Politechnika Śląska
Wydział Automatyki, Elektroniki i Informatyki
Instytut Informatyki

Daniel Kostrzewa

PRZESZUKIWANIE PRZESTRZENI ROZWIĄZAŃ W OPTYMALIZACJI PLANÓW ZAPYTAŃ DO BAZ DANYCH Z WYKORZYSTANIEM HEURYSTYCZNEGO ALGORYTMU IWO

Rozprawa doktorska napisana
pod kierunkiem
Prof. dr hab. inż. Stanisława Kozielskiego

Gliwice, 2015

Spis treści

1. Wprowadzenie 3
2. Problemy optymalizacyjne 4
2.1. Problem określenia kolejności złączeń w realizacji zapytań bazodanowych 4
2.2. Problem znajdowania minimum funkcji wielowymiarowych 6
2.3. Problem komiwojażera 6
3. Zmodyfikowany algorytm IWO (expanded IWO, exIWO) 6
3.1. Idea zmodyfikowanego algorytmu IWO 7
3.2. Modyfikacje IWO opracowane na potrzeby problemów optymalizacyjnych 11
3.2.1. Znajdowanie minimum funkcji wielowymiarowych 11
3.2.2. Problem komiwojażera 11
3.2.3. Określanie kolejności złączeń w realizacji zapytań w scentralizowanych bazach danych 12
4. Wyniki badań i ich analiza 14
4.1. Problem znajdowania minimum funkcji wielowymiarowej 15
4.1.1. Dobór funkcji testowych i metodyka przeprowadzonych badań 15
4.1.2. Badania z użyciem wybranych funkcji testowych 16
4.1.3. Porównanie exIWO z wersją oryginalną 17
4.1.4. Porównanie exIWO z algorytmem APSO 18
4.2. Problem komiwojażera 20
4.2.1. Dobór danych testowych i metodyka przeprowadzonych badań 20
4.2.2. Porównanie exIWO z algorytmem Meta-RaPS i innymi algorytmami heurystycznymi 21
4.2.3. Porównanie exIWO ze strategią sieci samoorganizujących 22
4.2.4. Zestaw Mona Lisa 23
4.3. Problem określania kolejności złączeń w realizacji zapytań 24
4.3.1. Dobór danych testowych i metodyka przeprowadzonych badań 24
4.3.2. Porównanie czasów wykonania zapytań w systemie SQL Server 2008 26
5. Wnioski końcowe 29
6. Bibliografia 31

1. Wprowadzenie

Intensywny rozwój systemów informatycznych wprowadził znaczące zmiany w zakresie gromadzenia i przetwarzania danych. Systemy oparte na bazach danych obecne są w niemal każdej gałęzi przemysłu i usług. Z tego względu kluczowym zagadnieniem stało się efektywne wyszukiwanie danych. Ma ono zasadnicze znaczenie dla szybkości dostępu do informacji w każdym systemie informatycznym.

Ta sytuacja spowodowała, że głównym przedmiotem rozważań niniejszej rozprawy stało się skuteczne wyszukiwanie danych w relacyjnych bazach danych.

Problem wydajnej realizacji zapytań bazodanowych jest niezwykle trudnym zagadnieniem badawczym. Złożoność tego zadania powoduje, że prowadzone prace koncentrują się wokół wielu problemów składowych, do których między innymi należy *optymalizacja zapytania*.

Zapytania poddawane optymalizacji formułowane są przez użytkownika w języku SQL (ang. *Structured Query Language*). Wykonanie zapytania SQL wymaga od systemu zarządzania bazą danych (SZBD) określenia zestawu operacji oraz kolejności ich realizacji. Celem optymalizacji jest minimalizacja czasu oczekiwania użytkownika na dane będące wynikiem końcowym [9, 49]. Większość zapytań może być przetworzonych na wiele sposobów. Kluczowym zagadnieniem optymalizacyjnym jest wyznaczenie jak najbardziej efektywnej strategii (*planu*) wykonania zapytania. Jako plan „optymalny” w tym przypadku rozumie się najlepsze z aktualnie osiągalnych, nie zaś najlepsze ze wszystkich możliwych rozwiązań.

Proces optymalizacji może się okazać czynnością czasochłonną. Z praktycznego punktu widzenia nie do zaakceptowania jest sytuacja, w której czas poszukiwania rozwiązania optymalnego znacznie przekracza czas realizacji zapytania według wyznaczonego planu. Istotniejsze jest zatem odnalezienie planu o zadowalającej jakości (i uniknięcie nieefektywnego, długotrwałego planu realizacji) niż dążenie do rozwiązania optymalnego globalnie.

Do operacji składowych realizacji zapytania zalicza się: skanowanie (relacji lub indeksu), selekcję, projekcję, złączenie, grupowanie, agregację. Wśród tych operacji niezwykle ważną rolę pełnią *złączenia*, które służą do łączenia zbiorów danych pochodzących z różnych relacji. W większości zapytań bazodanowych angażujących wiele relacji proces łączenia danych jest najbardziej złożonym i czasochłonnym elementem realizacji zapytania. W związku z tym minimalizacja czasu trwania tego procesu ma kluczowe znaczenie dla całego zadania optymalizacji zapytania kierowanego do bazy danych.

Opisana problematyka od wielu lat jest przedmiotem intensywnych prac badawczych, ale tematyka tych badań jest ciągle aktualna. Głównymi tematami prowadzonych badań są: dobór modelu kosztów oraz wyznaczenie kolejności realizacji operacji złączenia.

Mimo, że powstało bardzo wiele technik rozwiązywania wspomnianego problemu, to nadal poszukuje się nowych, skutecznych strategii.

W 2006 roku zaprezentowano algorytm heurystyczny o nazwie *Invasive Weed Optimization* (IWO). Metoda ta często jest wykorzystywana do rozwiązywania różnych problemów optymalizacyjnych [34, 44, 10, 11]. Przekonujące rezultaty otrzymywane przy użyciu wspomnianej techniki stanowiły dla autora rozprawy sugestię, aby zmodyfikować algorytm IWO, a następnie zba-

dać możliwość zastosowania go do problemu wyznaczenia optymalnej kolejności realizacji operacji złączenia.

W związku z tym **celem rozprawy stało się opracowanie zmodyfikowanej metody *Invasive Weed Optimization* dla potrzeb zadań optymalizacji, w szczególności wyznaczania kolejności złączeń w zadaniach optymalizacji zapytań do baz danych**.

Osiągnięcie tak postawionego celu wymagało zarówno dokładnego zapoznania się ze znanymi rozwiązaniami, jak i przeprowadzenia szczegółowej analizy algorytmu IWO. Czynniki te dały podstawę do opracowania metody autorskiej. Tezę rozprawy sformułowano w następującej postaci:

Zaproponowany w pracy zmodyfikowany algorytm *Invasive Weed Optimization* może zostać skutecznie wykorzystany w zadaniach optymalizacji, a w szczególności do wyznaczania kolejności złączeń w procesie realizacji zapytań w bazach danych.

2. Problemy optymalizacyjne

Jedno z praw algebry relacji, prawo łączności operacji złączenia ($R \bowtie (S \bowtie T) = (R \bowtie S) \bowtie T$, gdzie R, S, T oznaczają relacje, a symbol $\bowtie$ oznacza operację złączenia), pozwala na rozważenie wielu wariantów określania kolejności złączeń relacji biorących udział w zapytaniu. Poszukiwanie optymalnej kolejności złączeń jest kluczowym elementem optymalizacji zapytania i zalicza się je do klasy problemów NP-trudnych. Porządkowanie ciągów innych operacji dwuargumentowych (suma, przecięcia) ma mniejsze znaczenie, ponieważ rzadziej występują one w grupach, a ich wykonanie zajmuje zwykle mniej czasu niż realizacja złączenia [9].

Określanie kolejności złączeń ma szczególne znaczenie dla niniejszej rozprawy, dlatego zostanie ono szczegółowo omówione w podrozdziale 2.1.

2.1. Problem określenia kolejności złączeń w realizacji zapytań bazodanowych

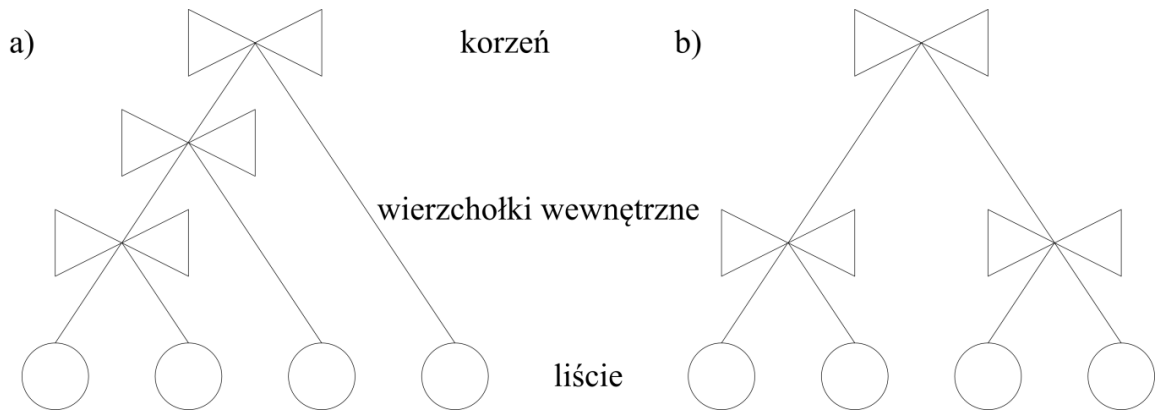
Zakłada się, że przed przystąpieniem do wyznaczania kolejności złączeń dane są wstępnie przetworzone. Oznacza to, że na tabelach bazy danych, biorących udział w zapytaniu, przeprowadzono, zgodnie z własnościami wyrażeń algebry relacji, operacje projekcji i selekcji. Dla określenia takich wstępnie przetworzonych danych, w niniejszej pracy stosuje się termin *zbiór rekordów*.

