STUDIA INFORMATICA 2013
Volume 34 Number 1 (104)

Ewa LACH, Marianna HETMAN
Politechnika Śląska, Instytut Informatyki

OCENA PODOBIEŃSTWA DRZEW STRATEGII GENEROWANYCH ZA POMOCĄ PROGRAMOWANIA GENETYCZNEGO

Streszczenie. W pracy zaprezentowano nowy algorytm oceny stopnia podobieństwa drzew strategii, opisujących zachowanie trójwymiarowych postaci wirtualnych. Algorytm został użyty w trakcie automatycznego generowania strategii postaci, za pomocą programowania genetycznego w celu otrzymania bardziej zróżnicowanych populacji strategii z większą liczbą dostępnych ścieżek poszukiwań.

Słowa kluczowe: programowanie genetyczne, uczenie maszynowe, animacja behawioralna, porównywanie drzew

THE SIMILARITY MEASURE FOR STRATEGY TREES GENERATED BY MEANS OF GENETIC PROGRAMMING

Summary. The paper proposes new algorithm for measuring a similarity of strategy trees, describing behaviors of the three-dimensional virtual characters. The obtain measure is used in automatic generation of characters' strategies by means of genetic programming. The aim of the algorithm is to create more diversified populations with many different paths to explore.

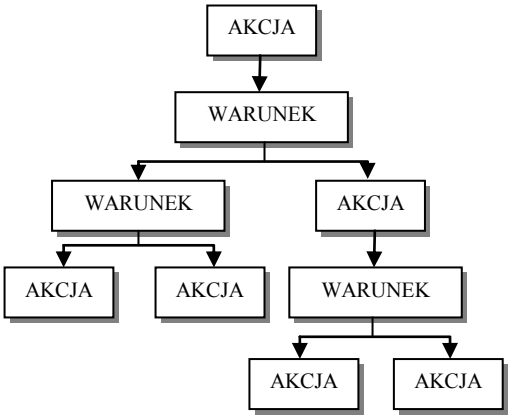
Keywords: genetic programming, machine learning, behavioural animation, tree comparison

1. Wprowadzenie

Na przestrzeni ostatnich lat można zaobserwować dynamiczną ewolucję symulatorów, aplikacji wirtualnych, gier komputerowych oraz komputerowych efektów specjalnych wykorzystywanych w przemyśle filmowym. Spowodowała ona wzrost zapotrzebowania na posta-

cie wirtualne działające w sposób symulujący zachowanie inteligentne – postępujące tak jak człowiek. Do opisu zachowania takich postaci są wykorzystywane złożone struktury (np. zbiory reguł decyzyjnych, maszyny stanów, sieci neuronowe) konstruowane na różne sposoby, w tym automatycznie. W niniejszej pracy zajęto się strategiami zachowania postaci przedstawionymi za pomocą struktur drzewiastych (drzew strategii), generowanych z użyciem programowania genetycznego. W szczególności skupiono się na analizie miar podobieństwa tworzonych drzew, które wpływają na jakość otrzymywanych zachowań wirtualnych postaci.

Drzewo strategii (rys. 1) opisuje plan działania wirtualnej postaci za pomocą predefiniowanych operacji warunkowych i akcji (węzłów drzewa). Operacje warunkowe sprawdzają stan postaci oraz jej otoczenia (nie zmieniając ich), warunkując ścieżkę jej kolejnych ruchów (wybór jednego z węzłów potomnych). Akcje zmieniają stan środowiska wirtualnego i postaci. Akcje dzielą się na proste i złożone. Akcje złożone są opisywane za pomocą kolejnych drzew strategii, których węzły są aktywowane sekwencyjnie lub równolegle. Akcje mogą być dowolnymi węzłami drzewa, podczas gdy operacje warunkowe nie mogą być liśćmi (węzłami drzewa nieposiadającymi węzłów potomnych). Drzewa strategii są oceniane za pomocą predefiniowanej funkcji oceny, która określa stopień wykonania założonego celu przez postać i jakość jej zachowań. Zdefiniowanie funkcji oceny stanowi jedno z większych wyzwań automatycznego generowania „inteligentnych" zachowań [6] i obecnie w większości jest wykonywane „ręcznie" przez animatora. Drzewa strategii mogą być tworzone ręcznie (przez człowieka) lub za pomocą dowolnej techniki pozwalającej na konstruowanie drzew z użyciem wcześniej zdefiniowanego zbioru węzłów (operacji warunkowych i akcji). Techniki te mogą opierać się na różnym stopniu wiedzy na temat środowiska i reguł podejmowania decyzji. W niniejszej pracy skupiono się na generowaniu drzew strategii za pomocą programowania genetycznego.



Rys. 1. Drzewo strategii
Fig. 1. Strategy tree

Programowanie genetyczne (PG) w procesie ewolucji populacji osobników (drzew strategii) dąży do optymalizacji funkcji przystosowania osobników (funkcji oceny strategii). Ewolucja opiera się na darwinowskiej zasadzie doboru naturalnego, wedle której przetrwają jedynie najlepiej ocenione osobniki. Istotnym problemem generacji drzew strategii przy użyciu programowania genetycznego jest spadek zróżnicowania populacji w trakcie procesu ewolucji. Aby zapobiec sytuacji, w której niektóre ścieżki rozwoju populacji zostałyby odrzucone na rzecz ścieżek już istniejących, można zastosować przy wyborze osobników potomnych algorytm określający ich podobieństwo. Obecnie do porównywania struktur będących wynikiem działania algorytmu programowania genetycznego najczęściej używa się ich wartości funkcji dopasowania. W zależności od sposobu obliczania funkcji dopasowania strategie o podobnej jej wartości mogą opisywać całkiem różne zachowania.

