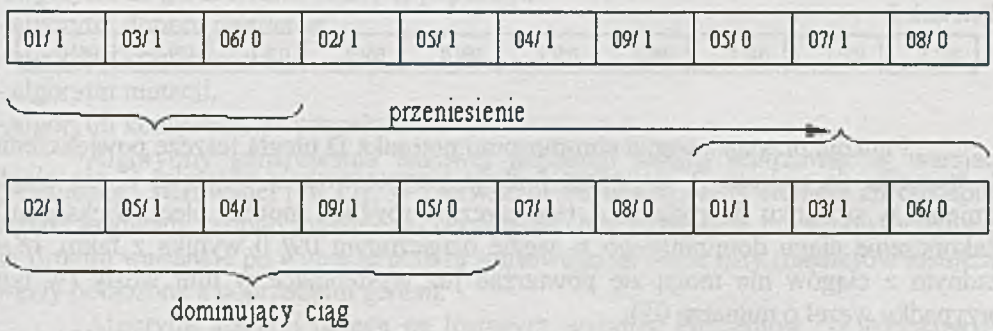
Dla oszacowania parametru priorytetu pr należy uwzględnić zarówno punkty początkowe jak i punkty końcowe transferu. Trzy ciągi przesyłowe przedstawione przez chromosom z rys.4. będą miały współczynniki priorytetowe odpowiednio równe:

Pr1 = 0,09 + 0,076 = 0, 166,
Pr2 = 0,041+ 0,09 = 0,131
Pr3 = 0,041.

Powyższy przykład stanowi jedynie możliwą wersję metody oszacowania funkcji przystosowania oraz oceny występujących w niej parametrów.

3.3 Operacje genetyczne wspomagające transfer w sieci

Operacja krzyżowania powinna być związana z ciągiem, któremu odpowiada funkcja przystosowania [21]. Punktem krzyżowania jest miejsce za dominującym ciągiem, po którym następuje wymiana genów. Etapy realizacji procedury krzyżowania można opisać następująco:
1. Etap przygotowania – przesunięcie dominujących ciągów na początek chromosomów ; elementy poprzedzające ciąg dominujący przemieszamy jako blok na koniec chromosomu.



2. Etap wyboru punktu krzyżowania. Miejsce krzyżowania znajduje się za dominującym ciągiem chromosomu o większej wartości funkcji przystosowania.

Chromosom A

01/ 1	04/ 1	09/ 1	08/ 0	06/ 1	09/ 1	05/ 0	03/ 1	07/ 1	08/ 0

x – punkt krzyżowania

Chromosom B

02/ 1	05/ 1	04/ 1	09/ 1	05/ 0	07/ 1	08/ 0	01/ 1	03/ 1	06/ 0

Zakładając, że funkcja przystosowania drugiego chromosomu jest większa od funkcji przystosowania pierwszego chromosomu FB>FA przyjmujemy więc, że punktem krzyżowania jest miejsce zakończenia dominującego ciągu drugiego chromosomu -x.

3. Etap wymiany genów przeprowadzany pod warunkiem:

$$\left.\begin{array}{l} GenA(x+1) = \text{ciągłość } GenB(x) \\ \text{lub} \\ (GenA(x+1) \neq \text{ciągłość } GenA(x) \ \Lambda \ GenB(x+1) = \text{ciągłość } GenA(x)) \end{array}\right\} \text{FB>FA}$$

3.6

$$\left.\begin{array}{l} GenB(x+1) = \text{ciągłość } GenA(x) \\ \text{lub} \\ (GenB(x+1) \neq \text{ciągłość } GenB(x) \ \Lambda \ GenA(x+1) = \text{ciągłość } GenB(x)) \end{array}\right\} \text{FA>FB}$$

Potomek C

01/ 1	04/ 1	09/ 1	08/ 0	06/ 0	07/ 1	08/ 0	01/ 1	03/ 1	06/ 0

x

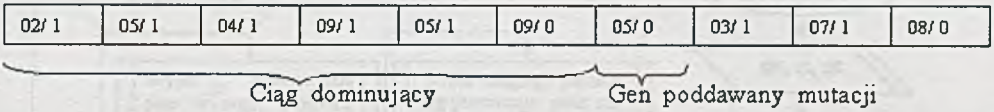
Potomek D

02/ 1	05/ 1	04/ 1	09/ 1	05/ 1	09/ 0	05/ 0	03/ 1	07/ 1	08/ 0

Funkcja przystosowania chromosomu potomka D uległa jeszcze powiększeniu w stosunku do rodzica B natomiast funkcja przystosowania potomka C nie uległa zmianie w stosunku do rodzica A (teoretycznie również mogła ulec zwiększeniu). Zakończenie ciągu dominującego w węźle oznaczonym 09/ 0 wynika z faktu, iż w żadnym z ciągów nie mogą się powtarzać już występujące w nim węzły (w tym przypadku węzeł o numerze 05).

Operacja mutacji z zadanym prawdopodobieństwem pm będzie przeprowadzona dla każdego chromosomów w miejscu gdzie znajduje się następny

gen po dominującym ciągu. Będzie to więc pierwsze miejsce z kodem zera lub jedynki w kodzie ciągłości po zerze oznaczającym zakończenie ciągu dominującego.



Mutacja z reguły będzie przeprowadzana po krzyżowaniu. Celem mutacji nie jest „wydostanie" się z lokalnego optimum lecz przyśpieszenie poszukiwania owego optimum, ponieważ w miejscu gdzie znajduje się wybrany gen mutacji możemy tylko wstawić kod węzła, który połączony jest z ostatnim węzłem ciągu dominującego. Tak więc w konkretnym przypadku będzie to węzeł o numerze 08. Jeśli takich węzłów będzie kilka to wylosowany będzie jeden z nich z prawdopodobieństwem proporcjonalnym do różnicy potencjałów (3.1).

$$\Delta\varphi(j,k) = \{\varphi(k) - \varphi(j)\} * kcon(j,k) \qquad 3.7$$

gdzie k = 1,..., m – numery węzłów sieci,
j – numer ostatniego węzła ciągu dominującego,
kcon(j,k) – współczynnik ciągłości z między węzłem j oraz k.

Po mutacji przeprowadzamy kompensację występujących po sobie tych samych węzłów oraz uzupełnienie pojawiających się w końcu chromosomu „wakatów" genów. Uzupełnienie będzie generowane zgodnie z zasadą istnienia powiązania z węzłem poprzednim i proporcjonalnie do różnicy potencjałów (3.7).

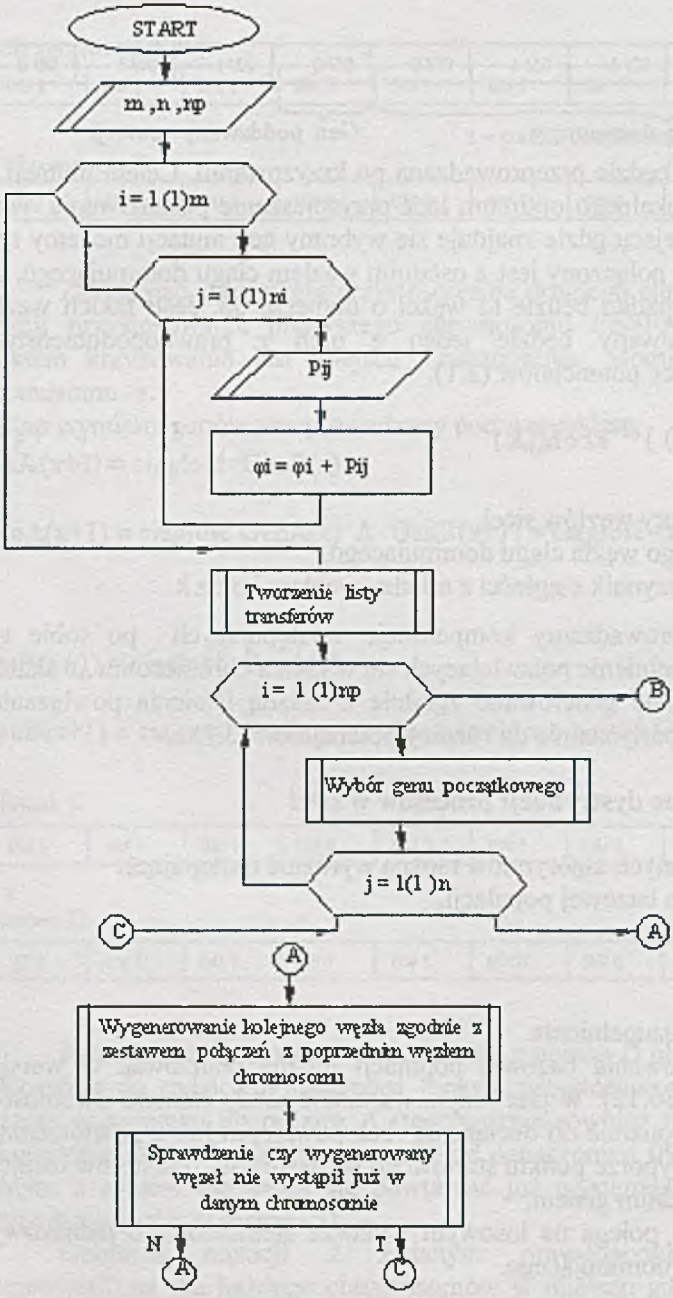
3.4 Algorytmy genetyczne dystrybucji procesów w sieci

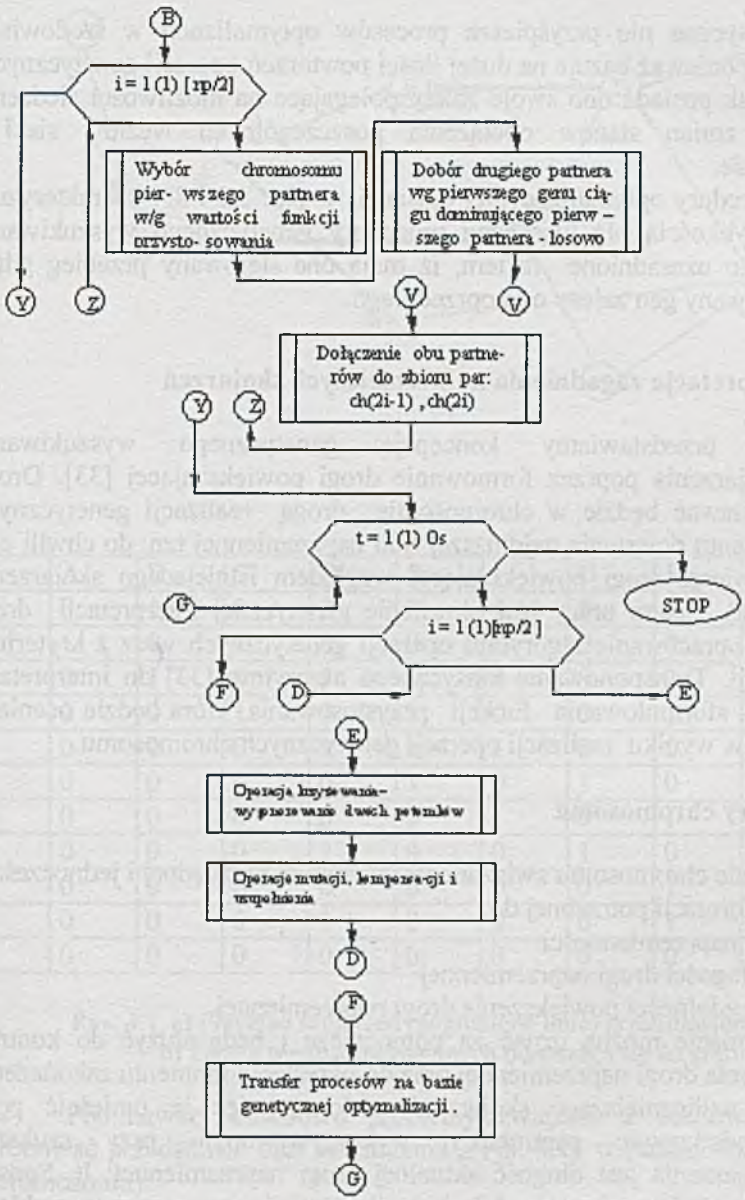
Wśród proponowanych algorytmów można wyróżnić następujące:
- algorytm do generowania bazowej populacji,
- algorytm doboru partnerów,
- algorytm krzyżowania,
- algorytm mutacji,
- algorytm kompensacji i uzupełnienia.

Algorytmy generowania bazowej populacji można realizować w wersjach niezależnej i sterowanej [10,12]. W pierwszym wariancie każdy element chromosomu będzie losowany proporcjonalnie do obciążenia bez powtórzeń lub z powtórzeniami. W drugim wariancie po wyborze punktu startowego na bazie listy transferów losujemy węzły połączone z poprzednim genem.

Algorytm selekcji polega na losowym wyborze elementów o jednakowych węzłach startowych ciągu dominującego.

Sterowanie genetycznymi procedurami ilustruje poniższy schemat:





Rys. 3.5. Algorytm zastosowania genetycznych narzędzi do optymalizacji transferu procesów w środowisku rozproszonym

Wnioski

1. Śledzenie genetyczne nie przyśpiesza procesów optymalizacji w środowisku rozproszonym, ponieważ bazuje na dużej ilości powtórzeń operacji genetycznych. Nie mniej jednak posiada ono swoje zalety polegające na możliwości śledzenia dynamicznych zmian stanów obciążenia poszczególnych węzłów sieci i reagowania na nie.
2. Zbieżność procedury optymalizacyjnej opisanej w pkt. 3.3 i 3.4 charakteryzuje się większą szybkością niż klasyczne procedury genetycznego wyszukiwania optimum. Jest to uzasadnione faktem, iż mają one sterowany przebieg gdzie aktualnie formowany gen zależy od poprzedniego.

4.Genetyczna interpretacja zagadnienia najliczniejszych skojarzeń

W pracy przedstawiamy koncepcję genetycznego wyszukiwania najliczniejszego skojarzenia poprzez formowanie drogi powiększającej [33]. Droga powiększająca powstawać będzie w chromosomie drogą realizacji genetycznych operacji, aż do momentu powstania najdłuższej linii naprzemiennej tzn. do chwili gdy graf nie będzie zawierał drogi powiększającej względem istniejącego skojarzenia (twierdzenie Berge'a). Celem pracy jest określenie genetycznej interpretacji drogi naprzemiennej oraz opracowanie algorytmu operacji genetycznych wraz z kryterium zakończenia ewolucji. Transponowanie klasycznego algorytmu [33] do interpretacji genetycznej wymaga sformułowania funkcji przystosowania, która będzie oceniana na bazie powstałego w wyniku realizacji operacji genetycznych chromosomu.

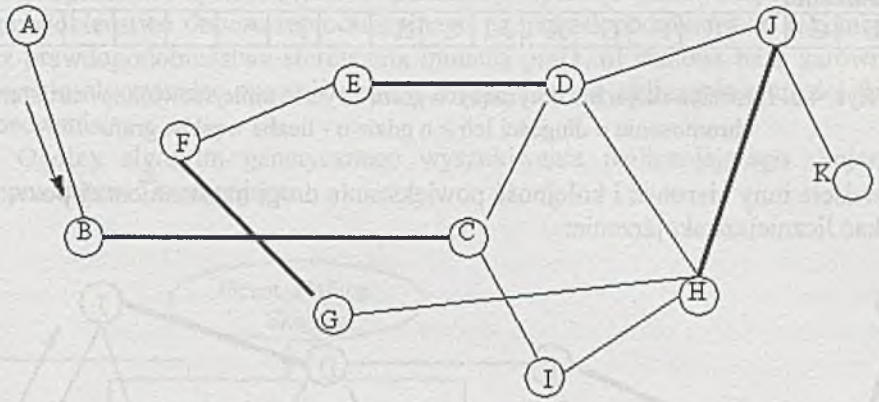
4.1 Projekt struktury chromosomu

Konstruowanie chromosomu związane jest wyborem niezbędnej i jednocześnie minimalnej ilości informacji potrzebnej do:

- sprawdzenia naprzemienności
- obliczenie długości drogi naprzemiennej
- sprawdzenie zdolności powiększenia drogi naprzemiennej.

Dwie ostatnie informacje można uznać za pomocnicze i będą służyć do kontroli przebiegu powiększania drogi naprzemiennej oraz do określenia momentu zakończenia procesu tworzenia najliczniejszego skojarzenia. Można więc je umieścić poza chromosomem. Dodatkowym parametrem wykorzystywanym przy szukaniu najliczniejszego skojarzenia jest długość aktualnej drogi naprzemiennej: lt. Sposób zapisu powstającego skojarzenia wygodnie jest zilustrować przy pomocy przykładu opisującego graf.

a)



b)

	A	B	C	D	E	F	G	H	I	J	K
A	0	1	0	0	0	0	0	0	0	0	0
B	1	0	1	0	0	0	0	0	0	0	0
C	0	1	0	1	0	0	0	0	0	0	0
D	0	0	1	0	1	0	0	1	0	1	0
E	0	0	0	1	0	1	0	0	0	0	0
F	0	0	0	0	1	0	1	0	0	0	0
G	0	0	0	0	0	1	0	1	0	0	0
H	0	0	0	1	0	0	1	0	1	1	0
I	0	0	0	0	0	0	0	1	0	0	0
J	0	0	0	1	0	0	0	1	0	0	1
K	0	0	0	0	0	0	0	0	0	1	0

Rys. 4.1. a) Przykład skojarzeń (pogrubione linie) przedstawionych na grafie,
b) Tabela węzłów incydentnych odnosząca się do grafu a)

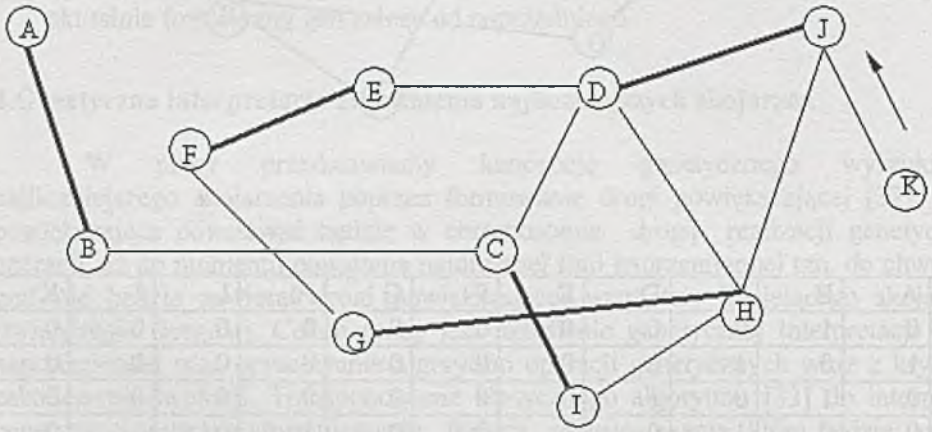
Pominąwszy chwilowo problemy związane z cechowaniem [4] możemy spróbować przedstawić opis skojarzenia z rys. 4.1a w postaci informacji genetycznej (chromosomu):

Chromosom

A	B	C	D	E	F	G	H	J	*	*	*	*	*	*	*	*	*	*

Rys. 4.2. Ilustracja skojarzeń dotyczących grafu z rys.1a umiejscowiona w strukturze chromosomu o długości lch ≥ n gdzie n - liczba węzłów grafu

Oczywiście inny kierunek i kolejność powiększania drogi naprzemiennej pozwoli nam uzyskać liczniejsze skojarzenie:



Chromosom

K	J	D	E	F	G	H	I	C	B	A	*	*	*	*	*	*	*	*	*

Rys. 4.3. Droga naprzemienna największej długości (bez możliwości powiększenia) oraz chromosom ją opisujący (gdzie * oznacza dowolny węzeł)

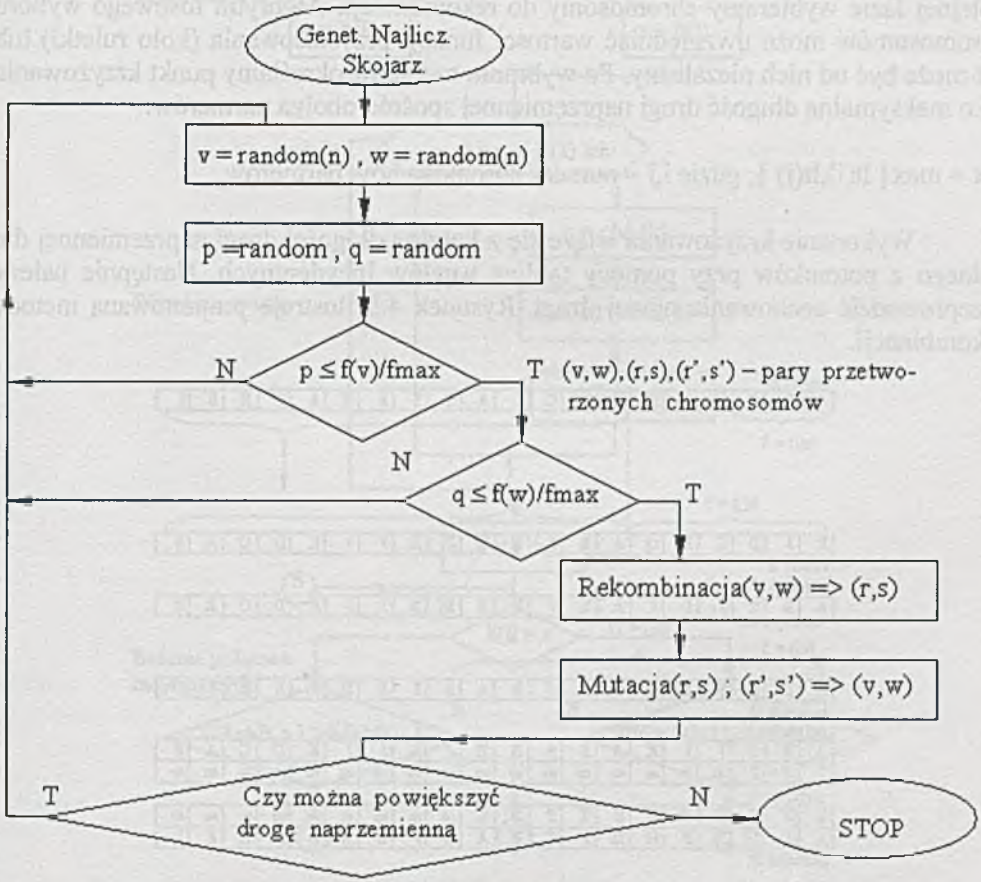
4.2 Parametry i algorytm generacji drogi naprzemiennej

Do tworzenia drogi naprzemiennej stosowana jest metoda cechowania [33]. Znane algorytmy [33] bazują na tablicy węzłów incydentnych i podstawach kielichów (cykli naprzemiennych nieparzystej długości). Dane te można zebrać w następujących tablicach i zmiennych:

WI[1..n,1..n] - macierz węzłów incydentnych,
MC[1..n,1..lch] - wektor naprzemienności, ich – wielkość populacji,
lt[1..i] - wektor długości drogi naprzemiennej dla i-go chromosomu,
S[1..n] - punkty startowe,
I - numer chromosomu, f(i)-funkcja przystosowania, fmax –aktualna maksymalna wartość funkcji przystosowania.

Uzupełnienie tych danych o parametry operacji genetycznych takich jak prawdopodobieństwo doboru reprodukcyjnego pr, prawdopodobieństwo rekombinacji pk oraz prawdopodobieństwa sterowania mutacją pm[1..n] stanowi bazę zarówno do opracowania algorytmów operacji genetycznych jak i do obliczania wartości funkcji przystosowania.

Ogólny algorytm genetycznego wyszukiwania najliczniejszego skojarzenia można przedstawić następująco:



Rys. 4.4. Algorytm genetycznego wyszukiwania najdłuższej drogi naprzemiennej

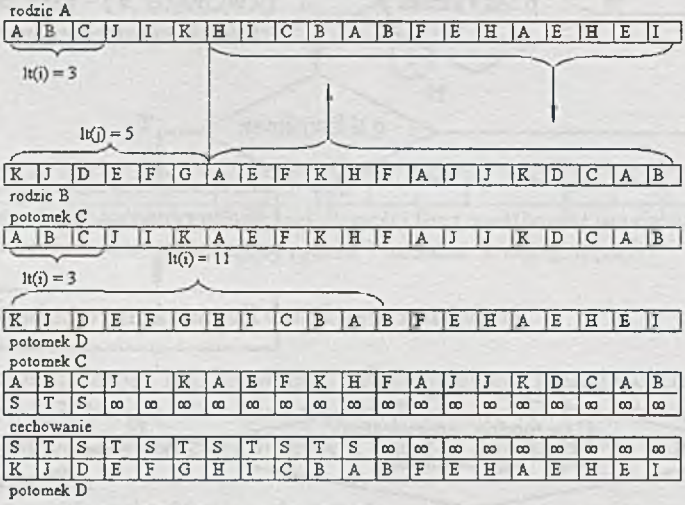
4.3 Algorytmy operacji genetycznych

Algorytm rekombinacji należy dostosować do zadania wyszukiwania drogi naprzemiennej. W tym celu należy wybrać wierzchołek startowy. Jeśli istnieje kilka

takich węzłów dla każdego tworzonego chromosomu losowo wybieramy jeden ze wskazanych punktów startowych S[i] (oczywiście z prawdopodobieństwem ps = 1/ls, gdzie ls- ilość punktów startowych). Pozostałe elementy chromosomu (geny węzły) możemy dobrać losowo ze wszystkich pozostałych węzłów lub spośród połączeń incydentnych (rys.4.1). W operacji krzyżowania wybieramy punkt krzyżowania. Najefektywniejszym rozwiązaniem jest przyjęcie punktu krzyżowania [1,12] miejsce zakończenia najdłuższej z aktualnych dróg naprzemiennych. Etapem poprzedzającym rekombinację będzie więc ocena długości początkowej drogi naprzemiennej lt(i). W kolejnej fazie wybieramy chromosomy do rekombinacji. Algorytm losowego wyboru chromosomów może uwzględniać wartości funkcji przystosowania (koło ruletki) lub też może być od nich niezależny. Po wybraniu rodziców określamy punkt krzyżowania jako maksymalną długość drogi naprzemiennej spośród obojga partnerów:

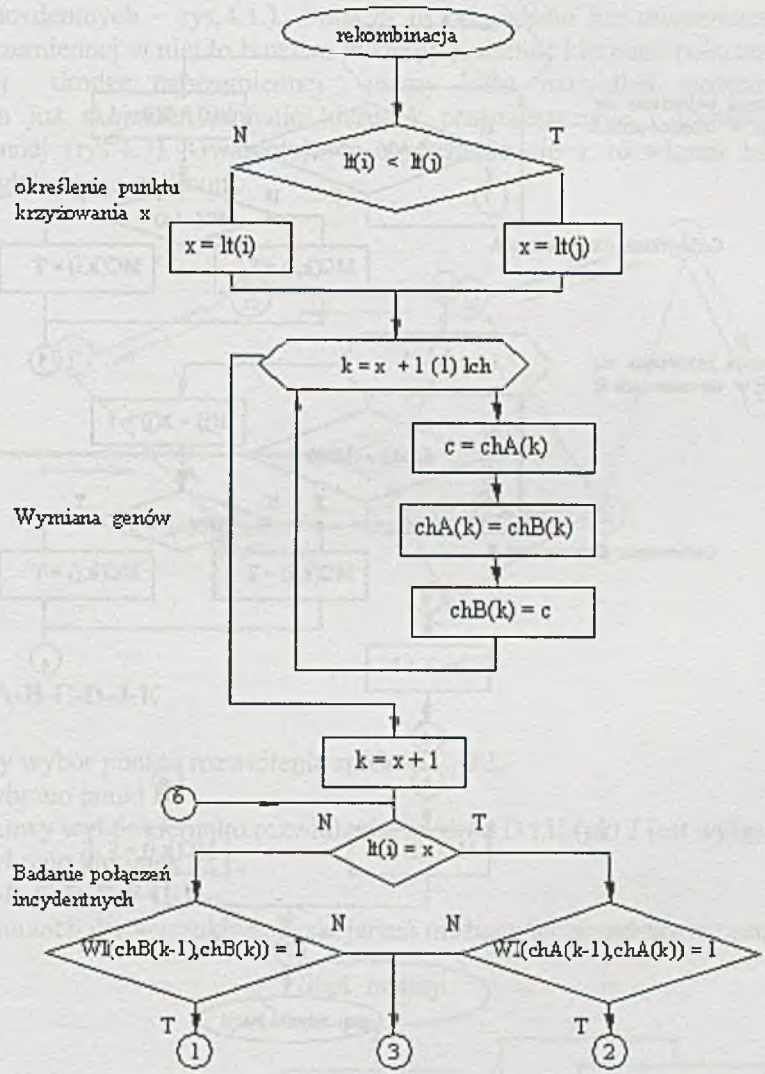
x = max{ lt(i),lt(j) }, gdzie i,j – numery chromosomów partnerów,

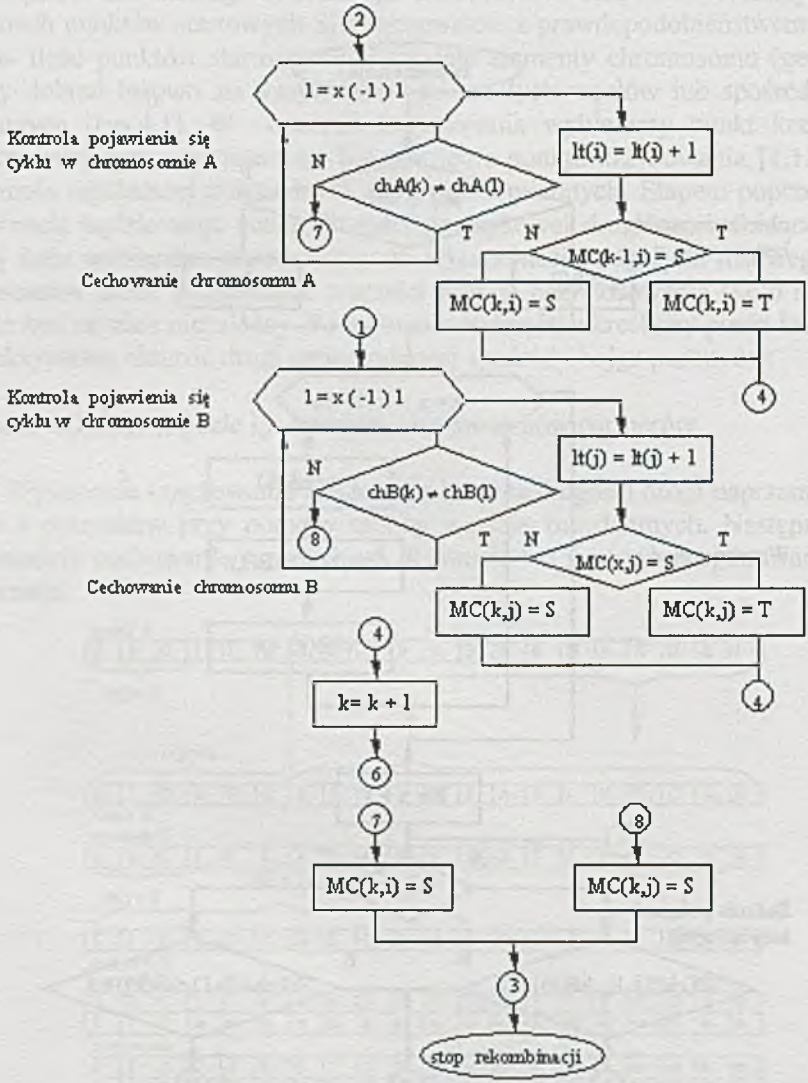
Wykonanie krzyżowania wiąże się z korektą długości drogi naprzemiennej dla jednego z potomków przy pomocy tablicy węzłów incydentnych. Następnie należy przeprowadzić cechowanie nowej drogi. Rysunek 4.5 ilustruje proponowaną metodę rekombinacji.



