WEKTOROWE KODOWANIE PERMUTACJI. NOWE OPERATORY GENETYCZNE

Mariusz MAKUCHOWSKI

Streszczenie: W pracy proponuje się alternatywny sposób kodowania permutacji. Prezentuje się szereg jego własności niewystępujących w klasycznym sposobie kodowania. Między innymi pokazuje się, iż w nowym modelu kodowania permutacji zawarta jest informacja zarówno o sekwencji jak i rozkładzie prawdopodobieństwa mutacji poszczególnych jej fragmentów. Ponadto wskazuje się nowe operatory mutacji i krzyżowania. Przedstawione w pracy operatory poddane zostały teoretycznej analizie matematycznej.

Słowa kluczowe: kodowanie permutacji, operatory mutacji, operatory krzyżowania.

1. Wstęp

Problemy optymalizacji dyskretnej polegają na wyborze najlepszego rozwiązania ze zbioru rozwiązań dopuszczalnych, zwanego przestrzenią rozwiązań. Postać rozwiązania oraz kryterium jego oceny zależą ściśle od analizowanego problemu. Zauważyć można, iż wśród problemów optymalizacji jest pewna grupa, w których rozwiązanie opisywane jest przy pomocy permutacji elementów pewnego zbioru. Pomimo, iż permutacje kodują różnorakie byty (zależnie od problemu permutacja może oznaczać np. sekwencję podawania zadań do systemu w permutacyjnym problemie przepływowym [7], czy kolejność odwiedzania miast w problemie komiwojażera [1]) można poddawać je tym samym transformacjom. Algorytmy popraw takie jak symulowane wyżarzanie [4] czy algorytmy genetyczne [3,2] zawierają element odpowiedzialny za losowe zaburzanie rozwiązania, są nimi operatory mutacji. Opracowanie nowych operatorów mutacji i zastosowanie ich w istniejących algorytmach może przynieść korzyści w postaci zwiększenia efektywności zmodyfikowanych w ten sposób algorytmów. Podobnie, opracowanie i zastosowanie nowym operatorów krzyżowania w istniejących algorytmach genetycznych może w niektórych przypadkach zwiększyć ich wydajność.

W pracy prezentuje się nowe operatory mutacji i krzyżowania działające na innowacyjnym wektorowym modelu kodowania permutacji. Proponowany wektorowy model kodowania permutacji zawiera informacje zarówno o kodowanej sekwencji jak i (co jest całkowicie innowacyjne) informacje o rozkładzie prawdopodobieństwa mutacji poszczególnych jej fragmentów. W efekcie takiego podejścia, rozwiązaniem analizowanym przez algorytm może być nie stricte permutacja (jak to jest klasycznie) ale np. sekwencja ze skłonnością do mutacji w końcowych jej fragmentach a „niechętne" mutująca elementy początkowe. Proponowane w pracy wektorowe kodowanie zawiera więc zewnętrzne cechy rozwiązania, będące permutacją właściwą, jak i wewnętrzne jej cechy będące informacją o rozkładzie prawdopodobieństwa mutacji poszczególnych jej fragmentów. Ponieważ, zarówno zewnętrzne jak i wewnętrzne cechy rozwiązania zakodowane są w wektorowym kodowaniu permutacji, to podczas pracy algorytmu pod wpływem operatorów zarówno jedne jak i drugie podlegają ewolucji.

2. Kodowanie permutacji

W wielu problemach optymalizacji dyskretnej rozwiązaniem lub częścią rozwiązania jest pewna permutacja (np. problem komiwojażera, problem przepływowy, problem wyznaczenia cyklu Hamiltona). Klasyczne kodowanie permutacji jest intuicyjne i powszechnie znane [1,5-9]. Jednakże, ze względu na porównanie do proponowanego alternatywnego kodowania, poniżej poddane analizie zostaną oba (tzn. klasyczne i alternatywne) podejścia. W obu przypadkach kodowaniu poddana zostanie permutacja n elementowego zbioru $J = \{1,2,\dots,n\}$ indeksów bytów odpowiednich do problemu (odwiedzanych miast, wykonywanych operacji, odwiedzanych wierzchołków w grafie, itd.).