Proces łączenia danych w zbiór rekordów stanowiący wynik końcowy zapytania adresowanego do scentralizowanej bazy danych polega na wykonaniu wielu złączeń. Celem każdego z nich jest złączenie dwóch zbiorów rekordów, z których każdy może być zbiorem pobranym bezpośrednio z bazy scentralizowanej (zbiór pierwotny) lub może stanowić wynik wcześniej przeprowadzonej operacji złączenia (zbiór pośredni, wynik pośredni).

W prowadzonych pracach skupiono uwagę na operacji złączenia. Celem niniejszego zadania optymalizacji jest wyznaczenie takiego porządku realizacji złączeń, aby wynik końcowy zapytania został przedstawiony użytkownikowi w jak najkrótszym czasie.

Przestrzenią poszukiwań dla rozpatrywanego zagadnienia jest zbiór planów realizacji zadanego ciągu złączeń. Pojedynczy plan realizacji może być reprezentowany przez drzewo realizacji złączeń (ang. *join processing tree*). W drzewie realizacji złączeń wyróżnia się następujące elementy (rys. 1): liście przedstawiające zbiory pierwotne; wierzchołki wewnętrzne – operacje złączenia; wierzchołek nazywany korzeniem (element, któremu przypisano złączenie realizowane jako ostatnie); drzewo o n liściach posiada n-1 wierzchołków wewnętrznych (symbolizujących n-1 złączeń); krawędzie łączące każdy wierzchołek wewnętrzny z dwoma niżej położonymi elementami drzewa (wierzchołkami wewnętrznymi lub liśćmi); każda taka trójka elementów wyraża pojedynczą operację złączenia i jej argumenty. Krawędzie ilustrują przepływ danych od poziomu liści aż do korzenia.

Układ krawędzi, wierzchołków i liści decyduje o kształcie drzewa. Wyróżnia się dwa rodzaje drzew realizacji złączeń: o sekwencyjnym układem wierzchołków (rys. 1a) oraz o równoległym układzie wierzchołków (rys. 1b).



Rys. 1. Kształty drzew realizacji złączeń o układzie wierzchołków: a) sekwencyjnym, b) równoległym.

Jedną z form opisu zapytania stanowi graf zapytania (ang. *query graph*). Graf $G = (\{R_1,\ldots,R_n\}, E)$, który reprezentuje wyłącznie ciąg operacji złączenia, nazywany jest grafem złączeń (ang. *join graph*). Stanowi on strukturę złożoną z wierzchołków połączonych nieskierowanymi krawędziami. Wierzchołki grafu są odpowiednikami zbiorów rekordów $R_1,\ldots,R_n$. Każda z krawędzi, reprezentowanych przez zbiór E, który jest zbiorem par (R_i, R_j), $1 \le i, j \le n$, $i \ne j$, symbolizuje fakt istnienia wyrażenia łączącego między jednostkami tworzącymi pojedynczą parę [45].

Literatura związana z tematem wyznaczania kolejności złączeń w realizacji zapytań bazodanowych jest bardzo bogata. Zadanie to jest rozwiązywane m.in. przy użyciu: Minimum Selectivity, Top-Down, KBZ, Relational Difference Calculus, AB [46], programowania dynamicznego [37], przeszukiwania lokalnego [47], metod probabilistycznych [50], algorytmu symulowanego wyżarzania [46], algorytmu iteracyjnego poprawiania [46], algorytmu z listą tabu [40], a także algorytmów genetycznych [8] i metod hybrydowych [13].

Opisane zadanie jest zagadnieniem bardzo trudnym do rozwiązania. Wydaje się więc zasadne w pierwszej kolejności obszerne przetestowanie rozważanego w niniejszej pracy algorytmu heurystycznego IWO, rozwiązując klasyczne zadania optymalizacji, a dopiero potem przystąpienie do eksperymentów dotyczących problemu wyznaczania kolejności złączeń.

W niniejszej pracy skupiono się na problemach znajdowania minimum funkcji matematycznej oraz komiwojażera. Wybór padł na te zadania ze względu na ich ogólność oraz bogatą literaturę.

2.2. Problem znajdowania minimum funkcji wielowymiarowych

Problem znajdowania minimum funkcji wielowymiarowych (funkcji wielu zmiennych) jest podstawowym zadaniem optymalizacji o charakterze ciągłym. Sprowadza się do wyznaczenia wartości argumentów funkcji, dla których posiada ona ekstremum globalne, w tym przypadku minimum. Należy zauważyć, że problem ten ma nieskończenie dużą liczbę potencjalnych rozwiązań. W związku z tym odnalezienie wartości optymalnych bez uwzględnienia dodatkowych założeń jest praktycznie niemożliwe.

2.3. Problem komiwojażera

Problem komiwojażera jest jednym z najbardziej znanych i rozpowszechnionych zadań optymalizacyjnych o charakterze dyskretnym [2]. Ma on bardzo wiele odmian o praktycznym zastosowaniu.

W najprostszej wersji tego zadania dana jest pewna liczba miejsc, które mają być odwiedzone przez wędrującego od miasta do miasta sprzedawcę (komiwojażera) dokładnie jeden raz, oraz koszt podróży między poszczególnymi lokalizacjami. Koszt ten wyrażany jest najczęściej za pomocą odległości, czasu podróży lub jego ceny. Celem zadania optymalizacji jest wyszukanie takiej trasy, prowadzącej przez wszystkie miasta, która ma najniższy koszt przejścia. Zwykle nakłada się dodatkowe założenie określające punkt startowy, będący zarazem miejscem docelowym komiwojażera.

Rozróżnia się dwie podstawowe wersje zagadnienia: symetryczną i asymetryczną. Symetryczny problem komiwojażera polega na tym, że koszt podróży między dwoma miastami w obydwu kierunkach jest jednakowy. Natomiast w odmianie asymetrycznej koszt ten może być różny. Symetryczna wersja jest zatem szczególnym przypadkiem asymetrycznej.

3. Zmodyfikowany algorytm IWO (expanded IWO, exIWO)

W rozdziale tym przedstawiono algorytm IWO, autorskie modyfikacje tej strategii oraz jego przystosowanie do poszczególnych problemów optymalizacyjnych, co również stanowi wkład własny autora.

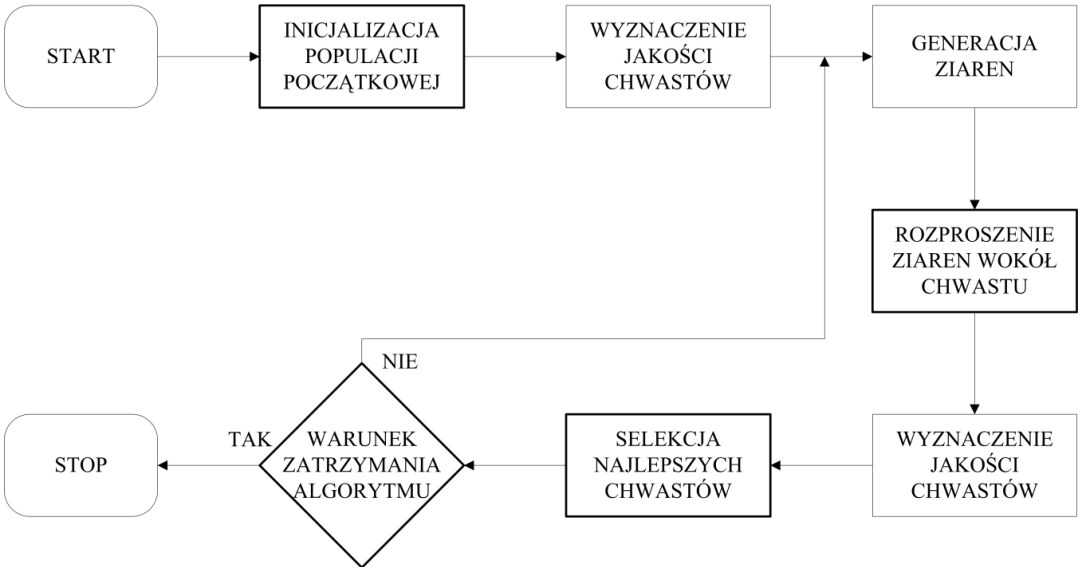
Algorytm IWO (ang. *Invasive Weed Optimization*) jest metodą optymalizacyjną, która została zainspirowana charakterystycznym dla chwastów gwałtownym rozprzestrzenianiem się oraz szybkim przystosowywaniem się do zmiennych warunków otoczenia. Algorytm ten został przedstawiony w 2006 roku przez naukowców Uniwersytetu w Teheranie, A. R. Mehrabiana i C. Lucasa [34]. IWO należy do rodziny algorytmów ewolucyjnych. Wprowadzono w nim nazewnictwo luźno odnoszące się do słownictwa związanego z chwastami:

- iteracja – pokolenie / populacja,

- rozwiązanie – chwast,
- rozwiązanie pochodne – ziarno,
- transformacja – rozproszenie.

