PROPOZYCJA STRATEGII EWOLUCYJNEGO GENEROWANIA STRUKTUR DANYCH OPARTYCH O POZIOME DRZEWA DANYCH DLA POTRZEB TESTÓW

Marek ŻUKOWICZ, Michał MARKIEWICZ
Politechnika Rzeszowska

Streszczenie: Celem artykułu jest prezentacja definicji matematycznego modelu obiektu, który w informatyce znany jest jako TreeList oraz wykorzystanie tego modelu do zaprojektowania algorytmu ewolucyjnego, którego zadaniem jest generowanie struktur opartych na obiekcie TreeList. Pierwszy rozdział wprowadza czytelnika w problem, jakim jest prezentacja danych za pomocą wspomnianego obiektu TreeList. Drugi rozdział opisuje problem testowania struktur danych opartych o TreeList. Rozdział trzeci natomiast prezentuje matematyczny model obiektu TreeList oraz miary, które można wykorzystać w celu określenia użyteczności struktur utworzonych za pomocą wspomnianych obiektów oraz w strategii ewolucyjnej, która generuje te struktury dla potrzeby ich testowania. Ostatni rozdział zawiera krótkie podsumowanie oraz plany przyszłych badań związanych z zaprezentowanym w artykule algorytmem.

Słowa kluczowe: obiekt TreeList, algorytmy ewolucyjne, strategia ewolucyjna, testowanie

WPROWADZENIE

Bardzo często w nauce i technice, również w dydaktyce ma miejsce taka sytuacja, że zjawisko czy proces jest opisane za pomocą modelu, który jest zbliżony do konkretnej struktury występującej w otaczającej człowieka przyrodzie. W niniejszym artykule skupiono się na opisie przechowywania danych za pomocą struktury, która nosi nazwę drzewa. W dalszych rozdziałach zostaną omówione drzewa poziome, które stanowią pewną klasę tradycyjnych drzew. W celu zachowania spójności w następnych trzech rozdziałach przedstawione zostały informacje na temat struktur, które nazwane są drzewami poziomymi, co pozwoli zrozumieć główny problem poruszony w publikacji.

Drzewa danych są klasą grafów znanych z matematyki. Z punktu widzenia testowania oprogramowania pod kątem użyteczności oraz prezentacji danych, bardzo ważnym aspektem jest ułożenie tych danych oraz prezentacja struktury danych jako całości. Dla użytkownika bardzo wygodną i przejrzystą formą prezentacji są struktury, które nazywane są TreeList [7, 8]. Właściwości struktury TreeList są analogiczne do własności drzew danych znanych w informatyce [1] z tą różnicą, że wykorzystuje się je do poziomej prezentacji danych, gdzie dane zależą od siebie w sposób hierarchiczny. Przykładowe takie drzewo prezentuje rysunek 1. Takie drzewa można uzyskać po prostu poprzez ułożenie drzew opisanych z w pozycji [1] od lewej strony do prawej strony ekranu. Daje to wizualny efekt poziomej prezentacji danych. Najczęściej wyżej przedstawiona struktura jest zaimplementowana tak, że użytkownik ma możliwość rozwijania i zwijania gałęzi. Daje to bardzo dobry wizualny efekt prezentacji danych, jeśli taka struktura wykorzystana jest do prezentacji kosztów, ponieważ koszty nie zawsze muszą być prezentowane na analitycznym poziomie. Wyżej przedstawione drzewo jest przykładem prezentacji kosztów. Dane, które ono zawiera, mogą być dowolne, ważne jest tylko to, żeby były od siebie hierarchicznie zależne.