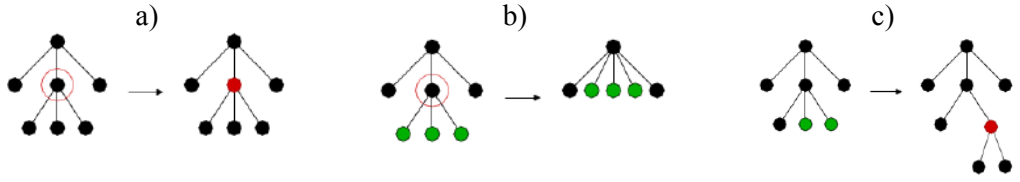
W celu poprawienia jakości strategii generowanych za pomocą PG w artykule zaproponowano zastosowanie innych niż funkcja dopasowania miar oceny podobieństwa strategii. W dalszej części pracy został opisany nowy algorytm STLD oceniający stopień podobieństwa drzew strategii oraz zostały przedstawione eksperymenty, które miały na celu ocenę algorytmu pod kątem jego zastosowania w procesie automatycznej generacji zachowań postaci wirtualnych z użyciem programowania genetycznego.

2. Miara podobieństwa drzew strategii

Większość metod porównywania struktur drzewiastych bazuje na następujących operacjach przeprowadzanych na węzłach:

- zamiana: modyfikowany jest opis węzła (rys. 2a).,
- usunięcie: kiedy węzeł jest usuwany, wszystkie jego dzieci stają się dziećmi jego ojca (rys. 2b),
- wstawienie: kiedy węzeł jest wstawiany, wszystkie węzły potomne jego ojca znajdujące się po jego lewej stronie stają się jego dziećmi (rys. 2c).

Dla każdych dwóch struktur drzewiastych jest możliwe znalezienie sekwencji operacji, która umożliwia przekształcenie jednego drzewa w drugie. Koszt sekwencji jest sumą kosztów zawierających się w niej operacji.

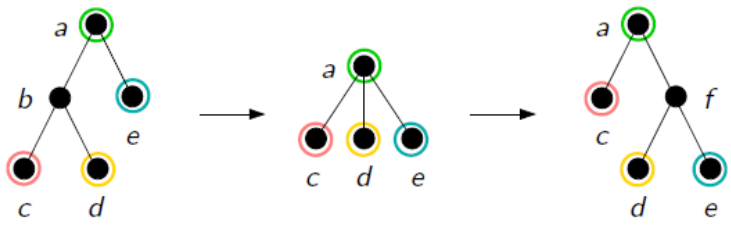


Rys. 2. Operacje na drzewach: a) zamiana węzła, b) usunięcie, c) wstawienie
Fig. 2. Tree operations: a) change, b) removal, c) insertion

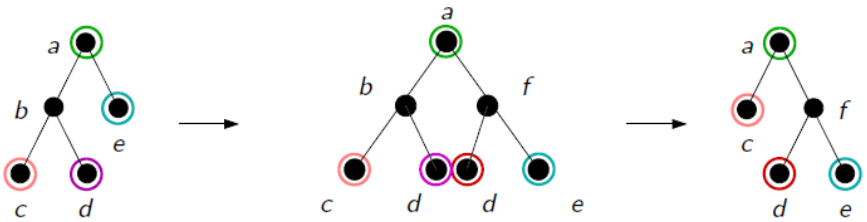
Do zagadnienia porównywania struktur drzewiastych stosuje się dwa główne podejścia:

- Poszukiwanie odległości edycyjnej (rys. 3), które polega na poszukiwaniu największego wspólnego poddrzewa i odnalezieniu najmniej kosztownej sekwencji operacji, która przekształca jedno drzewo w drugie (wszystkie przekształcenia są dozwolone).
- Sprawdzanie dopasowania (rys. 4), które polega na odnalezieniu drzewa, w które mogą być przekształcone porównywane drzewa jedynie z użyciem operacji zamiany i wstawienia, w sposób minimalizujący koszt wykonanych operacji. W przekształceniach wstawianie węzła powinno poprzedzać usuwanie węzła.

Pierwszy sprawny algorytm obliczający odległość dla drzew uporządkowanych został przedstawiony przez Zhanga i Shasha'e [1]. Dopasowanie drzew jako pierwsi zbadali Jiang, Wang i Zhang [3]. Następnie pojawiło się wiele modyfikacji wyżej wymienionych algorytmów, mających na celu zmniejszenie ich złożoności lub dostosowanie ich do konkretnych potrzeb [2, 4, 5].



Rys. 3. Przykładowe przekształcenie jednego drzewa na drugie metodą poszukiwania odległości edycyjnej [2]
Fig. 3. Optimal mapping for the two trees with the edit distance method [2]



Rys. 4. Przykładowe przekształcenie jednego drzewa na drugie metodą sprawdzania dopasowania [2]
Fig. 4. Optimal mapping for the two trees with the alignment method [2]

Analiza dostępnych metod porównywania struktur drzewiastych wykazała, że nie nadają się one do porównywania drzew strategii zachowania postaci. Głównym problemem, który czyni te metody nieprzydatnymi, jest dopuszczanie takich operacji, jak usunięcie czy przemieszczenie całego poddrzewa. Wynikiem przejścia przez strukturę reprezentującą strategię zachowania postaci jest linearna sekwencja ruchów. W przypadku przemieszczenia całego poddrzewa w inne miejsce w strukturze otrzymuje się tyle nowych wariantów sekwencji ruchów, ile dzieci ma korzeń przenoszonego drzewa. Dodatkowo są usuwane ścieżki związane z drzewem w jego poprzednim położeniu, a główna ścieżka jest ucinana.