Rys. 4.5. Koncepcja realizacji operacji krzyżowania (rekombinacji)

Algorytm krzyżowania można zilustrować w następujący sposób:

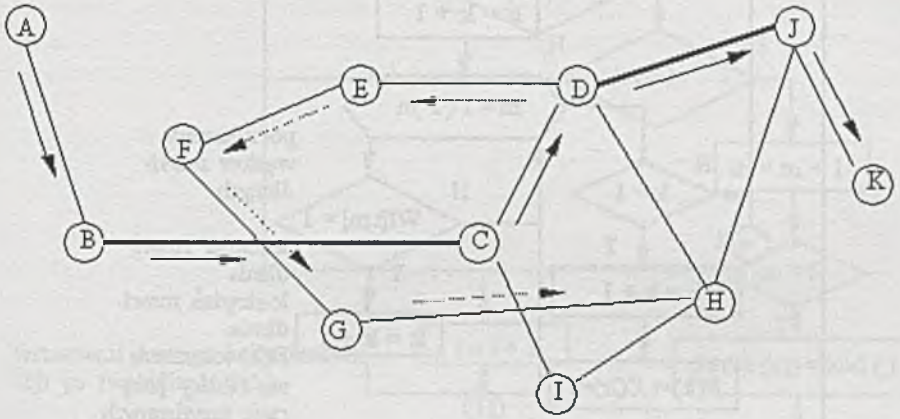




Rys. 4.6.Algortm krzyżowania z następującymi danymi: chromosomy rodziców (chA(*),chB(*)), numery tych chromosomów (i,j), aktualne długości dróg naprzemiennych (lt(i),lt(j)), aktualne cechowanie dróg (MC(*,i),MC(*,j)) oraz punkt krzyżowania (x)

Potomkowie są automatycznie włączani do populacji, a rodzice są z niej usuwani, ponieważ w wyniku działania powyższej operacji rekombinacji nie możemy uzyskać gorszych rezultatów niż w populacji bazowej [10].

Koncepcja operacji mutacji oparta jest na modyfikacji połączeń z węzłami, które dysponują alternatywną możliwością kontynuacji komunikacji (tzn. mają więcej węzłów incydentnych – rys.4.1.). Dotyczy to oczywiście już utworzonej, aktualnej drogi naprzemiennej w niej to bowiem możemy zmieniać kierunek połączeń. Jeżeli na utworzonej drodze naprzemiennej mamy kilka rozwidleń możemy kosztem powstałych już skojarzeń zmienić kierunek przemieszczania i formowania drogi naprzemiennej (rys.4.7). Gwarantuje to wydostanie się z rozwiązań lokalnych w kierunku globalnego optimum.



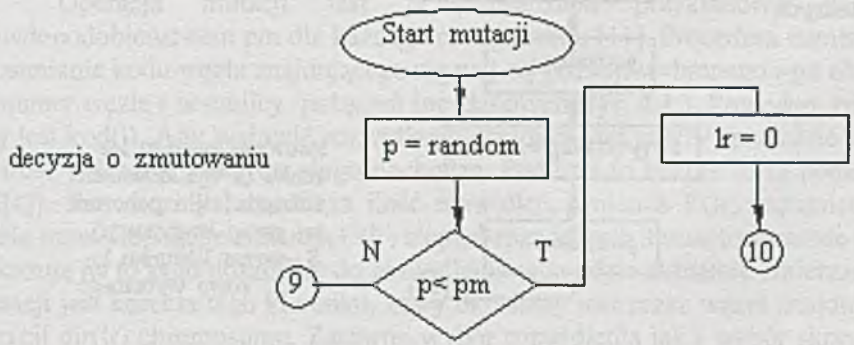
droga 1 – **A-B-C-D-J-K**

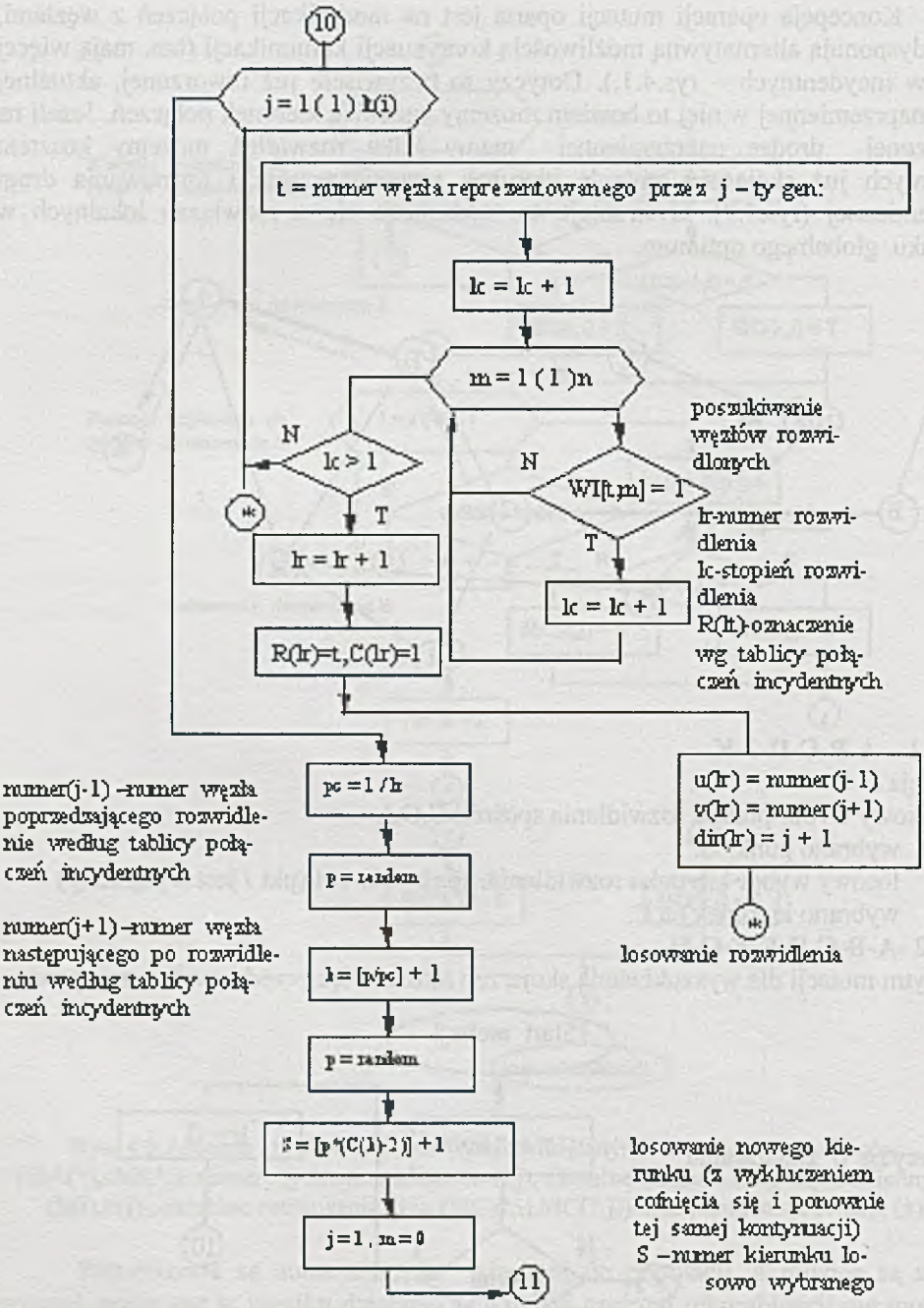
mutacja

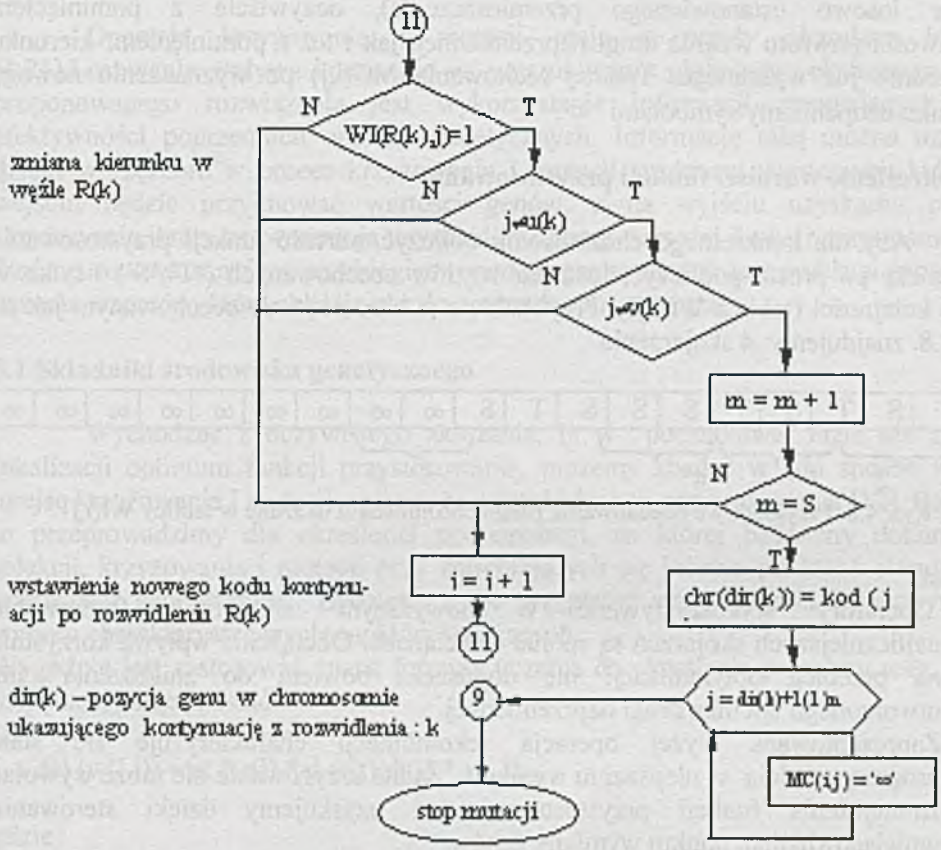
- losowy wybór punktu rozwidlenia spośród C,D,J,
  - wybrano punkt D,
  - losowy wybór kierunku rozwidlenia, spośród D i H (pkt J jest wyłączony)
  - wybrano kierunek na E,

droga2 **-A-B-C-D-E-F-G-H-...**

Algorytm mutacji dla wyszukiwania skojarzeń można więc przedstawić następująco:




10
j = 1 (1) k(i)
t = numer węzła reprezentowanego przez j – ty gen:
lc = lc + 1
m = 1 (1) n
N
lc > 1
T
lr = lr + 1
R(lr)=t, C(lr)=1
N
W[t,m] = 1
T
lc = lc + 1
poszukiwanie węzłów rozwidlonych
lr-numer rozwidlenia
lc-stopień rozwidlenia
R(lt)-oznaczenie wg tablicy połączeń incydentnych
pc = 1 / lr
p = random
l = [p*pc] + 1
p = random
S = [p*(C(l)-2)] + 1
j=1 , m=0
11
u(lr) = numer(j-1)
v(lr) = numer(j+1)
dir(lr) = j + 1
losowanie rozwidlenia
losowanie nowego kierunku (z wykluczeniem cofnięcia się i ponownie tej samej kontynuacji)
S –numer kierunku losowo wybranego
numer(j-1) –numer węzła poprzedzającego rozwidlenie według tablicy połączeń incydentnych
numer(j+1) –numer węzła następującego po rozwidleniu według tablicy połączeń incydentnych



Rys. 4.7. Algorytm mutacji dla wyszukiwania najliczniejszych skojarzeń

Operacja mutacji jest przeprowadzana przykładowo z zadanym prawdopodobieństwem pm dla każdego chromosomu [11]. Procedura numer (j) polega na zamianie kodu węzła znajdującego się na j-tej pozycji wybranego i-go chromosomu na numer węzła t w tablicy połączeń incydentnych (rys. 4.1.). Procedurą odwrotną do niej jest kod(j). Aby wyłowić rozwidlenia każdy węzeł należy sprawdzić ze względu na ilość połączeń, które do niego dochodzą. Podlicza to licznik lc za pomocą tablicy WI[i,j]. Inny licznik lr podlicza ilość rozwidleń, zmienna R(lr) zapamiętuje numer węzła rozwidlonego, a zmienna C(lr) stopień rozwidlenia. Ponadto zmienne u(lr) i v(lr) wskazują na to skąd doszliśmy do rozwidlenia oraz gdzie aktualnie zmierzamy. Celem mutacji jest korekta tego kierunku, który określany jest przez węzeł znajdujący się na pozycji dir(lr) chromosomu. Zarówno wybór rozwidlenia jak i wybór skorygowanego

kierunku przemieszczenia dokonywany jest losowo (k –losowo wybrany węzeł, S numer losowo ustanowionego przemieszczenia), oczywiście z pominięciem możliwości powrotu wzdłuż drogi naprzemiennej, jak i też z pominięciem kierunku poprzednio już wybranego. Tablicę cechowania MC(i,j) po wyznaczeniu nowego kierunku uzupełniamy symbolami '∞'.

4.4 Określenie wartości funkcji przystosowania

Aby dla konkretnego chromosomu obliczyć wartość funkcji przystosowania f(i) należy po prostu podliczyć ilość par węzłów ocechowanych ('T','S') i tylko w takiej kolejności (tablica WI(i,j)). Przykładowo w chromosomie ocechowanym jak na rys. 4.8. znajdujemy 4 skojarzenia.

S	T	S	T	S	T	S	S	S	T	S	∞	∞	∞	∞	∞	∞	∞	∞	∞
	1		2		3		4												

Rys. 4.8. Przykładowe ocechowanie i-tego chromosomu ukazane w tablicy WI(i,j)

Wnioski

1. Operatory wykorzystywane w powyższym algorytmie formowania najliczniejszych skojarzeń są spójne i obciążone. Obciążenie wpływa korzystnie na przebieg optymalizacji nie dopuszcza bowiem do „niszczenia" już utworzonego odcinka drogi naprzemiennej.
2. Zaprezentowana wyżej operacja rekombinacji charakteryzuje się stałą progresywnością w ulepszaniu wyników. Żadne krzyżowanie nie może wywołać zmniejszenia funkcji przystosowania co uzyskujemy dzięki sterowaniu umiejscowieniem punktu wymiany genów.
3. Operacja mutacji jest niezbędna, ponieważ pozwala zrezygnować z utartego szlaku w celu poszukiwania być może bardziej optymalnego rozwiązania. W jej wyniku część uformowanej drogi naprzemiennej zostaje odcięta oraz wybrany zostaje nowy kierunek przemieszczenia.
4. Aby nie stracić części drogi zmierzającej do rozwiązania optymalnego celowym byłoby zmniejszenie wpływu mutacji po osiągnięciu określonego poziomu funkcji przystosowania. Można to zrobić na kilka sposobów na przykład zmniejszając prawdopodobieństwo realizacji mutacji pm lub tworząc zbiór podpopulacji zachowujący aktualne najlepsze rozwiązania.
5. W niektórych przypadkach, na przykład kiedy chromosomy rozpoczynają się od różnych węzłów, celowym może się okazać zastosowanie operacji genetycznej inwersji. Przyśpieszyć to może formowanie drogi naprzemiennej.

5. Zastosowanie struktury neuronowej do wspomagania operacji genetycznych

Operacje krzyżowania i mutacji mają z reguły charakter losowy [4,21].Losowanie wpływa korzystnie na wyszukiwanie globalnego ekstremum. Ideą proponowanego rozwiązania jest wykorzystanie informacji stanowiących ślad efektywności poprzednich operacji genetycznych. Informację taką można uzyskać dzięki wprzężeniu w proces krzyżowania i mutacji struktury neuronowej, która na wejściu będzie przyjmować wartości genów, a na wyjściu uzyskamy punkty krzyżowania, które zagwarantują uzyskanie większej wartości funkcji przystosowania. Sieć neuronowa sterująca przebiegiem operacji genetycznych jest poddana procesowi uczenia wzorców, które zbliżają nas do optymalnych rozwiązań.

5.1 Składniki środowiska genetycznego

Wychodząc z oczywistego założenia, iż w początkowej fazie nie znamy lokalizacji optimum funkcji przystosowania, możemy zbadać w jaki sposób wybór miejsc krzyżowania i mutacji wpływa na wartość funkcji przystosowania [12]. Badanie to przeprowadzimy dla określonej podpopulacji, na której będziemy dokonywać selekcji, krzyżowania i mutacji przy zmieniających się losowo punktach sterujących operacjami genetycznymi. Będziemy więc przeplatać etapy uczenia i dokonywania wyboru charakterystycznych punktów sterujących.
Wygodnie jest zastosować znane formuły uczenia do określania przyrostu wag, które mają następującą postać:

$$del\,(w(i,j)) = et * r(j) * d\,\psi(t)/dt(j) * x(i,j), \qquad 5.1$$

gdzie
et – wskaźnik uczenia,

$$r(j) = s(j) - y(j)\ ;\ y(j) = \psi\left(\sum_{i=1}^{n} \{w(i,j) * x(i,j)\}\right);$$

n – ilość neuronów w warstwie sieci,

$$t = \sum_{i=1}^{n} \{w(i,j) * x(i,j)\};$$

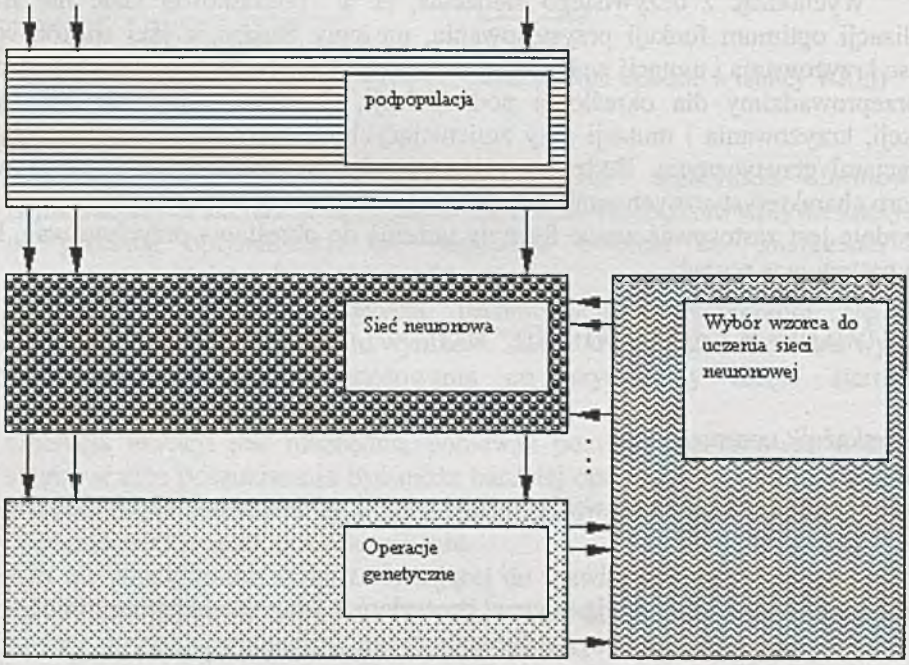
x(i,j) – sygnał przesyłany z j-tego do i-tego neuronu,
w(i,j) – waga połączenia synaptycznego,
ψ - funkcja odpowiedzi.

W części genetycznej możemy się posłużyć jednym typem operacji genetycznej na przykład krzyżowaniem lub krzyżowaniem połączonym z mutacją. Realizacja tych operacji ma charakter losowy jako że zarówno krzyżowanie jak i mutacja przeprowadzane są z pewnym, zadanym prawdopodobieństwem. Ponadto

element „losowości" do wyznaczania punktów krzyżowania. W tej sytuacji krzyżowanie ma również charakter wymienialny i „podwójnie losowy".

5.2 Dostosowanie charakteru operacji genetycznych do specyfiki funkcjonowania sieci neuronowej

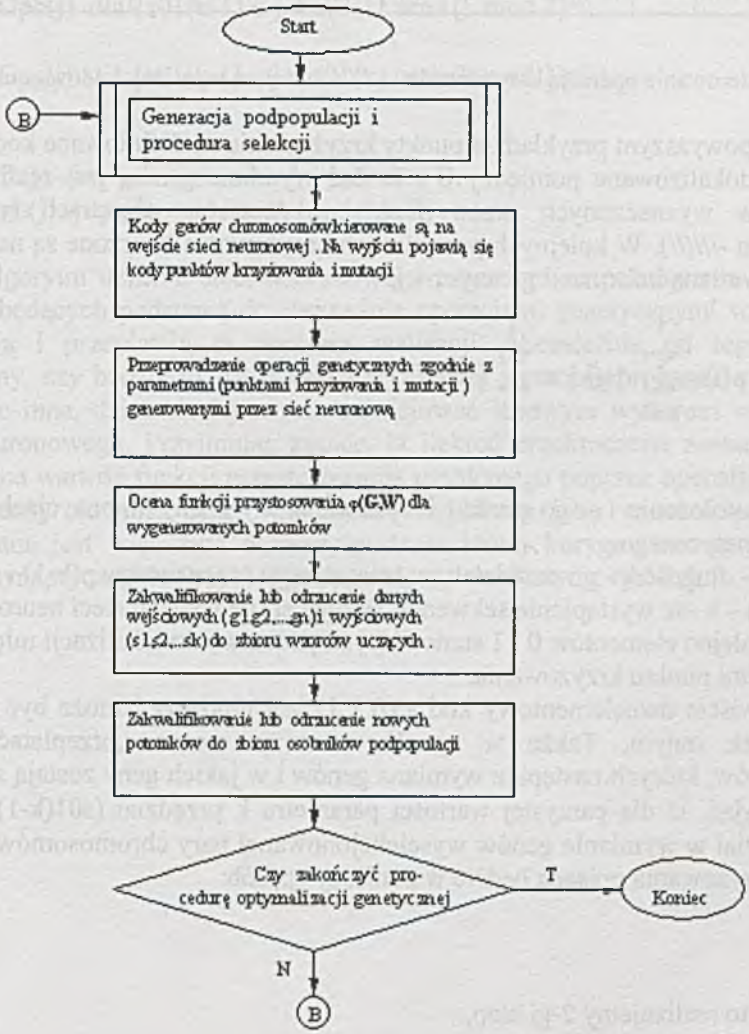
Zbiorem informacji wejściowej dla struktury neuronowej będą wartości (kody) poszczególnych genów poszczególnych chromosomów. Wyjściowe dane sieci neuronowej będą elementami sterującymi parametrami operacji genetycznych. W zależności od typu zadania i interpretacji genetycznej ilość danych wyjściowych sterującego układu neuronowego może odpowiadać ilości genów w chromosomie bądź też stanowić jego część (tylko tych, które mogą stanowić miejsca krzyżowania lub mutacji). Schemat ideowy proponowanego modelu może być przedstawiony jak na rys. 5.1.



Rys. 5.1. Schemat ideowy modelu sterownia operacjami genetycznymi za pomocy sieci neuronowej

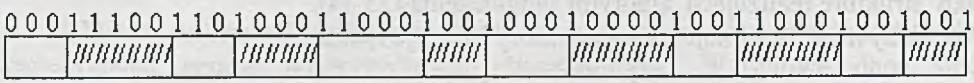
W proponowanym modelu znajduje się blok wyboru wzorca do uczenia sieci neuronowej, w którym kumulowane są dane wejściowe (g1,g2,...,gn) i wyjściowe (s1,s2,...sk) wybrane dla najwyższych, z zadaną tolerancją δ(m), wartości funkcji

przystosowania φ(G,W) (gdzie m – objętość podpopulacji, G – wektor danych wejściowych, W – macierz wag w połączeniach międzyneuronowych).Elementy macierzy wag międzyneuronowych zmieniają się jedynie w procesie uczenia sieci. Mając na uwadze to, iż w początkowej fazie pracy sieci neuronowej nie znamy wartości wzorców wykorzystywanych w procesie uczenia możemy sklasyfikować ją jako strukturę realizującą algorytm samouczenia [23,34].
Powyższy model realizuje algorytm, który ilustruje rys. 5.2.



Rys. 5.2. Algorytm działania modelu optymalizacji genetycznej sterowanej siecią neuronową

Operacja krzyżowania będzie miała charakter zmienno-punktowy tzn. miejsca wymiany wartości kodów ulegają przemieszczeniu w każdej kolejnej operacji. Na wyjściach sieci neuronowej w kolejnych etapach rozpoznawania pojawiają się kody binarne, które definiują punkty sterowania operacjami genetycznymi (krzyżowaniem lub mutacją).



Rys. 5.3. Sterowanie operacją krzyżowania - ////// - miejsca wymiany kodów genetycznych

Na powyższym przykładzie punkty krzyżowania są definiowane kodem 01 i są dokladnie zlokalizowane pomiędzy 0 i 1. Zaś wymiana genów jest realizowana na przemian w wyznaczonych przez punkty krzyżowania obszarach (oznacznych przykladowo -//////). W kolejnych procedurach krzyżowania tworzone są nowe punkty sterujące wymianą informacji genetycznej.

$$p(i) = \sum_{k=1}^{i} ds(k) = \sum_{k=1}^{i} (s01(k) - s01(k-1)) \qquad 5.2$$

gdzie

p(i) – położenie i – go punktu krzyżowania wyrażony w pozycjach kodu genetycznego,

ds(k) – długość k - go przedziału wyznaczonego przez punkty p(k-1) i p(k),

s01(k) – k –te wystąpienie sekwencji sterującej (na wyjściu sieci neuronowej) kolejno elementów 0 i 1 stanowiących podstawę do lokalizacji między nimi punktu krzyżowania.

Oczywiście dwuelementowy kod - (0 i 1) jest umowny i może być zastąpiony jakimkolwiek innym. Także w sposób umowny można „przeplatać" obszary chromosomów, których następuje wymiana genów i w jakich geny zostają zachowane. Można przyjąć, iż dla parzystej wartości parametru k przedział (s01(k-1) ÷ s01(k)) weźmie udział w wymianie genów wyselekcjonowanej pary chromosomów. Tak więc operacja krzyżowania opisana będzie w poniższy sposób:

1-szy etap

jeśli $\xi \leq pk$ to realizujemy 2-gi etap,

gdzie

ξ - liczba losowa z przedziału (0,1),

pk – prawdopodobieństwo krzyżowania.