3.1. Idea zmodyfikowanego algorytmu IWO

Na rys. 2 przedstawiono schemat blokowy algorytmu IWO z zaznaczeniem tych części, które zostały zmodyfikowane przez autora.



Rys. 2. Schemat blokowy algorytmu IWO z zaznaczeniem zmodyfikowanych jego części.

Inicjalizacja populacji początkowej

W oryginalnej wersji algorytmu IWO pierwszy krok polega na swobodnym rozproszeniu ziaren chwastów po całej przestrzeni poszukiwań. Odpowiada to losowemu wyborowi określonej liczby propozycji rozwiązań rozpatrywanego problemu. Liczba chwastów biorących udział w populacji jest jednym z parametrów algorytmu.

Podczas wstępnych badań zauważono, iż nie warto losowo wybierać osobników do populacji początkowej. Wstępna selekcja rozwiązań pozwala na osiągnięcie znacznie lepszych wyników lub rezultatów o podobnej jakości w zdecydowanie krótszym czasie.

Przeprowadzenie wstępnego doboru osobników jest ściśle zależne od optymalizowanego problemu. Do tego celu może być wykorzystany np. algorytm zachłanny.

Wyznaczenie jakości chwastów

Realizacja tego etapu jest równoznaczna z oceną rozwiązania reprezentowanego przez analizowany chwast. Sprowadza się to do obliczenia wartości pewnej funkcji związanej z kryterium optymalizacji, zwanej funkcją przystosowania. Jest ona budowana na podstawie modelu obliczeniowego, ściśle zależnego od rozwiązywanego zadania optymalizacyjnego.

Generacja ziaren

Każdy chwast tworzy pewną liczbę ziaren. Liczba ta jest powiązana z przystosowaniem do otoczenia generującego je osobnika. Im lepsze jest dane rozwiązanie, tym większą liczbę ziaren

może wytworzyć. Zależność ta ma charakter liniowy i można ją przedstawić następującym wzorem [34]:

$$S_{chw} = S_{min} + \left\lfloor (f_{chw} - f_{min}) \cdot \frac{S_{max} - S_{min}}{f_{max} - f_{min}} \right\rfloor \tag{1}$$

gdzie: S_{chw} – liczba ziaren wygenerowanych przez analizowany chwast; S_{min}, S_{max} – minimalna i maksymalna liczba ziaren, jaka może być wygenerowana przez chwast; f_{chw} – wartość funkcji przystosowania analizowanego chwastu; f_{min}, f_{max} – minimalna i maksymalna wartość funkcji przystosowania osobników w aktualnym pokoleniu chwastów.

Rozproszenie ziaren wokół chwastu

Etap ten jest centralną częścią IWO. Dlatego autor upatruje największych możliwości w modyfikacji tej części algorytmu.

Opisywana faza składa się z dwóch kroków: wylosowania metody rozpraszania oraz jej przeprowadzenia. Metoda rozpraszania jest losowana dla każdego z ziaren z osobna. Wartości prawdopodobieństwa przypisane poszczególnym metodom: rozwiewaniu ($p_{rozwiew.}$), rozsiewaniu ($p_{rozsiew.}$) i staczaniu ($p_{stacz.}$) stanowią parametry algorytmu oraz sumują się do jedności.

Rozwiewanie jest metodą znaną z oryginalnej wersji algorytmu IWO, która została wzbogacona o możliwość użycia rozkładu *t*-Studenta (którego charakter może zapobiegać utknięciu w lokalnym minimum) w miejsce rozkładu normalnego. Polega ona na wyznaczeniu kierunku i odległości miejsca upadku ziarna *Z* od chwastu macierzystego *M*. Odległość jest opisana rozkładem normalnym lub rozkładem *t*-Studenta (typ rozkładu jest parametrem metody). W przypadku rozkładu normalnego wartość oczekiwana wynosi 0, odchylenie standardowe zmienia się zgodnie ze wzorem [34]:

$$\sigma_{iter} = \left(\frac{iter_{max} - iter}{iter_{max}} \right)^m \left(\sigma_{init} - \sigma_{fin} \right) + \sigma_{fin} \tag{2}$$

gdzie: *iter* – aktualny numer iteracji algorytmu; $iter_{max}$ – łączna liczba iteracji algorytmu; σ_{iter} – odchylenie standardowe dla iteracji o numerze *iter*; σ_{init} – odchylenie standardowe początkowe; σ_{fin} – odchylenie standardowe końcowe; *m* – współczynnik modulacji nieliniowej. Rola współczynnika modulacji nieliniowej sprowadza się do określenia kształtu krzywej, według której zmienia się wartość odchylenia standardowego.

Rozkład *t*-Studenta jest scharakteryzowany przez określoną liczbę stopni swobody *p*. Wartość otrzymywana z generatora liczb pseudolosowych jest przemnażana przez współczynnik skalujący γ_{iter}. Współczynnik skalujący zmienia się wraz z każdą kolejną iteracją algorytmu, a formuła opisująca tę zmianę jest podobna do wzoru (1), określającego zmianę odchylenia standardowego w rozkładzie normalnym:

$$\gamma_{iter} = \left(\frac{iter_{max} - iter}{iter_{max}} \right)^m \left(\gamma_{init} - \gamma_{fin} \right) + \gamma_{fin} \tag{3}$$

gdzie: γ_{init} i γ_{fin} – początkowa i końcowa wartość współczynnika skalującego; *iter* – aktualny numer iteracji algorytmu; $iter_{max}$ – łączna liczba iteracji algorytmu; *m* – współczynnik modulacji nieliniowej.

Gdy algorytm rozwiązuje problem optymalizacyjny o charakterze ciągłym, konstrukcja nowego osobnika (będąca odzwierciedleniem rozproszenia ziarna) polega na wykonaniu na kopii chwastu macierzystego transformacji, oddalającej potomka, w dowolnym losowym kierunku. Prawdopodobieństwo wyboru każdego z kierunków jest takie samo. Dla problemów dyskretnych wykonuje się transformacje, w liczbie równej zdyskretyzowanej wartości wyznaczonej przez wybrany rozkład. Określenie losowego kierunku przesunięcia ziarna sprowadza się do zapewnienia równego prawdopodobieństwa każdej z transformacji generujących poprawne rozwiązanie.

Rozsiewanie to losowe rozrzucanie ziaren w całej przestrzeni poszukiwań. Zakłada się rozkład równomierny we wszystkich wymiarach przestrzeni. Działanie to sprowadza się zatem do utworzenia losowego osobnika. Metoda ta ma na celu zapobieganie utknięciom w minimach lokalnych.

Staczanie opiera się na badaniu sąsiedztwa chwastu macierzystego. Przez osobniki sąsiednie rozumie się takie rozwiązania, które różnią się od rozwiązania pierwotnego dokładnie o jedną transformację. Spośród tak zdefiniowanego zbioru osobników sąsiednich w losowy sposób wybiera się określoną liczbę k chwastów. W kolejnym kroku wyznaczana jest jakość każdego osobnika należącego do tego podzbioru oraz wybiera się najlepsze z nich. Następnie przeprowadza się analizę jego sąsiedztwa, ponownie wybierając rozwiązanie o najlepszej jakości. Procedurę tę wykonuje się m-krotnie. Dla uproszczenia przyjęto, że liczba analizowanych sąsiadów k oraz liczba przejść do najlepszego z nich m, stanowiąca liczbę iteracji staczania, są sobie równe i stanowią parametr metody.

Selekcja najlepszych chwastów

W związku z tworzeniem ziaren, których liczebność może kilkukrotnie przewyższać początkową liczbę chwastów, zachodzi konieczność selekcji osobników. Etap selekcji zawiera trzy metody do wyboru: wybór z całej populacji (znany z oryginalnej wersji algorytmu IWO), wybór z osobników potomnych oraz wybór w ramach jednej rodziny. Wszystkie sposoby mają charakter deterministyczny.

Ogólną definicję selekcji można przedstawić w następujący sposób:

Niech:

$\mathbf{Z}_p$ *oznacza zbiór osobników w pokoleniu p o liczebności* $n_p = |\mathbf{Z}_p|$

$\mathbf{Z}_{p+1}$ *oznacza zbiór osobników w pokoleniu* $p+1$ *o liczebności* $n_{p+1} = |\mathbf{Z}_{p+1}|$

Założenie:

$n_p \geq n_{p+1}$

Definicja:

Selekcja jest funkcją f przekształcającą zbiór $\mathbf{Z}_p$ *w zbiór* $\mathbf{Z}_{p+1}$ *taką, że* $\mathbf{Z}_{p+1}$ *zawiera się w* $\mathbf{Z}_p$:

$$f : \mathbf{Z}_p \rightarrow \mathbf{Z}_{p+1} \Leftrightarrow \mathbf{Z}_p \supseteq \mathbf{Z}_{p+1} \tag{4}$$

Dla uproszczenia opisu prezentowanych strategii selekcji wprowadzono następujące oznaczenia: $\mathbf{M}_p$ – zbiór osobników macierzystych w pokoleniu p oraz $\mathbf{P}_p$ – zbiór osobników potomnych stworzonych na podstawie osobników macierzystych $\mathbf{M}_p$ w pokoleniu p [35].