Powoduje to wiele zmian w ścieżkach wynikowych – więc w rozumieniu wymienionych algorytmów pojedyncza operacja, która sprawia, że drzewa są uznawane za podobne, może diametralnie zmienić zachowanie postaci. Eliminacja takiego poddrzewa może doprowadzić do sytuacji, w której algorytm straci istotną ścieżkę rozwoju, a tym samym zmniejszą się szanse na znalezienie optymalnej ścieżki prowadzącej do celu.

Powyższe problemy spowodowały, że w artykule zaproponowano rozpatrywanie struktury drzewa strategii zachowania postaci jako zbioru potencjalnych sekwencji ruchów, które może wykonać postać. Oznaczało to poszukiwanie miary podobieństw drzew strategii za pomocą metody porównywania ciągów. Techniki służące do odnajdywania odległości edycyjnej (najmniejszej liczby operacji, która pozwoli przeprowadzić jeden ciąg na drugi) pomiędzy ciągami znaków posiadają wiele zastosowań (np. telekomunikacja, teoria informacji, kryptografia) i długą tradycję użycia: odległość Hamminga została wprowadzona w 1950 roku [7]. Do konstrukcji miary podobieństwa drzew strategii w artykule zastosowano odległość Levenshteina [8], która mierzy liczbę działań prostych przekształcających jeden ciąg znaków w drugi. Działaniami prostymi są:

- wstawienie nowego znaku do ciągu,
- usunięcie znaku z ciągu,
- zamiana znaku w ciągu na inny.

Przykładowo, odległość Levenshteina pomiędzy wyrazami: *granat* i *granit* wynosi 1, ponieważ do przeprowadzenia pierwszego na drugi wystarcza jedno działanie: zamiana litery *a* na *i*. Odległość Levenshteina pomiędzy wyrazami: *orczyk* i *oracz* wynosi 3 – do przeprowadzenia pierwszego ciągu znaków na drugi potrzeba trzech działań: usunięcia liter *y* i *k* oraz wstawienia litery *a*.

Zaproponowany w artykule algorytm porównywania drzew strategii STLD (Strategic Trees' Levenshten Distance) ma następujący przebieg:

a) Zamiana drzewa na listę sekwencji ruchów:
   i. pobranie korzenia drzewa,
   ii. jeżeli rodzic posiada jeden węzeł potomny: pobranie węzła potomnego,
   iii. jeżeli węzeł ma więcej niż jedno dziecko: utworzenie tylu kopii sekwencji wynikowej, ile dzieci (nie następuje zróżnicowanie sekwencji w zależności od wybranego dziecka),
   iv. dla każdego z dzieci powrót do punktu a(ii),
   v. jeżeli węzeł nie ma dzieci: zamknięcie bieżącej sekwencji ruchów.
b) Porównanie każdej sekwencji z pierwszego drzewa z każdą sekwencją z drugiego drzewa przy użyciu odległości Levenshteina. Jeżeli dla danej sekwencji z pierwszego drzewa znajdzie się identyczną sekwencję w drzewie drugim, sekwencja z drzewa drugiego jest

usuwana (działanie to ma na celu redukcję liczby porównań i uniknięcie sytuacji, w której do jednej sekwencji z drzewa drugiego mogłoby być przyporządkowanych kilka sekwencji z drzewa pierwszego).

c) Normalizacja otrzymanych odległości:
   i. podział wartości odległości przez liczbę elementów dłuższej porównywanej sekwencji,
   ii. przypisanie wartości wynikowej wartości 1 (oznaczającej, że sekwencje są zupełnie niepodobne) albo odległości z kroku c(i) (jeżeli otrzymana wartość jest mniejsza od górnej granicy dopuszczalnej różnicy ustawianej jako parametr algorytmu (przykładowo: 0,25)).
d) Przyporządkowanie każdej sekwencji z drzewa pierwszej najniższej znormalizowanej odległości.
e) Zwrócenie wartości wynikowej *stld*: średniej arytmetycznej z odległości sekwencji drzewa pierwszego, wyrażonej w procentach.

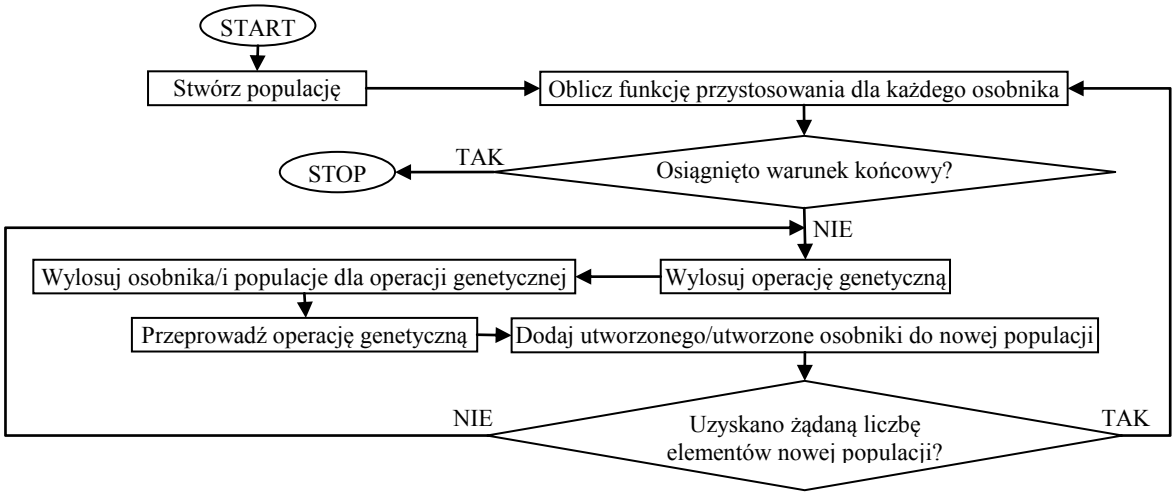
3. Zastosowanie algorytmu STLD w procesie ewolucji PG