2-gi etap

$Gda(j) = Gpa(j)$ dla $\{s01(k-1) \leq j < s01(k)$ oraz $(k \bmod 2)=1\}$,

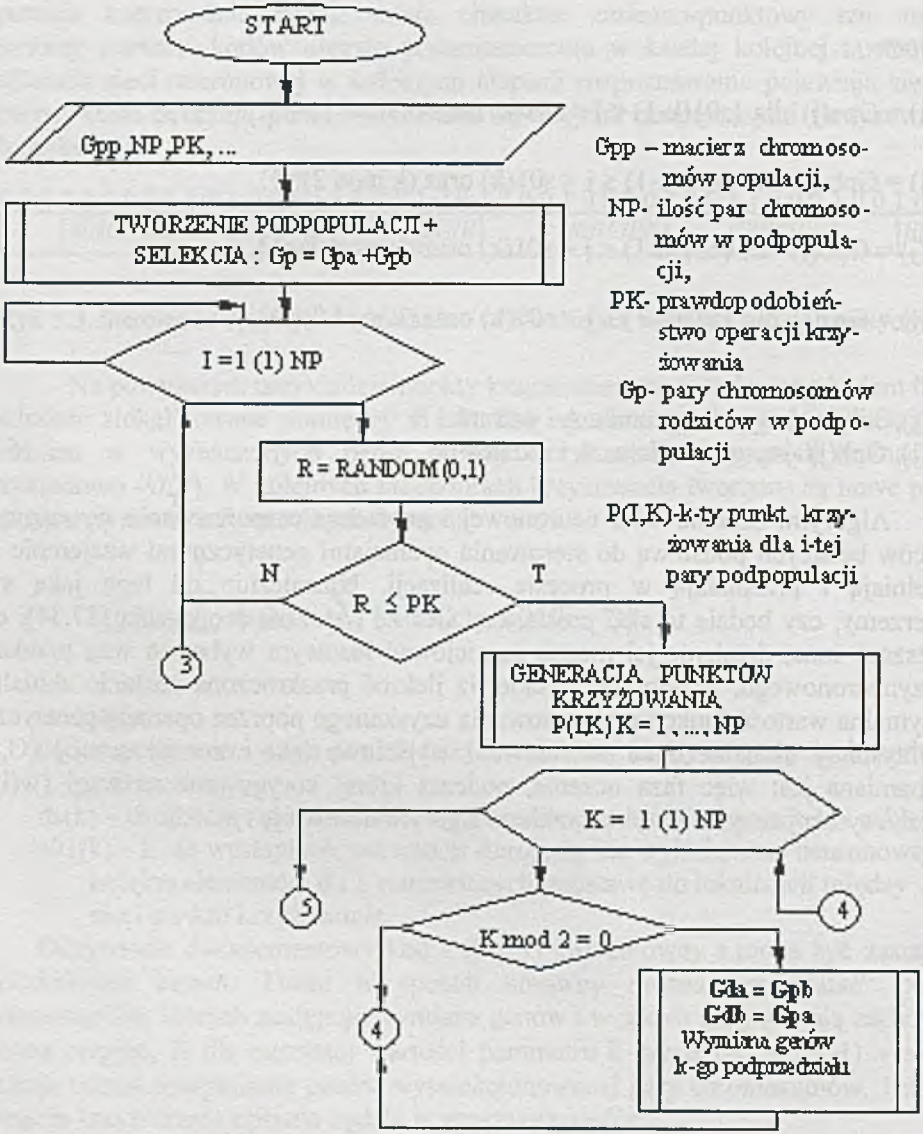
$Gda(j) = Gpb(j)$ dla $\{s01(k-1) \leq j < s01(k)$ oraz $(k \bmod 2)=0\}$,

$Gdb(j) = Gpb(j)$ dla $\{s01(k-1) \leq j < s01(k)$ oraz $(k \bmod 2)=1\}$,

$Gdb(j) = Gpa(j)$ dla $\{s01(k-1) \leq j < s01(k)$ oraz $(k \bmod 2)=0\}$,

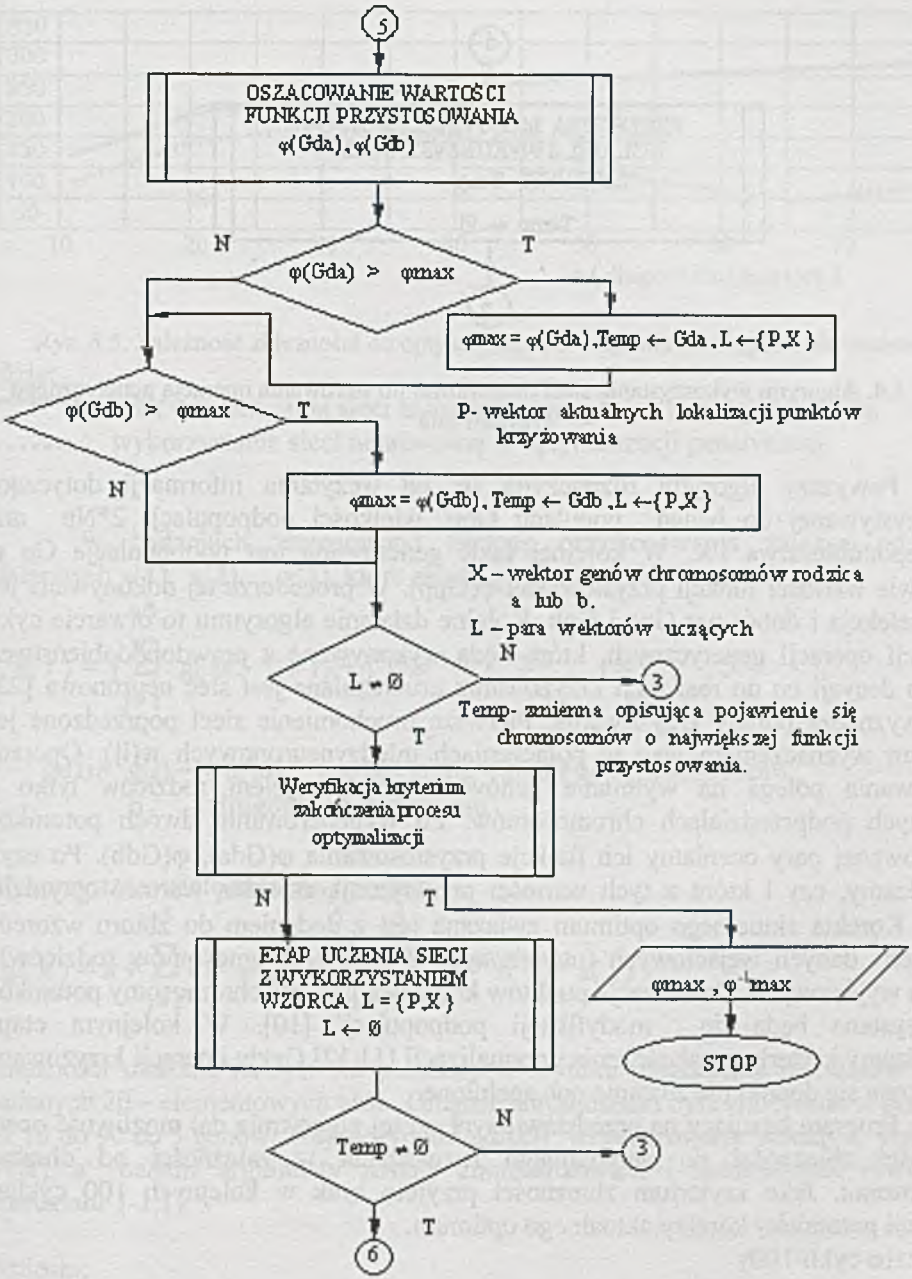
gdzie
Gda(j),Gdb(j) –j-ty gen potomka A i potomka B,
Gpa(j), Gpb(j) -j-ty gen rodzica A i rodzica B. (5.3)

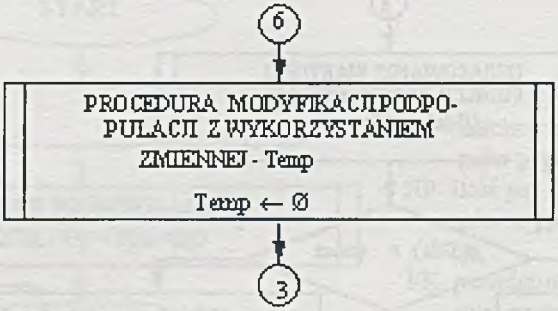
Algorytm uczenia sieci neuronowej i procedura rozpoznawania wyuczonych wzorców będących podstawą do sterowania operacjami genetycznymi wzajemnie się uzupełniają i przeplatają w procesie realizacji. Niezależnie od tego jaką sieć wybierzemy, czy będzie to sieć prosta, czy sieć ze wsteczną propagacją [27,34], czy też jeszcze inna, działanie jej można zainicjować losowym wyborem wag przekazu międzyneuronowego. Przyjmując zasadę, iż ilekroć przekroczona zostanie aktualnie maksymalna wartość funkcji przystosowania uzyskanego poprzez operację genetyczną wykorzystamy aktualne dane wejściowe i wyjściowe jako wzorzec uczący (G,S). Uruchamiana jest więc faza uczenia, podczas której korygowane są wagi (w(i,j)) połączeń synaptycznych. Przykład takiego algorytmu ilustruje rys. 5.3.



Gpp – macierz chromosomów populacji,
NP- ilość par chromosomów w podpopulacji,
PK- prawdopodobieństwo operacji krzyżowania
Gp- pary chromosomów rodziców w podpopulacji

P(I,K)-k-ty punkt krzyżowania dla i-tej pary podpopulacji

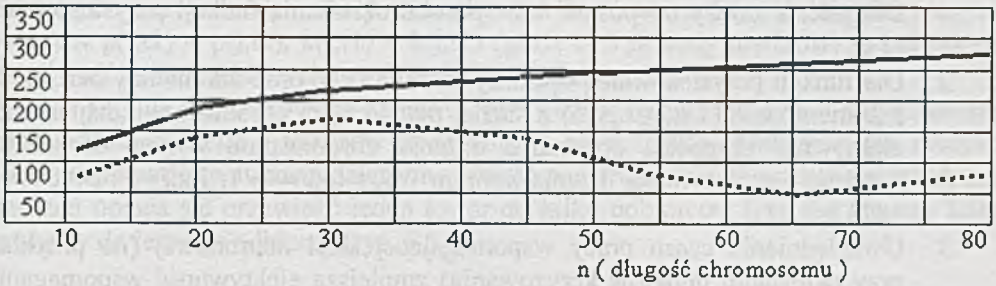

5
OSZACOWANIE WARTOŚCI FUNKCJI PRZYSTOSOWANIA φ(Gda), φ(Gdb)
N
T
φ(Gda) > φmax
φmax = φ(Gda), Temp ← Gda, L ←{P,X}
φ(Gdb) > φmax
T
N
P- wektor aktualnych lokalizacji punktów krzyżowania
φmax = φ(Gdb), Temp ← Gdb, L ←{P,X}
X – wektor genów chromosomów rodzica a lub b.
L – para wektorów uczących
N
L ≠ Ø
3
T
Temp- zmienna opisująca pojawienie się chromosomów o największej funkcji przystosowania.
Weryfikacja kryterium zakończenia procesu optymalizacji
N
T
ETAP UCZENIA SIECI Z WYKORZYSTANIEM WZORCA L = {P,X}
L ← Ø
φmax , φ max
STOP
Temp ≠ Ø
N
3
T
6



Rys. 5.4. Algorytm wykorzystania sieci neuronowej do sterowania operacją genetycznego krzyżowania

Powyższy algorytm rozpoczyna się od wczytania informacji dotyczącej wykorzystywanej do badań populacji Gpp, wielkości podpopulacji 2*Np oraz prawdopodobieństwa PK. W kolejnej fazie generowana jest podpopulacja Gp na podstawie wartości funkcji przystosowań φ(Gpp). W procedurze tej dokonywana jest także selekcja i dobór par Gpa i Gpb. Kolejne działanie algorytmu to otwarcie cyklu realizacji operacji genetycznych, które będą wykonywane z prawdopodobieństwem PK. Po decyzji co do realizacji krzyżowania uruchamiana jest sieć neuronowa [23], która wyznacza punkty krzyżowania. Pierwsze uruchomienie sieci poprzedzone jest losowym wyznaczeniem wag w połączeniach międzyneuronowych w(ij). Operacja krzyżowania polega na wymianie genów między obojgiem rodziców tylko w parzystych podprzedziałach chromosomów. Po wygenerowaniu dwóch potomków skrzyżowanej pary oceniamy ich funkcje przystosowania φ(Gda), φ(Gdb). Po czym sprawdzamy, czy i które z tych wartości przekraczają aktualną wartość optymalną φmax. Korekta aktualnego optimum związana jest z dodaniem do zbioru wzorców uczących danych wejściowych (utworzonych z genów chromosomów rodziców) i danych wyjściowych (lokalizacja punktów krzyżowania), zaś chromosomy potomków wykorzystane będą do modyfikacji podpopulacji [10]. W kolejnym etapie sprawdzamy kryterium zakończenia optymalizacji [11,12].Cykle operacji krzyżowania powtarzają się dopóki nie zostanie ono spełnione.

Program bazujący na przedstawionym wyżej algorytmie dał możliwość oceny szybkości zbieżności do optymalnego rozwiązania w zależności od długości chromosomu. Jako kryterium zbieżności przyjęto brak w kolejnych 100 cyklach generacji potomków korekty aktualnego optimum.
lop (liczba cykli/100)



Rys. 5.5. Zależność zbieżności do optymalnego rozwiązania od długości chromosomu, gdzie:

——— brak wspomagania sieci neuronowej w optymalizacji genetycznej,

..... wykorzystanie sieci neuronowej w optymalizacji genetycznej.

W badaniach zastosowano funkcję przystosowania zależną od pięciu zmiennych x(1), x(2),..., x(5), które zależą genów chromosomu:

$$x(i) = \{ \sum_{j=1}^{n/5} g(n/5 *(i-1)+j)*2^{j-1} \} - 2^{j-2} \; ; \; i = 1, \ldots, 5$$

gdzie g(i) – wartość i-tego genu badanego chromosomu, 5.4
n - długość chromosomu.

Wykorzystany wielomian miał postać:

$$\varphi(x) = a + \sum_{i=1}^{5} bi * x(i) + \sum_{i=1}^{5} \sum_{j=1}^{5} cij * x(i) * x(j) \qquad 5.5$$

Zależności ukazane na rys. 5.5 powstały w wyniku uśrednienia rezultatów analizy badanych 20 – elementowych prób. Długość chromosomu była zmieniana w przedziale od 10 do 90 co 5 genów. Współczynniki a,bi,cij wygenerowane zostały z przedziału [-1,1], a badane argumenty zostały znormalizowane i sprowadzone również do przedziału [-1,1].

Wnioski:

1. Wyniki badań wskazują na istnienie optymalnej długości chromosomu, dla której wykorzystanie wspomagania neuronowego jest najbardziej efektywne.

Długość ta zależy od postaci lub sposobu określania funkcji przystosowania $\varphi(x)$.

2. Dla funkcji przystosowania opisanej formułą (5.5) oraz dla metody określania argumentów x(1),x(2),...,x(5) z każdej piątej części chromosomu „najbardziej efektywną" długością optymalna długość chromosomu wynosi 65 genów (wartość współczynnika liczona wzorem WE = Iop(——) / Iop(▪▪▪) 249641= / 4873 = 5,12.
3. Uwzględnienie czasu pracy wspomagającej sieci neuronowej (na przykład przy określaniu punktów krzyżowania) zmniejsza efektywność wspomagania statystycznie o około 7,31% wartości maksymalnej WE.

6. Wielo-asortymentowe zadanie transportowe w ujęciu genetycznym

Realizując zadanie transportowe różnych asortymentów przewożonych towarów możemy skorzystać ze znanych [33] tablic jednostkowych cen przewozu powielając podobne struktury dla każdego rodzaju towaru. Dla każdego z towarów można ustalić listę optymalnych przewozów. Wykorzystanie posiadanego taboru do przewozu w ramach struktur dwudzielnych [33] to pierwszy etap analizy transportowej. Kolejną fazą badań jest przekształcenie grafu dwudzielnego w strukturę sieciową dającą szansę nazwijmy to „dostawy po drodze". Decydującym czynnikiem wspomagającym decyzję jest posłużenie się kryterium minimalnych kosztów przy każdorazowej rozbudowie trasy transportowej w ramach pojemności taboru przewozowego. Podejście genetyczne stwarza możliwość heurystycznego doboru etapów transportu, które zbliży nas do rozwiązania optymalnego co stanowi cel wspomagania decyzji organizacyjnych.

6.1 Założenia wielo-asortymentowego zadania transportowego

Dysponując tabelą danych c(i,j,k) zawierającą jednostkowe koszty przewozu z miejsca i do miejsca j towaru o kodzie k możemy korzystając z algorytmów o złożoności $O(n^2)$ [4] określić listy przewozów dla każdego z towarów:

p(1,1)>k(1,1)W(1,1) , p(1,2)>k(1,2)W(1,2), . . . , p(1,n)>k(1,n)W(1,n)
p(2,1)>k(2,1)W(2,1) , p(2,2)>k(2,2)W(2,2), . . . , p(2,n)>k(2,n)W(2,n) (6.1)
.....
p(m1,1)>k(m1,1)W(m1,1), p(m2,2)>k(m2,2)W(m2,2), ..., p(mn,n)>k(mn,n)W(mn,n)

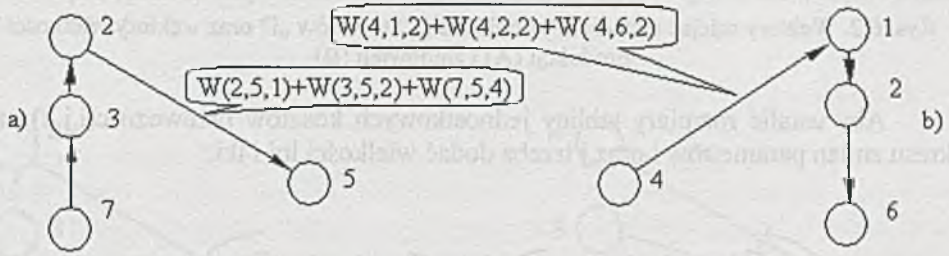
gdzie

p(i,k)>k(i,k)W(i,k) - przewóz towaru o kodzie k i wielkości W(i,k) z punktu p(i,k). do punktu k(i,k).

Jak widzimy ilość list przewozów odpowiada asortymentowi towarów k. Zaś ilość przewozów dla każdego z towarów może być inna m1≠m2≠... ≠mn. Jeżeli zaś

zaistnieje sytuacja kiedy punkty początkowy i końcowy pokrywają się dla różnych towarów p(i,t1) = p(j,t2), k(i,t1) = k(j,t2) (gdzie t1 i t2 kody towarów) to pojawi się możliwość dostarczenia tych towarów jednym przewozem.

Tablice jednostkowych kosztów przewozów c(i,j,k) mogą być przygotowane zarówno dla każdego z towarów oddzielnie w postaci dwuwymiarowej jak również jako macierz trójwymiarowa dająca możliwość podejmowania decyzji na przykład: czy bardziej opłaca się przewieźć jeden towar do kilku odbiorców (rys. 6.1.b) czy kilka towarów do jednego odbiorcy (rys. 6.1.a).



Rys. 6.1.Schemat przewozu towarów

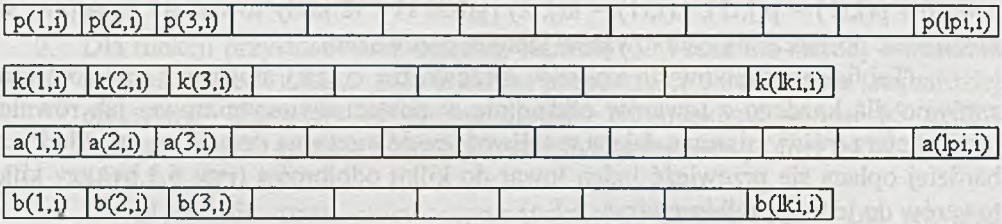
Na rys.6.1.a zbierając towary o kodach 1,2 i 4 u dostawców oznaczonych odpowiednio 2,3 i 7 dowozimy je do odbiorcy 5. Na rys. 6.1.b towar o kodzie 2 dostarczamy do odbiorców 1,2,6 od dostawcy o kodzie 4. Koszty transportu można ocenić następująco:

c(2,5,1)*W(2,5,1)+(c(3,2,2)+c(2,5,2))*W(3,5,2)+(c(7,3,4)+c(3,2,4)+c(2,5,4))*W(7,5,4)=>z1
c(4,1,2)*W(4,1,2)+(c(4,1,2)+c(1,2,2))*W(4,2,2)+(c(4,1,2)+c(1,2,2)+c(2,6,2))*W(2,6,2)=>z2

Zasada podejmowania decyzji oparta na wyborze minimalnego kosztu przewozu jako priorytetowego etapu działania polega, dla powyższego przykładu, na sprawdzeniu czy z1<z2. Jeżeli warunek jest spełniony to podejmowana jest decyzja o przewozie ilustrowanym przez rys.6.1.a, w przeciwnym wypadku decydujemy na rozwiązanie z rys.6.1.b. Po realizacji aktualnego etapu przewozu sytuacja dotycząca zamówień oraz wielkości towaru którym dysponuje producent ulega zmianie. Stąd też po wykonaniu odpowiednich korekt przeprowadzamy ponownie analizę powstałej sytuacji.

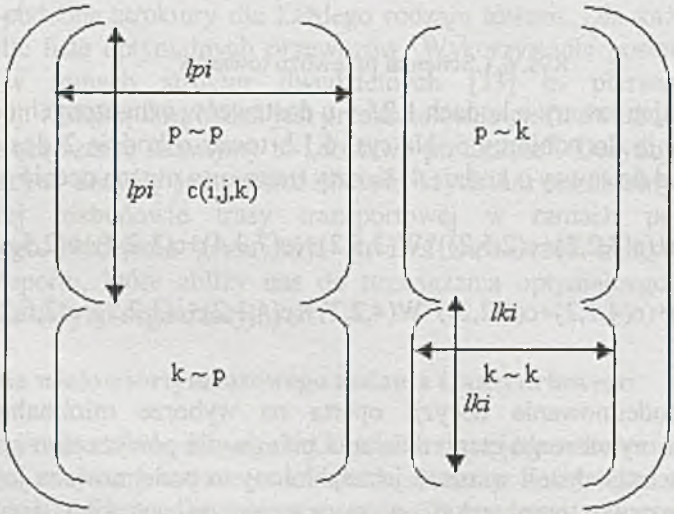
6.2 Struktura danych i dynamika ich zmian

Dla każdego towaru możemy określić zbiór producentów i odbiorców, a także wielkość produkcji i zamówień.



Rys. 6.2. Wektory miejsc nadania (P) i odbioru (K) towarów „i" oraz wektory wielkości produkcji (A) i zamówień (B)

Aby ustalić rozmiary tablicy jednostkowych kosztów przewozu c(i,j,k) czyli zakresu zmian parametrów i oraz j trzeba dodać wielkości lpi i lki.

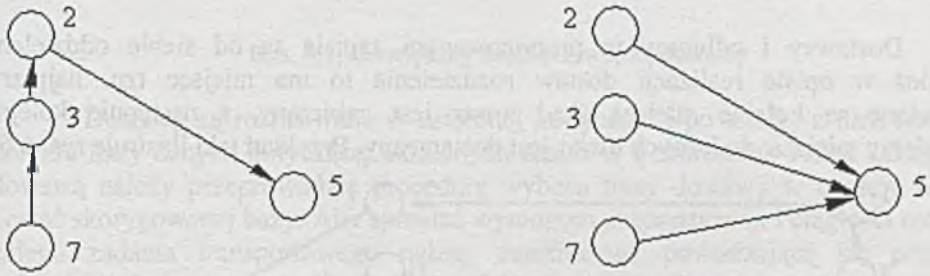


Rys. 6.3. Macierz kosztów jednostkowych przewozu towaru o kodzie k

W powyższej macierzy (rys. 6.3.) pole p ~ p przeznaczone jest dla transportu między producentami w sytuacji kiedy na przykład pojemność taboru pozwala na zabranie kilku rodzajów towarów wyprodukowanych w różnych miejscach. W polu p ~ k (klasyczny wariant) zawarte są jednostkowe koszy przewozów bezpośrednio od dostawcy do odbiorcy. W kwadracie k ~ k podróżując między odbiorcami pozostawiamy zamówiony towar. Ostatnie pole k ~ p, z reguły niewykorzystywane, można wypełnić na przykład jednostkowymi kosztami zwrotu towaru.

Podjęcie decyzji o przemieszczaniu się po polu p ~ p (pomiędzy dostawcami) wymaga przeprowadzenia analizy czy pobranie towarów od kilku dostawców i dostarczenie ich do jednego lub kilku odbiorców jest bardziej opłacalne niż transport niezależny (bezpośrednia dostawa). Algorytm takiej analizy winien zawierać następujące etapy:

- sprawdzenie czy aktualna pojemność taboru jest większa (równa) którymkolwiek aktualnym wielkościom zamówień,
- sprawdzenie czy korzystniejsze jest pobranie nowego towaru u innego producenta czy niezależne dostarczenie poszczególnych partii towarów (rys.6.4.),



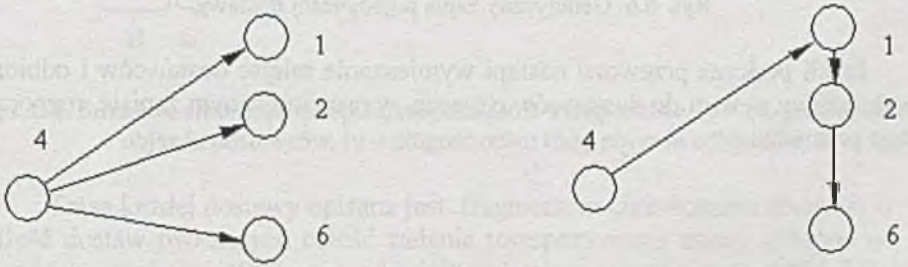
Rys. 6.4. Przedmiot analizy efektywności objazdu producentów:

Struktura zamówień: W(7,5,7), W(3,5,3), W(2,5,2)

Sprawdzenie warunku opłacalności (dla jednego z możliwych wariantów):

c(7,3,7)+c(3,2,7)+c(2,5,7))*W(7,5,7)+(c(3,2,3)+c(2,5,3))*W(3,5,3)+c(2,5,2)*W(2,5,2) < c(7,5,7)*W(7,5,7)+c(3,5,3)W(3,5,3)+c(2,5,2)*W(2,5,2)

- sprawdzenie czy korzystniejsze jest dostarczenie towarów do kilku odbiorców w jednym objazdowym przewozie czy w dostawach niezależnych (rys.6.5.),



Rys. 6.5. Przedmiot analizy efektywności objazdu odbiorców:

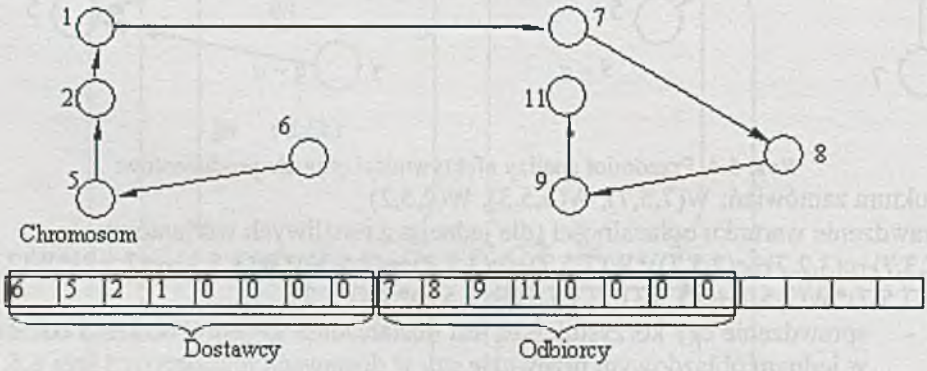
Struktura zamówień: W(4,1,4), W(4,2,4), W(4,6,4)
Sprawdzenie warunku opłacalności (dla jednego z możliwych wariantów):
c(4,1,4)*W(4,1,4)+c(4,2,4)*W(4,2,4)+c(4,6,4)*W(4,6,4)>
c(4,1,4)*(W(4,1,4)+W(4,2,4)+W(4,6,4))+c(1,2,4)*(W(4,2,4)+W(4,6,4))+c(2,6,4)*W(4,6,4)

- integracja rezultatów drugiego i trzeciego etapu i podjęcie finalnej decyzji o trasie przewozu i strukturze załadunku.

Po każdorazowym przewozie redukujemy tablice zamówień i zasobów. Ich rozmiary redukowane są w sposób w przybliżeniu liniowy w funkcji liczby dostaw przy założeniu, że wielkości produkcji i zamówień są tego samego rzędu.

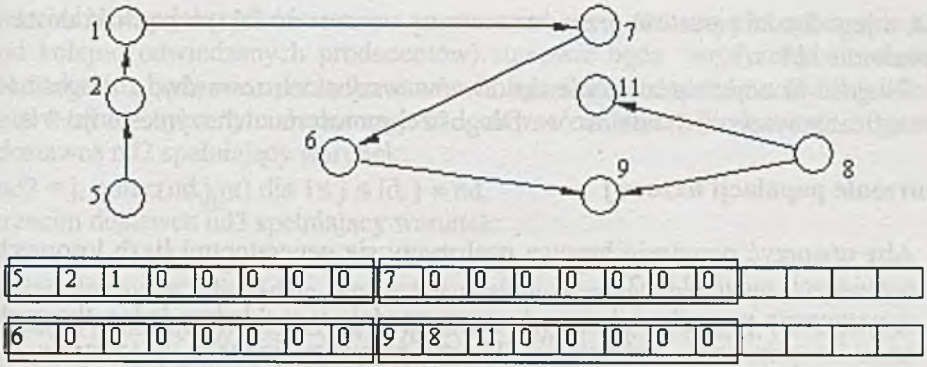
6.3 Genetyczny zapis dostaw

Dostawcy i odbiorcy w proponowanym zapisie są od siebie oddzieleni. Również w opisie realizacji dostaw rozdzielenie to ma miejsce tzn. najpierw zestawione są kolejne miejsca skąd towar jest zabierany, a następnie kolejno zapisujemy miejsca, do których towar jest dostarczany. Przykład taki ilustruje rys. 6.6.



Rys. 6.6. Genetyczny zapis pojedynczej dostawy

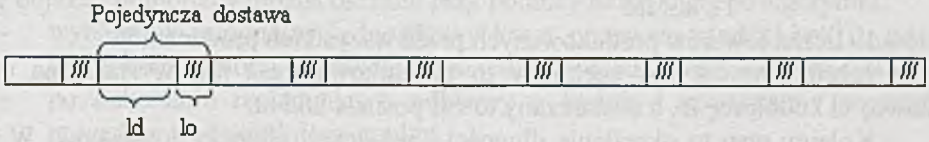
Jeżeli podczas przewozu nastąpi wymieszanie miejsc dostawców i odbiorców to każdorazowy powrót do dostawców oznacza, w proponowanym zapisie, rozpoczęcie nowego przewozu.



Rys. 6.7. Genetyczny zapis podwójnej dostawy

Dostawy są realizowane w ustalonej kolejności i po każdej z nich następuje korekta bazy danych dotyczącej aktualnych zasobów i zamówień. Przed każdą nową dostawą należy przeprowadzić procedurę wyboru trasy dostawy w oparciu o nową postać skorygowanej bazy. Aby sprostać wymogom automatyzacji i ciągłości realizacji całego zadania transportowego należy zunifikować powtarzającą się procedurę polegającą na przygotowaniu bazy zasobów i zamówień, dokonaniu wyboru trasy dostawy oraz określeniu wartości funkcji celu (funkcji przystosowania ; w wariancie genetycznym), którą jest koszt samej dostawy. Procedura winna zawierać także etap systematycznego „wyławiania" najlepszego rozwiązania.

Chromosom opisujący zadanie transportowe w proponowanym wariancie zawierać będzie opis wszystkich dostaw [1,4]. Przykładowa struktura takiego chromosomu przedstawiona jest na rys. 6.8.



Rys. 6.8. Struktura chromosomu opisującego zadanie transportowe, ld – długość opisu trasy objazdu dostawców, lo – długość opisu trasy objazdu odbiorców

Trasa każdej dostawy opisana jest fragmentem chromosomu długości ls = ld + lo. Ilość dostaw tworzących całość zadania transportowego zależy głównie od sumy wszystkich zamówień (SR) oraz od wielkości taboru dostawczego (ST). Minimalna ilość dostaw będzie miała miejsce przy każdorazowo pełnym wykorzystaniu taboru i będzie równa id = TRUNC(SR/ST + 0.5). Oznaczając współczynnik załadowania

przez kt, a jego średnią wartość przez kts możemy oszacować faktyczną ilość dostaw fid na poziomie id/kts.

Długość ld odpowiada liczbie dostawców wszystkich towarów, a długość lo jest równa liczbie wszystkich odbiorców. Długość chromosomu lch wyniesie fid * ls.

6.4 Tworzenie populacji bazowej

Aby utworzyć populację bazową posłużymy się generatorami liczb losowych dla nastę- pujących zadań [11,12]:

- generowanie początku i końca dostawy zgodnie z rozkładem jednostkowych kosztów przewozu (najmniejsze koszty przewozu jednego z towarów na konkretnej trasie to największe prawdopodobieństwo wyboru tej trasy),
- generowanie z zadanym rozkładem długości trasy kumulowania dostawy (odwiedzamy kolejnych producentów),
- generowanie pozostałych elementów trasy „dostawczej" z rozkładem jednostkowych kosztów przewozu,
- generowanie pozostałych odbiorców zgodnie z rozkładem minimalnych kosztów jednostkowych lub maksymalnego zaspokojenia zamówień.

Aby wybrać początkową trasę (początek i koniec dostawy) generujemy losową wartość ξ z przedziału [0,1), następnie losowo generujemy dostawcę nd i odbiorcę no. Jeżeli dostawca jest producentem kilku towarów należy wylosować jeden z nich, kod którego oznaczymy przez nt. O przyjęciu do populacji wartości nd i no zdecyduje sprawdzenie warunku:

$$\xi \geq c(nd,no,nt)/\max_{\substack{1\leq i\leq ld\\ 1\leq j\leq lo\\ 1\leq k\leq ltd}} c(i,j,k) \qquad (6.2)$$

gdzie ltd - liczba towarów produkowanych przez wszystkich producentów.

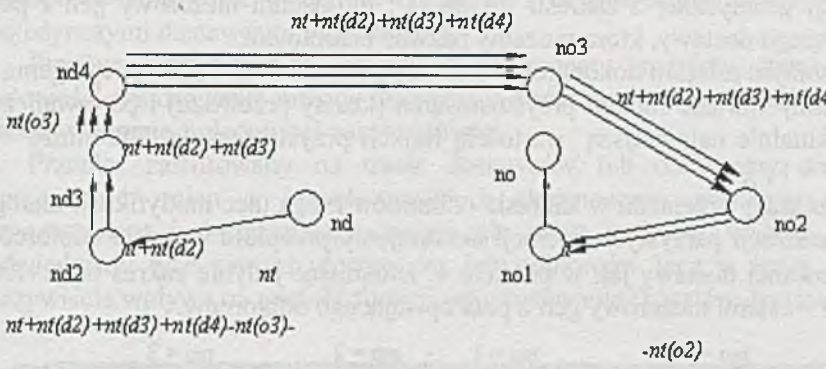
Jeżeli warunek jest spełniony to początkowa trasa jest wyznaczona przez dostawcę ld i odbiorcę lo, a dostarczany towar posiada kod nt.