Selekcja z całej populacji jest techniką znaną z oryginalnej wersji algorytmu IWO i polega na wyborze najlepiej przystosowanych osobników ze zbioru złożonego ze wszystkich chwastów macierzystych i potomnych [35]. Opisaną metodę można zdefiniować następująco:

$$f : \mathbf{Z}_p \rightarrow \mathbf{Z}_{p+1} \Leftrightarrow \max_{n_{p+1}}\left(\mathbf{M}_p \cup \mathbf{P}_p\right) = \mathbf{M}_{p+1} \tag{5}$$

gdzie $\max_{n_{p+1}}\left(\mathbf{M}_p \cup \mathbf{P}_p\right)$ oznacza funkcję wybierającą n_{p+1} najlepiej przystosowanych osobników ze zbioru $\mathbf{M}_p \cup \mathbf{P}_p$.

Selekcja z osobników potomnych sprowadza się do wyboru rozwiązań ze zbioru złożonego tylko z chwastów potomnych:

$$f : \mathbf{Z}_p \rightarrow \mathbf{Z}_{p+1} \Leftrightarrow \max_{n_{p+1}}\left(\mathbf{P}_p\right) = \mathbf{M}_{p+1} \tag{6}$$

Nie ma niebezpieczeństwa redukcji liczności nowego pokolenia, ponieważ łączna liczba ziaren przewyższa liczbę chwastów macierzystych. Metoda ta nie gwarantuje, iż nowe pokolenie nie będzie gorsze od aktualnego. Można jednak założyć, że część chwastów potomnych przystosuje się lepiej niż odpowiadające im osobniki macierzyste. Efektem tego założenia jest fakt, że zastosowanie wyboru z osobników potomnych jest oszczędnością czasu, gdyż nie ma potrzeby przeglądania zbioru wszystkich ($\mathbf{M}_p$ + $\mathbf{P}_p$) rozwiązań, a jedynie jego części. Co więcej, takie podejście zmniejsza szanse na stagnację przy nieoptymalnym rozwiązaniu.

Metoda selekcji w ramach jednej rodziny koncentruje się na wyborze lokalnym osobników [48]. Strategia ta wybiera do kolejnego pokolenia jeden chwast ze zbioru złożonego z jednej rośliny macierzystej i jej bezpośrednich potomków. Opisywana selekcja przeprowadzana jest dla każdej rodziny oddzielnie, zapewniając tym samym stałą liczebność nowej populacji chwastów:

$$f : \mathbf{Z}_p \rightarrow \mathbf{Z}_{p+1} \Leftrightarrow \underset{C \in \mathbf{M}_p}{\forall} \max_{1}\left(C + \mathbf{P}_p^C\right) = \mathbf{M}_{p+1} \tag{7}$$

gdzie: $\left(C + \mathbf{P}_p^C\right)$ jest zbiorem osobników złożonym z rośliny macierzystej C oraz jej bezpośrednich potomków w pokoleniu p ($\mathbf{P}_p^C$), natomiast $\max_{1}\left(C + \mathbf{P}_p^C\right)$ oznacza funkcję wybierającą jednego najlepiej przystosowanego osobnika ze zbioru $C + \mathbf{P}_p^C$.

Warunek zatrzymania algorytmu

Tradycyjnym warunkiem zatrzymania algorytmu jest z góry założona liczba iteracji. Element ten został wzbogacony o dodatkowe kryterium, jakim jest czas optymalizacji. Wprowadzono go w celu lepszej możliwości porównania różnych metod optymalizacyjnych, gdyż w związku z różnym czasem realizacji algorytmów bardziej miarodajnym kryterium zatrzymania algorytmu wydaje się być z góry założony interwał czasowy (zakładając wykonanie wszystkich eksperymentów na tej samej maszynie obliczeniowej).

Czas, jaki upłynął od początku działania algorytmu IWO, jest sprawdzany tylko na koniec iteracji. Pozwala to na dokończenie obliczeń związanych z danym pokoleniem chwastów.

3.2. Modyfikacje IWO opracowane na potrzeby problemów optymalizacyjnych

Do pełnego opisu algorytmu niezbędne jest zdefiniowanie jego części ściśle zależnych od rozwiązywanego problemu optymalizacyjnego. Konieczne jest przedstawienie reprezentacji pojedynczego rozwiązania zadania optymalizacji. Wymagane jest także zdefiniowanie metod inicjalizacji pierwszej populacji oraz transformacji pojedynczego osobnika. Ostatnim ważnym elementem jest zaprojektowanie modelu obliczeniowego oraz opartej na tym modelu funkcji kosztu, służącej do porównywania jakości otrzymywanych rozwiązań.

3.2.1. Znajdowanie minimum funkcji wielowymiarowych

Potencjalnymi rozwiązaniami zagadnienia znajdowania minimum funkcji *n*-wymiarowej (funkcji *n* zmiennych) są punkty w przestrzeni, dla których ta funkcja jest określona. Do jednoznacznego zdefiniowania takiego punktu należy znać jego współrzędne w każdym z wymiarów. Zatem chwast reprezentujący pojedyncze rozwiązanie jest *n*-elementową tablicą zawierającą współrzędne punktu w *n*-wymiarach.

Najprostszą formą wyboru osobników do populacji początkowej jest wybór losowy. Jedynym ograniczeniem staje się zakres możliwych wartości początkowych, w jakich ma się mieścić generowany chwast.

Transformacja osobnika, wykorzystywana w operatorach rozwiewania i staczania (rozsiewanie nie wymaga stosowania transformacji), polega na umiejscowieniu nowego chwastu w pewnej odległości od rośliny macierzystej. Odległość ta obliczana jest zgodnie z wybranym rozkładem (rozkładem normalnym lub *t*-Studenta). Ostatnim elementem jest wyznaczenie kierunku upadku ziarna. Odbywa się to w trzech etapach:

1. Dla każdego wymiaru *i* losuje się liczbę x_i o rozkładzie jednostajnym z zakresu <0; 1).
2. Wyznacza się składową dla każdego z wymiarów według następującego wzoru:

$$y_j = x_j \cdot dist \cdot \sqrt{\frac{1}{\sum_{i=1}^{n} x_i^2}}, \tag{8}$$

   gdzie: y_j – składowa dla wymiaru *j*; x_i (x_j) – liczba wylosowana dla *i*-tego (*j*-tego) wymiaru; *n* – liczba wymiarów; *dist* – odległość pomiędzy roślinami wyznaczona przy użyciu jednego z rozkładów. Formuła (8) powstała w wyniku przekształceń podstawowych reguł geometrii analitycznej.
3. Zwrot składowych każdego z wymiarów jest losowany z prawdopodobieństwem 0,5.

Model obliczeniowy w przypadku znajdowania minimum funkcji wielowymiarowej jest bardzo prosty i sprowadza się do wyznaczenia wartości tej funkcji w określonym punkcie.

3.2.2. Problem komiwojażera

Użycie algorytmu IWO do problemu komiwojażera wymaga dokonania wyboru postaci osobnika (chwastu) jako trasy stanowiącej pojedyncze rozwiązanie problemu. W literaturze proponuje się reprezentacje wektorowe: ścieżkową, porządkową i w postaci listy sąsiedztwa oraz reprezentacje macierzowe. Reprezentacja ścieżkowa, zgodnie z którą kolejne pozycje na liście

oznaczają kolejno odwiedzane miasta, wydaje się być najbardziej naturalną. Z tego też względu zdecydowano się na wybór tej reprezentacji.

Inicjalizacja pierwszej populacji chwastów odbywa się przy użyciu algorytmu zachłannego. Poszczególne rozwiązania konstruuje się w następujący sposób:

1. Losuje się miasto, które staje się miastem startowym.
2. Z pozostałych miast wybiera się to, które leży najbliżej ostatnio wybranego.
3. Miejsce wybrane w punkcie 2 jest usuwane ze zbioru miast nieodwiedzonych.
4. Punkty 2 i 3 powtarzane są do chwili, w której zbiór miast nieodwiedzonych jest pusty.

W pracy [48] opisano rozwiązanie problemu komiwojażera za pomocą algorytmu ewolucyjnego przy użyciu operatora o nazwie *inver-over*. Operator ten łączy w sobie cechy mutacji i krzyżowania. Rezultaty tych badań pozwoliły określić zaproponowany algorytm jako „prawdopodobnie najszybszy dotychczas opracowany algorytm ewolucyjny dla problemu komiwojażera" [35]. Z tego względu zdecydowano się na wykorzystanie operatora *inver-over* w algorytmie IWO. Prócz tego zastosowano dwa tradycyjne operatory: zamianę oraz inwersję [21].