2. 1. Klasyczne kodowanie permutacji

W klasycznych modelach permutacja n elementowa wyznaczająca kolejność elementów ze zbioru J, kodowana jest w postaci sekwencji indeksów;

$$\pi = \big(\pi(1), \pi(2), \dots, \pi(n)\big) \tag{1}$$

gdzie: $\pi(i) \in J,\ i = 1,\dots,n$ jest numerem obiektu na *i*-tej pozycji w permutacji π. Zbiór wszystkich rozwiązań oznaczany jest przez $\Pi(n)$, a zbiór wszystkich zbiorów $\Pi(n), n \in \mathbb{Z}^+$ oznaczamy przez Π. Ponadto elementy kodowania π są parami różne; $\pi(i) \neq \pi(j)$ dla $i \neq j$. Ponieważ każdej permutacji odpowiada dokładnie jedno kodowanie, liczność zbioru wszystkich możliwych kodowań odpowiada liczności wszystkich permutacji n elementowego zbioru J; $|\Pi(n)| = n!$.

2. 2. Wektorowe kodowanie permutacji

W wektorowym kodowaniu, sekwencja $\big(v(1), v(2), \dots, v(\mathrm{n})\big)$, n elementowego zbioru J, kodowana jest w postaci wektora *n* liczb rzeczywistych;

$$v = \big[v[1], v[2], \dots, v[n]\big] \tag{2}$$

gdzie: $v[i] \in (0,1), i \in J$ oznacza pozorną pozycję elementu o numerze i. Zbiór wszystkich wektorów kodujących oznaczamy V. Niech $poz(v,a), a \in J, v \in V$ oznacza rzeczywistą pozycję elementu a w kodowanej sekwencji; $v\big(poz(v,a)\big) = a$. Wektor v koduje sekwencję

$$(v) = \big(v(1), v(2), \dots, v(n)\big) \tag{3}$$

w niemalejącej kolejności pozornych pozycji, a w przypadku tych samych wartości, kodowana kolejność wynika z indeksów elementów;

$$poz(v,a) < poz(v,b) \Leftrightarrow (v[a] < v[b]) \cup \big((v[a] = [b]) \cap (a < b)\big) \quad \text{dla } a,b \in J, v \in V. \tag{4}$$

Rzeczywista pozycja $poz(v,a)$ elementu a w kodowanej przez wektor v sekwencji (v)

wynosi:

$$poz(v,a) = |\{b: (v[b] < v[b]), b \in J\} \cap \{c: (v[c] = v[a]) \cap (c \leq a), c \in J\}| \quad \text{dla } a \in J, v \in V. \tag{5}$$

W celu ułatwienia dalszej analizy, praktycznie nie zmniejszając ogólności, załóżmy, iż wartości pozornych pozycji w wektorze permutacji v są parami różne; $v[a] \neq v[b]$ dla $a \neq b$. W praktyce założenie to można osiągnąć poprzez dodanie wielokrotności niewielkiej wartości $\epsilon > 0$ do wartości pozornych pozycji $v[i] \coloneqq v[i] + i\epsilon$. Wartość ϵ należy wybrać mniejszą niż przyjęta dokładność zapamiętywanych wartości pozornych pozycji.

Ostatecznie, przy założeniu, $v[a] \neq v[b]$dla $a \neq b$, wzory (4) i (5) upraszczają się do:

$$poz(v,a) < poz(v,a) \Leftrightarrow v[a] < v[b] \text{ dla } a, b \in J, v \in V, \tag{6}$$

$$poz(v,a) = |\{b: v[b] < v[a], b \in J\}| + 1 \text{ dla } a \in J, v \in V. \tag{7}$$

Dla każdej permutacji zbioru J istnieje nieskończony, nieprzeliczalny zbiór wektorów $V(n) \subset V, n = |J|$, z których każdy koduje tę samą sekwencję.