Kolejny etap to określenie długości i sekwencji objazdu dostawców. W tym celu generujemy wartość losową ξ z przedziału [0,c(nd,no,nt)) i sprawdzamy kolejny warunek:

$$\xi \geq c(nd,j,k), \quad 1\leq j\leq ld\ ;\ j\neq nd,\ k = nt... \qquad (6.3)$$

Liczba sytuacji dla których spełniony jest warunek (6.3) wyznacza długość trasy objazdu dostawców lt. Kolejne kody dostawców które spełniają warunek (6.3) (dla nt <= k <= nt(lt-1), gdzie nt(lt-1) - kod towaru zabieranego od dostawcy

odwiedzanego jako przedostatni, a oznaczenie „ <= „ wskazuje na zabieranie towarów od kolejno odwiedzanych producentów) stanowić będą węzły objazdu dostawców. Jednakowoż będą one uporządkowane w kolejności rosnących kosztów. I tak kolejno: pierwszym wybranym dostawcą jest dostawca oznaczony kodem nd, drugim będzie dostawca nd2 spełniający warunek:
nd2 = j, min c(nd,j,nt) dla $1 \leq j \leq ld$, $j \neq nd$,
trzecim dostawca nd3 spełniający warunek:
nd3 = j, min (c(nd2,j,nt) + c(nd2,j,nt(d2))), $j \neq nd$, $j \neq nd2$,
i-tym dostawcą ndi będzie ten, który spełnia warunek: ndi = j, min(c(nd(i-1),j,nt)+ c(nd(i-1),j,nt(nt(d2)))+... +c(nd(i-1),j,nt(d(i-1)))), $j \neq nd$, $j \neq nd2$,..., $j \neq nd(i-1)$.
Opisaną sytuację przedstawia rys. 6.9.



Rys. 6.9. Przebieg realizacji dostawy (kursywą – kody dostarczanych oraz odbieranych towarów)

Zakładając, że kolejność rozładowania nie zależy od kolejności załadunku to trasę objazdu odbiorców można określić przy pomocy następującego algorytmu:

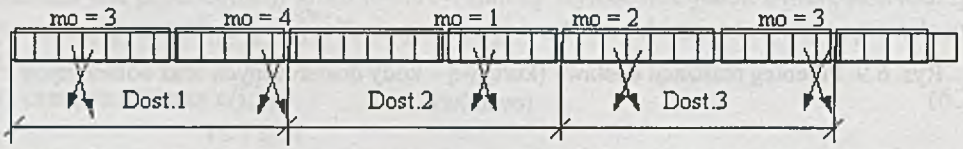
- wybieramy minimalny jednostkowy koszt przewozu c(nd(lt),j,nt(j)), gdzie j jest kodem odbiorcy zamawiającego jeden spośród lt dostarczanych towarów,
- po dokonaniu rozładunku u odbiorcy o kodzie j powtarzamy poprzednią procedurę z poprzedniego punktu.

Po zrealizowaniu całej dostawy korygujemy macierz zamówień i powtarzamy wszystkie procedury poczynając od warunku (6.2). Robimy to tak długo, aż wszystkie zamówienia będą zrealizowane. W taki sposób zostanie utworzony każdy chromosom bazowej populacji.

6.5 Algorytm operacji genetycznych realizujący zadanie transportowe

Ze względu na fakt, iż populacja bazowa zawiera już częściowo przygotowane pod względem ograniczenia kosztów dane zastosujemy tylko jedną operację genetyczną, której realizację można opisać następującym algorytmem:

1. losujemy numer chromosomu z populacji bazowej,
2. losujemy liczbę P z zakresu 1÷ [fid/km] ; km - współczynnik modyfikacji, fid - liczba pojedynczych dostaw, [.] - zaokrąglenie (round),
3. losujemy P razy numery c(i) pojedynczych dostaw w zakresie 1÷fid; i - numer pojedynczej dostawy, które będą modyfikowane,
4. dla każdej wylosowanej dostawy generujemy miejsce mo przeprowadzenia operacji genetycznej z zakresu 1÷(ld0-1) ; ld0-ostatni niezerowy gen z pola opisującego dostawy, którą możemy nazwać przeplotem,
5. w wybranym miejscu dokonujemy zamiany poprzedniego i następnego genu,
6. obliczamy wartość funkcji przystosowania (koszty przewozu) i porównujemy ją z aktualnie najmniejszą wartością funkcji przystosowania (FP<minFP => minFP = FP),
7. również trasy przejazdu w zakresie odbiorców mogą ulec modyfikacji dlatego przy realizacji parzystych operacji dokonujemy przeplotu w opisie odbiorców wylosowanej dostawy jak w punkcie 4, zmieniając jedynie zakres na (1÷l00-1); l00 – ostatni niezerowy gen z pola opisującego odbiorców.



Operacja pierwsza	**Operacja druga**
P = 2	P = 3
c(1) = 1, mo = 3,	c(1) = 1, mo = 4
c(2) = 3, mo = 2,	c(2) = 2, mo = 1
	c(3) = 3, mo = 3

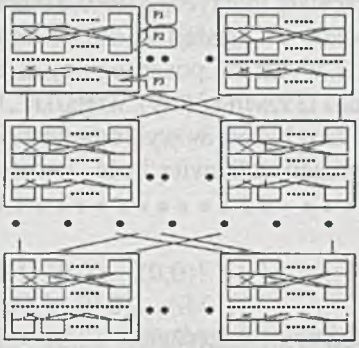
Rys. 6.10. Przebieg operacji genetycznego przeplotu

Organizację przeprowadzania operacji genetycznych można realizować w licznych wariantach. Można na przykład w trakcie realizacji jednej operacji wykonać dwa przebiegi: jeden odnoszący się do dostawców, a drugi do odbiorców. Jednakże, jak wykazują przeprowadzone symulacje nie wpływa to na przyśpieszenie wyszukiwania wartości optymalnej rozwiązania [11].
Wnioski:

1. Oryginalną koncepcją wydaje się wzbogacenie populacji bazowej o informację porządkującą przebieg zadania transportowego. Jest to w istocie rzeczy połączenie stosowanej praktyki algorytmicznej z metodykami genetycznymi. Jest to zarazem efektywne przygotowanie, wspierające i przyśpieszające dalsze poszukiwania optimum.
2. Trudno zakładać, że otrzymane w rozsądnym (i z góry założonym) czasie rozwiązanie jest ostateczne i w pełni optymalne; jest ono jednak na tyle zbliżone do optymalnego, że wykorzystanie go jako pomoc w podejmowaniu decyzji organizacyjnych może dać zdecydowane i wymierne korzyści materialne.
3.Genetyczne opisy tras odwiedzania dostawców i odbiorców jest korzystnym (z punktu widzenia minimalizacji kosztów) uzupełnieniem stosowanych rozwiązań zadań transportowych, gdzie mamy do czynienia z bezpośrednim połączeniem pomiędzy pojedynczymi dostawcami i odbiorcami.
4. Prostota proponowanej operacji genetycznego przeplotu stanowi o sensie i celowości zastosowania metody do wyszukiwania optymalnego wariantu dostarczenia towarów zgonie z złożonymi zamówieniami.
5. Przeplot zastosowany na trasie dostawców lub odbiorców nie wpływał na konieczność zmian w innych częściach chromosomu, ponieważ rodzaj i ilość przewożonego i dostarczanego towaru nie ulega zmianie. Wynika to z faktu, iż odwiedzamy tych samych dostawców lub odbiorców lecz w innej kolejności co oczywiście wpływa na wartość funkcji przystosowania (kosztów dostawy).

7. Koncepcja zagnieżdżonych struktur neuronowych

Celem analizy jest zbadanie możliwości sterowania przebiegiem procesów wewnątrz sieci neuronowej [23,34] przy pomocy zagnieżdżonych struktur neuronowych zgodnie z ilustracją przedstawioną na rys. 7.1. Położenie każdego neuronu przy trzecim stopniu zagnieżdżenia określać będą trzy pary współrzędnych {(x1,y1),(x2,y2),(x3,y3).



Rys.7.1. Schemat ideowy ilustrujący zagnieżdżenie struktur neuronowych. Pi – i-ta płaszczyzna

7.1 Opis zagnieżdżonej struktury neuronowej

W przedstawionej na rys. 7.1 strukturze stopień zagnieżdżenia wynosi trzy lecz teoretycznie może on być dowolny. Oznaczmy go przez r. Tymczasem ilość warstw w każdym zagnieżdżonym elemencie może być różna. Aby określić parametry elementów sieci neuronowej należy ukazać poziom zagnieżdżenia, lokalizację w poprzednich poziomach zagnieżdżeniach oraz lokalizację w właściwym swojemu umiejscowieniu poziomie zagnieżdżenia. Przykładowo taką informację można przeprowadzić następująco:

$$E\{(x1,y1),(x2y2),...,(xr,yr)\} \quad (7.1)$$

Przykłady lokalizacji zagnieżdżonego elementu sieci neuronowej

a) E{(3,4),(2,2),(1,4)}
b) E{(3,4),(2,2),(0,0)}
c) E{(3,4),(0,0),(0,0)}

W przykładzie a) opisano element z trzeciej płaszczyzny zagnieżdżenia o parametrach: 4-ty neuron 1-szej warstwy trzeciej płaszczyzny zagnieżdżenia, 2-giego elementu 2-giej warstwy drugiej płaszczyzny zagnieżdżenia, 4-go elementu 3-ciej warstwy pierwszej warstwy zagnieżdżenia.

W przykładzie b) opisano element z drugiej płaszczyzny zagnieżdżenia o parametrach:
2-gi element 2-giej warstwy drugiej płaszczyzny, 4-go elementu 3-ciej warstwy pierwszej płaszczyzny zagnieżdżenia.

W przykładzie c) opisano element z pierwszej płaszczyzny zagnieżdżenia o parametrach:
4-ty element 3-ciej warstwy pierwszej płaszczyzny zagnieżdżenia.

O przynależności do konkretnej płaszczyzny zagnieżdżenia decyduje ostatni niezerowy układ współrzędnych w opisie (7.1). Do analizy wybierzmy wariant, w którym i-te wyjście elementu poprzedniej warstwy połączone jest z wejściem o tym samym numerze kolejnej warstwy. Tym samym utworzona zostanie struktura będąca wielosieciową konstrukcją, charakteryzująca się identyczną ilością warstw oraz ilością elementów w poszczególnych warstwach w każdej z sieci. Pojawia się także ograniczenie związane z liczbą wejść i wyjść w poszczególnych elementach we wszystkich płaszczyznach zagnieżdżenia. Przyjmujemy więc, iż liczba ta jest wielkością stałą i wynosi k. Ilość wewnętrznych sieci w wybranym wariancie struktury zagnieżdżonej wyniesie więc k. W tej sytuacji całkowitą ilość warstw w każdej z k sieci można oszacować następująco:

$$N = n(1)*n(2)*... *n(r) \quad (7.2)$$

gdzie n(i) – liczba warstw na i-tej płaszczyźnie zagnieżdżenia.
Ilość neuronów w każdej warstwie jest równa:

$$M = k^r \tag{7.3}$$

gdzie k – ilość elementów w każdej warstwie,
r – stopień zagnieżdżenia.

Kolejność i sposób połączeń w zagnieżdżonej strukturze można opisać następująco:
Sygnał na wejściu o numerze 1

↓

E{ (1,1),(0,0),...,(0,0)},E{(1,2),(0,0),...,(0,0)},...,E{(1,k),(0,0),...,(0,0)}

↓

E{(1,1),(1,1),(0,0),...,(0,0)},E{(1,1),(1,2),(0,0),...,(0,0)},...,E{(1,1),(1,k),(0,0),...,(0,0)}
E{(1,2),(1,1),(0,0),...,(0,0)},E{(1,2),(1,2),(0,0),...,(0,0)},...,E{(1,2),(1,k),(0,0),...,(0,0)}
....
E{(1,k),(1,1),(0,0),...,(0,0)},E{(1,k),(1,2),(0,0),...,(0,0)},...,E{(1,k),(1,k),(0,0),...,(0,0)}

↓

E{(1,1),(1,1),(1,1),(0,0),...,(0,0)},E{(1,1),(1,1),(1,2),(0,0),...,(0,0)},...
...,E{(1,1),(1,1),(1,k),(0,0),...,(0,0)}
E{(1,1),(1,2),(1,1),(0,0),...,(0,0)},E{(1,1),(1,2),(1,2),(0,0),...,(0,0)},...
...,E{(1,1),(1,2),(1,k),(0,0),...,(0,0)}
....
E{(1,1),(1,k),(1,1),(0,0),...,(0,0)},E{(1,1),(1,k),(1,2),(0,0),...,(0,0)},...
...,E{(1,1),(1,k),(1,k),(0,0),...,(0,0)}

E{(1,2),(1,1),(1,1),(0,0),...,(0,0)},E{(1,2),(1,1),(1,2),(0,0),...,(0,0)},...
...,E{(1,2),(1,1),(1,k),(0,0),...,(0,0)}
E{(1,2),(1,2),(1,1),(0,0),...,(0,0)},E{(1,2),(1,2),(1,2),(0,0),...,(0,0)},...
...,E{(1,2),(1,2),(1,k),(0,0),...,(0,0)}
....
E{(1,2),(1,k),(1,1),(0,0),...,(0,0)},E{(1,2),(1,k),(1,2),(0,0),...,(0,0)},...
...,E{(1,2),(1,k),(1,k),(0,0),...,(0,0)}

• •

E{(1,k),(1,1),(1,1),(0,0),...,(0,0)},E{(1,k),(1,1),(1,2),(0,0),...,(0,0)},...
...,E{(1,k),(1,1),(1,k),(0,0),...,(0,0)}
E{(1,k),(1,2),(1,1),(0,0),...,(0,0)},E{(1,k),(1,2),(1,2),(0,0),...,(0,0)},...
...,E{(1,k),(1,2),(1,k),(0,0),...,(0,0)}

.....

E{(1,k),(1,k),(1,1),(0,0),...,(0,0)},E{(1,k),(1,k),(1,2),(0,0),...,(0,0)},...

...,E{(1,k),(1,k),(1,k),(0,0),...,(0,0)}

•

•

•

Z powyższego zapisu wynika, że transfer sygnału o kodzie 1 jest realizowany między elementami zawierającymi konfigurację współrzędnych następującej postaci:

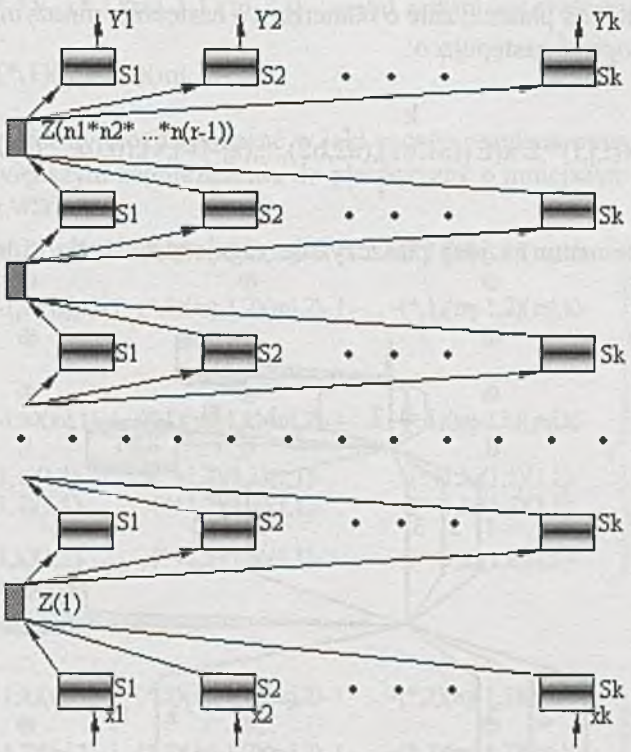
... (1,i),(0,0)... $1 \leq i \leq k$,

a sygnału o kodzie 2 między elementami z konfiguracją współrzędnych:

... (2,i),(0,0)... $1 \leq i \leq k$, itd.

Zagnieżdżenie winno być określone przez numer ostatniego niezerowego zestawu współrzędnych, przykładowo element E{(1,k),(1,k),(1,2),(0,0),...,(0,0)} znajduje się na trzeciej płaszczyźnie zagnieżdżenia, element E{(1,k),(0,0),...,(0,0)} jest na pierwszej płaszczyźnie zagnieżdżenia.

Wzajemne wymieszanie informacji w proponowanej wersji występuje w najgłębszym miejscu zagnieżdżenia. Ilustruje to rys.7.2., na którym nie widać płaszczyzn zagnieżdżenia, a jedynie wzajemne zależne i niezależne fragmenty wewnętrznych sieci. Ilość tych wewnętrznych struktur w danej wersji jest równa k.



Rys.7.2. Miejsca najbardziej zagnieżdżonych struktur jako miejsca wymiany informacji między wewnętrznymi sieciami neuronowymi. Z(i) – i-ta warstwa ostatniego poziomu zagnieżdżenia wewnętrznej sieci, Yi- wektor wyjść i-tej wewnętrznej sieci, xi – i-ty sygnał wejściowy, Si elementy i połączenia sieciowe na poziomach wyższych niż r (tj. na poziomach 1,2,...r-1)

7.2 Warianty zagnieżdżenia

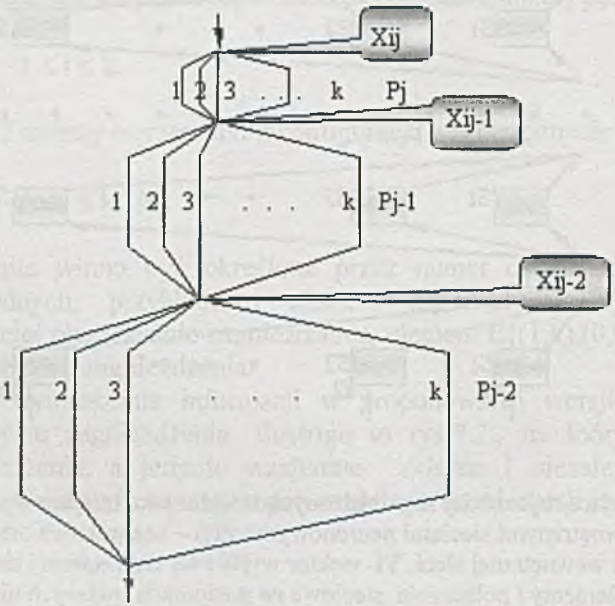
Na rysunku 7.2 przedstawiono widok, który można by określić jako „widok z boku" co pozwala wyodrębnić poszczególne sieci wewnętrzne. Na rys.7.3. przedstawiono charakter połączeń między poszczególnymi poziomami zagnieżdżeń (byłby to „widok z przodu").

Sygnał xi jest rozproszony na neurony pierwszej warstwy, następnie zostaje przetworzony wraz z innymi sygnałami i przekazany na drugą i kolejne warstwy zgodnie z klasyczną strukturą sieci neuronowych [18,34]. Po przejściu przez wszystkie warstwy j-tej płaszczyzny zagnieżdżenia kierujemy się na płaszczyznę j-1. Aby uzyskać sygnał wejściowy elementu j-1 płaszczyzny sumujemy sygnały wyjściowe o

tej samej numeracji na płaszczyźnie o numerze j, a następnie mnożymy tę sumę przez wagę. Można to opisać następująco:

$$x(i,j,l) = w(i,j,l)^* \sum_{v=1}^{k} x(E\{(a1,b1),(a2,b2),...,(aj-1=i,v),(0,0),...,(0,0)\}) \qquad (7.4)$$

gdzie l-numer elementu na j-tej płaszczyźnie zagnieżdżenia, do którego skierowany jest sygnał x(i,j,l).



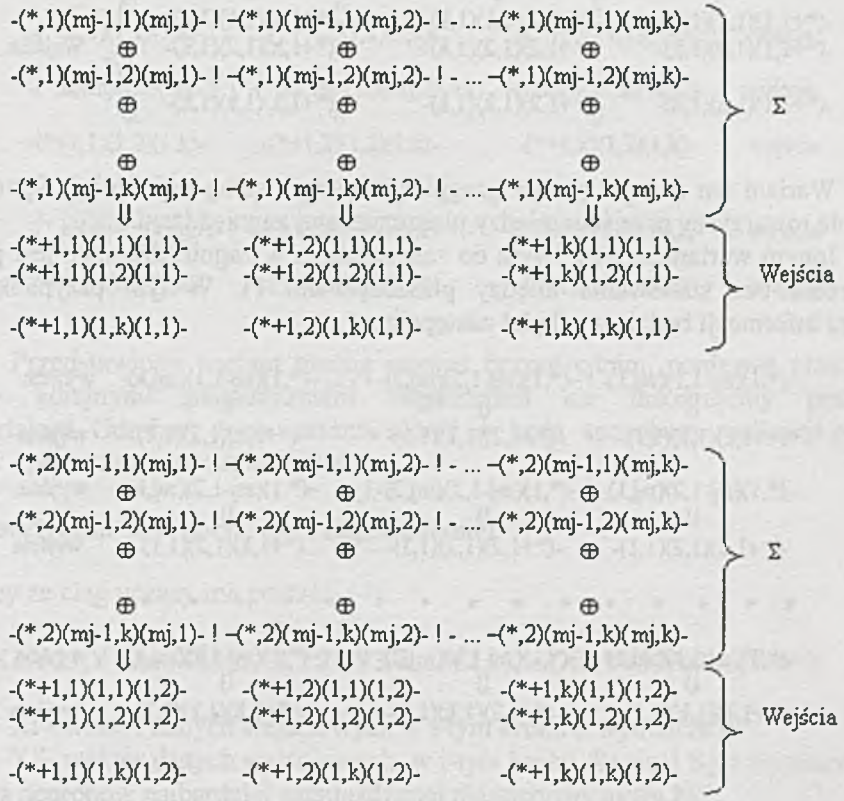
Rys. 7.3. Ilustracja sposobu przekazu i-tego sygnału pomiędzy poszczególnymi płaszczyznami zagnieżdżenia, Xij=(xij1,xij2,...,xijk)

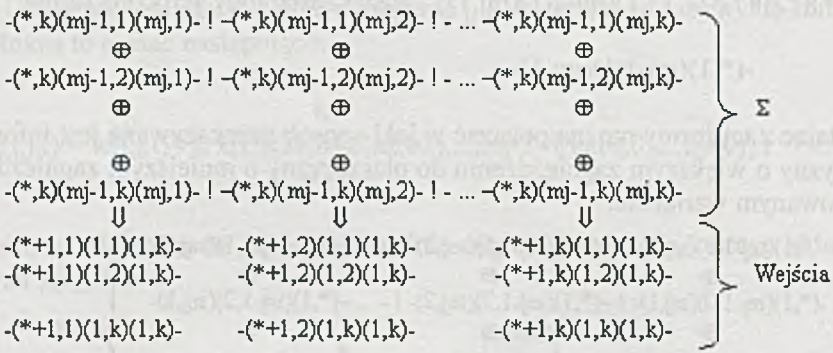
Aby opisać szczegółowiej prezentowaną formę zagnieżdżenia posłużymy się uproszczonym zapisem elementów zagnieżdżonej sieci neuronowej; miast opisywać wszystkie współrzędne zagnieżdżenia wybierzemy tylko trzy aktywne płaszczyzny tj. płaszczyznę zawierającą warstwę wyjściową (ostatnią o numerze mj) najgłębiej zagnieżdżoną, płaszczyznę pośrednią (j-1)-szą również zawierającą także ostatnią warstwę o numerze mj-1 oraz płaszczyznę najpłycej zagnieżdżoną zawierającą pierwszą warstwę, do której przekazywany jest przetworzony sygnał. Przykładowo zapis elementu przedstawiony następująco:

E{(*,*)(*,*)(*,*)... (*,1)(mj-1,1)(mj,1)} został zastąpiony skróconą formą:

-(*,1)(mj-1,1)(mj,1)-

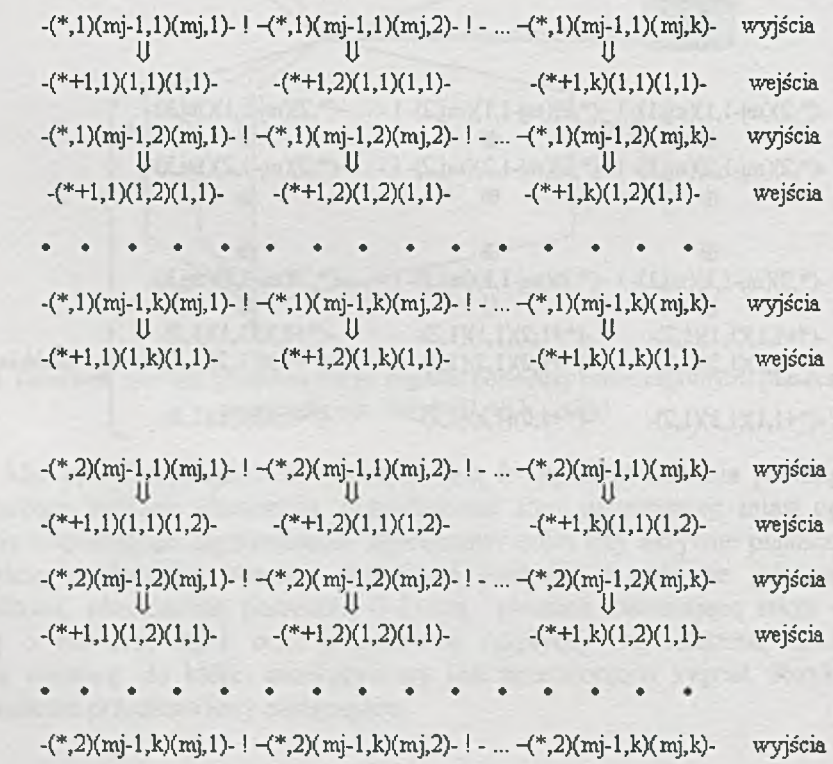
Korzystając z tej formy można pokazać w jaki sposób przekazywana jest informacja z płaszczyzny o większym zagnieżdżeniu do płaszczyzny o mniejszym zagnieżdżeniu w prezentowanym wariancie:

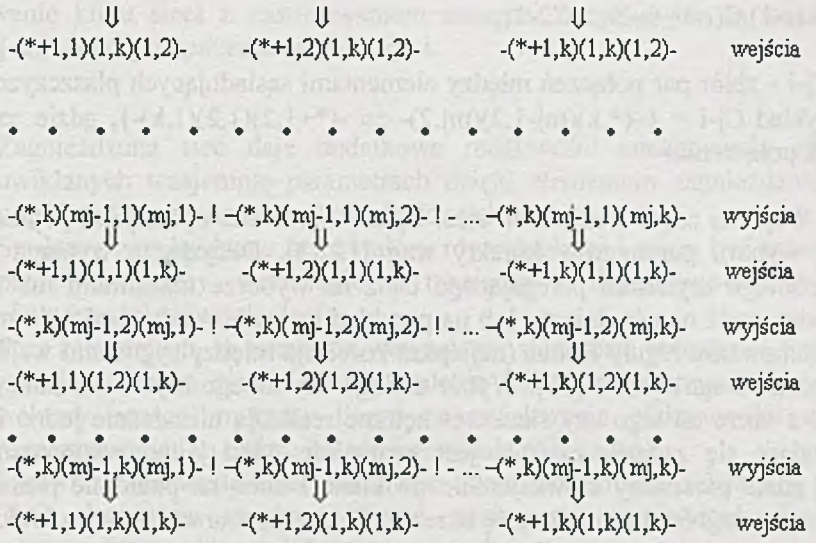




Wariant ten nazwać można „zagnieżdżeniem z sumą pośrednią". Sumowanie pośrednie towarzyszy przejściu między płaszczyznami zagnieżdżenia.

Innym wariantem możliwym do zastosowania w zagnieżdżeniach jest przekaz bezpośredni bez sumowania między płaszczyznami Pi. W tym przypadku opis transferu informacji będzie wyglądał następująco:





Przedstawiony wariant można nazwać bezpośrednim, ponieważ przechodząc między kolejnymi płaszczyznami zagnieżdżeń nie dokonujemy pośrednich przekształceń. Odmiany tego wariantu różnić się będą sposobem realizacji połączeń między - płaszczyznami zagnieżdżeń.

7.3 Teoretyczna koncepcja sieci zagnieżdżonej

Załóżmy że ciąg uczący ma postać:

$$L = \{(X1,Y11,Y12,...,Y1k),(X2,Y21,Y2,2,...,Y2,k),...,(Xn,Yn1,Yn2,...,Ynk)\}, \quad (7.5)$$

gdzie Xi - wektor danych wejściowych w i-tym kroku o wymiarze k,
Yij- wektor danych wyjściowych w i-tym kroku dla sieci S j o wymiarze k.
Wyjścia neuronów najbardziej zagnieżdżonej płaszczyzny mogą być:

- wyjściami zewnętrznymi jednej z k sieci,
- przejściem między najbardziej zagnieżdżonymi elementami poprzez płaszczyzny o niższych stopniach zagnieżdżenia,
- przejściem między neuronami sąsiednich warstw tej samej płaszczyzny.

W pierwszym przypadku sygnały wyjściowe mogą być jednakowe dla różnych sieci w tym samym kroku uczenia Y11=Y12=...=Y1k lub różne (na przykład gdy będziemy modelować różne zjawiska lecz wzajemnie powiązane). Do tego dodać należy reguły połączeń między płaszczyznami:

$$R = \{Cr\text{-}(r\text{-}1),C(r\text{-}1)\text{-}(r\text{-}2),...,C2\text{-}1\}, \quad (7.6)$$

gdzie Cj-i - zbiór par połączeń między elementami sąsiadujących płaszczyzn j oraz i, (na przykład Cj-i = {–(*,k)(mj-1,2)(mj,2)- ⇒ -(*+1,2)(1,2)(1,k)-}, gdzie ⇒ oznacza kierunek połączenia)

Wspólna część wszystkich sieci będzie sterowana operacjami poprzedzonymi decyzją wyboru parametrów korekty wag [18,23]. Decyzja ta wymaga przyjęcia odpowiedniego kryterium polegającego bądź na wyborze maksimum lub minimum zmian wag, bądź na uśrednieniu lub na przykład na zachowaniu tendencji zmian albo też na zachowaniu reguły Hebba (najlepsza korelacja między sygnałami wejściowymi, a „wzorcami wagowymi") [23]. Wybór takiego czy innego kryterium zależy od typu zadania, a także od tego czy sieci wewnętrzne realizują niezależnie jedno i to samo powtarzające się zadanie czy też jest generalnie tylko jedno wieloparametryczne zadanie gdzie parametry są wzajemnie uwikłane z sobą na poziomie płaszczyzny o największym zagnieżdżeniu. Regułę uczenia dla q-tego neuronu w i-tym kroku uczenia określamy w następujący sposób:

$$w(p,q)(St)(i+1) = w(p,q)(St)(i) + \eta x(p,q)(St)y(q)(St)(i), \quad (7.7)$$

gdzie

$$y(q)(St)(i) = \sum_{p=1}^{k} w(p,q)(St)(i)^* x(p,q)(St), \quad (7.8)$$

- p – numer neuronu do którego skierowany jest sygnał x(p,q),
- t – numer wewnętrznej sieci.