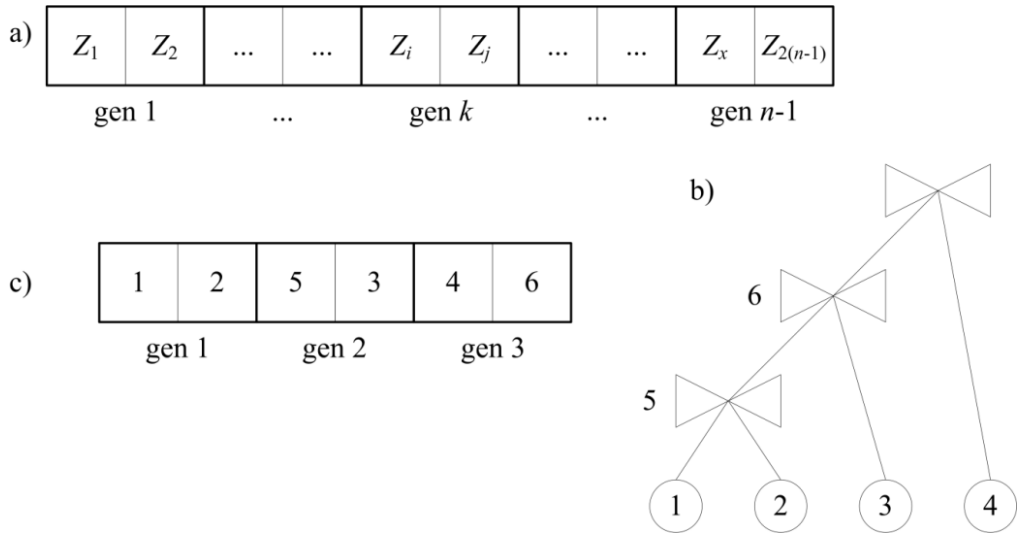
Ostatni element algorytmu, który wymaga sprecyzowania, to postać funkcji przystosowania osobnika. Dla rozważanego problemu optymalizacyjnego jest ona łączną długością trasy wyznaczonej przez kolejne numery miast zapisanych na kolejnych pozycjach chwastu.

3.2.3. Określanie kolejności złączeń w realizacji zapytań w scentralizowanych bazach danych

We wszystkich rozważaniach dotyczących problemu określania kolejności złączeń w realizacji zapytań kierowanych do baz danych zakłada się, że dane, które podlegają analizie, są danymi wstępnie przetworzonymi. Oznacza to, że dysponuje się zbiorami rekordów, powstałymi w wyniku zastosowania na tabelach bazodanowych operacji selekcji i projekcji.

Pojedynczy osobnik populacji chwastów stanowi odpowiednik pojedynczego rozwiązania problemu, którym jest określony porządek realizacji złączeń zbiorów rekordów. Jego reprezentację przedstawia rys. 3a.

Symbole Z_i, Z_j oznaczają numery zbiorów rekordów, które zostaną złączone w k-tym kroku. Dwójka symboli (Z_i, Z_j) będzie nazywana genem. Liczba genów w osobniku jest równa n–1, gdzie n jest liczbą zbiorów pierwotnych wymienionych w zapytaniu. Przez zbiór pierwotny rozumie się taki zbiór rekordów, który powstał w wyniku zastosowania operacji selekcji i projekcji na tabelach bazy danych. Zbiory pierwotne otrzymują kolejne numery od 1 do n. Układ genów odpowiada kolejności wykonywania poszczególnych złączeń. Wynik złączenia opisanego pierwszym genem będzie zbiorem rekordów o numerze n+1, a kolejne zbiory (wyniki pośrednie, zbiory pośrednie) otrzymają numery będące kolejnymi liczbami naturalnymi (n+2, n+3, …). Ponieważ wyniki pośrednie są wykorzystywane w dalszych złączeniach, będą występować w kolejnych genach. Jednakże musi być spełniony następujący warunek: gen zawierający numer zbioru stanowiącego wynik pośredni może w osobniku wystąpić dopiero po genie, który reprezentuje operację złączenia generującą ten wynik pośredni.



Rys. 3. Reprezentacja pojedynczego rozwiązania: a) symboliczna reprezentacja osobnika; b) przykładowe drzewo realizacji złączeń; c) osobnik odpowiadający drzewu z punktu b).

Konstruowanie opisanych osobników do populacji początkowej polega na określeniu w każdym z genów pary zbiorów rekordów, jakie mają być złączone. Należy pamiętać, że numery zbiorów pośrednich mogą wystąpić dopiero po genie reprezentującym złączenie, w którym ten zbiór powstaje. Zbiory rekordów dobiera się zatem do poszczególnych genów w taki sposób, aby nie wystąpiła sytuacja nieprawidłowa.

Transformacja rośliny macierzystej, prowadząca do powstania chwastu potomnego polega na zmianie jednego z pary łączonych ze sobą zbiorów. Mechanizm wykorzystany w algorytmie IWO odpowiada operatorowi genetycznemu mutacji, zastosowanemu w algorytmie ewolucyjnym opisanym w pracy [36].

Transformacja numerów zbiorów jest realizowana przez zamianę dwóch wylosowanych zbiorów należących do różnych genów. W ten sposób nie dojdzie do niedopuszczalnego wielokrotnego wystąpienia tego samego numeru zbioru w pojedynczym osobniku. Ponadto sposób przeprowadzenia transformacji zapewnia, że do genu trafia taki numer zbioru, dla którego gen, reprezentujący operację złączenia tworzącą ten zbiór, zajmuje w osobniku wcześniejszą pozycję niż gen poddawany transformacji.

Algorytm IWO wymaga do oceny jakości rozwiązań zdefiniowania funkcji celu, którą w rozważanym problemie jest szacunkowy koszt (czas) realizacji ciągu złączeń. W prowadzonych badaniach poddano modyfikacji model zastosowany w pracach [46, 20]. Wymaga on zdefiniowania formuł umożliwiających oszacowanie liczebności zbioru wynikowego operacji złączenia dla następujących przypadków: złączenie typu jeden do jeden (1:1), jeden do wielu (1:N) oraz wiele do wielu (N:M). Należy także dysponować możliwością obliczenia rozmiaru rekordu w zbiorze wynikowym oraz zbiorach pośrednich. Długość rekordu jest ważna ze względu na konieczność odczytania danych z dysku twardego podczas wykonania złączenia. W niniejszych rozważaniach zakłada się, że całkowity rozmiar zbioru wynikowego znacznie przekracza dostępną pamięć operacyjną. Stąd konieczność przechowywania danych na dysku. Istotnym elementem do wyznaczenia funkcji celu jest także koszt wykonania operacji złączenia. Szacuje się go według formuły opartej na pracach [5, 46]. Wzór ten został wzbogacony przez autora o człon

określający narzut czasowy związany z odczytem danych z dysku twardego komputera wykonującego złączenie.

Do obliczenia wartości funkcji celu dla pojedynczego rozwiązania, czyli określenia kolejności złączeń w realizacji zapytania skierowanego do bazy danych, wykorzystano strukturę drzewiastą. Liście struktury zawierają charakterystykę zbiorów pierwotnych, natomiast w wierzchołkach zapisane są informacje o poszczególnych operacjach złączenia i ich rezultatach. Pojedyncza krawędź łączy liść lub wierzchołek opisujący powstanie danego zbioru z wierzchołkiem przyporządkowanym złączeniu z udziałem tego zbioru. Każdy wierzchołek jest zwieńczeniem poddrzewa obejmującego określony fragment rozwiązania i zawiera informacje pozwalające na obliczenie wartości funkcji celu dla tego fragmentu. Przez stopniowe kumulowanie wartości funkcji celu kolejne elementy struktury są wypełniane w jednym przebiegu – od liści w kierunku korzenia.

Wartość funkcji przystosowania, odgrywająca w przypadku algorytmu IWO istotną rolę w określeniu liczby nasion rozsiewanych przez daną roślinę, obliczana jest jako odwrotność funkcji celu.

Jak wspomniano w podrozdziale 2.1, jedną z form reprezentacji zapytania jest graf złączeń. Jego wierzchołki odpowiadają zbiorom pierwotnym, a każda z krawędzi wyraża fakt istnienia między nimi wyrażenia łączącego. Jest to kryterium dopasowania rekordów tych zbiorów do siebie. Do charakterystycznych kształtów grafu złączeń zalicza się:

- graf gwiazdy, w którym jeden ze zbiorów rekordów występuje w każdym wyrażeniu łączącym,
- graf łańcuchowy, gdzie dwa zbiory rekordów występują tylko jednokrotnie w wyrażeniach łączących, natomiast pozostałe zbiory – dwukrotnie.

Podana charakterystyka stanowi podstawę do zaproponowania formuły pozwalającej na identyfikację kształtu grafu liczącego n wierzchołków, gdzie z każdym wierzchołkiem i związanych jest k_i krawędzi. Formuła ma następującą postać:

$$L = \frac{1}{n} \cdot \sum_{i=1}^{n} k_i \tag{9}$$

Dla grafu o charakterze gwiazdy liczba L wynosi $L = \frac{n^2 - 2n + 2}{n^2 - n}$, zaś dla grafu łańcuchowego $L = \frac{n+2}{2n}$. Pozostałe kształty określa się mianem grafów nieregularnych. Dzięki łatwej identyfikacji kształtu grafu złączeń istnieje możliwość lepszego doboru współczynników algorytmu.

4. Wyniki badań i ich analiza

Niniejszy rozdział jest poświęcony doborowi danych wybranych do realizacji testów. Przedstawiono również wyniki eksperymentów oraz ich analizę. Rozdział podzielono na trzy części zawierające zagadnienia dotyczące poszczególnych problemów optymalizacyjnych.

4.1. Problem znajdowania minimum funkcji wielowymiarowej

Problem znajdowania minimum globalnego funkcji wielowymiarowej (funkcji wielu zmiennych) jest jednym z klasycznych zagadnień optymalizacyjnych o charakterze ciągłym. Realne zestawy danych optymalizacyjnych często zawierają dziesiątki, setki, a czasami nawet tysiące wymiarów. Z tego względu ważnym wydaje się być wszechstronne przetestowanie algorytmu optymalizacyjnego.