Numer	Nazwa pozycji	Wartość pozycji	Wartość przychodów
1	**ROBOCIZNA**	**301 199,25 zł**	**436 738,91 zł**
1.1	**ROBOCIZNA WŁASNA**	**299 199,25 zł**	**433 838,91 zł**
1.1.1	Płace	45 645,00 zł	66 185,25 zł
1.1.2	Narzuty płacowe	45 654,00 zł	66 198,30 zł
1.1.3	Narzędzia osobiste	2 343,00 zł	3 397,35 zł
1.1.4	Przewozy pracowników	2 796,00 zł	4 054,20 zł
1.1.5	Delegacje - noclegi	13 424,50 zł	19 465,52 zł
1.1.6	Delegacje - diety i ryczałty	34 670,00 zł	50 271,50 zł
1.1.7	Posiłki i napoje regeneracyjne	2 249,00 zł	3 261,05 zł
1.1.8	Koszty BHP, odzieży	67 500,00 zł	97 875,00 zł
1.1.9	Badania okresowe	2 553,60 zł	3 702,72 zł
1.1.10	Szkolenia pracowników	76 700,00 zł	111 215,00 zł
1.1.11	Ekwiwalent za pranie	5 664,15 zł	8 213,02 zł
1.2	**ROBOCIZNA OBCA**	**2 000,00 zł**	**2 900,00 zł**
1.2.1	Umowy zlecenia - płace	1 000,00 zł	1 450,00 zł
1.2.2	Umowy zlecenia - narzuty płacowe	1 000,00 zł	1 450,00 zł
2	**MATERIAŁY**	**0,00 zł**	**0,00 zł**
3	**SPRZĘT, TRANSPORT**	**3 000,00 zł**	**4 350,00 zł**
3.1	Najem sprzętu	1 000,00 zł	1 450,00 zł
3.2	Usługi transportowe	1 000,00 zł	1 450,00 zł
3.3	Zakup sprzetu, amortyzacja	1 000,00 zł	1 450,00 zł
4	**PODWYKONAWCY**	**4 000,00 zł**	**5 800,00 zł**
5	**KOSZTY ORGANIZACJI BUDOWY**	**46 000,00 zł**	**66 700,00 zł**

Rys. 1 Drzewo prezentujące koszty i przychody w systemie B2B OPTIbud autorstwa firmy Opteam

PROBLEM TESTOWANIA DRZEW POZIOMYCH

Bardzo ważnym etapem projektowania testów jest przygotowanie danych testowych (danych wejściowych) oraz struktur dla testów. W artykule skupiono się głównie na wygenerowaniu struktur przeznaczonych do przyjmowania oraz przetwarzania danych wejściowych wielowymiarowych. Takie postępowanie jest czynnością, którą należy wykonać przed dostarczaniem gotowych danych wejściowych do struktury drzewa. Struktury te powinny być jak najlepiej dopasowane do specyfikacji testowanego programu. W artykule zostanie przedstawione podejście, które za pomocą ewolucyjnego algorytmu oraz podanych początkowych struktur tworzy nowe lepiej dopasowane do potrzeb testów struktury, na przykładzie klasy TreeListControl podczas projektowania budżetów budowlanych w systemie B2B OPTIbud. Algorytmy ewolucyjne mają sporo cech, które pozwalają w pewien sposób kontrować proces ewolucji [6], a dzięki temu istnieje duże prawdopodobieństwo, że wygenerowane struktury będą odznaczać się takimi cechami jak [3]:

- Zależność od systemu (jakość zależy od aspektów technologicznych),
- Zupełność (stopień, w jakim te dane mają wartość dla wszystkich atrybutów w systemie),
- Aktualność,
- Wiarygodność,
- Spójność (np. niesprzeczność pomiędzy danymi testowymi),
- Precyzyjność (dotyczy testów na liczbach, w aplikacjach które wymagają dużej dokładności np. przeliczanie miar czy jednostek),
- Zależność od kontekstu.

Muszą one być spełnione dla poprawnych danych testowych [2, 4, 5]. Niewiele jest takich publikacji, które opisywałaby strategię testowania struktur danych opartych na

poziomych drzewach. W związku z tym autorzy postanowili opisać wspomniany problem i zaproponować pewne rozwiązanie generowania drzew poziomych tak, aby w wyniku procesów ewolucyjnych dostarczone zostały struktury odpowiadające preferencjom osób projektujących testy.

MATEMATYCZNY OPIS STRUKTUR SKŁADAJĄCYCH SIĘ Z POZIOMYCH DRZEW I MIARY ICH UŻYTECZNOŚCI

Projektując algorytm ewolucyjny, należy najpierw przyjąć pewne założenia co do struktury danych opartej o drzewa poziome oraz wielkości, które można zmierzyć w celu sformułowania funkcji dopasowania.

Założenia dotyczące struktury D złożonej z poziomych drzew są następujące:

a) istnieje co najmniej jedno drzewo (prezentujące dane w poziomie),
b) węzeł (w tym również korzeń) posiada co najmniej jedną cechę, która opisana jest liczbowo (liczbą rzeczywistą dodatnią),
c) obowiązuje następująca zasada: wartość konkretnej cechy liczbowej węzła na poziomie n jest sumą wartości dzieci tego węzła, ale tylko tych, które mają poziom (n + 1),
d) dowolna liczbowa cecha węzła x (w tym korzenia) może zostać podzielona (ale nie musi) na m okresów o wartościach x_i, a w takim przypadku suma wartości dla poszczególnych okresów jest równa, czyli:

$$x = \sum_{i=1}^{m} x_i \tag{1}$$

Ilość możliwości wygenerowania wyżej opisanych struktur, jakie wynikają z założeń jest nieskończona. W praktyce, w systemach informatycznych struktury posiadają skończony rozmiar. Wobec tego potrzebna jest strategia, która dostarczy struktury, które z dużym prawdopodobieństwem zostaną zastosowane w praktyce. Aby osiągnąć taki cel, należy określić pewne miary związane z wyżej opisaną strukturą. Przed wygenerowaniem struktur wzięte zostaną pod uwagę następujące ograniczenia:

1) ilość drzew k w strukturze D,
2) poziom s_k zagnieżdżenia k-tego drzewa (poziom na którym znajdują się liście danego drzewa),
3) łączna ilość dzieci t_{kj} dla węzła na poziomie j w k-tym drzewie,
4) ilość cech liczbowych u – wspólna własność dla wszystkich drzew w D,
5) ilość cech nie liczbowych v – wspólna własność dla wszystkich drzew D.

Kodowanie informacji opisujących pojedyncze drzewo będzie realizowane za pomocą wektora:

$$d_k = [s_k\{t_{k1},\ldots,t_{ksk}\}] \tag{2}$$

gdzie:

d_k – numer drzewa,
s_k – poziom zagnieżdżenia drzewa k,
t_k – łączna liczba węzłów na poziomie j dla drzewa k (poziom korzenia jest równy 0).

Niech dla pojedynczego drzewa zdefiniowana będzie następująca funkcja F_k mierząca użyteczność:

$$F_k(d_k) = \frac{1}{w_1 + w_2}\left[w_1 f(s_k) + \frac{w_2}{s_k}\left(\sum_{j=1}^{s_k} \frac{g_j(t_{kj})}{2} \right) \right] \tag{3}$$

gdzie:
funkcje $f, g : \mathrm{N} \rightarrow (0,1]$, natomiast w_1, w_2 są wagami.
Łatwo zauważyć, że funkcja F_k również przyjmuje wartości z przedziału (0,1]. Struktura, która będzie składała się z sumy pojedynczych drzew będzie miała następujący opis:

$$D_i(u_i,v_i) = \left\{ \begin{bmatrix} s_1, \{t_{11},....,t_{1s_1}\} \end{bmatrix} \begin{bmatrix} s_2, \{t_{21},...,t_{2s_2}\} \end{bmatrix} \begin{bmatrix} s_k, \{t_{k1},...,t_{ks_k}\} \end{bmatrix} \right\} \tag{4}$$

gdzie:
i – numer struktury składającej się z drzew poziomych,
u_i, v_i – cechy liczbowe i nieliczbowe wspólne dla drzew w D_i.

Struktura ta będzie nazywana w dalszej części artykułu macierzą struktury drzew poziomych.

Niech $D = \{D_1,\ldots,D_n\}$ będzie zbiorem struktur złożonych z poziomych drzew.
Funkcja dopasowania $F : D \rightarrow (0,1]$ dla struktury D_i dana jest wzorem:

$$F[D_i(u_i,v_i)] = \frac{w_c(h(u_i) + l(v_i)) + \frac{w_d}{n}\left(\sum_{k=1}^{n} F_k(d_k)\right)}{w_c + w_d} \tag{5}$$

gdzie:
w_c, w_d – wagi dla funkcji F,
n – liczba drzew w strukturze $D_i(u_i\ v_i)$.
Mając zdefiniowane funkcje dopasowania, można zaprojektować algorytm ewolucyjny, który będzie opisany w następnym rozdziale [6].

PROJEKT ALGORYTMU EWOLUCYJNEGO W ZBIORZE *D*

W algorytmach ewolucyjnych istnieje zjawisko krzyżowania elementów oraz ich mutacja [6]. Wobec tego do zaprojektowania algorytmu genetycznego potrzeba jeszcze dwóch funkcji: jedna odpowiada za krzyżowanie elementów ze sobą, a druga to operator mutacji. Wobec tego podany jest algorytm: Algorytm KM (Krzyżowanie i mutacja drzew poziomych). Niech $\mathrm{D}_1, \mathrm{D}_2$ będą strukturami złożonymi z drzew poziomych. Niech D_1 posiada n drzew natomiast D_2 posiada k drzew. Algorytm KM zdefiniowany jest jako funkcja

$$\mathrm{X}(\mathrm{D}_1, \mathrm{D}_2) : \mathrm{D} \times \mathrm{D} \rightarrow \mathrm{D}, \tag{6}$$

dana za pomocą warunków:

1) Weź resztę z dzielenia n przez 2, dodaj 1, zapisz obliczoną wartość jako x;
2) Weź resztę z dzielenia k przez 2, dodaj 1, zapisz obliczoną wartość jako y;
3) Stwórz nową strukturę poprzez połączenie x początkowych drzew, zaczynając od pierwszego wiersza w macierzy struktury D_1, z wierszami ze struktury D_2 zaczynając od wiersza o numerze y;
4) zapisz liczbę drzew w nowej strukturze jako s;
5) Dla każdego wiersza w nowopowstałej strukturze zapisz liczby $(\mathrm{t}_{i1}), \ldots, (\mathrm{t}_{ik})$ binarnie;
6) for $(\mathrm{i} = 1; \mathrm{i} \leq \mathrm{s}; \mathrm{s}++)$

7) do{
8) k = 1;
9) Przeprowadź mutację w elementu t_{ik} z prawdopodobieństwem q, bit po bicie;
10) Jeśli po mutacji element t_{ik} mieści się w dziedzinie odpowiedniej funkcji, której jest argumentem, to zaakceptuj mutację;
11) Jeśli po mutacji element t_{ik} nie mieści się w dziedzinie odpowiedniej funkcji, to nie zostaw element bez zmian;
12) k ++;
13) }While (t_{ik} istnieje w drzewie s);
14) Ilość cech liczbowych ze struktur D_1, D_2 wybierz losowo z wartości u_1 oraz u_2 i zapisz jako u;
15) Ilość cech nie liczbowych oraz liczbowych wybierz losowo ze struktur D_1, D_2 z wartości v_1 oraz v_2 i zapisz jako v;
16) Zapisz nowo powstałą strukturę z wylosowanymi parametrami u oraz v;

Mając zdefiniowany algorytm realizujący proces krzyżowania oraz mutacji, można podać algorytm ewolucyjny, którego dziedziną jest zbiór D: Algorytm ALGEN $(w_1, w_2, w_c, w_d, q, n, m, c, fv)$ (Algorytm ewolucyjny dla struktur $D_1, \dots, D_n$).

{

1. Wylosuj m struktur i oznacz je jako $D'_1, D'_2, D'_3, \dots, D'_m$;

for(j = 1; j < c; j + +)

{

2. Twórz nowe struktury poprzez zastosowanie algorytmu KM:

$D''_1 = X(D'_1, D'_2)$,
$D''_2 = X(D'_2, D'_3)$,
$D''_3 = X(D'_3, D'_4)$,
.....
$D''_m = X(D'_m, D'_1)$,

3. Porównaj wyrażenia $s = \sum_{i=1}^{m} F[D'_i(u_i, v_i)]$ oraz $l = \sum_{i=1}^{m} F[D''_i(u_i, v_i)]$

a) Jeśli $s \geq l$, to rozpocznij następną iterację pętli;
b) Jeśli $s < l$ to zamień elementy $D'_1, D'_2, D'_3, \dots, D'_m$ odpowiednio na $D''_1, D''_2, D''_3, \dots, D''_m$,
c) Jeśli $l > fv$, to przerwij pętlę i zapamiętaj struktury $D'_1, D'_2, D'_3, \dots, D'_m$

}

4. Zwróć wygenerowane pokolenie $D'_1, D'_2, D'_3, \dots, D'_m$.

}

gdzie:

w_1, w_2, w_c, w_d – wagi funkcji mierzących użyteczność (opisane w poprzednim rozdziale),

q – prawdopodobieństwo mutacji,

n – ilość wszystkich dostarczonych struktur,

c – ograniczenie ilości iteracji,

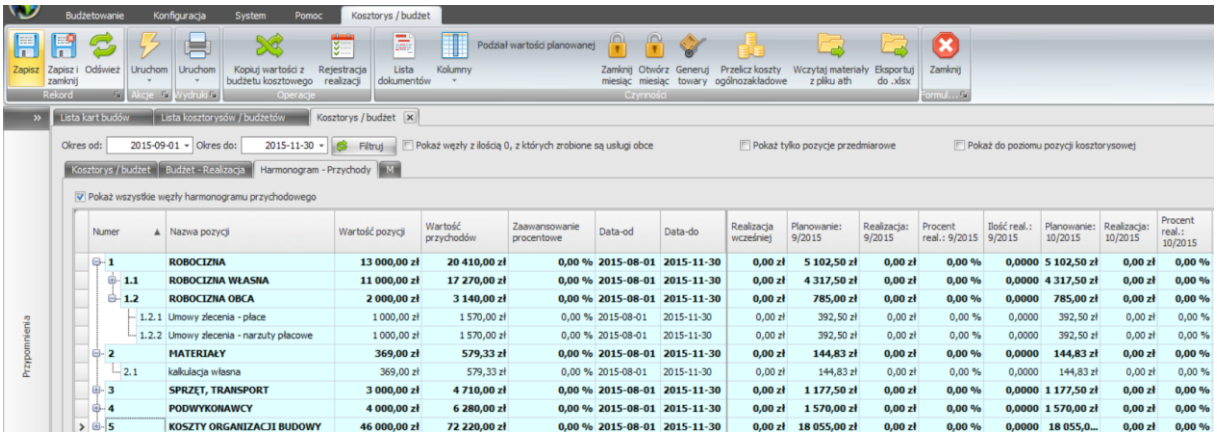
fv – wartość, po której przekroczeniu algorytm zatrzymuje się i zwraca wynik.