3. Operatory

Generalnie, wszystkie operatory zdefiniowane dla pewnego typu (w naszej pracy jest to wektorowa postać permutacji) można podzielić ze względu na liczbę przyjmowanych argumentów. Najczęściej wykorzystywane w literaturowych algorytmach [2,3] są:

- operatory jednoargumentowe, w których jedna permutacja, oraz parametry sterujące generują nową permutację,
- operatory dwu argumentowe, które przekształcają dwie permutacje oraz parametry sterujące w trzecią permutację.

W algorytmach genetycznych do mutacji czyli losowego zaburzenia rozwiązania stosuje się operatory jednoargumentowe z losowymi parametrami sterującymi i stąd wywodzi się ich nazwa - operatory mutacji. Podobnie w tych samych algorytmach w celu krzyżowania rozwiązań stosuje się najczęściej operatory dwuargumentowe które w literaturze nazwane zostały operatorami krzyżowania.

3. 1. Operatory mutacji

Najpopularniejsze operatory jednoargumentowe (dla klasycznego sposobu kodowania permutacji) to $Insert(\pi, i, j)$ oraz $Swap(\pi, i, j)$, $\pi \in \Pi,\ i \in J,\ j \in J$. Pierwszy z nich przekłada element $\pi(i)$ z pozycji i na pozycję j w wejściowej permutacji π. Utworzoną w ten sposób permutację oznaczać będziemy przez $\langle Insert(\pi, i, j)\rangle$. Drugi z nich zamienia miejscami elementy $\pi(i)$ i $\pi(j)$ znajdujące się odpowiednio na pozycjach i i j w wejściowej permutacji π tworząc analogicznie permutację $\langle Swap(\pi, i, j)\rangle$. Inne operatory mutacji są złożeniem przedstawionych przekształceń, lub wynikają z nałożenia dodatkowych ograniczeń np. $|i - j| < k$ dla ustalonego k. Na bazie przedstawionych operatorów działa większość algorytmów popraw. Stosuje się je w algorytmach bazujących na metodzie symulowanego wyżarzania, poszukiwania z zabronieniami, algorytmach genetycznych i innych.

Przy proponowanym w pracy wektorowym kodowaniu permutacji, w łatwy sposób

można osiągnąć opisane powyżej klasyczne przekształcenia typu $Insert$ i $Swap$. Załóżmy, iż permutacja klasycznie kodowana przez $\pi \in \Pi$ opisana jest wektorem $v \in V$ w kodowaniu wektorowym. Niech ruch $InsertVect(v, a, r)$ wykonany na wektorze $v \in V$ polega na zmianie jednej wartości $v[a], a \in J$, w wektorze kodującym na wartość $r \in (0,1)$. Ruch taki powoduje zmianę typu $Insert$ w kodowanej sekwencji, lub pozostawia ją niezmienioną. Aby uzyskać dokładnie permutację $\langle Insert(\pi, i, j)\rangle$, należy w wektorze kodującym v zmienić wartość $v[a]$ na pozycji $a = \pi(i)$ na wartość z przedziału $(v[\pi(j)], v[\pi(j+1)])$ dla $i < j$ lub wartość z przedziału $(v[\pi(j-i)], v[\pi(j)])$ dla $i > j$, gdzie $v[\pi(0)] = 0, v[\pi(n+1)] = 1$.

Wykonanie natomiast ruchu typu $SwapVect(v, a, b)$ polegającego na zamianie miejscami dwóch wartości z pozycji a i b, w wektorze permutacji $v \in V$ powoduje klasyczną zmianę typu $Swap$. Aby uzyskać dokładnie permutację $\langle Swap(\pi, i, j)\rangle$, należy w wektorze v zamienić ze sobą wartości na pozycjach $a = \pi(i), b = \pi(j)$.

W algorytmach popraw, takich jak symulowane wyżarzanie, czy algorytmy genetyczne, operatory jednoargumentowe np. $Insert(\pi, i, j)$ stosowane są z losowymi parametrami $i \in J, j \in J\backslash\{i\}$. Parametry i wybiera się losowo ze zbioru J, przy czym prawdopodobieństwo wybory każdego z elementów jest takie same i wynosi ono $1/n$. Podobnie z wyborem parametru j z tym, iż wybiera się go losowo z zbioru $J\backslash\{i\}$, wybierając każdy element z prawdopodobieństwem $1/(n-1)$.