4.1.1. Dobór funkcji testowych i metodyka przeprowadzonych badań

Z całego wachlarza funkcji do testów postanowiono wybrać pięć: pierwszą funkcję De Jonga [6] oraz funkcje Rosenbrocka [43], Rastrigina [41], Griewanka [12] i Ackleya [1]. Minimum globalnym dla wszystkich wymienionych funkcji jest wartość równa 0.

Pierwsza funkcja De Jonga [6] jest jedną z najprostszych funkcji testowych. Wybrano ją ze względu na jej popularność i prostotę. Gdyby analizowana metoda optymalizacyjna nie pozwalała na uzyskanie wyników zbliżonych do globalnego minimum funkcji, bezcelowa byłaby kontynuacja badań dla funkcji bardziej złożonych.

Pierwszą funkcję De Jonga można przedstawić za pomocą następującego wzoru:

$$f(x) = \sum_{i=1}^{n} x_i^2 \tag{10}$$

Obszar testowy zwykle zawężany jest do hiperkostki o wymiarach $-5{,}12 \le x_i \le 5{,}12$, dla $i = 1, \ldots, n$, gdzie n jest liczbą wymiarów [38].

Minimum globalne funkcji Rosenbrocka [43], zwanej także: doliną Rosenbrocka, funkcją bananową lub drugą funkcją De Jonga leży wewnątrz długiej, wąskiej doliny o płaskim dnie. Dotarcie do minimum globalnego jest trudne z powodu powolnego spadku dna doliny do minimum. Klasyczna funkcja Rosenbrocka jest dwuwymiarowa i jednomodalna. Jednak jej wielowymiarowy odpowiednik posiada dwa minima.

Obszar optymalizacji zawiera się zwykle w przedziale $-2{,}048 \le x_i \le 2{,}048$, dla $i = 1, \ldots, n$, gdzie n jest liczbą wymiarów. Dla liczby wymiarów $n > 1$ funkcja ta opisana jest następującą formułą:

$$f(x) = \sum_{i=1}^{n-1} \left[100 \cdot \left(x_{i+1} - x_i^2 \right)^2 + \left(1 - x_i \right)^2 \right] \tag{11}$$

Funkcja Rastrigina [41] jest odmianą pierwszej funkcji De Jonga wzbogaconej o czynnik kosinusowy. Powoduje on utworzenie bardzo dużej liczby równomiernie rozmieszczonych minimów lokalnych. Podobnie jak w przypadku pierwszej funkcji De Jonga, obszar optymalizacji zawęża się do zakresu $-5{,}12 \le x_i \le 5{,}12$, dla $i = 1, \ldots, n$. Definicję funkcji Rastrigina można przedstawić za pomocą wzoru:

$$f(x) = 10n + \sum_{i=1}^{n} \left[x_i^2 - 10\cos\left(2\pi x_i\right) \right] \tag{12}$$

Funkcja Griewanka [12] jest scharakteryzowana formułą:

$$f(x) = \frac{1}{4000} \cdot \sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1 \tag{13}$$

Wygląd funkcji zmienia się wraz z odległością, z jakiej się ją obserwuje. Przypomina ona pierwszą funkcję De Jonga, natomiast jej przybliżenia ukazują liczne minima lokalne. Analizowana jest zwykle hiperkostka o wymiarach $-600 \le x_i \le 600$, dla $i = 1, \ldots, n$ [38].

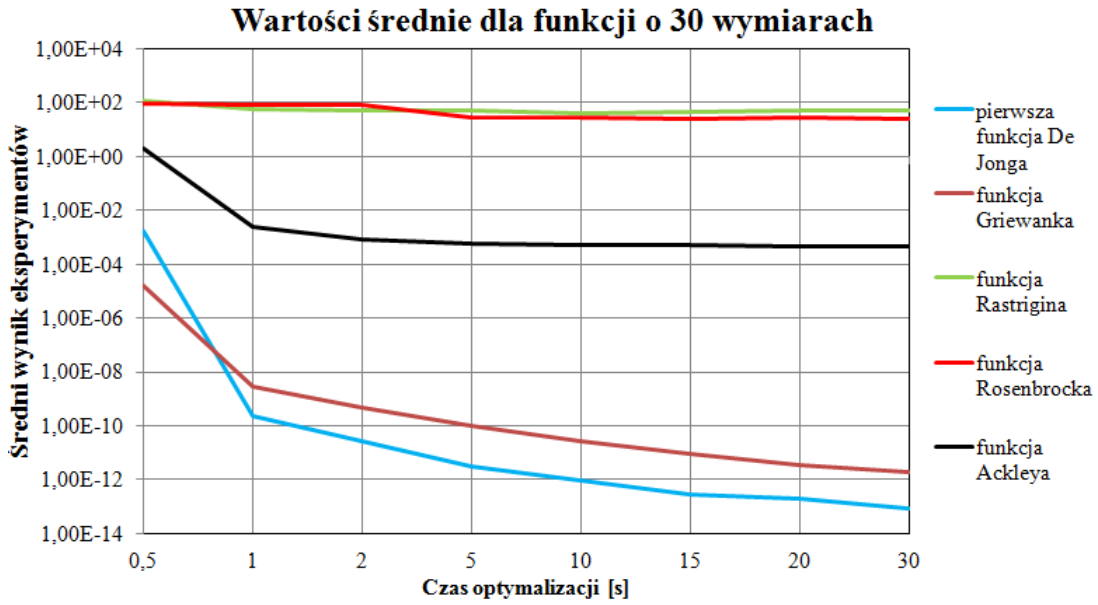
Ostatnią wybraną funkcją, zawierającą bardzo dużą liczbę minimów lokalnych, jest funkcja Ackleya [1], którą można przedstawić w następujący sposób [38]:

$$f(x) = -a \cdot \exp\left(-b \cdot \sum_{i=1}^{n} x_i^2\right) - \exp\left(\frac{1}{n} \cdot \sum_{i=1}^{n} \cos(c \cdot x_i)\right) + a + \exp(1) \tag{14}$$

gdzie zwyczajowo współczynniki przyjmują następujące wartości: $a = 20$; $b = 0{,}2$; $c = 2\pi$. Przestrzeń testowa ograniczona jest do przedziału $-32{,}768 \le x_i \le 32{,}768$, dla $i = 1, \ldots, n$.

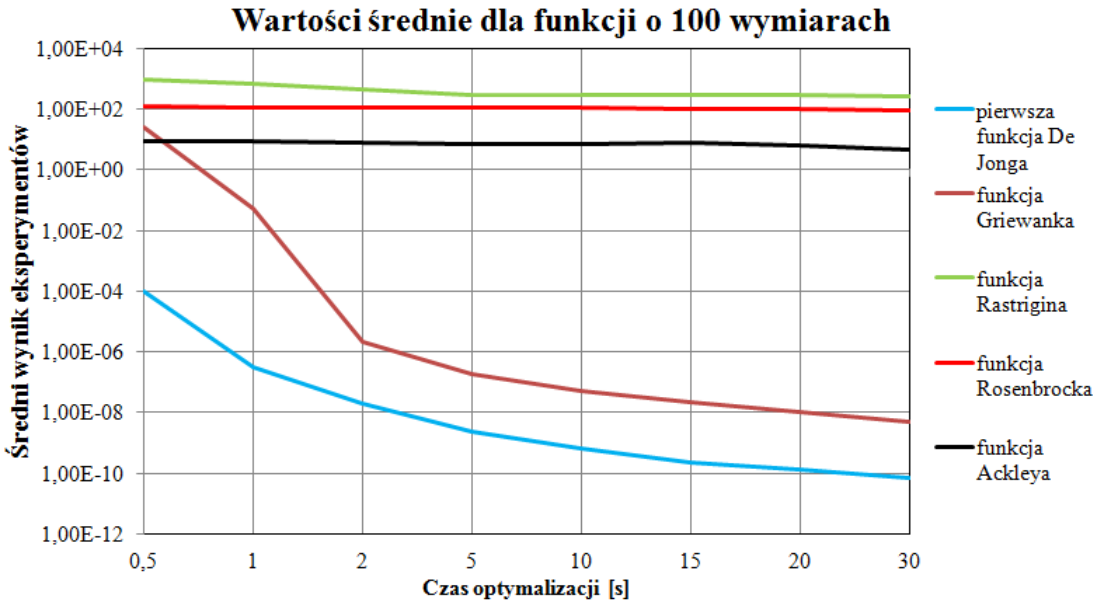
Wszystkie badania przeprowadzono dwuetapowo. Wstępną fazą było wykonanie eksperymentów mających na celu wyznaczenie jak najlepszych współczynników algorytmu dla dalszych testów. Po określeniu współczynników realizowano badania właściwe.

4.1.2. Badania z użyciem wybranych funkcji testowych



Rys. 4. Wykresy prezentujące średnie wyniki obliczeń przeprowadzone dla funkcji o 30 wymiarach.

Pierwsza faza badań skupiła się wokół wyznaczenia wartości minimów globalnych dla następujących funkcji testowych: pierwsza funkcja De Jonga, funkcje Griewanka, Rastrigina, Rosenbrocka i Ackleya. Obliczenia wykonano dla funkcji o 30 i 100 wymiarach. Wszystkie testy przeprowadzono przy użyciu autorskiej, zmodyfikowanej wersji IWO, a testy powtórzono stukrotnie. Eksperymenty przeprowadzono dla typowych zakresów początkowych tych funkcji [38]. Zastosowano dwa kryteria zatrzymania algorytmu – interwał czasowy oraz liczbę iteracji, z różnymi wartościami.