Ten parametr decyduje o tym, czy generować struktury w kolejnej iteracji, czy nie. Należy również zwrócić uwagę, że wartości l, v nie mogą być większe niż liczna m.

ZASTOSOWANIE ALGORYTMU ALGEN DO TESTÓW BUDŻETÓW W SYSTEMIE B2B OPTIBUD

Budżety w systemie B2B OPTIbud

Moduł Budżety w systemie B2B OPTIbud, którego twórcą jest firma Opteam, służy do tworzenia kosztorysów budowlanych. System B2B OPTIbud stworzony został dla polskich firm budowlanych. Jego podstawowe funkcjonalności, to: tworzenie budżetów dla prowadzonych projektów w oparciu dane z wielu źródeł, przekształcanie budżetów do harmonogramów oraz odzwierciedlenie na osi czasu realizacji projektu [3]. Do modułu wbudowane zostaną mechanizmy grupowania, scalania, rozbijania pozycji budżetowych oraz możliwość tworzenia różnego rodzaju budżetów: uproszczonych lub pełnych, śledzenie stopnia realizacji projektu od strony kosztowej jak również czasowej. Główny formularz budżetu uproszczonego w systemie B2B OPTIbud przedstawia rysunek 2.



Numer	Nazwa pozycji	Wartość pozycji	Wartość przychodów	Zaawansowanie procentowe	Data-od	Data-do	Realizacja wcześniej	Planowanie: 9/2015	Realizacja: 9/2015	Procent real.: 9/2015	Ilość real.: 9/2015	Planowanie: 10/2015	Realizacja: 10/2015	Procent real.: 10/2015
1	ROBOCIZNA	13 000,00 zł	20 410,00 zł	0,00 %	2015-08-01	2015-11-30	0,00 zł	5 102,50 zł	0,00 zł	0,00 %	0,0000	5 102,50 zł	0,00 zł	0,00 %
1.1	ROBOCIZNA WŁASNA	11 000,00 zł	17 270,00 zł	0,00 %	2015-08-01	2015-11-30	0,00 zł	4 317,50 zł	0,00 zł	0,00 %	0,0000	4 317,50 zł	0,00 zł	0,00 %
1.2	ROBOCIZNA OBCA	2 000,00 zł	3 140,00 zł	0,00 %	2015-08-01	2015-11-30	0,00 zł	785,00 zł	0,00 zł	0,00 %	0,0000	785,00 zł	0,00 zł	0,00 %
1.2.1	Umowy zlecenia - płace	1 000,00 zł	1 570,00 zł	0,00 %	2015-08-01	2015-11-30	0,00 zł	392,50 zł	0,00 zł	0,00 %	0,0000	392,50 zł	0,00 zł	0,00 %
1.2.2	Umowy zlecenia - narzuty płacowe	1 000,00 zł	1 570,00 zł	0,00 %	2015-08-01	2015-11-30	0,00 zł	392,50 zł	0,00 zł	0,00 %	0,0000	392,50 zł	0,00 zł	0,00 %
2	MATERIAŁY	369,00 zł	579,33 zł	0,00 %	2015-08-01	2015-11-30	0,00 zł	144,83 zł	0,00 zł	0,00 %	0,0000	144,83 zł	0,00 zł	0,00 %
2.1	kalkulacja własna	369,00 zł	579,33 zł	0,00 %	2015-08-01	2015-11-30	0,00 zł	144,83 zł	0,00 zł	0,00 %	0,0000	144,83 zł	0,00 zł	0,00 %
3	SPRZĘT, TRANSPORT	3 000,00 zł	4 710,00 zł	0,00 %	2015-08-01	2015-11-30	0,00 zł	1 177,50 zł	0,00 zł	0,00 %	0,0000	1 177,50 zł	0,00 zł	0,00 %
4	PODWYKONAWCY	4 000,00 zł	6 280,00 zł	0,00 %	2015-08-01	2015-11-30	0,00 zł	1 570,00 zł	0,00 zł	0,00 %	0,0000	1 570,00 zł	0,00 zł	0,00 %
5	KOSZTY ORGANIZACJI BUDOWY	46 000,00 zł	72 220,00 zł	0,00 %	2015-08-01	2015-11-30	0,00 zł	18 055,00 zł	0,00 zł	0,00 %	0,0000	18 055,0...	0,00 zł	0,00 %