Programowanie genetyczne należy do klasy algorytmów heurystycznych naśladujących procesy naturalne: dziedziczenie genetyczne, prawa doboru naturalnego oraz darwinowską walkę o przetrwanie. Poczynając od danego zbioru rozwiązań (populacji osobników), są tworzone zbiory rozwiązań coraz lepszych. Nowe rozwiązania są generowane z zastosowaniem operatorów genetycznych, symulujących naturalne procesy reprodukcji [9].

Rysunek 5 przedstawia ogólny schemat ewolucji PG. Na początku jest tworzona (najczęściej losowo) początkowa populacja osobników. Pojedynczy osobnik jest reprezentowany za pomocą struktury drzewiastej. Następnie każdy z osobników jest oceniany za pomocą funkcji przystosowania *ff*. Po dokonaniu oceny jest sprawdzane wystąpienie warunków końcowych (np. otrzymanie osobnika o wartości *ff* większej lub równej zadanemu progowi lub osiągnięcie maksymalnej generacji). Jeżeli warunek końcowy nie jest spełniony, jest tworzona nowa generacja osobników za pomocą operatorów genetycznych: selekcji, krzyżowania i mutacji. Operator selekcji wybiera osobniki, które są przenoszone do nowej populacji, operator mutacji wybiera osobniki, których poddrzewa są usuwane lub zastępowane nowowygenerowanymi, operator krzyżowania wybiera pary osobników, które wymieniają się [losowo określonymi] poddrzewami. Nowo utworzona generacja jest poddawana ocenie za pomocą *ff*.

Algorytm STLD może wspomóc zwiększenie zróżnicowania populacji na każdym etapie programowania genetycznego (np. przy generacji populacji początkowej lub przy wyborze

osobników w operatorach genetycznych). Może być także zastosowany do oceny zróżnicowania kilku osobników lub całej populacji przy określaniu warunków końcowych algorytmu lub przy wyliczaniu *ff* (np. w celu przyspieszenia PG nie liczy się *ff* dla podobnych osobników). W ramach przeprowadzonych testów STLD został wykorzystany w operatorze selekcji turniejowej. Selekcja turniejowa polega na losowym wyborze n osobników populacji, z których do nowej populacji przechodzi osobnik o najwyższej funkcji przystosowania. Zaletą selekcji turniejowej jest większe zróżnicowanie osobników w nowej populacji, wadą: brak pewności, że osobniki najlepsze przejdą do nowej populacji.



Rys. 5. Ogólny schemat algorytmu genetycznego
Fig. 5. Genetic Programming algorithm

Zaproponowany w artykule operator genetyczny SelectionCompareGPOp przy inicjalizacji algorytmu PG tworzy tablicę wielkości równej liczbie osobników w testowanej populacji – tak aby każdemu osobnikowi przypadało jedno miejsce w tablicy. Jeżeli któreś z drzew populacji zostanie wylosowane, wartość odpowiadającej mu komórki w tablicy jest inkrementowana. Każde nowo wylosowane drzewo porównuje się za pomocą algorytmu STLD z innymi drzewami, które już zostały wcześniej wylosowane. Porównywane drzewa są uznawane za podobne, jeżeli ich podobieństwo – wartość *stld* – przekracza ustaloną wartość (domyślnie: 85%). Dodatkowo wprowadzono limit osobników podobnych, występujący w dwóch wariantach:

- Dla całej populacji. Jeżeli odnajdzie się osobnika podobnego do któregoś z już zaznaczonych, sprawdza się, czy maksymalna liczba podobnych osobników w populacji została osiągnięta. Jeżeli nie, jest inkrementowana wartość komórki tablicy wynikowej odpowiadającej wylosowanemu osobnikowi, a liczba podobnych osobników w populacji jest zwiększana. Jeżeli tak – przeprowadza się losowanie osobnika jeszcze raz. Tym sposobem osobniki podobne zostają częściowo wyeliminowane.

- Dla każdego osobnika. Jeżeli odnajdzie się osobnika podobnego do któregoś z już zaznaczonych, sprawdza się, czy ten wcześniej zaznaczony osiągnął już limit zaznaczeń. Jeżeli nie – jest inkrementowana wartość komórki tablicy wynikowej, odpowiadającej wcześniej zaznaczonemu osobnikowi, jeżeli tak – losuje się kolejnego osobnika. Ze względu na możliwość zaznaczenia osobnika podobnego, lecz nie identycznego do aktualnie porównywanego, zaleca się zwiększyć minimalne podobieństwo, które pozwala uznać osobniki za podobne (w ramach testów zwiększono podobieństwo do 95%). Sprawdzenie limitu zaznaczeń odbywa się również dla osobników, które nie zostały uznane za podobne. Jeżeli liczba zaznaczeń danego osobnika jest równa maksymalnej, jest losowany kolejny osobnik.