Rys. 5. Wykresy prezentujące średnie wyniki obliczeń przeprowadzone dla funkcji o 100 wymiarach.

Rysunki 4 i 5 prezentują wyniki obliczeń dla poszczególnych funkcji testowych w formie wykresów. Niestety na podstawie zamieszczonych wyników nie ma możliwości jednoznacznego stwierdzenia, czy exIWO pozwala na otrzymanie dobrych rezultatów. Aby w pełni potwierdzić to twierdzenie, wymagane jest przeprowadzenie testów porównawczych z innymi algorytmami optymalizacyjnymi.

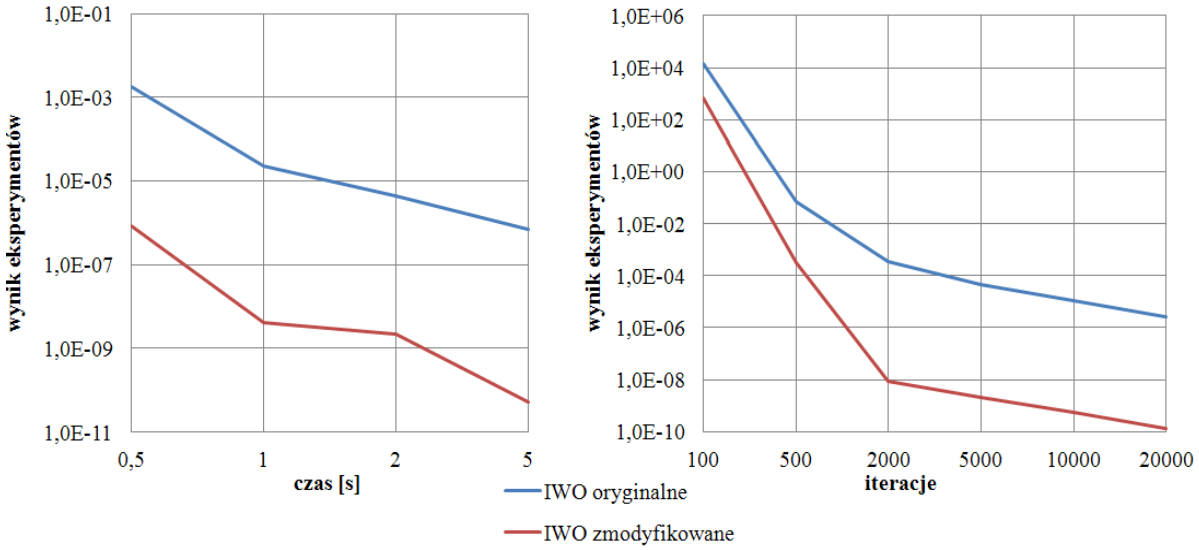
4.1.3. Porównanie exIWO z wersją oryginalną

Uzyskane wyniki badań zmodyfikowanego algorytmu IWO porównano z wynikami dla wersji oryginalnej, przedstawionej w pracy [34]. Oparto się na funkcjach Griewanka i Rastrigina o 30 wymiarach. Przestrzeń testową ograniczono zgodnie z [34] do hiperkostek o wymiarach odpowiednio: $-512 \le x_i \le 512$ oraz $-100 \le x_i \le 100$ dla $i = 1, \ldots, n$. Badania przeprowadzono dla następujących kryteriów stopu algorytmu: czasu (5, 2, 1 i 0,5 sekundy) oraz liczby iteracji (20 000, 10 000, 5000, 2000, 500 i 100 iteracji). Współczynniki algorytmu dla oryginalnej wersji IWO również zaczerpnięto z pracy [34]. Jako że algorytmy heurystyczne charakteryzują się wysoką niedeterministycznością, wszystkie obliczenia powtórzono stukrotnie. Rysunki 6 i 7 ilustrują wyniki numeryczne otrzymane przy użyciu oryginalnej i zmodyfikowanej wersji algorytmu IWO.

Z danych zestawionych na rysunkach 6 i 7 wynika, że dla badanych funkcji testowych, przy założonych kryteriach stopu, zmodyfikowana wersja algorytmu IWO pozwala na otrzymanie zdecydowanie lepszych jakościowo wyników. Na uwagę zasługuje fakt, że osie pionowe na obu wykresach zostały opisane skalą logarytmiczną, co potęguje przewagę wersji autorskiej nad oryginalną.

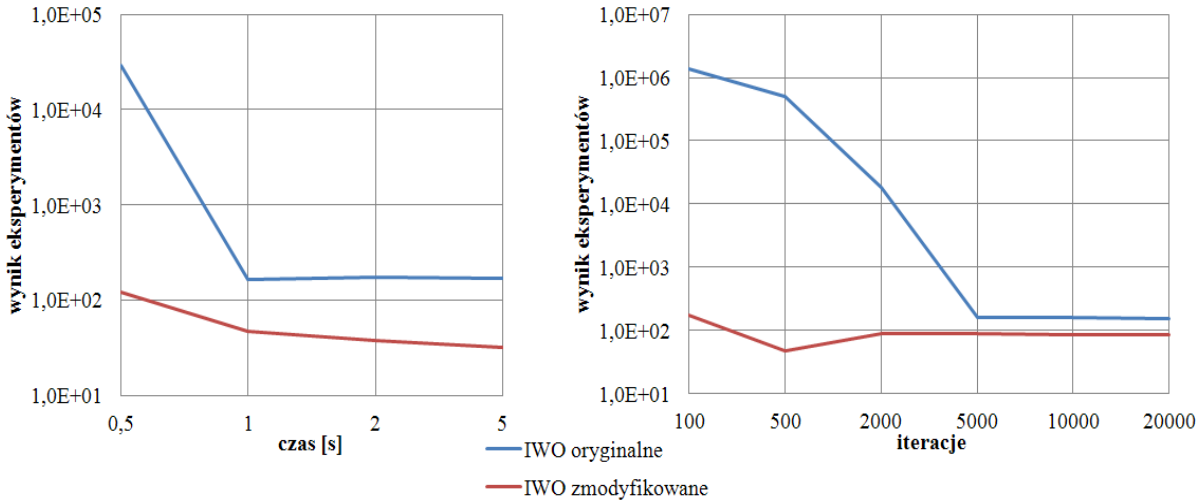
Obydwie wersje IWO cechują się różnymi złożonościami obliczeniowymi. Czas potrzebny na realizację jednej iteracji exIWO jest dłuższy niż w przypadku oryginalnej. Jednak ten nakład czasowy jest opłacalny, gdyż w jego wyniku otrzymuje się rozwiązania lepszej jakości. Przewa-

gi wersji autorskiej upatruje się przede wszystkim w rozbudowanym rozpraszaniu ziaren wokół chwastu macierzystego.



Rys. 6. Porównanie wyników otrzymanych przy użyciu oryginalnej i zmodyfikowanej wersji IWO dla 30 wymiarowej funkcji Griewanka.

W związku z różnym czasem realizacji jednej iteracji bardziej miarodajnym kryterium zatrzymania algorytmu wydaje się być z góry założony interwał czasowy.



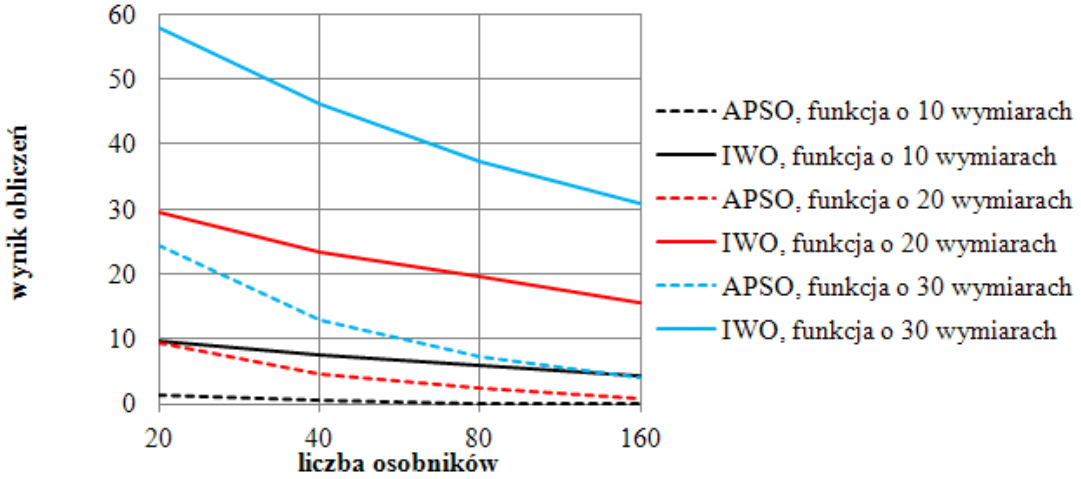
Rys. 7. Porównanie wyników otrzymanych przy użyciu oryginalnej i zmodyfikowanej wersji IWO dla 30 wymiarowej funkcji Rastrigina.