W przypadku wektorowego kodowania permutacji, proponujemy losowy ruch $InsertVect(v, a, r)$. Parametr a, określający numer modyfikowanej pozycji, należy wybrać ze zbioru J. Prawdopodobieństwo wyboru każdego z elementów powinno być takie same. Wartość parametru r, która zostanie przypisana bytowi $v(a)$ należy wartość losowo z przedziału $(0,1)$.

W efekcie tej operacji, otrzymujemy nowy wektor $\langle InsertVect(v, a, r)\rangle$, który pomimo zmienionych wartości może kodować tę samą sekwencję co wektor v. Dla danego wektora v i przy ustalonej wartości a oraz losowej wartości r wylosowanej z rozkładu jednostajnego na przedziale (0,1), prawdopodobieństwo $P(v, a, k)$ i przemieszczenie elementu a, na pozycję k wynosi odpowiednio:

$$P(v,a,k) = \begin{cases} v[v(k+1)] - v[v(k)] & dla \quad k > a \\ v[v(k)] - v[v(k-1)] & dla \quad k < a \\ v[v(k+1)] - v[v(k-1)] & dla \quad k = a \end{cases} \tag{8}$$

gdzie $v[v(0)] = 0, v[v(n+1)] = 1$.

Klasyczny losowy ruch typu $Insert$ przenosił wylosowany element z pozycji $i \in J$ na pozycje $j \in J\backslash\{i\}$ z równym prawdopodobieństwem. Natomiast przy wektorowym kodowaniu, i losowych ruchach typu $InsertVect$ w wektorze zakodowana jest nie tylko informacja o sekwencji (v), ale i także informacja o prawdopodobieństwie przełożenia losowego elementu na daną pozycję.

Bardziej interesujący z losowych ruchów, jest operator mutacji $TwoRand(v, RAND, r)$, w którym element zmieniany a nie jest losowany z jednakowym prawdopodobieństwem z zbioru J, lecz wybierany w sposób przedstawiony poniżej:

$$a \in arg\, min_{x \in J} |v[x] - RAND|, \tag{9}$$

gdzie $RAND$ jest liczbą losową z przedziału $(0,1)$.

Przy takim losowaniu w wektorze zakodowana jest nie tylko informacja o permutacji, ale także informacja o prawdopodobieństwie zmian na poszczególnych pozycjach. Prawdopodobieństwo wybrania elementu a w wektorze v wynosi:

$$P(v,a) = \begin{cases} \big(v(a^+) - v(a^-)\big)/2 & dla \quad 1 < poz(v,a) < n \\ \big(v(a^+) + v(a)\big)/2 & dla \quad poz(v,a) = 1 \\ 1 - \big(v(a^-) + v(a)\big)/2 & dla \quad poz(v,a) = n \end{cases} \tag{10}$$

gdzie elementy a^+, a^- są indeksami elementów leżącymi bezpośrednio za i przed elementem a w sekwencji (v); $a^+ = v(poz(v,a)+1), a^- = v(poz(v,a)-1)$.

Przykład 1

Załóżmy, iż wektor kodujący ma postać $v = [0.80, 0.78, 0.21, 0.84, 0.56]$. Koduje on sekwencję $(v) = (3,5,2,1,4)$. Wykonanie ruchu $TwoRand(v, Rand, r)$ dla $RAND = 0.31$ i $r = 0.62$ powoduje wybranie elementu 3, ponieważ jego pozorna pozycja jest najbliższa wartości $RAND$, i nadanie jej wartości 0.62. Powstały wektor ma więc postać $v' = [0.80\ 0.78, 0.62, 0.84, 0.56]$, a kodowana sekwencja to: $(v') = (5,3,2,1,4)$.

Zauważmy ponadto, że w kolejnym ruchu typu $TwoRand$, wybranie elementu nr 1 jako przekładanego, odbędzie się z prawdopodobieństwem tylko 0.03, a elementu 5 z prawdopodobieństwem aż 0.59. Załóżmy dalej, że przekładany element został wybrany i niech przykładowo będzie nim 4. Element ten zostanie przełożony na pozycję 1 (przed element nr 5) z prawdopodobieństwem wynoszącym aż 0.56, a z prawdopodobieństwem wynoszącym zaledwie 0.02 na pozycję 4 (pomiędzy element 2 i 1).