4. Przeprowadzone eksperymenty i analiza wyników

W celu przebadania efektywności proponowanego algorytmu przeprowadzono eksperymenty porównujące jakość miary podobieństwa wyliczanej za pomocą algorytmu STLD i na podstawie wartości funkcji dopasowania oraz testujące wpływ STLD na proces PG. Testy przeprowadzono przy użyciu środowiska fACT (framework for Animation of virtual Characters in Three dimensions) – aplikacji służącej do animacji wirtualnych postaci w przestrzeni trójwymiarowej [10]. Obecnie udostępnia ona trzy metody generowania zachowania postaci: za pomocą konsoli (użytkownik wprowadza kolejne polecenia dla postaci), za pomocą predefiniowanego pliku XML opisującego drzewo strategii zachowania postaci oraz za pomocą programowania genetycznego (z wykorzystaniem silnika Open BEAGLE [11]). Na potrzeby testów do aplikacji dołączono dwa genetyczne operatory selekcji: SelectionCompareGPOp, wykorzystujący algorytm STLD oraz SelectionCompareFitnessGPOp, porównujący drzewa strategii na podstawie funkcji dopasowania. We wszystkich eksperymentach zdecydowano się na prostą animację zakładającą podejście postaci z losowego miejsca sceny trójwymiarowej do drzwi przy wykorzystaniu akcji prostych: MOVE (przesunięcie postaci do przodu), ROTATE_L (obrót w lewo o 15^o), ROTATE_R (obrót w prawo o 15^o) oraz operacje warunkowe: ORIENTATION (sprawdzającą orientację postaci w stosunku do celu – drzwi), POSITION (sprawdzającą odległość postaci od celu – drzwi), IF_OPCTRL_*AKCJA_PROSTA* (sprawdzającą, czy wykonanie danej akcji (przesunięcia, obrotu w lewo lub prawo) spowoduje wystąpienie kolizji). Wykorzystana scena to prostokątna sala szkolna (otoczona ścianą z pojedynczymi drzwiami) z biurkiem i ławkami, przy których ustawiono krzesła (rys. 6).

Funkcja oceny strategii *ff* jest wyliczana ze wzoru:

$$ff = \frac{1}{\sqrt{d_K + p_K + p_S + p_O} + 1} \tag{1}$$

gdzie d_K określa odległość postaci od pozycji przed drzwiami, p_K – punkty karne za pojawiające się kolizje, p_S – punkty przyznawane za wykonane operacje (w celu ograniczenia ruchów nadmiarowych), p_O – punkty karne przyznawane za niewłaściwą orientację postaci w stosunku do drzwi. Wartość funkcji *ff* należy do przedziału (0,1). Im wyższa wartość *ff*, tym lepsze zachowanie postaci.

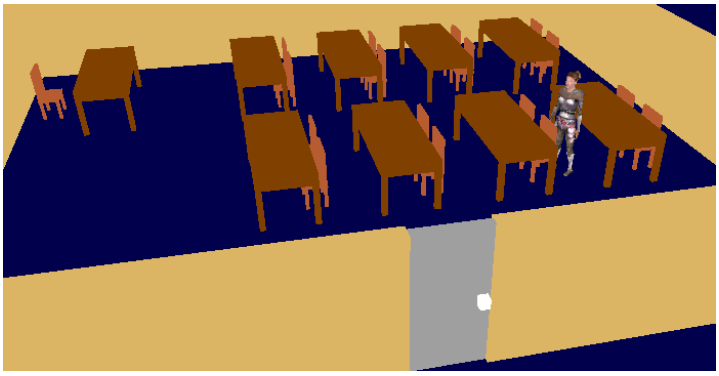
Tabela 1

Parametry algorytmu programowania genetycznego

Parametr	Wartość
wielkość populacji	20
maksymalna liczba generacji	50
funkcja przystosowania	wzór 1
minimalna głębokość drzewa	7
maksymalna głębokość drzewa	50
liczba testowanych ustawień	3
prawdopodobieństwo mutacji	0.2
prawdopodobieństwo krzyżowania	0.9
prawdopodobieństwo selekcji	0.3
liczba osobników w selekcji turniejowej	3

Dla każdego eksperymentu wykonano 20 ewolucji PG, zgodnie z parametrami podanymi w tabeli 1. Niewielka liczba osobników w populacji (20) pozwala na dokładną analizę różnorodności populacji, nadal zapewniając znalezienie strategii, która doprowadzi postać do zadanego celu. Tabela 2 opisuje 11 wykonanych eksperymentów. Przeprowadzono testy procesu generowania drzew strategii bez użycia operatora porównywania, z użyciem operatora selekcji SelectionCompareFitnessGPOp (SCFop) oraz SelectionCompareGPOp (SCop). Dla operatora SelectionCompareGPOp przetestowano dwa typy limitu podobnych osobników: dla całej populacji (cp) i dla każdego osobnika (ko). Sprawdzono również zachowanie algorytmu programowania genetycznego dla różnych wartości limitów cp i ko (tabela 2).

Na początku badań zaproponowana miara podobieństwa drzew została skonfrontowana z dostępną miarą porównującą wartości funkcji *ff*. Z drzew, będących wynikiem eksperymentu BO, to znaczy klasycznego procesu programowania genetycznego (tabela 2), wyłoniono pary drzew należące do tej samej populacji i o takiej samej wartości *ff*. Następnie wyliczono dla nich wartość podobieństwa *stld*. Otrzymane wyniki *stld* (m.in. 100%, 93%, 80%, 35%, 28%, 7%, 5%, 0%) znajdowały się w całym dostępnym przedziale – od 0% do 100%. Oznacza to, że wyniki obu miar nie zgadzają się ze sobą.



Rys. 6. Trójwymiarowa klasa w środowisku fACT
Fig. 6. Three-dimensional classroom from fACT framework

W celu sprawdzenia, jak obliczone podobieństwa pokrywają się z faktycznym ruchem postaci, wykonano dodatkowy eksperyment: za pomocą środowiska fACT prześledzono ruch postaci sterowanej drzewami o podobnym rozmiarze i takiej samej funkcji dopasowania (podobieństwo na podstawie *ff* wynosiło 100%), dla których wartość *stld* wynosiła 0%.