Jednym ze sposobów określenia przyrostu współczynnika wagi może być wyrażenie:

$$\Delta w(p,q)(St) = \eta^*\Delta y(q)(St)^*d\varphi t(\varepsilon)/d\varepsilon(q)^* x(p,q)(St), \quad (7.9)$$

gdzie $\Delta y(q)(St) = Y(q)(St) - y(q)(St)$, Y(q)(St)- element wektora uczącego dla wewnętrznej struktury sieci neuronowej oznaczonej St,
wartość pochodnej można łatwo określić dla funkcji logistycznej

$$\varphi t(\varepsilon) = 1/(1+ \exp(-\beta\varepsilon)),$$

dla której

$$d\varphi t(\varepsilon)/y(q)(St)\, d\varepsilon(q) = y(q)(St)\,(1 - y(q)(St)).$$

Modelowanie „rozszczepionej" sieci neuronowej (schizoneuronu) o wspólnych zagnieżdżeniach nie nastręcza większych trudności i przypomina jednoczesne

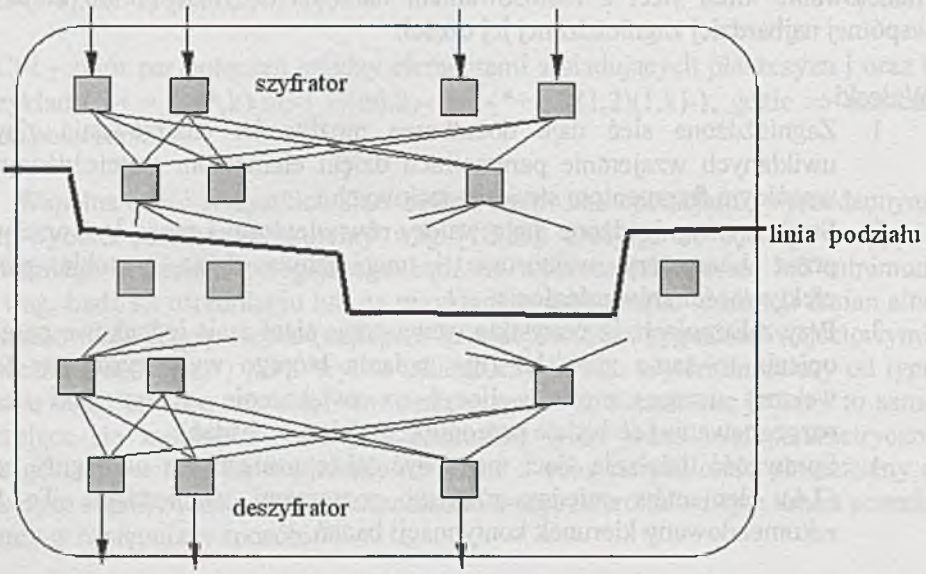
modelowanie kilku sieci z zastosowaniem narzędzi decyzyjnych do korekty wag wspólnej najbardziej zagnieżdżonej jej części.

Wnioski

1. Zagnieżdżona sieć daje dodatkowe możliwości analizowania zjawisk o uwikłanych wzajemnie parametrach dzięki elementom zagnieżdżonym oraz wspólnym fragmentom struktur sieciowych.
2. Struktury zagnieżdżone mają walory równoległości i mogą być przetwarzane przez komputery wektorowe i macierzowe dając wysokie parametry efektywności zrównoleglenia.
3. Przy założeniach, iż wszystkie wewnętrzne sieci mają jednakowe parametry i opisują to samo zjawisko (dla zadania którego wykorzystujemy te same wektory uczące) możemy liczyć na zwiększenie efektywności uczenia i rozpoznawania (co będzie przedmiotem dalszych badań).
4. Sprawność działania sieci może być także uzależniona od reguły połączeń (7.6) elementów między różnymi poziomami zagnieżdżeń. To kolejny rekomendowany kierunek kontynuacji badań.

8. Koncepcja neuronowej kryptografii

Istotą kryptografii jest użycie narzędzi szyfrujących przez wysyłającego wiadomość oraz deszyfrujących przez odbiorcę. Zastosowanie neuronów do szyfrowania lub deszyfrowania może być zrealizowane na wiele różnych sposobów . Koncepcja prezentowana przez autorów zakłada rozdzielenie struktury neuronowej na dwie lub więcej części (rys.8.1). Części te zostaną zgrupowane w dwa zbiory : zbiór szyfrujący oraz zbiór deszyfrujący. Zbiory te są całkowicie niezależne.



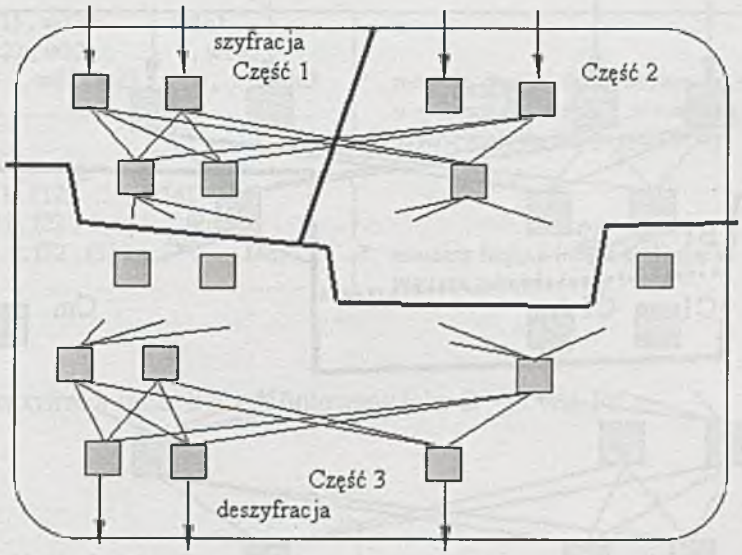
Rys.8.1. Idea rozdziału struktury neuronowej na układ szyfrujący i deszyfrujący

Pozostaje kwestia analizy jakości szyfrowania i analizy czy tak zaszyfrowane dane nie będą mogły zostać z łatwością rozszyfrowane bez pomocy deszyfratora[16]. Porównanie z klasycznymi wariantami szyfrowania powinno dać podobne rezultaty. Uczenie sieci neuronowej w pełnym strukturalnym ujęciu może zakładać : tożsamość danych wejściowych , przejście w inny układ liczbowy, zmianę skali przedstawienia danych , przesunięcie oraz inne modyfikacje danych wejściowych.

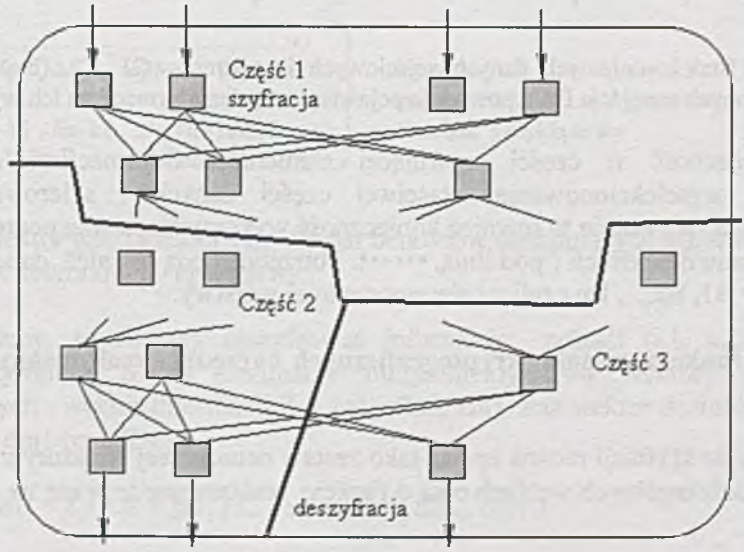
Podzielmy przykładową strukturę neuronową (rys.8.1) na dwie części :

$$\left\{\begin{matrix}1\ 2\ 3\ 4\ 5\ 6\ 7\ 8\ 9\\ 1\ 2\ 3\ 4\ 5\\ 1\ 2\ 3\ 4\ 5\ 6\ 7\\ 1\ 2\ 3\ 4\ 5\ 6\ 7\ 8\end{matrix}\right\} = \left\{\begin{matrix}1\ 2\ 3\ 4\ 5\ 6\ 7\ 8\ 9\\ 1\ 2\ 3\ \ldots\ldots\ldots\\ \ldots\ldots\ldots\ldots 5\ 6\ 7\\ \ldots\ldots\ldots\ldots\end{matrix}\right\} + \left\{\begin{matrix}\ldots\ldots\ldots\ldots\ldots\ldots\\ \ldots\ldots\ldots\ 4\ 5\\ 1\ 2\ 3\ 4\ldots\ldots\ldots\\ 1\ 2\ 3\ 4\ 5\ 6\ 7\ 8\end{matrix}\right\}$$

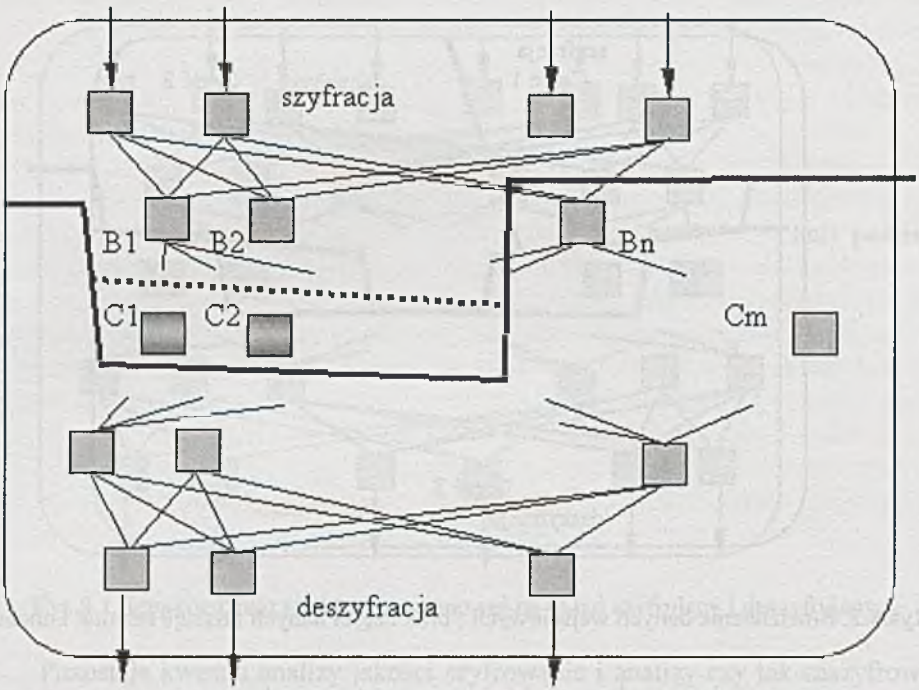
Skrótowo można to zapisać $\boxed{A} = \boxed{B} + \boxed{C}$ Strukturę A można rozbić na kilka części. Przy takim rozbiciu może dojść do sytuacji, kiedy będziemy mieli dostarczane dane z kilku źródeł (rys.8.2) lub też taką, kiedy złączenie kilku kluczy deszyfrujących daje finalny rezultat (rys.8.3). Może też zaistnieć sytuacja kiedy niepełne dane stworzą błędne pośrednie rezultaty (rys.8.4)



Rys.8.2. Rozdzielenie danych wejściowych ; brak części danych fałszuje rezultat końcowy



Rys. 8.3. Podział części deszyfrującej ; brak jednej z części uniemożliwia deszyfrację



Rys. 8.4. Brak kompletnych danych wejściowych dla neuronów C1 , C2...(brak na przykład danych z wyjścia Bn) powoduje pojawienie się dezinformacji na ich wyjściu

Obecność w części szyfrującej elementów informacji i dezinformacji wymusza wyselekcjonowanie właściwej części danych i skierowanie ich do deszyfratora. Wywołuje to również konieczność wykorzystania wag neuronów C1 , C2 ... do procesu deszyfracji (pod linią. ▪▪▪▪▪). Potrzebne będą również dane wyjściowe z neuronów B1, B2,..., Bn czyli z całej poprzedniej warstwy.

8.1 Opis funkcjonowania kryptograficznych narzędzi i realizowanych przez nie procedur

Narzędzie do szyfracji można opisać jako zestaw neuronowej struktury uzupełnionej o wagi w poszczególnych węzłach oraz o funkcje realizowane te przez węzły S = < Ws, Fs >,
gdzie:

$$Ws = \begin{cases} w11, w12, \ldots, w1n1 \\ w21, w22, \ldots, w2n2 \\ 0, w32, w33, \ldots, 0, \ldots, w3n3 \\ \ldots\ldots\ldots\ldots \end{cases}$$

macierz wag z wyłączeniem wierszy o wszystkich wagach zerowych (część szyfrująca)-wariant przykładowy

$$Fs = \begin{cases} f11, f12, \ldots, f1n1 \\ f21, f22, \ldots, f2n2 \\ *, f32, f33, \ldots, *, \ldots, f3n3 \\ \ldots\ldots\ldots\ldots \end{cases}$$

macierz funkcji (część szyfrująca) wariant przykładowy

Podobnie deszyfrator może być zdefiniowany jako D = < Wd, Fd >,

gdzie

$$Wd = \begin{cases} \ldots\ldots\ldots\ldots \\ w31, 0, 0, \ldots, w3i, \ldots, 0 \\ \ldots\ldots\ldots\ldots \\ wm\text{-}11, wm\text{-}12, \ldots, wm\text{-}1nm\text{-}1 \\ wm1, wm2, \ldots, wmn1 \end{cases}$$

macierz wag z wyłączeniem wierszy o wszystkich wagach zerowych (część deszyfrująca)- wariant przykładowy

$$Fd = \begin{cases} \ldots\ldots\ldots\ldots \\ f31, 0, 0, \ldots, w3i, \ldots, f3n3 \\ \ldots\ldots\ldots\ldots \\ fm\text{-}11, fm\text{-}12, \ldots, fm\text{-}1nm\text{-}1 \\ fm1, fm2, \ldots, fmnm \end{cases}$$

macierz funkcji (część deszyfrująca) - wariant przykładowy

tutaj wij to wektor wag określonych dla par neuronów sąsiadujących warstw [23].
ni – liczba neuronów i – ej warstwy

Załóżmy, że chcemy zaszyfrować informację postaci (x1, x2,..., xn1) i użyjemy szyfratora S. W rezultacie otrzymujemy dwa wektory rezultatów zaszyfrowanych : wektor informacji Z = {z1, z2,..., zni} oraz wektor dezinformacji DZ = {d1, d2,..., dnj} (rys. 8.4.).

$$S(x1, x2,..., xn1) = Z + DZ = \{z1, z2,..., zni\} \cup \{d1, d2,..., dnj\} \quad (8.1)$$

gdzie
zk – wartość na wyjściu k – go neuronu i –tej warstwy

Dane , które stanowią elementy wektora informacji Z poddajemy procedurze deszyfracji D (pomijamy w tej chwili problem odszukania tych elementów ze zbioru połączonego).

$$D(z1, z2,..., zni) = \{y1, y2,..., ynm\} \quad (8.2a)$$
$$D(z1, z2,..., zni, d1, d2,..., dnj) = \{y1, y2,..., ynm\} \quad (8.2b)$$

Załóżmy, iż liczba neuronów pierwszej warstwy szyfrującej n1 oraz ostatniej warstwy deszyfrującej nm są sobie równe (n1 = nm) i mamy do czynienia z transformacją typu „jeden do jednego" tzn. dane do szyfracji pokrywają się z rezultatami deszyfracji). Ponadto ustalmy, iż struktura neuronowa działa w obu kierunkach (atrybut wstecznej propagacji). Wówczas $\{x1, x2,..., xn1\} = \{y1, y2,..., ynm\}$

$$S'(y1, y2,..., ynm) = \{z1, z2,..., znj\} \cup \{d1, d2,..., dni\} \quad (8.3a)$$

$$D'(z1, z2,..., znj, d1, d2,..., dni) = D'(z1, z2,..., znj) = \{x1, x2,..., xn1\} \quad (3b)$$

$$D(S(x1, x2,..., xn1)) = \{y1, y2,..., ynm\} \quad (8.4a)$$

$$D'(S'(y1, y2,..., ynm) = \{x1, x2,..., xn1\} \quad (8.4b)$$

gdzie S', D' – szyfracja i deszyfracja wsteczna.

Wprowadźmy pojęcie funkcji kryptograficznej KF(*) = D(S(*)) (lub KF'(*)= D'(S'(*)) dla propagacji wstecznej).Oczywiście KF(KF'(*)) = X oraz KF'(KF (*)) = Y. Ponadto zdefiniujmy grupę neuro- kryptograficzną jako szóstkę { X, Y, S, D, +, * } . Tym razem X oznacza zbiór danych wejściowych, Y – zbiór danych wyjściowych, S = <Ws,Fs> (szyfrator), D = <Wd,Fd> (deszyfrator) , +, * - podstawowe działania w sieci neuronowej [34]. Aby przedstawić etap szyfracji użyjemy zapisu z pomocą zdefiniowanej grupy {X, Z, S, 0, +,* }. Fragmenty szyfracji i deszyfracji (części 1, 2, 3 na rys. 3) opiszemy następująco : {X1, Z1, S1, 0,+, * }, {Z2, Y2, 0, D2, +, * } oraz {Z3, Y3, 0, D3, 0, + }. Jeśli fragmenty te traktować odrębnie to składowe Z1, Y2, Y3 są typową dezinformacją jeśli zaś łącznie to będą one częściami „ właściwej" informacji. Brak któregokolwiek z fragmentów czyni układ szyfrator – deszyfrator bezużytecznym. Również „brak" choćby jednego neuronu dyskwalifikuje układ całkowicie.

$$S = <Ws1,Fs1> + <Ws2,Fs2> +...+ <Wsp,Fsp> \quad (8.5)$$
$$D = <Wd1,Fd1> + <Wd2,Fd2> +...+ <Wdr,Fdr>,$$

gdzie p – liczba części szyfratora,
r – liczba części deszyfratora.

Podobne zależności określić można dla propagacji wstecznej.Oczywiście części struktury neuronowej dotyczące szyfracji i deszyfracji podlegają zamianie, zmienia się także kierunek formowania sum warstwowych [16]. W zależności od sposobu przygotowania sieci podczas etapu uczenia wagi prostej i wstecznej propagacji mogą być albo równe sobie w obu kierunkach albo też mogą się różnić od siebie. W pierwszej wersji sygnały wejściowe rozdzielane są proporcjonalnie do wag określonych dla propagacji prostej [29]. W drugiej wersji uczenie w obu kierunkach traktowane jest niezależnie zamianie podlegają jedynie dane wejściowe i wyjściowe (ze zbiorów uczących).

8.2 Ocena jakości zabezpieczenia kryptograficznego

Wprowadźmy pojęcie wrażliwości szyfrowania neuronowego sw. Byłaby to minimalna wartość zmiany wagi , która wywołuje jakąkolwiek zmianę kodu wynikowego.

$$sw = \min_{\substack{1 \le i \le m \\ 1 \le j \le nm}} \{ wij - w'ij \; ; \; fij - f'ij \neq 0 \}, \qquad (8.6)$$

j – numer neuronu w i- tej warstwie,
w', f ' – wartość skorygowanej wagi oraz wartość zmiany kodu na wyjściu neuronu wywołana korektą wybranej wagi
m – ilość warstw, ni – ilość neuronów w i- tej warstwie.

Definicję wrażliwości oparto na założeniu, iż wartości funkcji kodowych f i f ' zmieniają się dyskretnie dla określonego układu liczbowego.
Miarą jakości zabezpieczenia kryptograficznego może być liczba wariacji stanów wagowych dla wszystkich neuronów oraz dla pełnego zakresu zmian wag (z krokiem równym wrażliwości szyfrowania).

$$CQS = [1/sw]^{nw} \; , \; nw = n0*n1 + n1*n2 + ... + nm-1* nm, \qquad (8.7)$$

gdzie CQS – współczynnik jakości zabezpieczenia, nw – ilość wag.

Przykładowo jeśli sw = 0.01 oraz liczba neuronów w układzie szyfrująco - deszyfrującym wynosi 50 (10 + 10 + 10 + 10 + 10, tzn. po 10 neuronów w pięciu warstwach) to współczynnik jakości zabezpieczenia kryptograficznego wyniesie 100 do potęgi 400, a więc liczba 801 – cyfrowa.

W oszacowaniu nie uwzględniono jeszcze roli dezinformacji, która zwiększa jakość zabezpieczenia. Ocena chaosu wprowadzonego przez wymieszanie informacji prawdziwej i dezinformacji mogła by być przeprowadzona na różne sposoby.

Wyraźmy to na przykład stosunkiem liczby wariacji stanów wag wykorzystywanych do tworzenia dezinformacji do liczby wariacji stanów wag zastosowanych do generowania pełnej informacji. Załóżmy, iż ostatnia pełna warstwa szyfratora ma numer j oraz że ilość warstw niepełnych wynosi k.Liczba neuronów w warstwach niepełnych wynosi odpowiednio : ne1, ne2,..., nek. Stąd ilość połączeń wagowych wykorzystywanych do tworzenia dezinformacji wynosi :

$$md = n0*n1 + n1*n2 +... + nj-1*nj + nj*ne1 + ne1*ne2 +...+ nek-1*nek, \quad (8.8)$$

a do tworzenia pełnej informacji

$$mi = n0*n1 + n1*n2 +... + nj-1*nj \quad (8.9)$$

Współczynnik oddziaływania chaosu będzie większy od jedności i można go wyrazić następująco:

$$CCH = (ws*[1/sw]^{md-mi} + [1/sw]^{mi})/ [1/sw]^{mi} = 1 + ws*[1/sw]^{md-2mi} \quad (8.10)$$

gdzie ws –współczynnik skali prezentacji dobierany dla zakresu (md – 2mi). W tablicy 8.1 „uwypuklono" zakres md = [20 – 16], mi = 10, stosując współczynnik ws = 1000.

Tablica 8.1.Wielkości oddziaływania chaosu informatycznego

mi = 10 sw / md	Współczynnik oddziaływania chaosu informatycznego CCH						
	20	18	16	14	12	10	8
0,002	1001	1,004	1	1	1	1	1
0,004	1001	1,016	1,0000003	1	1	1	1
0,006	1001	1,036	1,0000013	1	1	1	1
0,008	1001	1,064	1,0000041	1	1	1	1
0,01	1001	1,1	1,00001	1	1	1	1
0,02	1001	1,4	1,00016	1	1	1	1
0,03	1001	1,9	1,00081	1,000001	1	1	1

Współczynnik CCH wskazuje na nikłość zabezpieczenia , które wynika z wprowadzenia dezinformacji (CCH ≅ 1).

Znacznie bardziej wyraziste oddziaływanie chaosu informacyjnego uzyskamy dzięki zastosowaniu współczynnika zdefiniowanego jako stosunek sumy liczby neuronów w warstwach wyjściowych informacyjnej i dezinformacyjnej do liczby neuronów w wyjściowej warstwie informacyjnej :

$$CCN = (ni + nk)/ni = 1 + nk/ni \quad (8.11)$$

Aby uzyskać ostateczny poziom zabezpieczenia przemnażamy współczynnik jakości (8.7) przez współczynnik oddziaływania chaosu (10 lub 11):

$$CS = CQS * CCH \text{ (lub } CCN \text{)} \qquad (8.12)$$

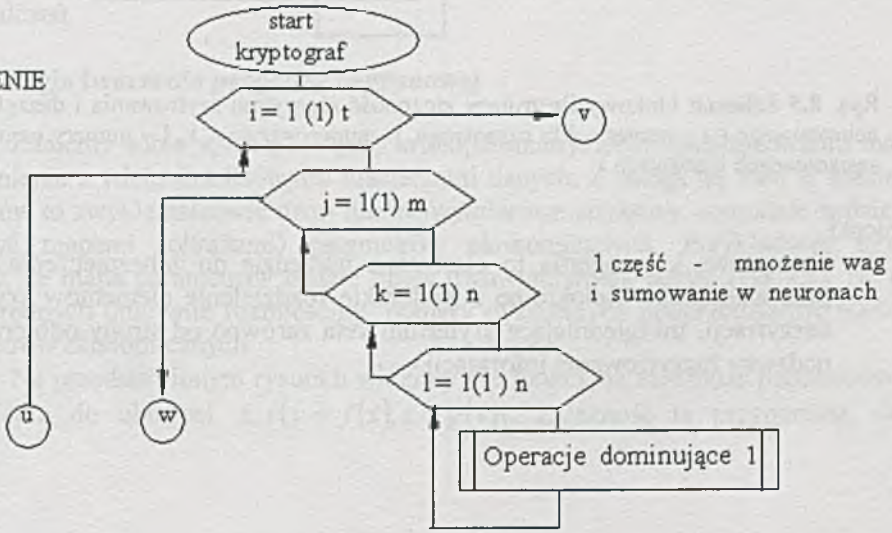
8.3 Złożoność (czasowa) algorytmu neuronowej szyfracji

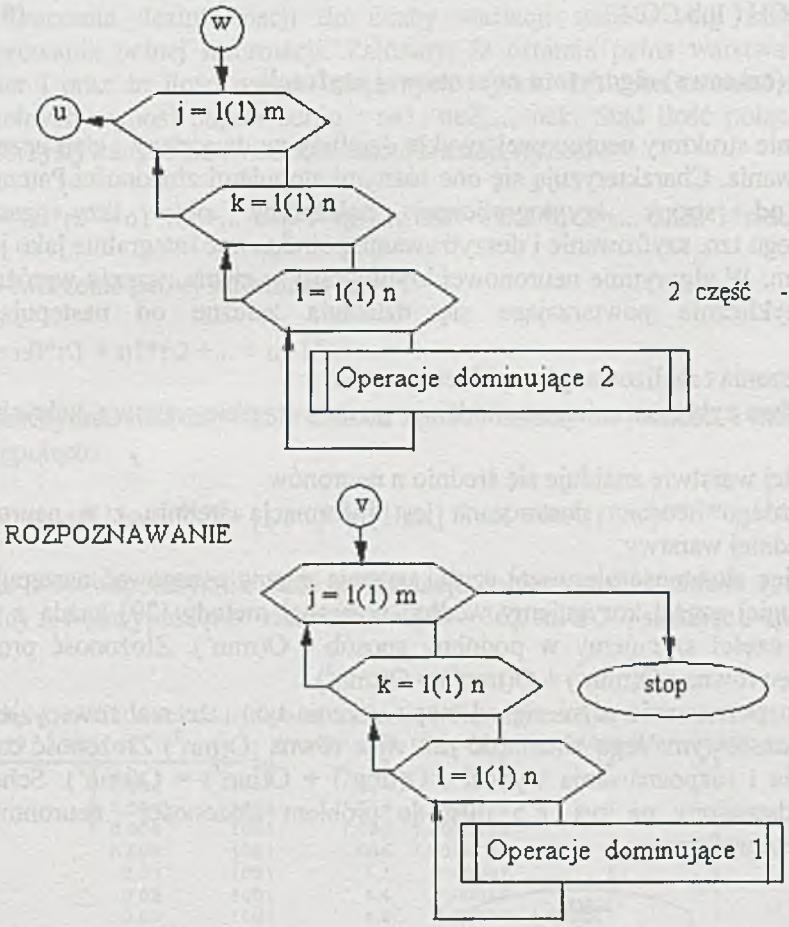
Działanie struktury neuronowej zwykle dzielimy na dwa etapy : etap uczenia i etap rozpoznawania. Charakteryzują się one różnymi stopniami złożoności.Patrząc na zagadnienie od strony kryptograficznej należałoby obie fazy zadania kryptograficznego tzn. szyfrowanie i deszyfrowanie potraktować integralnie jako jeden spójny algorytm. W algorytmie neuronowej kryptografii w etapie uczenia wyróżnimy następujące cyklicznie powtarzające się działania zależne od następujących parametrów :

- czas uczenia t realizowany w cyklach uczenia
- w każdym cyklu uczenia przechodzimy przez wszystkie warstwy, których jest m
- w każdej warstwie znajduje się średnio n neuronów
- do każdego neuronu dostarczana jest informacja średnio z n neuronów poprzedniej warstwy

Tak więc złożoność pierwszej części uczenia można oszacować następująco: $O(tmn^2)$.W drugiej części korygujemy według wybranej metody [29] każdą z wag. Złożoność tej części szacujemy w podobny sposób : $O(tmn^2)$. Złożoność procesu uczenia jest więc równa : $O(tmn^2) + O(tmn^2) = O(tmn^2)$

Etap rozpoznawania różni się od etapu uczenia tym , że realizowany jest w jednym cyklu czasowym. Jego złożoność jest więc równa :$O(mn^2)$ Złożoność całego procesu uczenia i rozpoznawania wynosi : $O(tmn^2) + O(mn^2) = O(tmn^2)$. Schemat blokowy przedstawiony na rys. 8.5 ilustruje problem złożoności neuronowego algorytmu kryptograficznego.





Rys. 8.5 Schemat blokowy ilustrujący złożoność algorytmu szyfrowania i deszyfrowania neuronowego (i – numer cyklu czasowego, j- numer warstwy, k, l – numery neuronów w sąsiadujących warstwach)

Wnioski

1. Neuronowa kryptografia to elastyczne narzędzie do zabezpieczenia danych pozwalające na wielokrotne i wielorakie rozdzielenie elementów szyfracji i deszyfracji, uwzględniające kryterium veta zarówno od strony odbiorcy jak i nadawcy zaszyfrowanej informacji.

2. Jakość zabezpieczenia (8.7) poprzez zastosowanie klucza neuronowego jest porównywalna z jakością zabezpieczenia przy pomocy znanych jawnych i prywatnych kluczy[16]
3. Złożoność algorytmu szyfracji (deszyfracji) bazującego na strukturze neuronowej jest o rząd wyższa $\{O(mn^2) > O(n^2)\}$ niż złożoność klasycznych algorytmów [29] co oczywiście wpływa na wydłużenie czasu operacji kryptograficznych.
4. Elastyczność modyfikowania kluczy neuronowych jest o dwa rzędy większa niż w klasycznych algorytmach co wynika z modyfikowalnej liczby wag , którą oszacować można następująco: $O(mn^2)$ $\{O(mn^2) >> O(m)\}$
5. Wpływ oddziaływania dezinformacji na stopień zabezpieczenia danych jest niewielki (8.10) i (8.11) i możemy go pominąć przy analizie jakości procedur kryptograficznych.