3. 2. Operatory krzyżowania

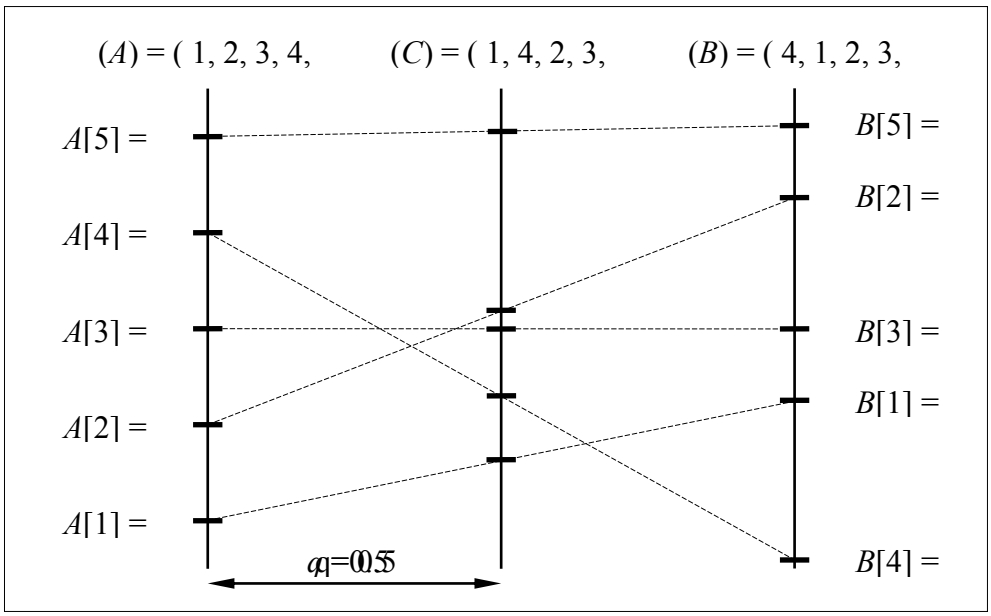
Idea operatora krzyżowania, polega na tym, aby z rozwiązań (najczęściej dwóch) zwanych rodzicami, wygenerować rozwiązanie zwane potomkiem w taki sposób, aby dziedziczyło ono cechy swoich rodziców. Cechami rozwiązania w przypadku permutacji, są np. relacje kolejnościowe w kodowanej sekwencji. Można także wskazać szereg innych cech permutacji, jednakże zostaniemy przy analizie tylko wspomnianych relacji kolejnościowych. W klasycznych operatorach krzyżowania PMX (ang. Partial Mapped Crossover), OX (ang. Order Crossover), dochodzi do skopiowania fragmentu sekwencji jednego z rodziców, (może to być zarówno początkowy, środkowy jak i końcowy fragment), a pozostałą część uzupełnia się elementami, sugerując się kolejnością w jakiej występują w drugim z rodziców. W ten sposób potomek posiada zarówno cechy (w postaci relacji kolejnościowych) zarówno jednego jak i drugiego rodzica. Dokładny model klasycznych operatorów krzyżowania znany jest w literaturze [8,2,3,6,5,9] i nie będzie tu szerzej przytaczany.

W proponowanym wektorowym modelowaniu permutacji, mamy większą dowolność krzyżowania danych rodziców, ze względu na mniejsze restrykcje ograniczające poprawny wektor. Proponowany operator krzyżowania $Wght(w^1, v^1, w^2, v^2, \dots, w^k, v^k)$ składa się z k par, z których i-ta para składa się z wagi $w^i \in [0,1]$; $\sum_{i=1}^{k} w^i = 1$ i wektora kodującego $v^i \in V$. W wyniku jego działania otrzymujemy wektor v', będący sumą warzoną poszczególnych wektorów wejściowych z odpowiadającym im wagą; $v'[j] = \sum_{i=1}^{k}(w^i v^i[j]),\ j \in J$. Ograniczmy się jednak do przypadku tworzenia potomka z $k = 2$

rozwiązań, nazwanych rodzicami A i B, krzyżowanymi odpowiednio z wagami w^A, w^B. Zapis $Wght(w^A, A, w^B, B)$ uprościmy do postaci $Wght(q, A, B)$, gdzie $q = 1 - w^A = w^B$, a powstałego w wyniku krzyżowania potomka oznaczymy krótko literą $C = [C[1], \dots, C[n]]$; $C[i] = (1-q)A[i] + qB[i]; i \in J$.