Tabela 2

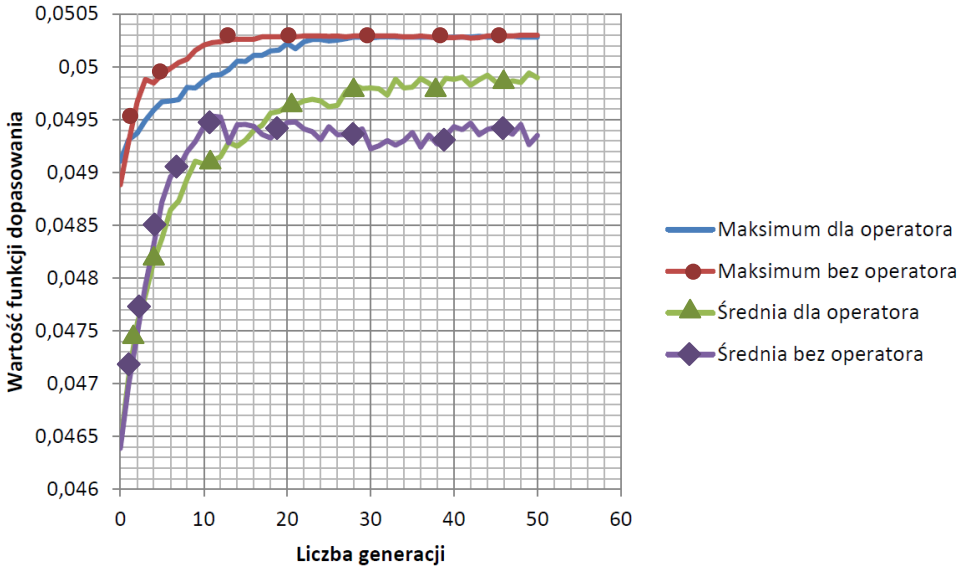
Opis przeprowadzonych eksperymentów

Eksperyment	Operator	Typ limitu	Wartość limitu [% populacji]	Próg zaliczenia osobników jako podobne
OP-P0-85	SCop	cp	0%	85%
OP-P15-85	SCop	cp	15%	85%
OP-P20-85	SCop	cp	20%	85%
OP-P25-85	SCop	cp	25%	85%
OP-P40-85	SCop	cp	40%	85%
OP-O10-95	SCop	ko	10%	95%
OP-O20-95	SCop	ko	20%	95%
OP-O25-95	SCop	ko	25%	95%
OP-O50-95	SCop	ko	50%	95%
OPF-P20-85	SCFop	cp	20%	85%
BO	brak	brak	brak	brak

W przypadku pierwszego drzewa postać wykonała następujące ruchy: podejście do biurka, obrót w stronę biurka, podejście do drzwi, podejście do ławki na środku sceny, powrót do drzwi, podejście do przeciwległej ściany, obrót w prawo, do stojącego najbliżej stolika, podejście do tego stolika, powrót do drzwi. Dla drugiego drzewa postać wykonała następujące ruchy: podejście do stołu, obrót w lewo dookoła własnej osi (360 stopni), podejście do drzwi, podejście do ławki na środku sceny, kilka obrotów i ruchów w przód i w tył obok ławki, powrót do drzwi, podejście do ściany, obrót w lewo dookoła własnej osi (360 stopni), powrót do drzwi.

Zachowanie postaci wyraźnie się różniło dla obu drzew, zatem można stwierdzić, iż faktyczne podobieństwo ruchów postaci było lepiej reprezentowane przez wartość *stld* niż *ff*.

Jak widać na przedstawionym przykładzie, wartości funkcji dopasowania danych drzew mogą diametralnie różnić się od ich faktycznego podobieństwa. Jeżeli osobniki miałyby być usuwane z populacji na podstawie podobieństwa ich funkcji dopasowania, wiele ścieżek rozwoju populacji mogłoby zostać utraconych.



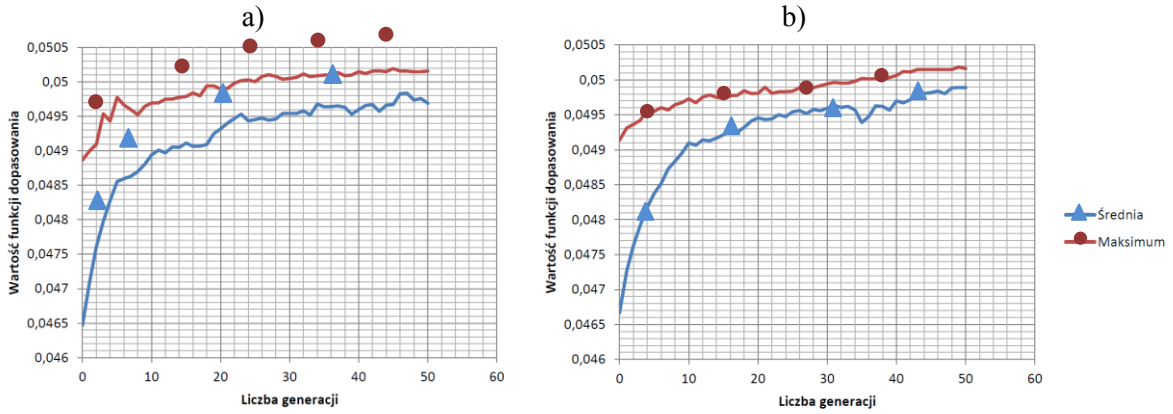
Rys. 7. Wartości funkcji dopasowania (średnia i maksymalna) dla eksperymentów OP-P20-85 (z operatorem) i BO (bez operatora)
Fig. 7. Fitness function (average and maximum) for experiments OP-P20-85 and BO

Drugi etap badań polegał na przetestowaniu wpływu zastosowania operatora selekcji, biorącego pod uwagę podobieństwo osobników, na ewolucję programowania genetycznego. Proces ewolucji był oceniany na podstawie średniej i maksymalnej wartości funkcji przystosowania dla danej generacji wyliczanej jako średnia z 20 wykonanych ewolucji.