9. Neuronowe prognozowanie scenariuszy wspomagające zarządzanie

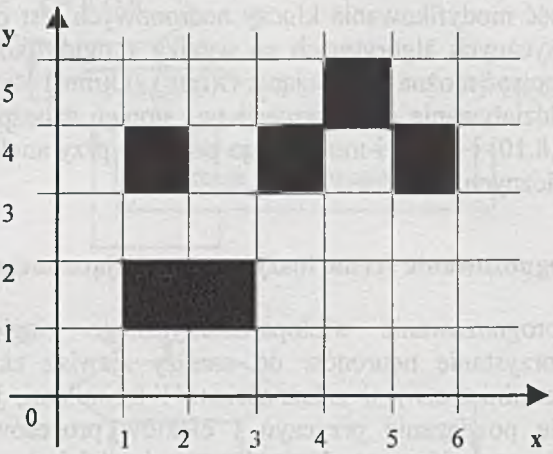
Metody prognozowania wieloparametrycznego ciągle się rozwijają i modyfikują. Wykorzystanie neuronów do analizy zjawisk ekonomicznych, które trudno opisać deterministycznymi zależnościami i formułami, jest w dużej mierze uzasadnione. Takie powiązanie przyczyn i efektów procesów ekonomicznych z powodzeniem może zastąpić pracę ekspertów tym bardziej, że struktury neuronowe potrafią się same uczyć oraz są w stanie wyciągać wnioski na podstawie niepełnej informacji wejściowej. Scenariuszem w danym przypadku będzie sekwencja wartości wybranych parametrów ekonomicznych Si = {x1, x2,..., xn}, gdzie x1 - znaczenia parametrów lub wskaźników. Badania można rozbudować, poszukując naj-bardziej optymistycznego i najbardziej pesymistycznego scenariusza (Sopt, Spes).Jak w każdej prognozie, danymi wejściowymi będą informacje dotyczące bezpośrednio przeszłych doświadczeń.

9.1 Strategia tworzenia prognozy neuronowej

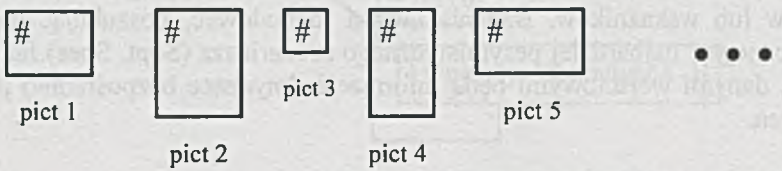
Zdajemy sobie sprawę, że przy wieloparametrycznym prognozowaniu mamy do czynienia z wielo-indeksowymi macierzami danych. Z uwagi na fakt, iż macierze neuronów to zwykle zaledwie dwu- lub trójwymiarowe struktury, wygodnie będzie się posłużyć mapami (obrazami) parametrów ekonomicznych. Przykładowo można założyć, że mapa parametrów będzie miała dwuwymiarową postać (rys.9.1). Na obu współrzędnych umownie rozmieścimy numery dyskretnych podprzedziałów wartości parametrów ekonomicznych.

Na przedstawionym rysunku widzimy nie funkcyjną zależność parametrów w odniesieniu do obu osi $x, y(y \neq f(x), x \neq g(y))$. Zależność ta przypomina obraz

stanowiący zwykle strukturę wejściową - wyjściową w procedurach roz-poznawania obrazów. Jednocześnie zdajemy sobie sprawę, iż takich struktur danych może być kilka, kilkanaście itd. w zależności od typu i zakresu prognozy (rys. 9.2).
Z każdym z wyżej przedstawionych układów związżemy strukturę neuronową rozpoznającą obrazy [23,34] nawet na podstawie niepełnych lub skażonych chaosami danych.



Rys.9.1. Dwuwymiarowa postać zależności parametrów



Rys.9.2. Układy danych wejściowych i wzorcowych, gdzie # symbolizuje obraz zależności parametrów ekonomicznych

Dane wzorcowe dla każdej struktury neuronowej to zbiór macierzy $W_{ij}^{(k_l)}$, gdzie k_l - numer wzorca l-tego układu, a i, j to współrzędne parametrów tworzących mapę. Elementy macierzy mogą przyjmować wartości (1,0) lub (1,–1) zgodnie z zapisem:

$$\overline{W}_{ij}^{(k_l)} = \begin{cases} 1 \text{ - kolor czarny} \\ 0 \text{ - kolor biały} \end{cases} \qquad \overline{\overline{W}}_{ij}^{(k_l)} = \begin{cases} 1 \text{ - kolor czarny} \\ 0 \text{ - kolor biały} \end{cases}$$

Przyjmując, że zakres zmian parametrów pomocniczych można przedstawić następująco:

$$1 \le i \le \max i(l) \qquad 1 \le l \le v$$
$$1 \le j \le \max j(l)$$
$$1 \le k \le \max k$$

gdzie:

l - numer układu rozpoznającego mapę (obraz),
v - ilość układów,
$\max k$ - ilość wzorców map parametrów ekonomicznych.

Na bazie powyższej informacji można oszacować ilość danych wejściowych wg następującej formuły:

$$N_{inp} = N_{inppict} + N_{inps} + N_{inpact} =$$

$$= \max k \cdot \sum_{l=1}^{V} \max i(l) \cdot \max j(l) + \sum_{m=1}^{p^+} n_m^+ + \sum_{m=1}^{p^-} n_m^- +$$

$$+ \sum_{l=1}^{V} \max i(l) \max j(l) = (\max k + 1) \sum_{l=1}^{V} \max i(l) \max j(l) + \qquad (1)$$

$$+ 2 \sum_{m=1}^{p} n_p \quad \left\{ \begin{array}{ll} N_{inppict} & \text{- ilość danych do opisu map wzorcowych} \\ N_{inps} & \text{- ilość danych do opisu scenariuszy wzorcowych} \\ N_{inpact} & \text{- ilość danych dot. aktualnej, badanej sytuacji} \end{array} \right\}$$

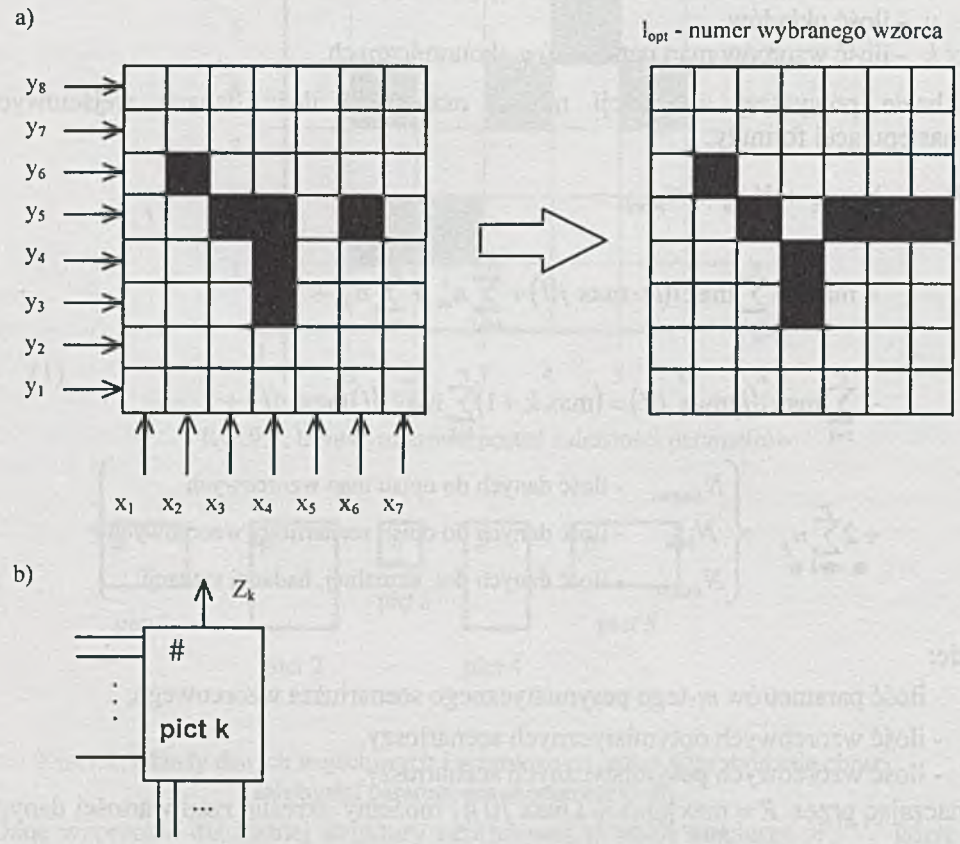
gdzie:

n_m^+ - ilość parametrów m-tego pesymistycznego scenariusza wzorcowego,
p^+ - ilość wzorcowych optymistycznych scenariuszy,
p^- - ilość wzorcowych pesymistycznych scenariuszy.

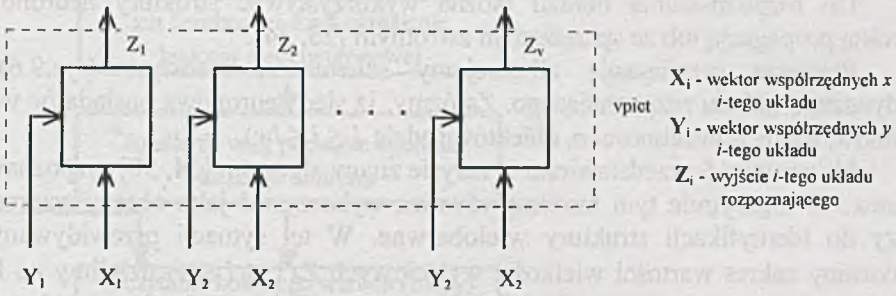
Oznaczając przez $R = \max_{1 \le l \le V} \{\max i(l), \max j(l)\}$, możemy określić rząd wartości danych wejściowych jako $\Theta(\max k \cdot V \cdot R^2)$.

Dane dotyczące aktualnych parametrów wejściowych, których rząd można oszacować jako $\Theta(V \cdot R^2)$, podawane będą na odpowiednie wejścia neuronowej sieci każdego z V układów rozpoznających obrazy. Sieć neuronowa zgodnie z „wyuczonymi" wzorcami wybiera ten, który jest najbliżej aktualnie badanej sytuacji ekonomicznej (rys. 9.3).Kompleks układów rozpoznających mapy parametrów

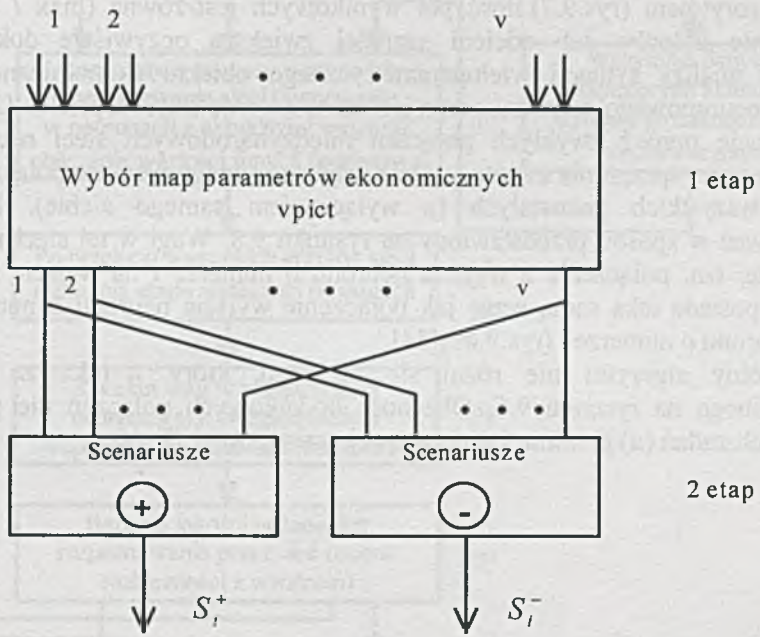
ekonomicznych zbioru struktur neuronowych na danym etapie można przedstawić schematycznie (rys. 9.4).Struktura przedstawiona na rysunku 9.4 realizuje pierwszy etap procedury prognozującej, dostarczając wektor danych $\{Z_1,Z_2,...,Z_n\}$, na bazie którego wybierzemy dwa skrajne scenariusze dotyczące opisu przewidywanej przyszłościowej sytuacji. Zgrupowanie scenariuszy wzorcowych w dwóch zbiorach (przewidywań optymistycznych i pesymistycznych) nie tylko pozwoli na bardziej szczegółową i wyważoną ocenę sytuacji, lecz także przyspieszy proces konwersji. Etapy realizacji zadania można więc przedstawić jak na rysunku 9.5.



Rys. 9.3. Wybrany wzorzec mapy parametrów ekonomicznych (a) oraz umowny schemat wyboru wzorca najbardziej zbliżonego do badanej sytuacji ekonomicznej (b)



Rys. 9.4. Schemat ideowy układów rozpoznających mapy parametrów ekonomicznych



Rys. 9.5. Realizacja procedury wyboru najbardziej prawdopodobnego optymistycznego i pesymistycznego scenariusza

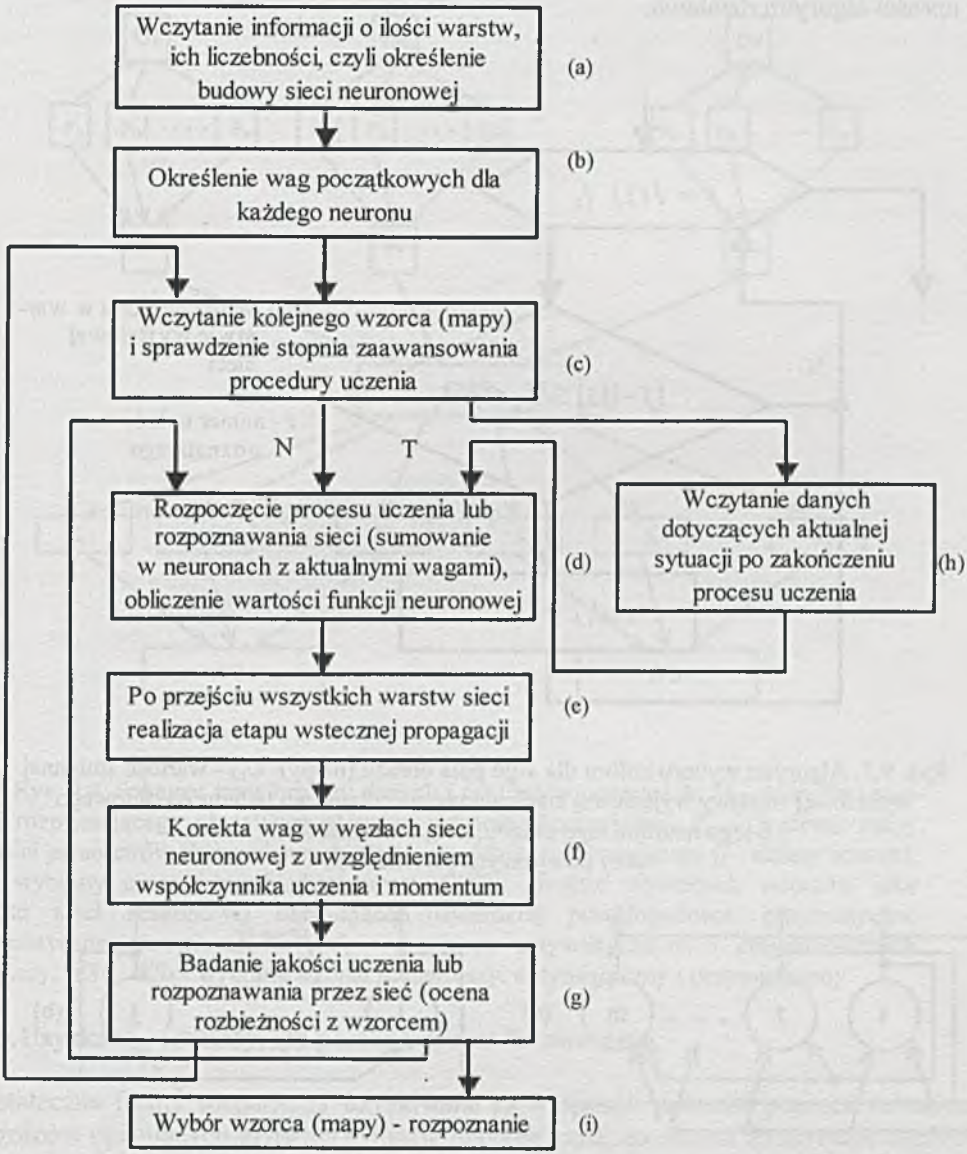
9.2 Realizacja neuronowych klasyfikatorów obrazów i scenariuszy

Do rozpoznawania obrazu można wykorzystywać struktury neuronowe z odwrotną propagacją lub ze sprzężeniem zwrotnym [23, 34].

Pierwsze rozwiązanie zilustrujemy schematem ideowym (rys.9.6) dla pojedynczego układu rozpoznającego. Załóżmy, iż sieć neuronowa posiada *lw* warstw neuronów, każda o liczebności n_i obiektów (gdzie $1 \le i \le lw$).

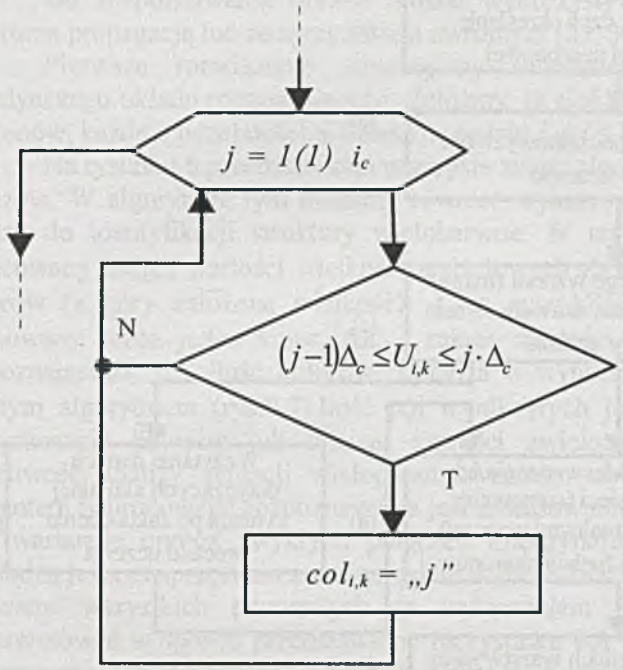
Na rysunku 6 przedstawiono w zarysie znany algorytm [34, 30] rozpoznawania obrazów. W algorytmie tym możemy również wykorzystać jako obrazy wzorcowe i obrazy do identyfikacji struktury wielobarwne. W tej sytuacji przewidywany lub oszacowany zakres wartości wielkości wyjściowych $Z_i (1 \le i \le v)$ dzielimy na liczbę kolorów (z góry założoną wielkość): $\Delta_c = \max(\Delta Z_i)/i_c$, gdzie Δ_c - podprzedział zajmowany przez jeden kolor, ΔZ_i - zakres wartości wyjściowych *i*-tego układu rozpoznającego, i_c - ilość kolorów. Decyzja o wyborze koloru może być opisana prostym algorytmem (rys.9.7).Ilość pól wynikowych jest równa (max i × max j). Dopuszczenie kolorów lub odcieni szarości zwiększa oczywiście dokładność i wnikliwość analizy sytuacji wieloparametrycznego obiektu ekonomicznego.Innym wariantem neuronowego rozpoznawania jest zastosowanie sieci Hopfielda [23, 27]. W tym wariancie oprócz zwykłych połączeń międzynarodowych sieci neuronowych dochodzą jeszcze sprzężenia zwrotne, gdzie wyjście każdego neuronu połączone jest z wejściem wszystkich pozostałych (z wyłączeniem samego siebie). Można to zinterpretować w sposób przedstawiony na rysunku 9.8. Wagi w tej sieci muszą być symetryczne, tzn. połączenie z wyjścia neuronu o numerze *i* na wejście neuronu o numerze *j* posiada taką samą wagę jak połączenie wyjścia neuronu o numerze *j* na wejście neuronu o numerze *i* (rys.9.8b)[34].

Ogólny algorytm nie różni się od tego, który wynika ze schematu przedstawionego na rysunku 9.6. Obecność dodatkowych połączeń nie przysparza dodatkowych zadań (*d*) pomimo każdorazowej zmiany wagi W_{ji} (*f*).

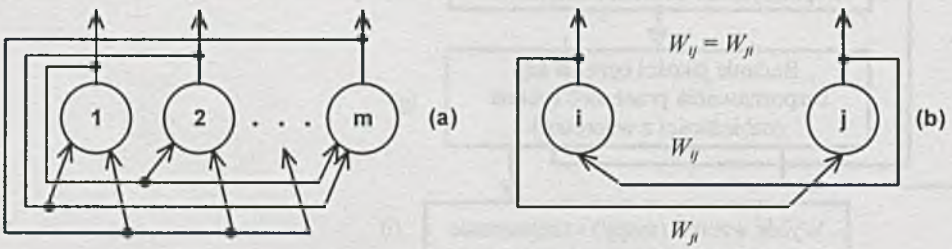


Rys. 9.6. Schemat ideowy neuronowego rozpoznawania obrazów

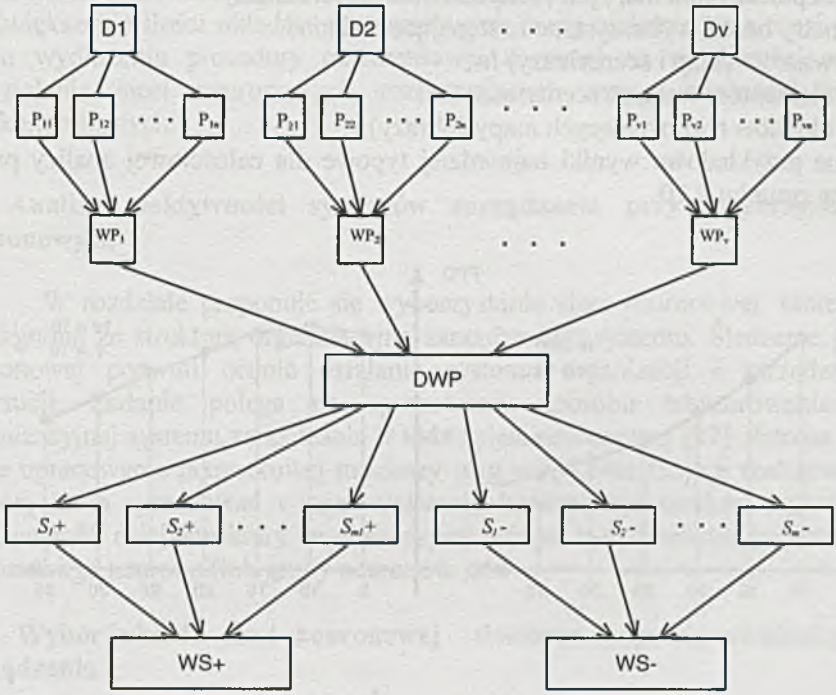
Obecność sprzężeń zwrotnych skłania do wyeliminowania etapu wstecznej propagacji, co uprości algorytm działania.



Rys. 9.7. Algorytm wyboru koloru dla k-go pola obrazu (mapy): $U_{i,k}$ - wartość zmiennej wyjściowej warstwy wyjściowej sieci neuronowej dla i-tego układu wyjściowego i k-tego neuronu tego układu, $col_{i,k}$ - wartość kodu koloru przy powyższych parametrach



Rys. 9.8 Połączenia sieci Hopfielda (ze sprzężeniami zwrotnymi)



Rys. 9.9. Schemat transformacji danych i rezultatów pośrednich: D_i - dane dla i-tego układu rozpoznającego obrazujące aktualną sytuację ekonomiczną, P_{ij} – wzorzec mapy zależności parametrów ekonomicznych dla i-tego układu rozpoznającego (j - numer wzorca), WP_i - wybrany przez i-ty układ wzorzec, DWP - wektor wybranych wzorców jako dane do sieci neuronowej określającej najbardziej prawdopodobne pesymistyczne i optymistyczne scenariusze, S_i+, S_i - - wzorce optymistycznych i pesymistycznych scenariuszy, WS+, WS- - wybrane wzorce scenariuszy: optymistyczny i pesymistyczny

9.3 Uzyskane rezultaty, ich postać i proces formowania

Ostateczne formy scenariuszy uzyskiwane są w sposób pośredni poprzez zaistnienie wzorców opisujących aktualną sytuację ekonomiczną, co można zilustrować ogólnym schematem transformacji (rys. 9.9).

Wnioski

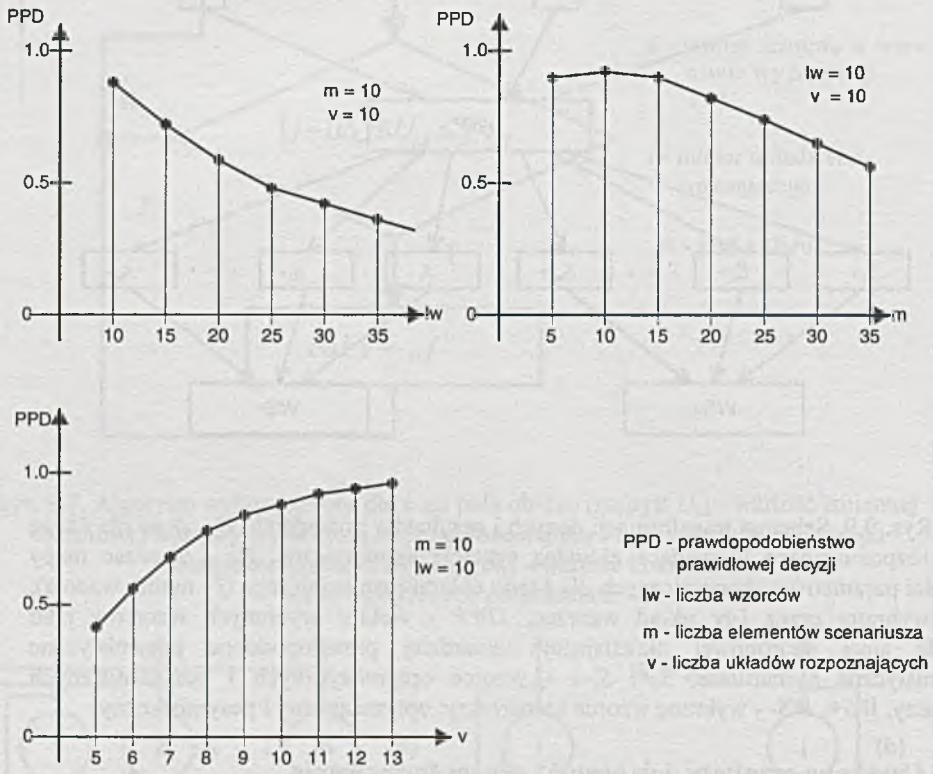
1. Dla przeprowadzenia badań posłużono się dwoma typami sieci neuronowej opisanymi w rozdziale 3 i rozpoznającymi wzorce. Jako efektywność działania

przyjęto prawdopodobieństwo podejmowania właściwych decyzji zarówno na etapie rozpoznawania map, jak i rozpoznawania scenariuszy.

Jako parametry badań wykorzystano następujące zmienne:

- liczbę wzorów (map i scenariuszy) *lw*,
- liczbę elementów wektora scenariusza *m*,
- liczbę układów rozpoznających mapy (obrazy) *v*.

Wybrane przykładowe wyniki najbardziej typowe dla całościowej analizy przedstawiono na rysunku 9.10.



Rys. 9.10. Generalne tendencje zachowania neuronowej struktury rozpoznającej prognostyczne scenariusze

2. Wykorzystanie do prognozowania „podwójnego" wnioskowania dotyczącego rozpoznawania zarówno map, jak i scenariuszy wydłuża dwukrotnie procedurę działania, lecz znacznie (*v*-krotnie) zwiększa ilość pośrednich informacji, które mogą być wykorzystane do głębszej analizy ekonomicznej.
3. Pomimo celowego wzbogacenia zasobu informacji stanowiącej dane do drugiego etapu rozpoznawania (etapu rozpoznawania scenariuszy) efektywność działania

struktury sieciowej nieznacznie zmalała, co wynika ze zwiększenia ilości operacji przetwarzania kumulujących sekwencyjne błędy.

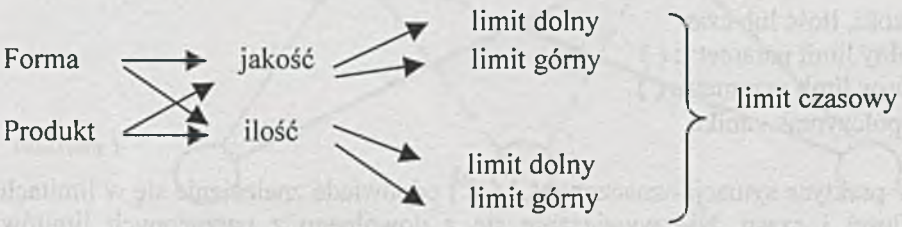
4. Zwiększenie ilości układów rozpoznających mapy niekoniecznie musi prowa-dzić do wydłużenia procedury obliczeniowej z uwagi na możliwości paralelizacji działania sieci neuronowych rozpoznających mapy zależności parametrów ekonomicznych.

10. Analiza efektywności systemów zarządzania przy wykorzystaniu sieci neuronowych

W rozdziale proponuje się wykorzystanie sieci neuronowej, której struktura jest zgodna ze strukturą organizacyjną zarządzanego systemu. Śledzenie pracy sieci neuronowej pozwoli ocenić działanie systemu organizacji i zarządzania danej instytucji. Zadanie polega na opracowaniu sposobu odwzorowania struktury organizacyjnej systemu zarządzania w układ sieci neuronowej [27] . Istotną rzeczą jest także opracowanie początkowej macierzy wag w(i,j) oraz funkcji realizowanej przez neuron, która syntezować winna działanie konkretnego modułu organizacyjnego. Efektywność całej struktury organizacyjnej winna być oceniona poprzez działanie wyjściowego neuronu (lub grupy neuronów).

10.1 Wybór układu sieci neuronowej dostosowanego do struktury systemu zarządzania

W każdej strukturze organizacyjnej poszczególne działy winny wykonywać określone funkcje, których rezultatem będą różnego formy lub produkty. Pod pojęciem forma w tym rozdziale będą rozumiane różnego rodzaju dokumenty, rezultaty realnych działań i analiz. Pod pojęciem produktu będzie rozumiana usługa lub towar. Zarówno forma jak i produkt będą tutaj się charakteryzować konkretną oceną jakości o ilości oraz odpowiadające im limity. Innym istotnym limitem będzie obarczony również czas:



Rys.10.1. Charakterystyki strategiczne form i produktów

Powyższe definicje i uproszczenia mają służyć uzyskaniu większej klarowności i jednorodności przy konwersji struktury organizacyjnej w układ sieci neuronowej. Taki sposób uproszczonego opisu każdego modułu struktury organizacyjnej prowadzi do uzyskania możliwości scalania współpracujących ze sobą modułów, jeżeli tworzą grupę generującą określone formy lub produkty i nie są związane bezpośrednio z innymi funkcjonalno – organizacyjnymi modułami. Takie połączenie zmniejsza szanse oceny wewnętrznych zintegrowanych ze sobą działów lub modułów organizacyjnych.

Kolejnym zadaniem staje się zgeneralizowanie funkcji, którą posłuży się neuron do opisu efektywności działania. Jedną z propozycji takiej funkcji może być następująca zależność :

$$f(x) = k_j \cdot k_i \cdot \delta_j \cdot \delta_i \cdot \delta_t . \qquad (10.1)$$

gdzie :

k_j -współczynnik jakości,
k_i - współczynnik ilości,
δ_j - współczynnik limitu jakości,
δ_i - współczynnik limitu ilości,
δ_t - współczynnik limitu czasu,

$$\delta_j = \begin{cases} 1 - \varepsilon \ \text{dla} \ \{ a_{(\,)} \le (\,) \le b_{(\,)} \}, \\ \varepsilon \quad \text{dla} \ \{ (\,) < a_{(\,)} \ \text{lub} \ (\,) > b_{(\,)} \} , \end{cases}$$

gdzie :

( ) - jakość, ilość lub czas
$a_{(\,)}$ - dolny limit parametru ( )
$b_{(\,)}$ - górny limit parametru ( )
ε - współczynnik zaniku.