Przykład 2

Załóżmy iż, mam dwa wektory kodujące permutacje; $A = [0.10, 0.30, 0.50, 0.70, 0.90]$ i $B = [0.34, 0.78, 0.46, 0.02, 0.92]$. Odpowiadające im sekwencje to: $(A) = (1, 2, 3, 4, 5)$ i $(B) = (4, 1, 3, 2, 5)$. W wyniki wykonania krzyżowania $Wght(0.5, A, B)$ otrzymujemy wektor $C = [0.22, 0.54, 0.48, 0.36, 0.91]$. Wektor ten koduje sekwencję $(C) = (1, 4, 3, 2, 5)$. Warto zauważyć, iż gdyby za wektora B, podstawić wektor $B' = [0.34, 0.58, 0.46, 0.02, 0.92]$, kodujący tę samą sekwencję $(B) = (B')$ to w wyniku krzyżowania go z wektorem A otrzymany potomek $C' = [0.22, 0.44, 0.48, 0.36, 0.91]$ kodował by już inną sekwencję niż potomek C; $(C) \neq (C') = (1, 4, 2, 3, 5)$.



Rys.1. Interpretacja graficzna operatora $Wght(q, A, B)$ dla danych z przykładu 1

Rysunek 1 przedstawia graficzną interpretację proponowanego operatora krzyżowania. Pionowymi liniami ciągłymi oznaczono przedziały pozornych pozycji odpowiednio dla wektorów A, B i powstałego w wyniku krzyżowania wektora C. Nad każdą z tych linii umieszczono informację o kodowanej przez nią sekwencji. Skośne przerywane linie oznaczają wartości interpolacji pozornych pozycji w potomku. W miarę zwiększania parametru q od 0 do 1 środkowa linia przesuwa się kolejno od linii odpowiadającej wektorowi A do linii odpowiadającej wektorowi B, potomek C zmienia zaś się stopniowo od postaci A do B. Podczas zwiększania parametru q, linia odpowiadająca wektorowi C, przechodzi kolejno przez węzły, gdzie węzłem nazywamy punkt przecięcia się dwóch

skośnych linii. Każdy węzeł odpowiedzialny jest za zamianę dwóch sąsiednich elementów w sekwencji (C).

W literaturze opisanych jest wiele podobnych operatorów mutacji np. MSX (ang. Multi Step Crossover) [5] czy $MSXF$ (ang. Multi Step Crossover Fusion) [9], w którym systematycznie wykonuje się przekształcenia doprowadzające z rodzica A do rodzica B, jednakże nie definiuje on kolejności ich wykonywania. W proponowanym operatorze, kolejność ta jest ściśle ustalona i dla danych z rysunku 1 zamiana elementów nastąpi w kolejności (3,4), (2,4), (2,3) i (1,4).

Poniżej przedstawione zostaną własności operatora krzyżowania $Wght(q, A, B)$, które można jednak łatwo rozszerzyć na ogólniejszy operator $Wght(w^1, v^1, w^2, v^2, \dots, w^k, v^k)$.

Własność 1

Jeżeli dwa elementy a i b wstępują w tej samej kolejności w sekwencjach kodowanych przez wektor A jak i B to w wyniku krzyżowania tych osobników otrzymany potomek posiada elementy a i b w tej samej sekwencji co jego rodzice, dla dowolnego $q \in [0,1]$.