Pierwszych pięć eksperymentów (tabela 2) sprawdzało działanie operatora SelectionCompareGPOp z limitem podobnych osobników ustawianym dla całej populacji i równym odpowiednio 0%, 15%, 20%, 25% i 40% liczebności populacji. Najlepsze wyniki zanotowano dla eksperymentu OP-P20-85 (rys. 7), w którym osiągnięto najwyższe średnie i maksymalne wartości funkcji dopasowania. Słabsze wyniki dla eksperymentów OP-P0-85 i OP-P15-85 wynikały ze zbyt dużego nacisku na zróżnicowanie populacji w stosunku do presji funkcji przystosowania. Silne zróżnicowanie wiązało się z dużą liczbą gorszych osobników przechodzących do nowej populacji. Z kolei dla eksperymentów z wyższym limitem podobnych osobników (25% i 40%) nieco gorsze wyniki mogą być spowodowane silniejszą specjalizacją populacji.

Kolejne cztery eksperymenty dotyczyły zastosowania operatora SelectionCompareGPOp z limitem podobnych osobników ustawianym dla każdego osobnika i równym odpowiednio 10%, 20%, 25% i 50% liczebności populacji. Eksperyment OP-010-95 cechował się bardzo dużym zróżnicowaniem populacji i dopuszczaniem do kolejnych generacji każdego rodzaju

osobników – zarówno tych o lepszej, jak i gorszej funkcji dopasowania. Spowodowało to równomierny wzrost dopasowania, jednak z niskim wynikiem końcowym. Zróżnicowanie populacji było w tym wypadku zbyt duże. Dla eksperymentów z limitem wielkości 20% i 25% zaobserwowano polepszenie wyników dla maksymalnej i średniej wartości funkcji przystosowania (rys. 8). Wyniki uległy ponownemu pogorszeniu dla eksperymentu OP-O50--95, co może być spowodowane spadkiem zróżnicowania populacji.



Rys. 8. Wartości funkcji dopasowania (średnia i maksymalna) dla eksperymentów: a) OP-O20-95, b) OP-O25-95

Fig. 8. Fitness function (average and maximum) for experiments: a) OP-O20-95, b) OP-O25-95

Eksperyment, który uzyskał najwyższe wartości funkcji *ff* dla limitu liczonego dla każdego osobnika (ko) – OP-025-95 – uzyskał niższe wartości niż eksperyment OP-P20-85 (tabela 2). Może to sugerować, że limit na całą populację cp jest lepszy od limitu na każdego osobnika ko, ponieważ pozwala na uzyskanie wyższych wartości *ff*. Na obecnym etapie badań jest to jednak zbyt wczesny wniosek, któremu mogą przeczyć dwie poczynione obserwacje. Po pierwsze, dla eksperymentu OP-O20-95 zostało wygenerowane drzewo strategii o najwyższej funkcji *ff*, z wszystkich drzew otrzymanych we wszystkich pozostałych eksperymentach. Po drugie, stały wzrost wartości *ff* przez wszystkie 50 generacji, który możemy zaobserwować na wykresie z rysunku 8b, może oznaczać, że zwiększenie maksymalnej liczby generacji pozwoli na osiągnięcie lepszych wyników. Dla eksperymentu OP-P20-85 w ostatnich 10 generacjach nie jesteśmy w stanie dostrzec istotnych zmian w wynikach (rys. 7).

W ramach eksperymentu OPF-P20-85 sprawdzono zachowanie algorytmu PG dla operatora selekcji, porównującego osobniki na podstawie ich funkcji dopasowania. Otrzymane wyniki były niższe od osiągniętych przez wszystkie pozostałe 10 eksperymentów, co potwierdza wyniki pierwszego badania, iż faktyczne podobieństwo porównywanych osobników może się diametralnie różnić od podobieństwa określanego przez funkcję *ff*.

Wykres na rysunku 7 przedstawia porównanie wartości najlepszego eksperymentu z eksperymentów dokonujących selekcji na podstawie podobieństwa drzew strategii z klasycznym procesem programowania genetycznego (BO). Jakość całej populacji opisana wartościami

średnimi jest większa dla eksperymentu OP-P20-85. Uzyskane wartości maksymalne są podobne, ale nie należy zapominać, że najwyższa funkcja przystosowania została uzyskana za pomocą eksperymentu z operatorem selekcji SelectionCompareGPOp. Pozwala to podejrzewać, że zwiększenie zróżnicowania populacji powoduje przetestowanie większej liczby ścieżek i znalezienie lepiej przystosowanych osobników. Zastosowanie operatora sprawia także wolniejsze polepszanie osobników w populacji, ale zapobiegając specjalizacji i nie utyka na określonym poziomie. Brak lepszych wartości maksymalnych funkcji dopasowania dla populacji z eksperymentu OP-P20-85 w stosunku do BO, może wynikać z prostoty generowanego zachowania. W tym wypadku specjalizacja populacji nie przeszkadzała w generacji dobrych strategii.