W praktyce sytuacji oznaczonej (1 - ε) odpowiada znalezienie się w limitach jakości, ilości i czasu. Nie wywiązanie się z dowolnego z narzuconych limitów powoduje zbliżenie wartości funkcji f(x) do zera . W takie sytuacji musi być uwzględniona skala zróżnicowania minimalnego progu współczynników k_j i k_i oraz

wielkości współczynnika zaniku ε. Jeżeli progi współczynników jakości i ilości oznaczymy odpowiednio przez Δ_j i Δ_i to możemy oszacować ich wartość:

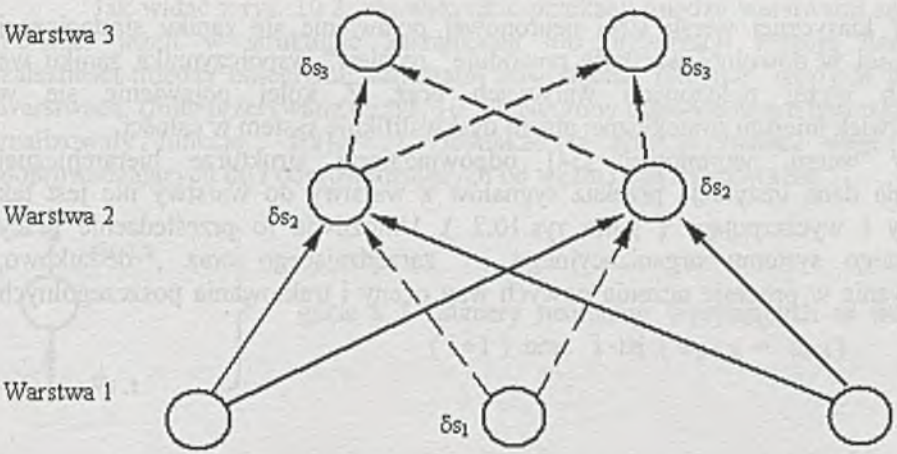
$$\Delta_j = (\max\{ k_j \} - \min\{ k_j \}) / n_j \,,$$
$$\Delta_i = (\max\{ k_i \} - \min\{ k_i \}) / n_i \,, \qquad (10.2)$$

gdzie

n_j, n_i - liczba progów jakościowych i ilościowych.

Z powyższego wynika, iż $\Delta_j >> \varepsilon$ oraz $\Delta_i >> \varepsilon$. Zależności te można określić jako zasadę skutecznej eliminacji to jest jako efekt przekroczenia limitów.

Jeżeli w strukturze organizacyjnej przekroczone zostaną limity strategiczne powoduje to dyskwalifikację systemu, jeżeli zaś przekroczone zostaną limity będące na przykład naruszeniem pewnych planowanych lub prognozowanych założeń, nie burzących strategii działania systemu , takie stany można uznać za dopuszczalne. Wynika z tego , iż żadnym dodatkowym działaniem nie można zrekompensować naruszenia limitów strategicznych, w przeciwieństwie do naruszenia limitów niestrategicznych. Wobec tego musi nastąpić przekaz zaniku wartości do warstw odpowiadających wyższym poziomom hierarchii organizacyjnej :



Rys10.2. Schemat zaniku przekazu w strukturze modelu neuronowego (δ_k - współczynnik przekazu strategicznego dla k – tej warstwy)

Z rysunku wynika, że zanik strategicznej skuteczności winien być przekazywany do wyższych warstw. Przekaz ten może być realizowany w różny sposób. Przykładowo :

a) $\delta s_k = \delta s_{k-1}$ c) $\delta s_k = (\delta s_{k-1})^2$ (10.3)

b) $\delta s_k = (\delta s_1)^k$ d) $\delta s_k = (k-1)/k * \delta s_{k-1}$

Powyższe przykłady sugerują, iż można również klasyfikować strategiczne znaczenie danego produktu lub formy oraz związanego z nimi działu systemu organizacyjnego. Niezależnie od intensywności zaniku efektywności działania przy przekazywaniu go do kolejnych wyższych warstw stosujemy zasadę rozsiewu współczynników δs_k (rys.10.1.)[27].
Funkcja opisująca skuteczność działania poszczególnych działów i realizowana przez odpowiadające im neurony (1) może być zmodyfikowana w następujący sposób:

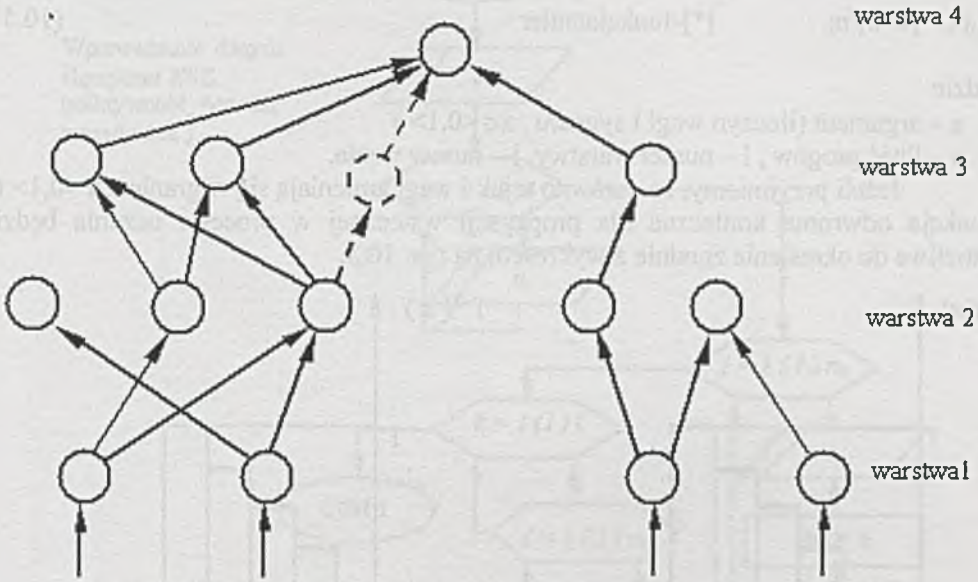
$$f(x) = k_j * k_i * \delta_j * \delta_i * \delta_t * \delta s_k . \quad (10.4)$$

gdzie

k – numer warstwy, w której zlokalizowany jest neuron opisujący funkcjonowanie danego działu

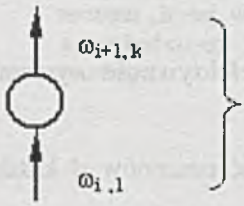
W klasycznej wersji sieci neuronowej pojawienie się zaniku strategicznej efektywności w dowolnej warstwie powoduje „rozsiew" współczynnika zaniku we wszystkich wyżej położonych warstwach oraz. Z kolei pojawienie się w którymkolwiek miejscu strategicznej atrofii dyskwalifikuje system w całości.

W wersji neuronowej [34] odpowiadającej strukturze hierarchicznej zarządzania daną instytucją przekaz sygnałów z warstwy do warstwy nie jest tak kompletny i wyczerpujący (patrz rys.10.2). Umożliwia to prześledzenie pracy rzeczywistego systemu organizacyjnego i zarządzającego oraz , dodatkowo, wypracowanie w procesie uczenia nowych wag oceny i traktowania poszczególnych działów.



Rys 10.2. Neuronowy odpowiednik struktury działania

Jak widać z rys. 10.2. nie wszystkie przekazy między warstwami są aktywne. Ponadto jeżeli w strukturze zarządzania lub organizacji istnieją bezpośrednie zależności między odległymi warstwami powołujemy fikcyjne węzły w pośrednich warstwach (linie przerywane na rys. 10.2). Neurony odpowiadające tym węzłom będą realizowały funkcję f(x) = 1. Jednakże w tym przypadku wagi sygnałów doprowadzających do i odprowadzających od węzła będą równoważne.



gdzie k ,l – numery neuronów wysyłających w warstwach (i+1) oraz i -tej ($\omega_{i+1,k} = \omega_{i,1}$) .

W tej sytuacji wagi nie mają szczególnego znaczenia, ponieważ dany neuron opisuje nieistniejący dział organizacyjny. Wprowadzenie tego neuronu jest istotne z uwagi na proces uczenia sieci neuronowej bazujący na przekład na wstecznej

propagacji [34]. Funkcja konwersji stosowana w naszym wariancie sieci neuronowej to funkcja wieloprogramowa określona w następujący sposób:

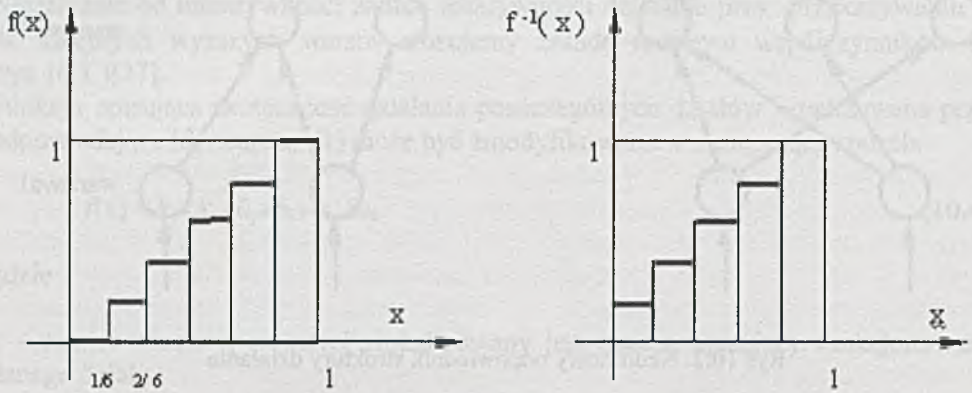
$$f_{i,j}(x)=[x*n]/n; \qquad [*]\text{-funkcjaentier} \qquad (10.5)$$

gdzie

x – argument (iloczyn wagi i sygnału , $x \in <0,1>$)

n – ilość progów , i – numer warstwy, j – numer węzła.

Jeżeli przyjmiemy, że zarówno x jak i wagi zmieniają się w granicach <0,1> to funkcja odwrotna konieczna dla propagacji wstecznej w procesie uczenia będzie możliwa do określenia zgodnie z wykresem na rys. 10.3.

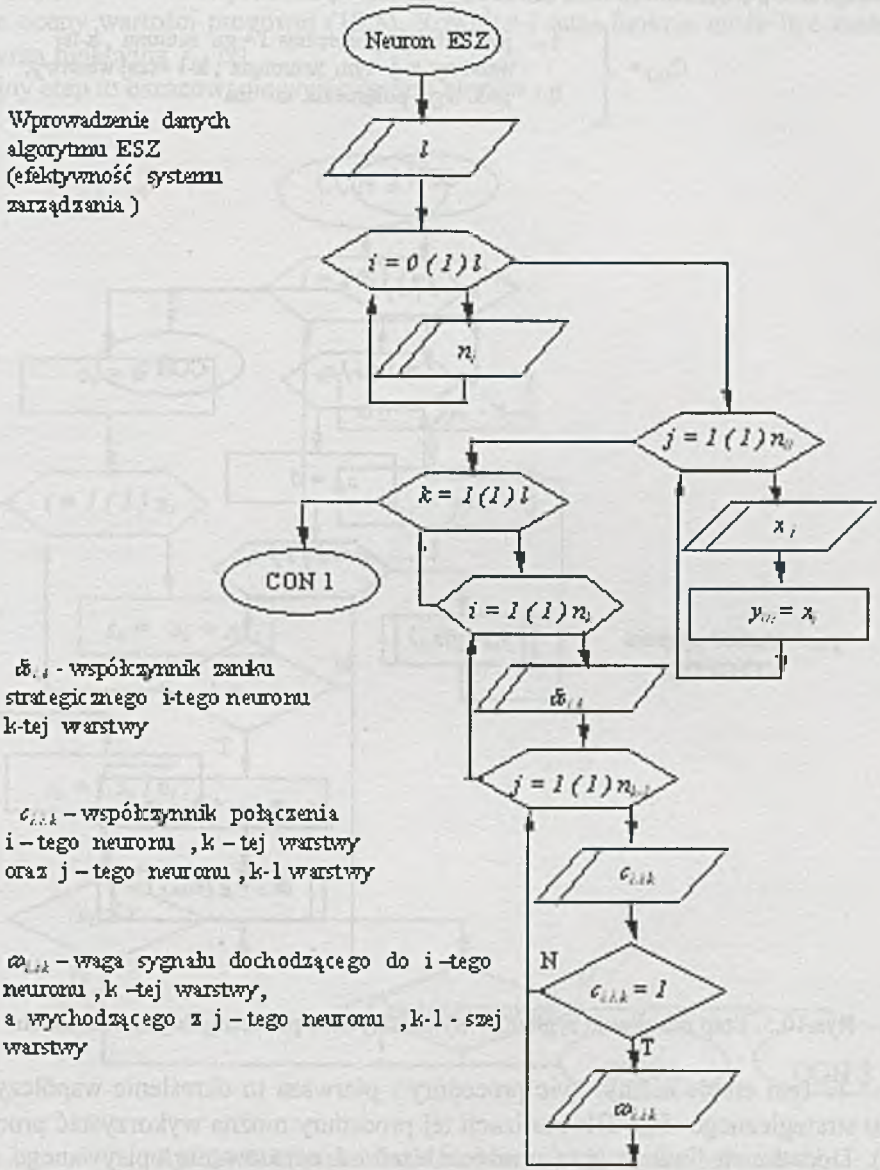


Rys. 10.3. Progowe funkcje realizowane przez neurony

Jednakże w procesie uczenia zarówno argumenty wejściowe, jak również wielkości wyjściowe oraz wagi osiągają wartości przekraczające jedność [23,34]. W takiej sytuacji obliczanie wartości funkcji odwrotnej będzie prostsze i szybsze niż procedury skalowania i określania nowych zakresów zmiennych wejściowych i wyjściowych.

10.2 Algorytm uczenia sieci neuronowej analizującej efektywność systemu zarządzania.

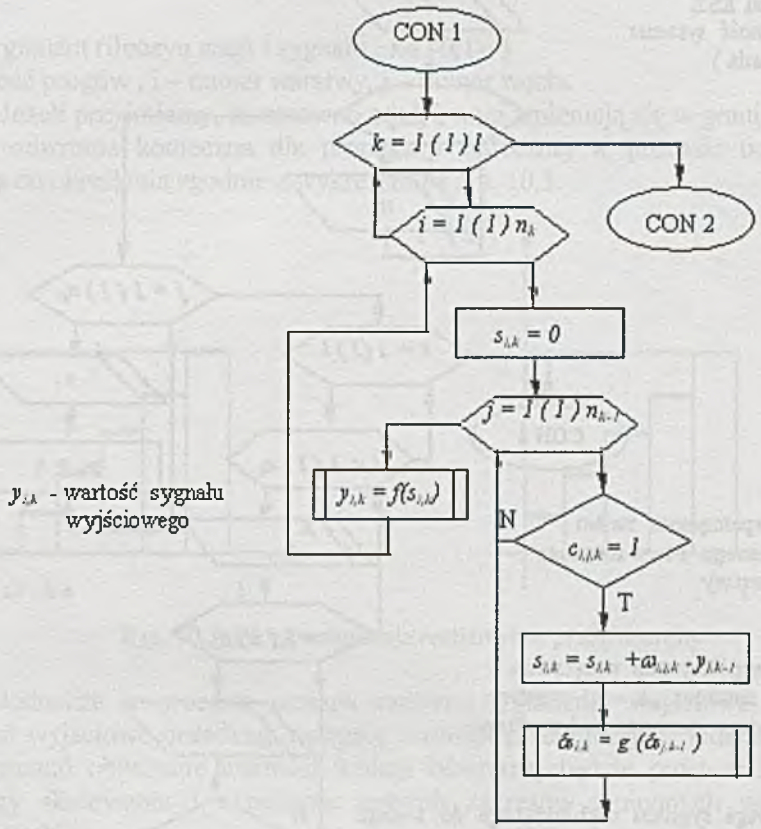
W pierwszym etapie określamy ilość warstw oraz ilość neuronów w każdej warstwie. Ponadto wprowadzamy sygnały wejściowe.



Rys. 10.4. Etap wprowadzania danych do sieci neuronowej

Współczynnik połączenia można określić w prosty sposób :

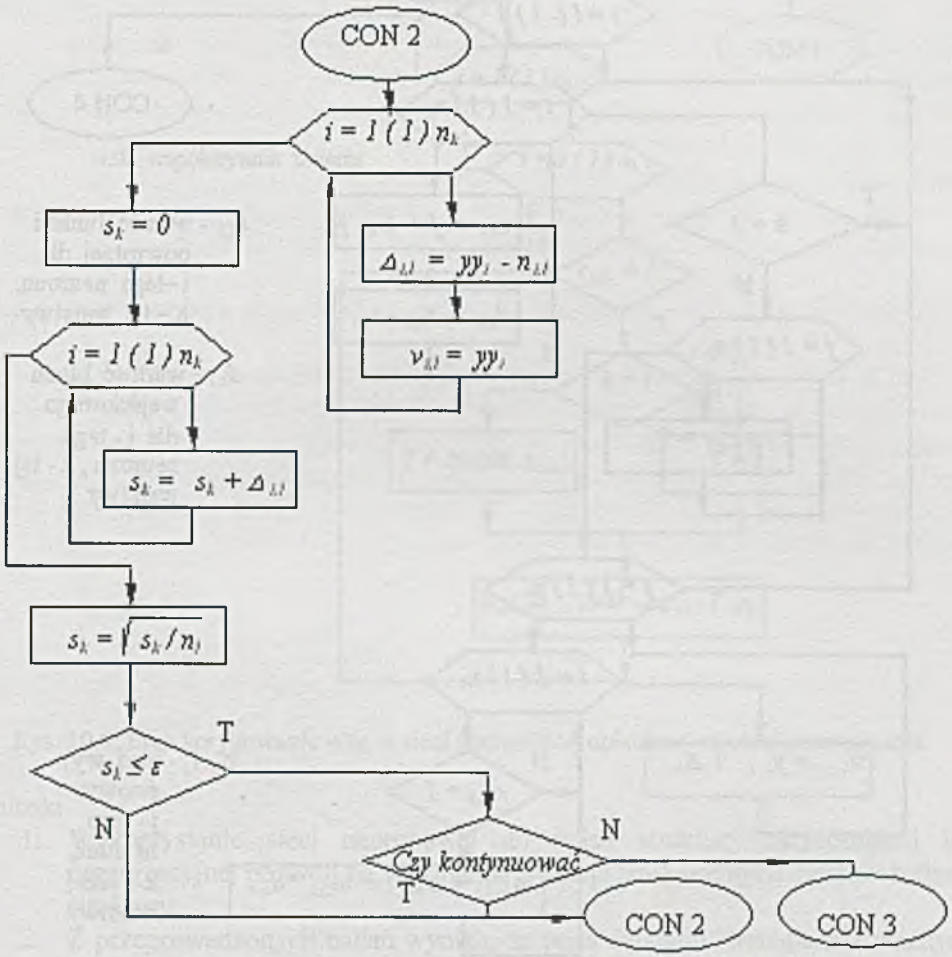
$C_{i,j,k} = \begin{cases} 1 - & \text{jeśli istnieje połączenie i – go neuronu , k-tej warstwy z j –tym neuronem , k-1 –szej warstwy,} \\ 0 - & \text{jeśli tego połączenia nie ma} \end{cases}$



Rys.10.5. Etap określania sygnałów wyjściowych w poszczególnych neuronach

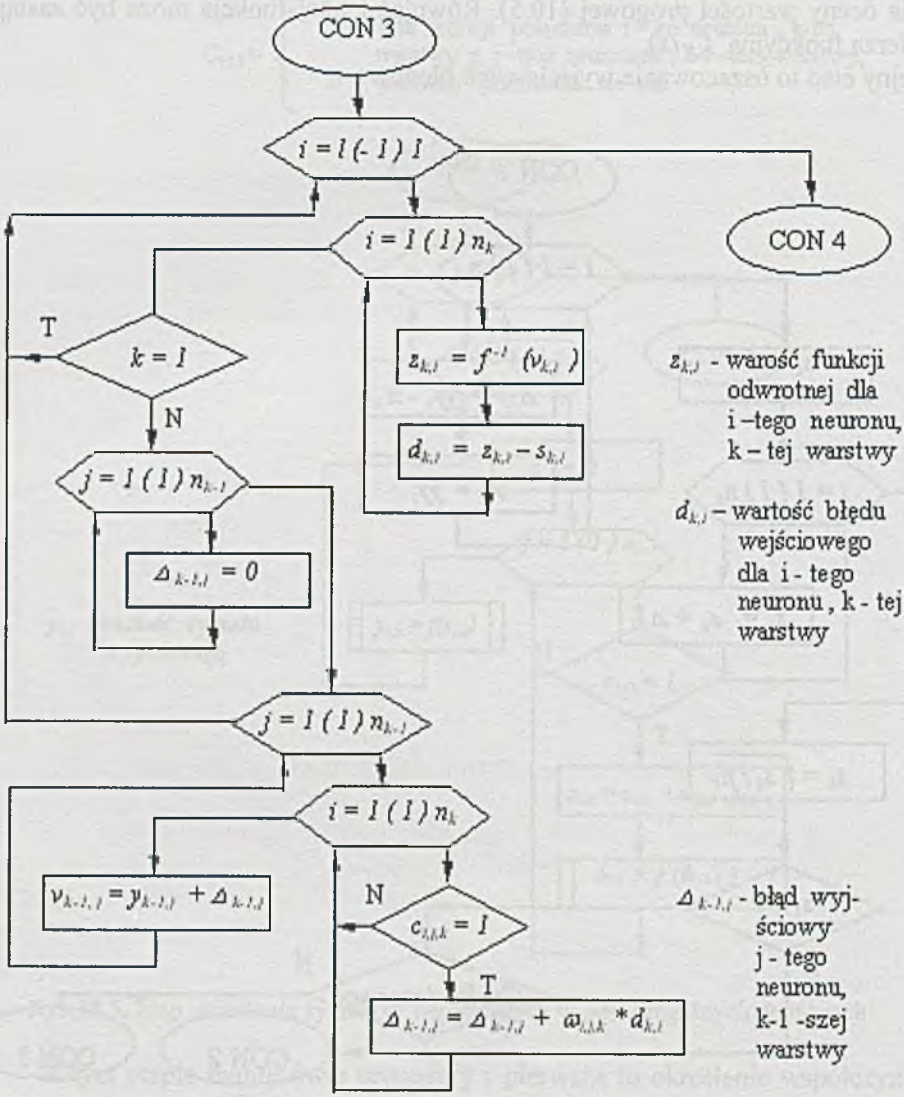
W tym etapie mamy dwie procedury : pierwsza to określenie współczynnika zaniku strategicznego $\delta s_{i,k}$. Dla realizacji tej procedury można wykorzystać procedury (10.3). Dodatkowo funkcja *g (x)* może zależeć od usytuowania opisywanego działu struktury organizacyjnej, a więc od parametrów *i* oraz *k*. W takiej sytuacji funkcja *g (x)* może zmienić może zmienić się w macierz funkcji $g_{i,k}$ *(x)*. Przy określaniu wartości funkcji *f (x)* najwygodniejszym rozwiązaniem jest przyjęcie takiej samej gradacji dla wszystkich współczynników jakości i ilości poszczególnych neuronów (działów struktury). Oczywiście nie jest to konieczne, ponieważ każdy neuron zmienia dowolne

dane wejściowe w funkcjonalne kaskady stosowanej do charakterystycznej dla danego węzła oceny wartości progowej (10.5). Również i tutaj funkcja może być zastąpiona macierzą funkcyjną $f_{i,k}(x)$.
Kolejny etap to oszacowanie wyjściowych błędów :



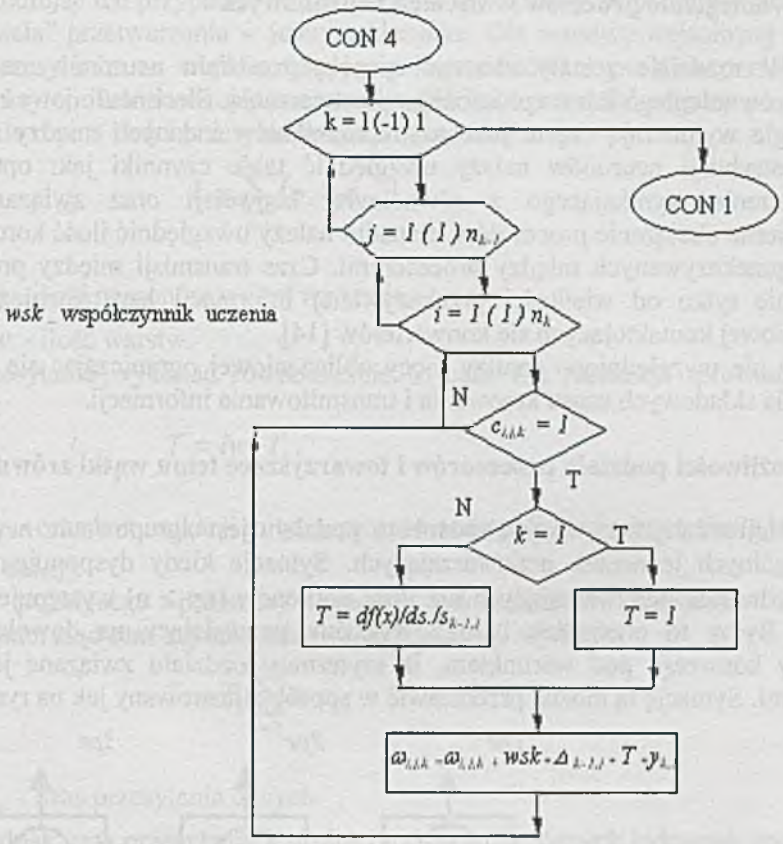
Rys. 10.6. Etap określania błędów na wyjściu sieci neuronowej

W tym momencie rozpoczyna się wsteczna propagacja ; począwszy od ostatniej warstwy szacujemy kolejno : błędy na wejściu i rozdział błędów na neurony niższej warstwy [23,34].



Rys. 10.7. Etap określania błędów wyjściowych i wejściowych dla wszystkich neuronów

Ostatni etap zakłada korektę wag, które mogą być wykorzystane do zmiany oceny strategicznych walorów opisywanego przez dany neuron działu.



Rys. 10.8. Etap korygowania wag w sieci neuronowej opisującej strukturę zarządzania.

Wnioski

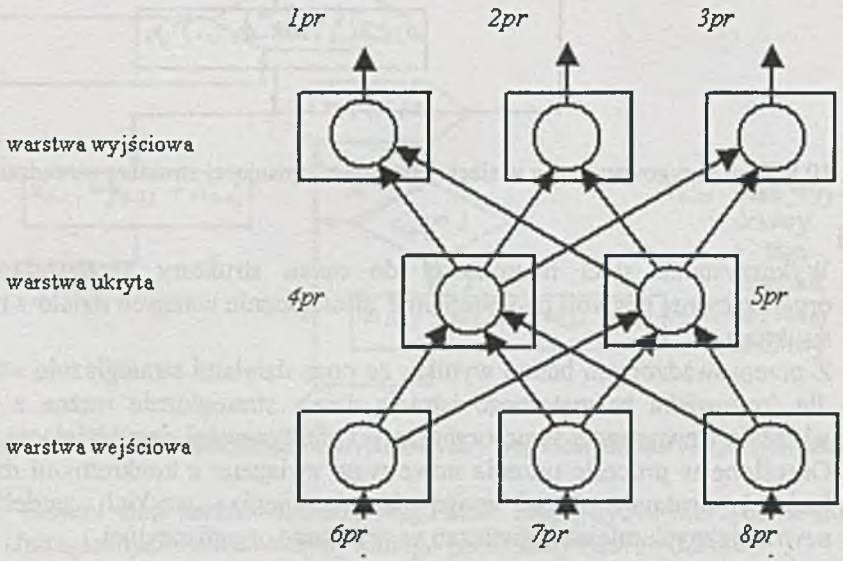
1. Wykorzystanie sieci neuronowej do opisu struktury zarządzającej lub organizacyjnej pozwoli na wyłonienie strategicznie ważnych działów badanej struktury.
2. Z przeprowadzonych badań wynika, że poza działami strategicznie ważnymi dla środowiska zewnętrznego istnieją działy strategicznie ważne z punktu widzenia wewnętrznej samo-organizacji i efektywności współdziałania.
3. Określone w procesie uczenia nowe wagi związane z konkretnymi działami badanej struktury służyć mogą do ujawnienia „wąskich gardeł" oraz newralgicznych miejsc i powiązań w strukturze organizacyjnej.

11. Zrównoleglenie procesów w sieciach neuronowych

W rozdziale zostały ukazane sposoby rozdziału neuronów na procesory struktur równoległego lub rozproszonego przetwarzania. Sieci neuronowe realizowane równolegle wymuszają częste przekazy komunikatów i danych między neuronami. Przy dystrybucji neuronów należy uwzględnić takie czynniki jak: optymalizacja przyspieszenia wynikającego z równoległej konwersji oraz związane z tym równomierne obciążenie procesorów. Ponadto należy uwzględnić ilość komunikatów i danych przekazywanych między procesorami. Czas transmisji między procesorowej zależy nie tylko od wielkości przekazywanej informacji lecz również od mocy obliczeniowej kontaktujących się konwerterów [14].
W pracy nie uwzględniono analizy mocy obliczeniowej ograniczając się jedynie do określenia składowych czasu kreowania i transmitowania informacji.

11.1 Możliwości podziału procesorów i towarzyszące temu wątki zrównoleglenia.

Najbardziej oczywistym sposobem podziału jest zgrupowanie neuronów dla poszczególnych jednostek przetwarzających. Sytuacje kiedy dysponujemy większą ilością jednostek przetwarzających niż ilość neuronów (np ≥ n) występuje niezwykle rzadko. Byłby to przypadek bardzo wygodny pozwalający na dowolne rozbicie struktury konwersji pod warunkiem, iż kryterium rozdziału związane jest tylko z neuronami. Sytuację tą można przedstawić w sposób zilustrowany jak na rys. 11.1.



Rys. 11.1. Dystrybucja neuronów na oddzielne procesory.

Studiując ten przypadek można określić ilość przesyłań pomiędzy procesorami dla „przejścia" przetwarzania w jednym kierunku. Dla warstwy wejściowej mamy po jednym przesyłaniu na wejście (przyjęcie argumentów) i po dwa przesyłania do warstwy ukrytej. Tak analizując można określić ilość przesyłań między procesorami, która będzie równa ilości przesyłań między neuronami:

$$L = \sum_{i=1}^{lw-1} (n_i \times n_{i+1}), \tag{11.1}$$

gdzie

n_i - ilość neuronów w i-tej warstwie,

lw - ilość warstw.

Jeżeli przesyłania wykonać równocześnie to czas ich realizacji sprowadzi się do wielkości:

$$T = lw \cdot T_k, \tag{11.2}$$

gdzie

T_k - czas komunikacji między procesorami (połączenie i przesłanie jednej danej).