Dowód: bez zmniejszania ogólności załóżmy, że element a jest na wcześniejszej pozycji niż element b w sekwencji (A) i (B); $poz(A,a) < poz(A,b)$, $poz(B,a) < poz(B,b)$. Z wzoru (6) oraz wcześniejszej nierówności wynika, że: $A[a] < A[b]$, oraz $B[a] < B[b]$. W wyniku krzyżowania w potomku C wartości pozornych pozycji elementów a i b wynoszą: $C[a] = (1-q)A[a] + qB[a]$, $C[b] = (1-q)A[b] + qB[b]$ więc $C[a] < C[b]$. Z ostatniej nierówności oraz wzoru (6) wynika, iż w sekwencji (C) pozycja elementu a jest mniejsza niż pozycja elementu b; $poz(C,a) < poz(C,b)$, co kończy dowód. ■

Własność 2

Jeżeli dwa elementy a i b występują w różnej kolejności w sekwencjach (A) i (B) to w wyniku ich krzyżowania otrzymany potomek posiada elementy a i b w kolejności odpowiadającej sekwencji (A) dla $q \in [0, x)$ oraz w kolejności odpowiadające sekwencji (B) dla $q \in (x, 1]$, gdzie $x = (A[a] - A[b])/(A[a] - A[b] + B[b] - B[a])$.

Dowód: bez zmniejszania ogólności załóżmy, że element a jest na wcześniejszej pozycji, niż element b w sekwencji (A) oraz w odwrotnej kolejności w sekwencji (B); $poz(A,a) < poz(A,b)$, $poz(B,a) > poz(B,b)$. Ze wzoru (6), oraz wcześniejszych nierówności wynika iż, $A(a) < A(b)$, oraz $B(a) > B(b)$. W potomnym C rozwiązaniu elementy a i b występują w takiej samej kolejności jak w sekwencji rodzica (A); $poz(C,a) < poz(C,b)$ wtedy gdy $C[a] < C[b]$. Rozpisując ostatnią nierówność otrzymujemy: $(1-q)A[a] + qB[a] < (1-q)A[b] + qB[b]$. Przekształcając tę nierówność otrzymujemy: $q < (A[a] - A[b])/\big(A(a) - A(b) + B(b) - B(a)\big)$. Przy uwzględnieniu powyższej nierówności oraz założenia, iż $q \in [0,1]$ otrzymujemy przedział [0,x) dla którego krzyżowanie generuje sekwencję elementów a i b taką samą jak w przypadku permutacji (A). Analogiczne postępowanie dowodzi, iż odwrotną sekwencję elementów a, b otrzymujemy dla krzyżowania z parametrem q należącym do przedziału $(x, 1]$, co ostatecznie dowodzi wykazywanej własności. ■

Własność 3

Jeżeli sekwencje (A) i (B) są identyczne do pozycji k; $A(i) = B(i)$ dla $i \leq k$ to w wyniku krzyżowania tych rozwiązań otrzymujemy rozwiązanie o tej samej początkowej sekwencji.

Dowód: własność ta wynika bezpośrednio z własności 1. Pierwszy element w sekwencji kodowany przez wektor A i $B; A(1) = B(1)$ pozostaje zawsze przed wszystkimi innymi elementami w potomnym rozwiązaniu C, niezależnie od parametru q. Jest więc pierwszym elementem w (C); $C(1) = A(1) = B(1)$. Dalszą analizę, i-tego elementu $A(i) = B(i)$, przeprowadza się identycznie jak dla pierwszego tylko nie uwzględnia się już $i-1$ elementów początkowych; $C(k), k < i$. Ostatecznie dopóki $A(k) = B(k)$, $k \leq i$ element $C(i) = A(i) = B(i)$. ■

Własność 4

Jeżeli sekwencje (A) i (B) są identyczne od pozycji k; $A(i) = B(i)$ dla $i \geq k$ to w wyniku krzyżowania tych rozwiązań otrzymujemy rozwiązanie o tej samej sekwencji końcowej.

Dowód własności 4 wykorzystuje własność 1 jest analogiczny do dowodu własności 3 więc zostanie tu pominięty.