Rys. 2 Formularz kosztów przykładowego budżetu budowlanego w systemie B2B OPTIbud

Struktura prezentująca koszty jest analogiczna do struktury opisanej w rozdziale Matematyczny opis struktur składających się z poziomych drzew i miary ich użyteczności. Dzięki temu ta struktura będzie dobrym przykładem do zaprezentowania działania algorytmu ALGEN. Cechy liczbowe widoczne na obrazku to kolumny: Wartość pozycji, Wartość przychodów, Zaawansowanie procentowe, Realizacja wcześniej, Realizacja 9/2015, Planowanie 9/2015, Procent real.: 9/2015,itd. Cechy nie liczbowe natomiast to kolumny Numer, Nazwa Pozycji, Data-od, Data-do. Kolumny zaczynające się od słów Planowanie, to wartości powstałe poprzez podzielenie planowanych przychodów z budżetu przez 4, ponieważ tak jest zaplanowany budżet (Data-od i Data-do). Ilość drzew jest równa 5. Poziom drzewa, które go korzeń ma numer 1 wynosi 2 (1 korzeń, węzły 1.1 oraz 1.2 – poziom I, natomiast węzły 1.2.1 i 1.2.2 – poziom II). W poprzednim rozdziale zostały podane definicje funkcji mierzących dopasowanie struktur do potrzeb testów. Wzory (3) oraz (5) zawierają w sobie funkcje f, g, h, l. Niżej przedstawione są definicje tych funkcji, które zostaną wykorzystane w implementacji algorytmów KM i ALGEN dla systemu OPTIbud:

1) $f(x) = -\frac{1}{4}|x-4|+1, \quad$ dla $1 \leq x \leq 8,$ (7)

2) $$g_j(x) = \begin{cases} -\frac{1}{3}|x-3|+1, & \text{dla } 1 \le x \le 6, \quad \text{j} = 1 \\ -\frac{1}{6}|x-6|+1, & \text{dla } 3 \le x \le 12, \ \text{j} = 2 \\ -\frac{1}{12}|x-12|+1, & \text{dla } 8 \le x \le 24, \ \text{j} = 3 \\ -\frac{1}{24}|x-24|+1, & \text{dla } 12 \le x \le 48, \text{j} = 4 \\ -\frac{1}{30}|x-30|+1, & \text{dla } 20 \le x \le 48, \text{j} > 4 \end{cases}, \quad (8)$$

3) $$h(u) = -\frac{1}{6}|u-6|+1, \quad \text{dla } 1 \le u \le 12, \quad (9)$$

4) $$l(v) = -\frac{1}{3}|v-3|+1, \quad \text{dla } 1 \le v \le 6, \quad (10)$$

Przepisy wyżej przedstawionych funkcji zostały celowo dobrane w taki sposób, aby nie osiągały wartości większych od 1 oraz tak, aby algorytm ewolucyjny generował możliwie jak najlepsze struktury dla potrzeb testowania (nie posiadające olbrzymiej liczby węzłów).

Działanie algorytmu ALGEN w module Budżety w systemie B2B OPTIbud

Niech początkowa populacja wejściowa D będzie składała się z następujących budżetów:

$$D_1 = \begin{Bmatrix} [4,\{1,2,4,8\}] \\ [3,\{3,7,18\}] \\ [4,\{5,6,12,24\}] \\ [2,\{7,8\}] \end{Bmatrix}, \quad D_2 = \begin{Bmatrix} [4,\{1,2,4,8\}] \\ [3,\{3,7,18\}] \\ [4,\{5,6,12,24\}] \\ [2,\{7,8\}] \end{Bmatrix}, \quad D_3 = \begin{Bmatrix} [3,\{2,12,24\}] \\ [2,\{6,8,16\}] \\ [3,\{4,8,12\}] \\ [4,\{2,4,8,16\}] \end{Bmatrix},$$

$$D_4 = \begin{Bmatrix} [2,\{9,92\}] \\ [2,\{7,29\}] \\ [2,\{4,60\}] \\ [2,\{2,18\}] \end{Bmatrix}, \quad D_5 = \begin{Bmatrix} [2,\{1,4\}] \\ [2,\{2,3\}] \\ [3,\{4,20,60\}] \\ [3,\{2,9,16\}] \end{Bmatrix}, \quad D_6 = \begin{Bmatrix} [3,\{9,27,92\}] \\ [3,\{7,16,29\}] \\ [2,\{4,60\}] \\ [2,\{2,18\}] \end{Bmatrix},$$