Jednakże pojedyncze procesory pracują oddzielnie i przesyłają informacje sekwencyjnie, stąd czas zsynchronizowanej transmisji wyniesie:

$$T = T_p \sum_{i=2}^{lw} n_i, \tag{11.3}$$

gdzie

T_p - czas przesyłania danych.

Zwykle jednak czas przesyłania jest niejednakowy dla różnych jednostek sprzężonych ze sobą. W związku z tym wyrażenie określające czas przyjmie postać:

$$T = \sum_{i=2}^{lw} \left\{ \max_{1 \le k < n_{i-1}} \sum_{j=1}^{n_i} T_p^{(i)}(k, j) \right\}, \tag{11.4}$$

gdzie

$T_p^{(i)}(k, j)$ - czas przesyłania między procesorem o numerze k w warstwie $(i-1)$, a procesorem j w warstwie i.

Oczywiście czas ten odnosi się do grupy lw warstw. Problem należy rozpatrywać z uwzględnieniem ilości danych między poszczególnymi procesorami. W najprostszym wariancie ilość tych danych to tylko jedna wejściowa wartość zmiennej przesyłanej z warstwy niższej do wyższej. Jednakże nie zawsze tak jest, ponieważ przesyłane mogą również być zmniejszające się współczynniki uczenia, momentum etc. W takich sytuacjach wydłuża się czas komunikacji. Tym niemniej realizowane jest to równolegle i zasada opisana wzorem (11.4) pozostanie w mocy.

W sytuacji gdy dysponujemy mniejszą ilością procesorów niż liczba neuronów w sieci, problem sprowadza się do podziału pomiędzy procesory całej ilości neuronów, tak aby zapewnić spełnienie następujących warunków:

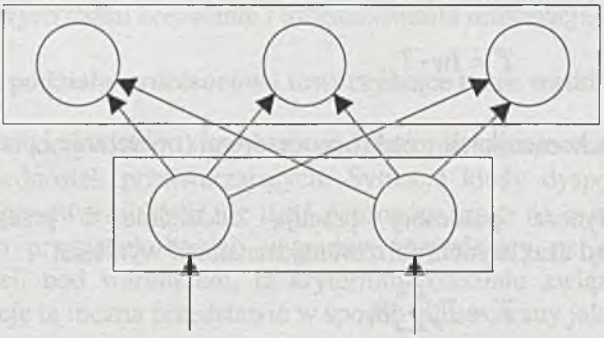
- równomierność obciążenia procesorów,
- minimalny czas tracony na komunikację między procesorami,
- minimalny czas realizacji zadania.

Sposób podziału neuronów między procesorami może mieć charakter:

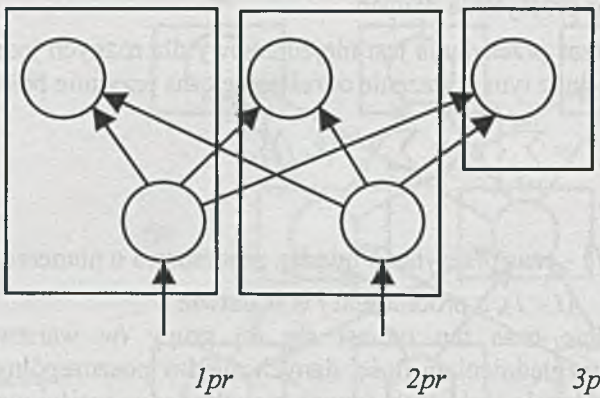
- warstwowy (warstwy lub grupy warstw obsługiwane są przez jeden procesor),
- międzywarstwowy (części warstwy lub części kilku warstw obsługiwane są przez jeden procesor).

Powyższe sytuacje można zilustrować w następujący sposób:

a)

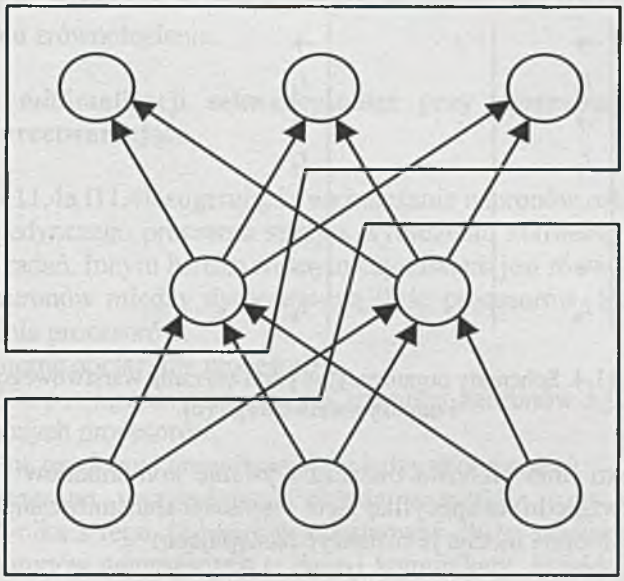


b)



Rys. 11.2. Sposoby rozdziału neuronów między procesorami

Równie często mogą wystąpić rozdziały o charakterze mieszanym. Ilustruje to rys. 11.3.

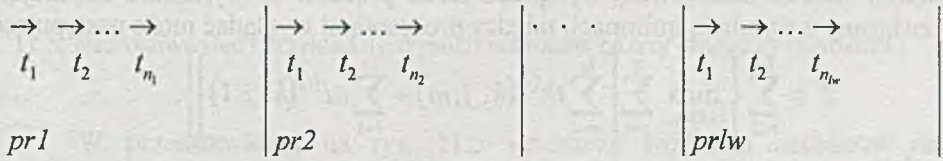


Rys.11.3. Mieszany i warstwowo – międzywarstwowy sposób rozdziału neuronów.

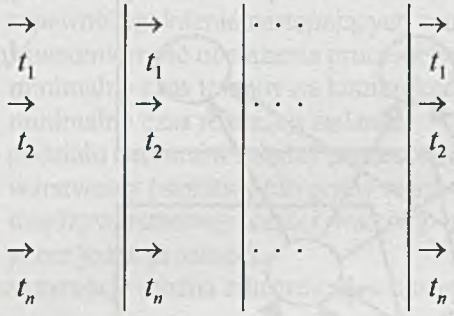
Specyfika sieci neuronowych określa sekwencyjny sposób konwersji między warstwami. Stąd też grupowanie warstwowe neuronów nie wpłynie na przyspieszenie obliczeń, ponieważ procesor obsługujący warstwę musi obsługiwać neurony sekwencyjnie, a jednocześnie kolejna warstwa musi oczekiwać na rezultaty uzyskane w warstwie poprzedniej. Stąd też będziemy mieli do czynienia tylko z przetwarzaniem sekwencyjnym, a ponadto czas konwersji wydłuży komunikacja między procesorami – warstwami [13].

Dystrybucja międzywarstwowa daje lepsze wyniki, mimo że i tu obsługa neuronów w kolejnych warstwach przebiega sekwencyjnie. Rys.11.4 ukazuje różnice w organizacji przetwarzania przy dystrybucji warstwowej i międzywarstwowej.

a)



b)



Rys. 11.4. Schematy organizacyjne przetwarzania warstwowego (a) i międzywarstwowego (b).

Na rysunku linia pionowa oznacza wysłanie komunikatów i danych między procesorami. Ze względu na specyfikę sieci wielkości transmitowanych informacji są jednakowe lub podobne i można je oznaczyć następująco:

$$t_i = \sum_{m=1}^{l_k} tk_m + \sum_{l=1}^{ld} td_l, \tag{11.5}$$

gdzie

t_i - czas transmisji komunikatów i danych,
tk_m - czas przekazu j – go komunikatu,
l_k - ilość komunikatów,
td_l - czas przekazu l – tej danej,
ld - ilość danych.

Uściślając możliwe sytuacje trzeba jednak stwierdzić, iż czas przesyłania czy transmisji nie musi być jednakowy dla możliwych połączeń między procesorami danej warstwy. Przejdziemy więc od typowego (lub średniego) znaczenia t_i do czasu transmisji zależnego od komunikujących się procesorów $\left(t_i \to T_p^{(i)}(k,j)\right)$. Gdyby każdy neuron obsługiwany był przez jeden procesor to wyrażenie estymujące czas związany z wymianą informacji między procesorami wyglądać może następująco:

$$T = \sum_{i=2}^{lw} \left\{ \max_{1 \le k \le n_{i-1}} \sum_{j=1}^{n_i} \left[\sum_{m=1}^{l_k} tk^{(i)}(k,j,m) + \sum_{l=1}^{ld} td^{(i)}(k,j,l) \right] \right\}, \tag{11.6}$$

gdzie

$tk^{(i)}(k,j,m)$ - czas wysyłania m – go komunikatu z procesora o numerze k w warstwie $(i-1)$ do procesora j w warstwie i – tej.

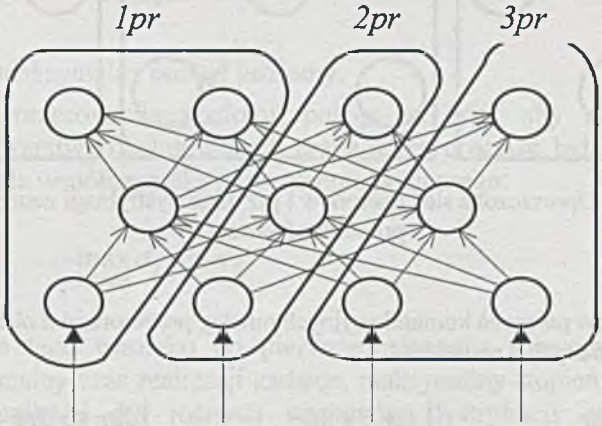
Do tego czasu należałoby dodać czas właściwej konwersji realizowanej przez procesory, które w zależności od powierzonych im neuronów i ich konfiguracji pracują w różnym stopniu zrównoleglenia.

11.2 Strategia minimalizacji sekwencyjności przy dystrybucji neuronów na jednostki przetwarzające.

Rysunki 11.4a i11.4b sugerują, iż wyznaczanie neuronów należących do jednej warstwy dla pojedynczego procesora sprzyja wydłużeniu konwersji zwiększając listę sekwencyjnych zadań. Innym bardzo ważnym czynnikiem jest równomierne Rozdzielenie neuronów między dysponowaną ilość procesorów. Stąd ogólne zasady strategii obciążenia procesorów:

- równomierne obciążenie procesorów,
- minimalne powiązania poziome; jak najmniej neuronów z jednej warstwy dla pojedynczych procesorów,
- minimalne przekazy komunikacyjne między procesorami.

W ostatnim kryterium uwzględniamy przede wszystkim czas przekazu między procesorami. Wynika z tego, iż lepiej wyeliminować dłużej trwające przekazy dzięki zgrupowaniu neuronów generujących te dane i komunikaty. Przykładowo, jeśli mamy sieć neuronową o następującej strukturze (rys.11.5) i dysponujemy 3 procesorami:



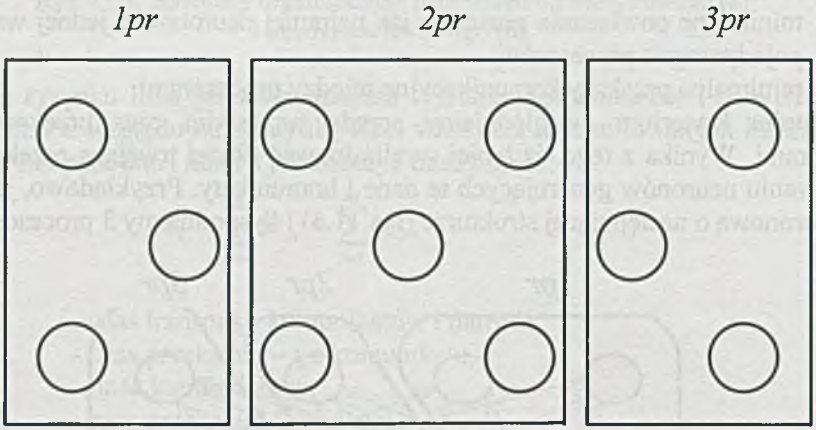
Rys. 11.5. Neuronowa sieć i przykład dystrybucji neuronów na trzy procesory (wariant 1)

W przedstawionej na rys. 11.5 strukturze rozdziału neuronów na pojedyncze procesory pojawią się następujące przekazy komunikatów i danych:

Tablica 11.1. Zestaw połączeń komunikacyjnych między procesorami o obciążeniach konwersyjnych zgodnych z rysunkiem 5.

1pr ↔ 2pr	*1pr ↔ 3pr*	*2pr ↔ 3pr*
(1,1) → (2,2)	(1,1) → (2,3)	(1,2) → (2,3)
(2,1) → (3,3)	(2,1) → (3,4)	(2,2) → (3,4)
(2,1) → (1,2)	(2,3) → (3,1)	(2,2) → (1,3)
(3,1) → (2,2)	(2,1) → (1,4)	(2,2) → (1,4)
(3,2) → (2,2)		(3,3) → (2,3)

Jeżeli tą samą sieć neuronową rozdzielimy w inny sposób (rys.11.6) to otrzymamy zestaw połączeń przedstawiony w tablicy 11.2.



Rys. 11.6. Uproszczona sieć neuronów i przykład dystrybucji neuronów na trzy procesory (wariant 2)

Tablica 11.2. Zestaw połączeń komunikacyjnych między procesorami o obciążeniach konwersyjnych zgodnych z rysunkie11.6.

1pr ↔ 2pr	*1pr ↔ 3pr*	*2pr ↔ 3pr*
(1,1) → (2,2)	(1,1) → (2,3)	(1,2) → (2,3)
(2,1) → (3,2)	(2,1) → (3,4)	(1,3) → (2,3)
(2,1) → (3,3)	(2,1) → (1,4)	(2,2) → (3,4)
(2,1) → (1,2)	(3,1) → (2,3)	(2,2) → (1,4)
(2,1) → (1,3)		(3,2) → (2,3)
(3,1) → (2,2)		(3,3) → (2,3)

W przedstawionych tablicach symbole (*i, j*) oznaczają neurony z *i* – tej warstwy o numerze *j*. Kierunek komunikacji między procesorami zaznaczony jest strzałką → lub ←. Zbadajmy przypadek równoległego rozdziału neuronów między warstwami. Przeprowadzając pełną analizę dochodzimy do wniosku, iż ilość połączeń komunikacyjnych zależy od ilości procesorów przy spełnieniu warunku rozproszenia poziomego, który przedstawiamy poniżej. Wprowadźmy oznaczenia:

$$d = \left[\frac{\sum_{i=1}^{lw} n_i}{np \cdot lw} + 1 - \varepsilon \right], \tag{11.7}$$

gdzie

d - współczynnik rozproszenia poziomego,
$[\cdot]$ - funkcja Entier,
np - ilość procesorów,
ε - mała wartość dodatnia.

Ponadto oznaczmy ilość neuronów z każdej warstwy obsługiwanych przez konkretne procesory przez $c\ (i, j)$ – gdzie i – numer procesora, j – numer warstwy. Następnie określmy maksymalna wielkość $c\ (i, j)$ dla każdego procesora:

$$\hat{d}_i = \max_{1 \le j \le lw} (c(i, j)), \tag{11.8}$$

gdzie

$\hat{d}_i$ - maksymalny odchył poziomy.

Warunek rozproszenia poziomy polega na tym, aby maksymalna ilość neuronów jednej warstwy obsługiwanych przez jeden procesor była minimalna czyli aby nie przekraczała współczynnika rozproszenia poziomego:

$$\max_{1<i \le np} \hat{d}_i == d, \tag{11.9}$$

Spełnienie tego warunku po pierwsze zapewnia równomierne obciążenie procesorów, minimalny czas realizacji zadania, maksymalny stopień zrównoleglenia i stałą ilość komunikacji dla różnych wariantów dystrybucji neuronów między procesorami.

Należy jednakowoż podkreślić, iż dotyczy to tylko sytuacji równomierności rozdzielenia neuronów pomiędzy warstwami (rys.11.5).
Nierzadko mamy do czynienia z dysproporcjami w rozdziale neuronów między warstwami.

W takiej sytuacji należy wybrać warstwę o największej szerokości (ilość neuronów w jednej warstwie) i dokonać równomiernego rozdziału neuronów w

wybranej warstwie między procesorami. W tym wypadku współczynnik rozproszenia poziomego można określić następująco:

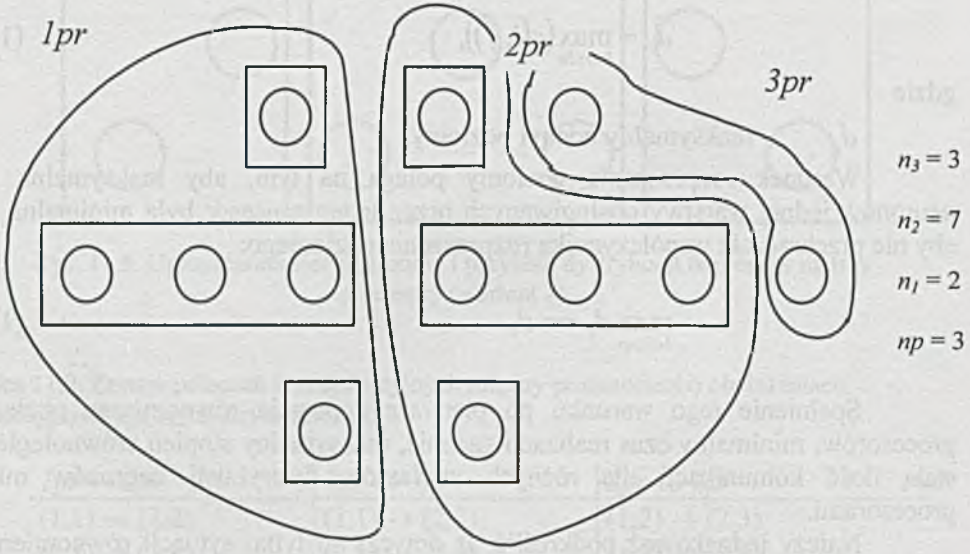
$$dd = \left[\frac{\max_{1 \le i \le lw}(n_i)}{np} + 1 - \varepsilon \right] \tag{11.10}$$

Maksymalny odchył poziomy $\hat{d}_i$ będzie równy wartości

$$dd\left(\hat{d}_i = dd \quad dla \quad i\left(n_i = \max_{1<j\le np}(n_j) \right) \right),$$

Z powyższych rozważań wynika, iż każdą z warstw należy rozpatrywać oddzielnie i przypisać każdemu procesorowi (jeśli to możliwe): $\left[n_j / np + 1 - \varepsilon \right]$ neuronów, $1 \le j \le lw$.

Przykład 1.



Rys. 11.7. Dystrybucja neuronów w neuronowych sieciach rozproszonych (pod względem ilości neuronów w poszczególnych warstwach).

Wybierając wartości $\varepsilon = 0.01$ możemy wstępnie oszacować:

$$c(i,1) = \left\lfloor \frac{n_1}{np} + 0{,}99 \right\rfloor = \left[\frac{2}{3} + 0{,}99 \right] = 1 \qquad 1 \le i \le 2$$

$$c(i,2) = \left\lfloor \frac{n_2}{np} + 0{,}99 \right\rfloor = \left[\frac{7}{3} + 0{,}99 \right] = 3 \qquad 1 \le i \le 2$$

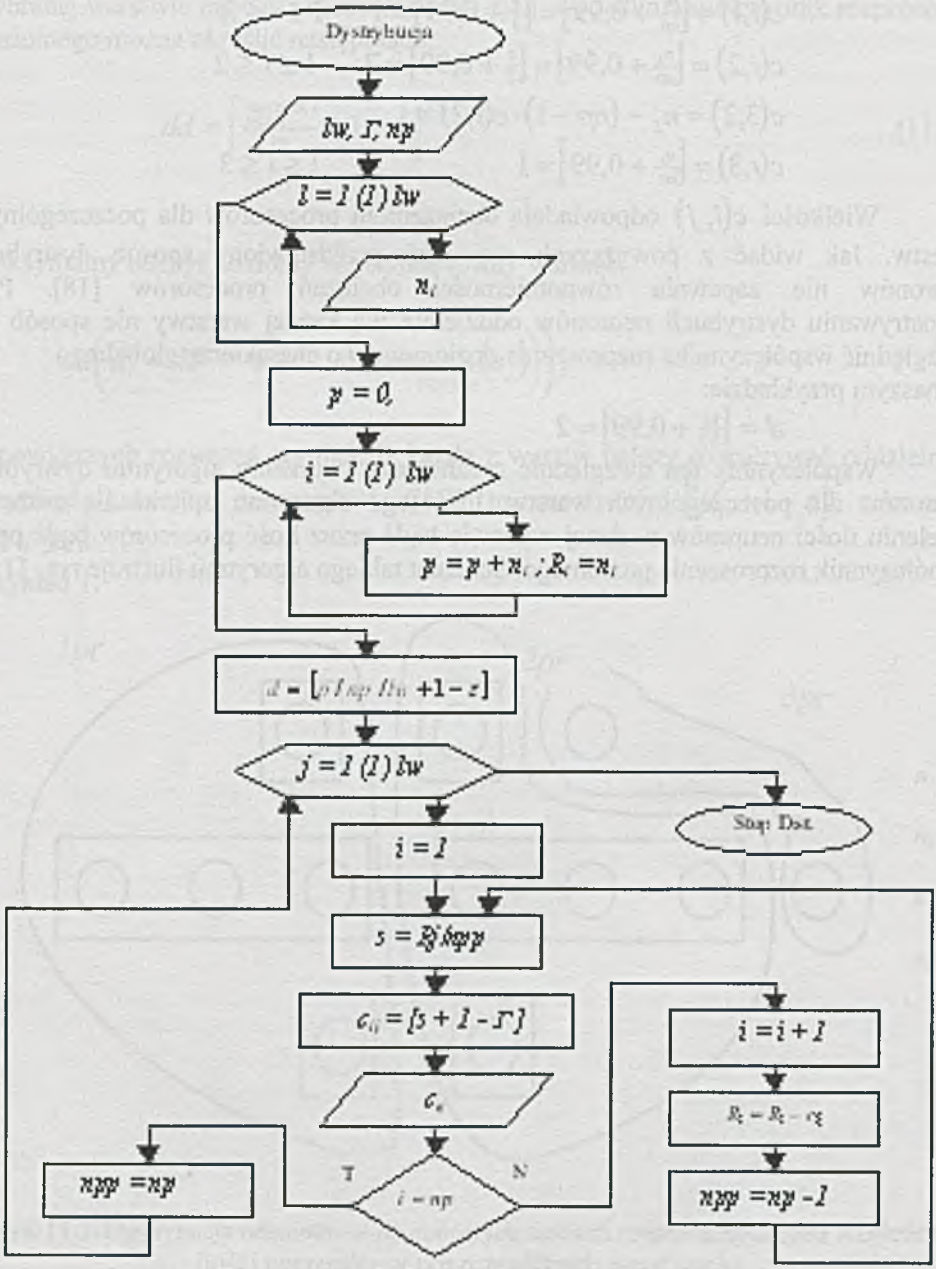
$$c(3,2) = n_2 - (np - 1) \cdot c(i,2) = 1$$

$$c(i,3) = \left[\frac{n_3}{np} + 0{,}99 \right] = 1 \qquad 1 \le i \le 3$$

Wielkości $c(i, j)$ odpowiadają obciążeniom procesorów dla poszczególnych warstw. Jak widać z powyższych rozważań przedstawiony sposób dystrybucji neuronów nie zapewnia równomierności obciążeń procesorów [18]. Przy rozpatrywaniu dystrybucji neuronów oddzielnie dla każdej warstwy nie sposób nie uwzględnić współczynnika rozproszenia poziomego d o charakterze globalnym.
W naszym przykładzie:

$$d = \left[\frac{12}{3 \cdot 3} + 0{,}99 \right] = 2$$

Współczynnik ten uwzględnić można przy tworzeniu algorytmu dystrybucji neuronów dla poszczególnych warstw. Idea tego algorytmu opierać się może na dzieleniu ilości neuronów w danej warstwie bądź przez ilość procesorów bądź przez współczynnik rozproszenia poziomego. Schemat takiego algorytmu ilustruje rys. 11.8.



Rys. 11.8. Algorytm rozdziału neuronów między procesorami dla poszczególnych warstw.

Algorytm (rys.11.8) polega na warstwowym rozdziale neuronów między procesory. Po każdorazowym przydziale neuronów do konkretnego procesora, rozpatrujemy pozostałe neurony danej warstwy (R_i) jako bazę dystrybucyjną i rozdzielamy ją pomiędzy pozostałe procesory. Każdorazowo szacujemy przydział neuronów jako iloraz pozostałych elementów sieci przez pozostałą ilość procesorów.

Wnioski:

1. Duży wpływ na rezultaty paralelizacji ma rozproszenie neuronów w poszczególnych warstwach. Im większe jest to rozproszenie tym wolniej będzie przebiegał proces równoległej konwersji. Jeżeli jest to możliwe celowym jest zmniejszenie rozproszenia neuronów w poszczególnych warstwach.

2. Grupowanie neuronów jednej warstwy w jednym procesorze mija się z celem paralelizacji, ponieważ wymusza to sekwencyjną obsługę kolejnych neuronów.

3. Największe efekty paralelizacji uzyskujemy gdy maksymalne odchyły poziome (11.8) dla poszczególnych warstw są równe współczynnikowi rozproszenia poziomego (11.7):

$$\bar{d}_i = d, \qquad i = 1,2,\ldots,lw$$

4. Efekty paralelizacji sieci neuronowych zależą od równomierności dystrybucji neuronów na poszczególne procesory w poszczególnych warstwach, ilości przesyłanych między procesorami komunikatów i danych, rozproszenia mocy obliczeniowej procesorów.

Literatura

1. Arabas J.:Wykłady z algorytmów ewolucyjnych. WNT, Warszawa 2001.g
2. Brdyś Mieczysław, Ruszczyński Andrzej : Metody optymalizacji w zadaniach. Warszawa, WNT 1985.
3. Cieślak M.: Prognozowanie gospodarcze.PWN.Warszawa 1997.
4. Cytowski J.:Algorytmy genetyczne. Podstawy i zastosowania. Akademicka Oficyna Wydawnicza PLJ, Warszawa 1996.
5. Czogała E.,Perdycz w.: Elementy i metody teorii zbiorów rozmytych. PWN, Warszawa, 1985
6. Davis L.: Genetic Algorithms and Simulated, A. Morgan Kaufmann Pub., Inc.,1988
7. Davis L.: Handbook of Genetic Algorithms, Nostrand Reinhold, 1991.
8. De Groot M.H.: Optimal statistic decisions, New York, 1970, McGraw - Hill.
9. Feller W.: An introduction to probability theory and its application. New York, Wiley, 1968.
10. Goldberg D.: Algorytmy genetyczne. WNT, Warszawa 1995.
11. Grygiel K.: Przegląd teorii algorytmów ewolucyjnych W: Warsztaty naukowe, KAE i OG'99, 1999.
12. Gwiazda T. D.: Algorytmy genetyczne. Zastosowania w finansach, WSPIZ L.Koźmiński, 1998.
13. Handbook of genetic algorithms. Red. Davis Lawrence. New York, Van Nostrand Reinhold, 1991.
14. Jog P., Van Gucht D.: Parallelisation of probabilistic sequential search algorithms, Genetic algorithms and their applications , 1987, p.170-176.
15. Karg R.L., Thompson G.L.: A heuristic approach to solving traveling salesman problems, Management Science,1964, v.10(2), p.225-248.
16. Koblitz N.: Algebraiczne aspekty kryptografii. Wydawnictwo Naukowo Techniczne, Warszawa, 2000.
17. Konosala R.: Zastosowania metod sztucznej inteligencji w inżynierii produkcji, WNT, Warszawa, 2002.
18. Korbicz J., Obuchowicz A., Uciński D.:Sztuczne sieci neuronowe – podstawy i zastosowania.Akademicka Oficyna Wydawnicza PLJ, Warszawa 1994.
19. Kozieł S., Michalewicz Z.: Evolutionary algorithms, homomorphous mappings, and constrained parameter optimization. Evolutionary computation.1999. No.7 p. 19-44.
20. Leks D., Piech H., Ptak A., Kuczyński Ł.: Zastosowanie struktury neuronowej do wspomagania operacji genetycznych, Konferencja Krajowa PTI : Systemy Informatyczne, zastosowania i wdrożenia,WNT, Warszawa-Szczyrk, 2002.-referat
21. Michalewicz Zb.:Algorytmy genetyczne + struktury danych = programy ewolucyjne, WNT, 2002.

22. Michalewicz Z.: Genetics algorithms, numerical optimization, and constrains.W:ICGA'95. 1995.s.151 – 155.
23. Osowski S., Sieci neuronowe w ujęciu algorytmicznym, WNT, Warszawa 1996.
24. Podsiadło M.: Równoległe algorytmy genetyczne - przegląd tematyki W.:KAEiOG'97. 1997, s. 219 – 226.
25. Pokraśniewicz J.: Adaptacja parametrów w strategiach ewolucyjnych – analiza właściwości i nowe koncepcje.Rozprawa doktorska. Warszawa, Politechnika Warszawska.1998.
26. Radcliffe A.: A problem solving technique based on genetics, Creative Computing, 1981, v.3(2), p.78-81.
27. Rutkowska D., Piliński M., Rutkowski L.: Sieci neuronowe, algorytmy genetyczne i systemy rozmyte, PWN, Warszawa-Łódź, 1997.
28. Rzewuski M, Szreter M, Arabas J.: Poszukiwanie efektywniejszych operatorów genetycznych dla strategii ewolucyjnych.W:KAEiOG,97. 1997., s.237-243.
29. Schneier B.: Kryptografia dla praktyków. WILEY,Wydawnictwo Naukowo Techniczne, Warszawa 2002.
30. Skarbek W., Metody reprezentacji obrazów cyfrowych, Akademicka Oficyna Wydawnicza, Warszawa 1993.
31. Słomka M.: Wpływ metody inicjalizacji populacji początkowej na zbieżność strategii ewolucyjnych.Praca magisterska. Warszawa. Politechnika Warszawska. 1998.
32. Suh J., Van Gucht D.: Distributed genetic algorithms, Bloomington: Indiana University, Computer Science Department.
33. Sysło M., Deo N.,Kowalik J.:Algorytmy optymalizacji dyskretnej, Wydawnictwo Naukowe PWN, Warszawa 1995.
34. Tadeusiewicz R., Sieci neuronowe, Akademicka Oficyna Wydawnicza, Warszawa 1993.
35. Takahashi Y., Robins M.J, Auslander D.M.: Control and dynamic systems, Reading, MA: Addision-Wesley, 1970.
36. Wetzeel A.: Evaluation of the effectiveness of genetic algorithms in combinatorial optimization , University of Pittsburgh,1983.
37. Wieczorek R., Zieliński R.: Komputerowe generatory liczb losowych, WNT, Warszawa, 1997
38. Wilson S.W.: Classifier systems and the Animant problem. Machine Learning, 1987, v.2(3), p.199-228.
39. Włodarczyk M., Piech H. : A Distibuted Neural Network, Computer Science, IMI, Politechnika Częstochowska, 2002, No 1.

ISBN 83-204-3015-1