Tabela 3

Grupy osobników podobnych dla przykładowych populacji trzech eksperymentów

Nr osobnika	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
BO	a	b	b	b	b	b	b	c	d	e	b	f	b	b	b	b	g	h	i	b
OP-P15-85	a	b	c	d	e	f	g	c	h	i	d	j	k	l	c	m	h	n	o	m
OP-O20-95	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	r	r

Ostatni etap badań zakładał dokładniejsze sprawdzenie wpływu operatorów na zróżnicowanie populacji. Sprawdzono podobieństwa drzew z ostatnich generacji dla trzech eksperymentów: BO, OP-P15-85 i OP-O20-95. W tabeli 3 przedstawiono przykładowe trzy populacje: osobniki bliźniacze (o podobieństwie wyższym niż 85%) dla danej populacji przyporządkowano do tej samej grupy (np. a, b lub c).

Dla eksperymentu BO za bliźniacze zostało uznanych 12 osobników z 20. W eksperymencie OP-P15-85 pojawiły się cztery grupy podobnych osobników z 2 – 3 osobnikami w grupie. Dla eksperymentu OP-O20-95 odnaleziono jedynie 3 bliźniacze osobniki. Dla obu przypadków wykonania algorytmu z użyciem operatora sprawdzającego podobieństwo osobników znaleziono o wiele mniej osobników bliźniaczych, niż dla wykonania algorytmu bez użycia operatora. Taki wynik badań dowodzi, iż użycie algorytmu STLD pozwala na stworzenie populacji o zdecydowanie wyższym zróżnicowaniu.

5. Podsumowanie

Przedstawiony w artykule algorytm STLD pozwala na określanie stopnia podobieństwa drzew strategii opisujących zachowanie postaci wirtualnych. Przeprowadzone eksperymenty wykazały, że zastosowany do automatycznego generowania drzew strategii w procesie programowania genetycznego zapobiega powstawaniu specjalizacji populacji, pozwalając na ich silniejsze zróżnicowanie. Użyty w genetycznym operatorze selekcji PG spowodował polep-

szenie średniej jakości generowanych populacji. Fakt, że maksymalne wartości funkcji dopasowania były podobne lub słabsze dla PG z algorytmem STLD, może wynikać z prostoty generowanego zachowania. Silniejsze zróżnicowanie populacji nie było tutaj konieczne do osiągnięcia celu. Dalsze prace powinny zawierać eksperymenty dla bardziej złożonych zachowań postaci.

BIBLIOGRAFIA

1. Zhang K., Shasha D.: Simple fast algorithms for the editing distance between trees and related problems. SIAM Journal on Computing, Vol. 18, No. 6, Philadelphia 1989.
2. Touzet H.: Comparing similar ordered trees in linear-time. Journal of Discrete Algorithms, Elsevier Science Publishers, Vol. 5, No. 4, Amsterdam 2007.
3. Jiang T., Wang L., Zhang K.: Alignment of trees—an alternative to tree. Theoretical Computer Science, Elsevier, Vol 143, No. 1, 1995.
4. Nierman A., Jagadish H. V.: Evaluating Structural Similarity in XML Documents. ACM-SIGMOD Intl. Workshop on Web and Databases, Madison 2002.
5. Neamtiu I., Foster J. S., Hicks M.: Understanding source code evolution using abstract syntax tree matching. ACM SIGSOFT Software Engineering Notes, Vol 30, No. 4, New York 2005.
6. Lach E., Chaja T.: Składowe oceny animacji wirtualnych postaci ludzkich. Studia Informatica, Wyd. Pol. Śląskiej, Vol. 33, No. 1 (104), Gliwice 2012.
7. Hamming R. W.: Error detecting and error correcting codes. Bell System Technical Journal 29, 1950.
8. Levenshtein V. I.: Binary codes capable of correcting deletions, insertions, and reversals. Soviet Physics Doklady, Vol. 10, No. 8, 1966.
9. Koza J.R.: On the programming of computers by means of natural selection. MIT Press, 1996.
10. Lach E.: fACT – animation framework for generation of virtual characters. International Conference on Information Technology, 2008.
11. Gagné Ch., Parizeau M.: Genericity in Evolutionary Computation Software Tools: Principles and Case Study. International Journal on Artificial Intelligence Tools, Vol. 15, No. 2, 2006.

Wpłynęło do Redakcji 30 listopada 2012 r.

Abstract

The paper proposes new algorithm STLD for measuring a similarity of strategy trees, describing behaviors of the three-dimensional virtual characters. The algorithm uses a Levenshtein's distance [8] to compare strategy tree's sequences of predefined character's actions and decisions. The obtain measure is used in automatic generation of strategy trees by means of genetic programming [9]. The STLD is applied in genetic selection operators to restrict the choice of similar behaviors for the next generations. Two kinds of limits are proposed: for whole population and for each strategy tree. The aim of the operators is to create more diversified populations with many different paths to explore.

The paper presents, also, experiments conducting automatic generation of virtual characters behaviors using genetic programming with new selection operators GP-STLD. Table 2 describes 11 conducted experiments with strategy trees comparison and without. Acquired results show that using the STLD algorithm in selection operators of genetic programming produces better and more diversified populations. The obtained maximum values of fitness function are similar for the best populations generated with GP-STLD and with GP. This result may be the effect of the simplicity of the learned behavior (going to the door from the different spaces of the classroom). Future work should include conducting experiments with more complex learned behaviors.

Adresy

Ewa LACH: Politechnika Śląska, Instytut Informatyki, ul. Akademicka 16, 44-100 Gliwice, Polska, ewa.lach@polsl.pl .
Marianna HETMAN: Politechnika Śląska, Instytut Informatyki, ul. Akademicka 16, 44-100 Gliwice, Polska, marianna.hetman@gmail.com.