$$D_7 = \begin{Bmatrix} [4,\{1,2,9,92\}] \\ [5,\{1,2,7,17,29\}] \\ [4,\{4,8,16,60\}] \\ [3,\{2,18,44\}] \end{Bmatrix}, \quad D_8 = \begin{Bmatrix} [3,\{9,9,92\}] \\ [3,\{2,7,29\}] \\ [3,\{2,4,60\}] \\ [3,\{3,18,44\}] \end{Bmatrix}, \quad D_9 = \begin{Bmatrix} [2,\{9,92\}] \\ [2,\{7,33\}] \\ [2,\{4,44\}] \\ [2,\{2,18\}] \end{Bmatrix},$$

$$D_{10} = \begin{Bmatrix} [4,\{1,5,9,92\}] \\ [3,\{7,16,29\}] \\ [2,\{4,60\}] \\ [3,\{2,7,18\}] \end{Bmatrix},$$

odpowiednio z parametrami:

$u_1 = 6, v_1 = 2, u_2 = 8, v_2 = 2, u_3 = 7, v_3 = 3, u_4 = 7, v_4 = 3, u_5 = 4, v_5 = 2, u_6 = 5,$
$v_6 = 2, u_7 = 6,$
$v_7 = 2, u_8 = 7, v_8 = 3, u_9 = 8, v_9 = 3, u_{10} = 10, v_{10} = 4.$

Niech parametry wejściowe algorytmu ALGEN będą miały wartości:

$w_1 = 2, w_2 = 3, w_c = 2, w_d = 1, q = 0{,}2, n = 10, m = 4, c = 30, fv = 3{,}7.$

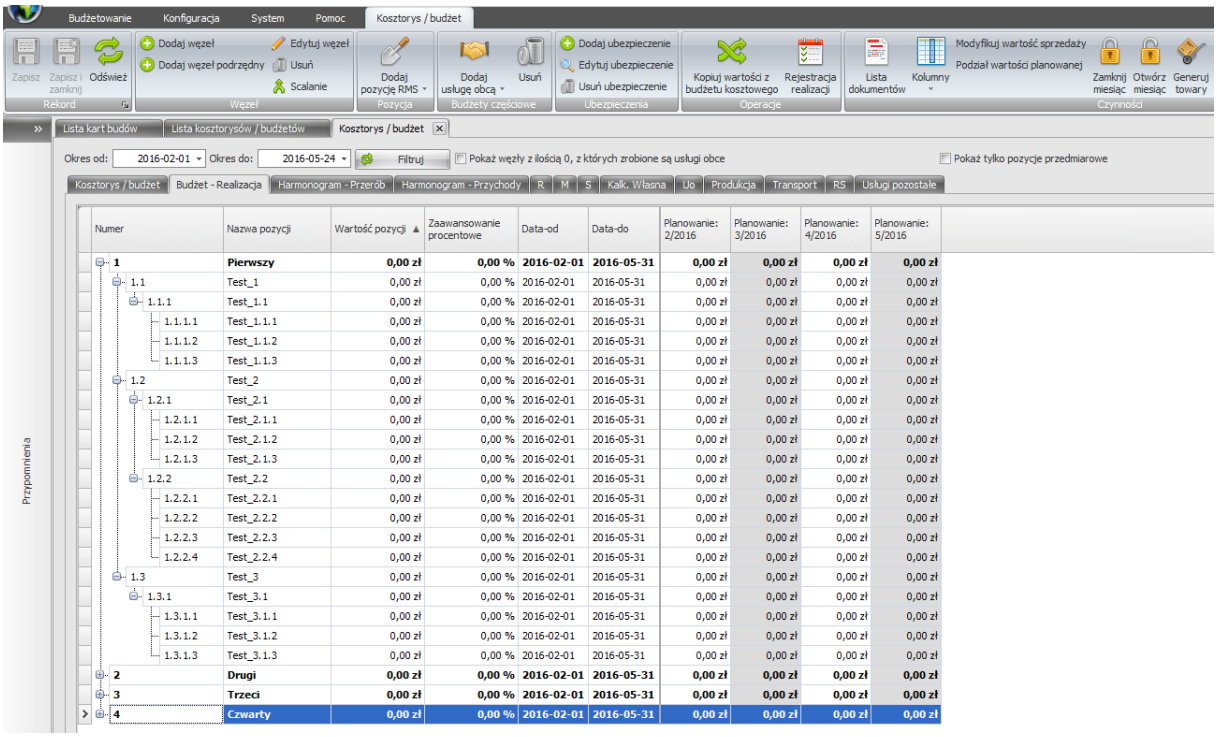
Wyniki zwrócone przez algorytm już po 6 iteracjach są następujące:

$$D_1' = \begin{Bmatrix} [4,\{1,3,4,13\}] \\ [3,\{2,3,3\}] \\ [4,\{1,4,15,5\}] \\ [2,\{3,2\}] \end{Bmatrix}, \quad D_2' = \begin{Bmatrix} [2,\{5,46\}] \\ [2,\{1,5\}] \\ [3,\{1,4,4\}] \\ [4,\{2,2,1,2\}] \end{Bmatrix}, \quad D_3' = \begin{Bmatrix} [4,\{1,2,6,14\}] \\ [5,\{1,1,4,17,20\}] \\ [4,\{4,3,2,8\}] \\ [2,\{3,5\}] \end{Bmatrix},$$

$$D_4' = \begin{Bmatrix} [3,\{3,1,19\}] \\ [3,\{3,11,22\}] \\ [2,\{6,12\}] \\ [2,\{1,23\}] \end{Bmatrix},$$

z parametrami $u_1' = 6, v_1' = 3, u_2' = 8, v_2' = 2, u_3' = 7, v_3' = 3, u_4' = 7, v_4' = 3$, przy wartościach $s = 3{,}5003, l = 3{,}7223$.

Przykładowo pierwsze drzewo w budżecie D_1' w systemie B2B OPTIbud przedstawiono na rysunku 3.



Rys. 3 Formularz kosztów o budżetu budowlanego D_1' w systemie B2B OPTIbud

Istnieje nieskończenie wiele możliwości testowania algorytmu ALGEN. Celem artykułu jednak było wskazanie podejścia metodycznego, dlatego wyniki związane z badaniem parametrów algorytmu ukażą się w kolejnej publikacji.

PODSUMOWANIE

Celem artykułu było opisanie struktur, prezentujących dane w celu zaprojektowania algorytmu ewolucyjnego, który generuje struktury tak, aby ich dopasowanie do testów było jak najlepsze. Cel ten został osiągnięty poprzez zaprojektowanie modelu matematycznego, za pomocą którego została opisana struktura składająca się z drzew poziomych. Funkcje dopasowania zostały celowo tak zdefiniowane, aby ich wartości mieściły się w przedziale (0,1], ponieważ można je szybko porównać do wartości procentowych.

W przyszłości autorzy zamierzają rozwinąć zagadnienie zawarte w tytule niniejszej pracy, modyfikując podany algorytm i porównać jego wyniki z algorytmem ALGEN. Autorzy zamierzają również opublikować artykuł prezentujący zachowanie algorytmu ze względu na różne kombinacje jego parametrów wejściowych.

LITERATURA

1. L. Banachowski, K. Diks and W. Rytter. *Algorytmy i struktury danych*, Warszawa: Wydawnictwa Naukowo-Techniczne, 2011.
2. D. Farley and J. Humble. *Ciągłe dostarczanie oprogramowania*, Gliwice: Helion, 2015.
3. M. Łobaziewicz. „Standard architektury modelu systemu B2B wspomagającego zarządzanie procesami budowlanymi", in *Od procesów do oprogramowania: badania i praktyka*, P. Kosciuszenko, M. Śmiałek and J. Swacha, Warszawa: Wydawnictwo Polskie Towarzystwo Informatyczne, 2015, pp. 111-120.
4. A. Piaskowy and R. Smilgin. *Dane Testowe: teoria i praktyka*, Gliwice: Helion, 2011.
5. A. Roman. *Testowanie i jakość oprogramowania*, Warszawa: Wydawnictwo Naukowe PWN, 2015.
6. D. Rutkowska, M. Piliński and L. Rutkowski. *Sieci neuronowe, algorytmy genetyczne i systemy rozmyte*, Warszawa: Wydawnictwo Naukowe PWN, 1997.
7. WinForms Tree List, [Online]. Available: https://www.devexpress.com/products/net/controls/winforms/tree_list/
8. WPF Tree List, [Online]. Available: https://www.devexpress.com/products/net/controls/wpf/tree_list/

mgr Marek Żukowicz, mgr Michał Markiewicz
Politechnika Rzeszowska, Wydział Elektrotechniki i Informatyki
Katedra Automatyki i Informatyki
ul. Wincentego Pola 2, 35-959 Rzeszów, Polska
e-mail: bobmarek@o2.pl; mmarkiewicz13@gmail.com

Data przesłania artykułu do Redakcji: 02.2016
Data akceptacji artykułu przez Redakcję: 04.2016