4. Zewnętrzne i wewnętrzne cechy kodowanej wektorowo permutacji

W algorytmach genetycznych, rozwiązanie kodowane jest za pomocą paczki danych zwanej genotypem. Podobnie jak w naturze, także w algorytmach genetycznych, genotyp koduje fenotyp (postać osobnika). Genotyp zawiera fragmenty informacji niebiorące bezpośrednio udziału podczas tworzenia osobnika (geny uśpione). Uśpione geny w bieżącym osobniku a przekazane przez dziedziczenie potomkom mogą okazać się istotne i decydować bezpośrednio o wyglądzie i przystosowaniu potomków powstających w kolejnych pokoleniach. Podobnie w wektorowym kodowaniu permutacji tyko część informacji bezpośrednio odpowiedzialna jest za cechy zewnętrzne rozwiązania czyli za kodowaną sekwencję. Pozostała część informacji zwana cechami wewnętrznymi, steruje bezpośrednio działaniem operatorów mutacji oraz ma wpływ na rezultat działania operatorów krzyżowania.

W przypadku operatorów mutacji cechy wewnętrzne wektora permutacji decydują o prawdopodobieństwie mutacji poszczególnych składowych wektora. Skutkuje to tym, że cechy wewnętrzne mają bezpośredni wpływ na numer indeksu wybieranego elementu oraz na pozycję w którą ten element zostanie przesunięty w kodowanej sekwencji (patrz przykład 1). Ilustracja wpływu cech wewnętrznych wektora permutacji na operator krzyżowania osobników pokazana jest natomiast w przykładzie 2. Opisane jest tam krzyżowanie dwóch par osobników (A, B) i (A, B'), gdzie wektory permutacji B i B' różnią się od siebie tylko cechami wewnętrznymi, cechy zewnętrzne czyli kodowana sekwencja, są identyczne; $(B) = (B')$. W wyniku tych dwóch krzyżowań otrzymane zostają odpowiednio osobniki C i C', które nie tylko różnią się od siebie cechami wewnętrznymi, ale i kodowaną sekwencją $(C) \neq (C')$.

5. Podsumowanie

W pracy zaprezentowano alternatywny sposób kodowania permutacji. Przedstawiono teoretyczne uzasadnienie jego stosowania oraz wskazano na pewne nowe własności nieistniejące w przy klasycznym sposobie kodowania. Wstępne, nieopublikowane, wyniki badań numerycznych potwierdzają, iż zastosowanie wektorowego kodowania permutacji z przedstawionymi operatorami mutacji i krzyżowania zwiększa wydajność algorytmów w stosunku do ich klasycznych odpowiedników.

Literatura

1. Croes G.: A method for solving traveling-salesman problems. Operations Research, 1958, s. 791–812.
2. Goldberg D.E.: Algorytmy genetyczne i ich zastosowania. WNT, Warszawa, 1995.
3. Holland J.H.: Genetic Algorithms. Scientific American, 44, 1992.
4. Kirkpatrick S., Gelatt C.D. Vecchi M.P.: Optimisation by simulated annealing. Science 220, 1983, s. 671-680.
5. Yamada T., Nakano R.: A GA with multi-step crossover for job-shop scheduling problems. Proc. of Int. Conf. on GAs in Engneering Systems: Innovations and Applications, 1995, 146-151.
6. Nagata Y.: New EAX crossover for large TSP instances. Parallel Problem Solving from Nature-PPSN IX, 2006, s. 372–381.
7. Nawaz M., Enscore Jr. E.E., Ham I.: A heuristic algorithm for the m-machine, n-job flow-shop sequencing problem. OMEGA International Journal of Management Science, 11, 1983, s. 91-95.
8. Poon P.W., Carter J.N.: Genetic algorithm crossover operators for ordering applications. Computers & Operations Research, V22, 1995, s. 135-147.
9. Reeves C. R., Yamada T.: Solving the Csum Permutation Flowshop Scheduling Problem by Genetic Local Search. IEEE International Conference on Evolutionary Computation, 1998, s. 230–234.

Dr inż. Mariusz Makuchowski
Instytut Informatyki Automatyki i Robotyki
Politechnika Wrocławska
50-372 Wrocław, ul. Janiszewskiego 11/17
tel./fax: (0-71) 320-29-61
e-mail: mariusz.makuchowski@pwr.wroc.pl