4.1.4. Porównanie exIWO z algorytmem APSO

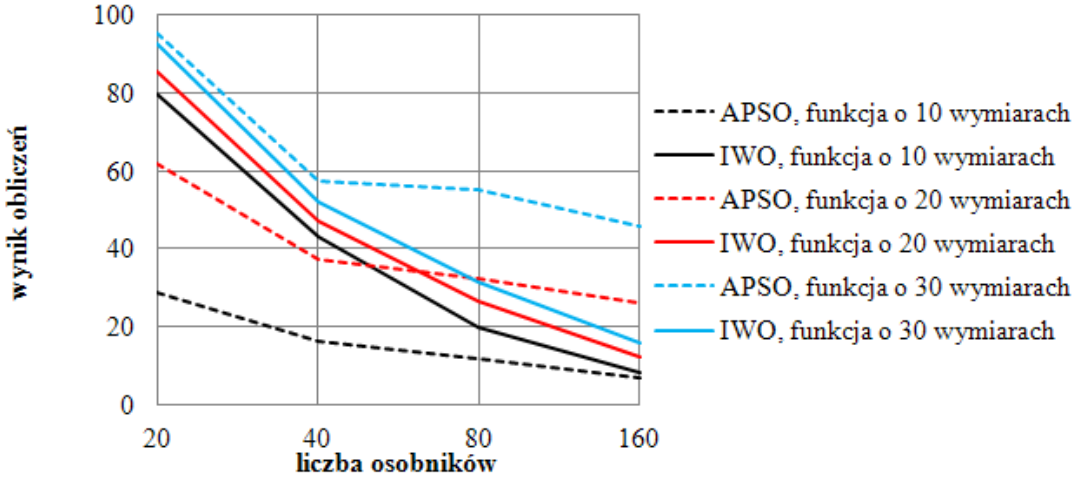
Ostatnią grupę stanowią badania porównawcze z adaptacyjnym algorytmem roju cząstek (ang. *adaptive particle swarm optimization*, APSO) [51]. Wybrano ten algorytm ze względu na jego wysoką skuteczność oraz szerokie rozpowszechnienie w literaturze.

Podstawą eksperymentów stały się funkcje: Rastrigina, Rosenbrocka i Griewanka. Zgodnie z [51], w każdej iteracji znajdowało się 20, 40, 80 lub 160 osobników. W przypadku algorytmu APSO liczba osobników oznacza liczbę cząstek w populacji, natomiast dla zmodyfikowanego

IWO – liczbę chwastów w pokoleniu. Funkcje zbudowane były z 10, 20 lub 30 wymiarów. Liczbie wymiarów ściśle odpowiadały liczby iteracji, równe odpowiednio: 1000, 1500 oraz 2000. Wszystkie obliczenia powtórzono 500 razy.



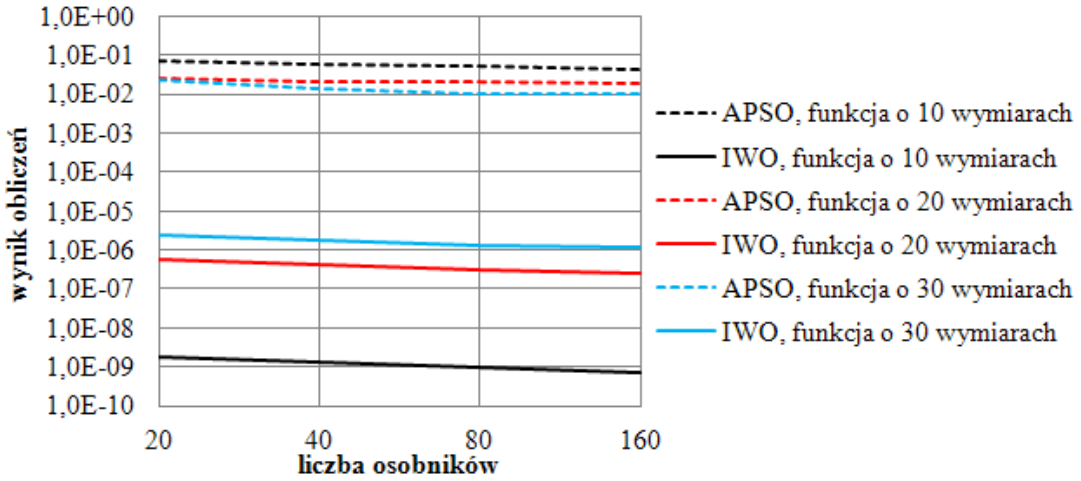
Rys. 8. Porównanie wyników uzyskanych przy pomocy APSO i exIWO na podstawie funkcji Rastrigina.



Rys. 9. Porównanie wyników uzyskanych przy pomocy APSO i exIWO na podstawie funkcji Rosenbrocka.

Przedstawione na rys. 8, 9 i 10 wyniki nie pozwalają jednoznacznie określić, który algorytm jest lepszy. Badania dotyczące funkcji Rastrigina wskazują na wyższość algorytmu APSO, natomiast funkcji Griewanka – zmodyfikowanego algorytmu IWO. Należy zwrócić uwagę, że osie pionowe dla funkcji Rastrigina i Rosenbrocka mają skalę liniową, a dla funkcji Griewanka – logarytmiczną. Fakt ten może delikatnie przechylać szalę na korzyść exIWO.

Przeprowadzone trzy serie eksperymentów dowodzą, że dla problemu znajdowania wartości minimalnej funkcji wielowymiarowej autorska, zmodyfikowana wersja algorytmu IWO generuje zdecydowanie lepsze wyniki niż wersja oryginalna. Może również z powodzeniem konkurować z innymi znanymi i szeroko rozpowszechnionymi w literaturze heurystycznymi algorytmami optymalizacyjnymi.



Rys. 10. Porównanie wyników uzyskanych przy pomocy APSO i exIWO na podstawie funkcji Griewanka.

Uzyskanie bardzo dobrych wyników w klasycznym problemie o charakterze ciągłym sugeruje, że opracowane metody mogą być równie wartościowe w przypadku dyskretnych problemów optymalizacyjnych.

4.2. Problem komiwojażera

Problem komiwojażera jest ciągle ważnym i aktualnym klasycznym problemem optymalizacyjnym o charakterze dyskretnym. Jego odmiany spotyka się m.in. w produkcji elektronicznych płytek drukowanych, jak i w firmach przewozowych. Wyniki przeprowadzonych eksperymentów będą bardzo ważne ze względu na ogólność i rozpowszechnienie niniejszego problemu optymalizacyjnego.

4.2.1. Dobór danych testowych i metodyka przeprowadzonych badań

Źródłem danych testowych stały się zbiory udostępnione przez *Research Group Combinatorial Optimization* z Ruprecht-Karls-Universität w Heidelbergu (Niemcy) [42]. Wybrano te zestawy testowe z dwóch przyczyn: dla znacznej większości zestawów znane są wartości optymalne oraz istnieje wiele odniesień w literaturze światowej. Fakt ten pozwala na konfrontację opracowanych metod z wynikami uzyskanymi przy użyciu innych popularnych i skutecznych algorytmów heurystycznych.

Odległości dróg łączących poszczególne miasta obliczane są w zestawach danych na kilka sposobów: euklidesowy, zaokrąglony euklidesowy, pseudo-euklidesowy, geograficzny oraz odległości podane wprost w postaci macierzowej [42].

Podobnie jak w przypadku problemu określania wartości minimalnej funkcji wielowymiarowej, wszystkie eksperymenty prowadzono dwuetapowo. Pierwszą fazą było wykonanie testów mających na celu wyznaczenie współczynników algorytmu dla poszczególnych grup. Następnie zrealizowano badania właściwe.

4.2.2. Porównanie exIWO z algorytmem Meta-RaPS i innymi algorytmami heurystycznymi

Tabela 1

Porównanie rezultatów uzyskanych za pomocą IWO i innych strategii

Nazwa metody	**Nazwa zbioru testowego**				
	kroA100	**kroB100**	**kroC100**	**kroD100**	**kroE100**
Meta-RaPS TSP (DePuy i in. [7])	0	0,25	**0**	**0**	0,17
Priority rule (DePuy i in. [7])	0,5	2,46	0,82	1,43	1,1
GRASP (DePuy i in. [7])	0	0,55	0,31	0,42	0,37
Christofides & 2opt 3opt (Lawler i in.)	2,51	1,4	1,53	0,17	3,03
Convex hull & 3opt (Lawler i in.)	0,37	1,46	1,06	**0,04**	2,46
Nearest neighbour & 2opt 3opt (Lawler i in.)	0,14	1,46	1,06	0,73	2,46
2opt 3opt (Lawler i in.)	0,81	1,44	0,53	1,74	0,18
Lin-Kernighan (Padberg i Rinaldi, 1991)	0,26	0	0,7	0,17	0,16
Modified Lin-Kernighan (Mark i Morton, 1992)	0	0,17	**0**	**0**	0,21
Composite heuristic, CCAO (Golden i Stewart, 1985)	0	0,97	0,5	0,97	2,54
Delaunay triangulation (Krasnogor i in., 1995)	0,51	2,13	2,79	3,81	2
I^3 (Renaud i in., 1996)	0	0,9	0,5	2	2,6
P-SEC (Rego, 1998)	0	0,32	**0,02**	0,75	0,33
F-SEC (Rego, 1998)	0	0	**0**	**0**	0
Guided local search (Voudouris i Tsang, 1999)	0	–	**0**	–	–
Genetic algorithm, A (Chatterjee i in., 1996)	0,70	–	1,78	1,45	–
Genetic algorithm, B (Chatterjee i in., 1996)	1,80	–	2,10	1,30	–
Neural net (Modares i in., 1999)	0,31	1,43	–	–	–
Neural net (Matsuyama i in., 1992)	1,81	3,37	–	–	–
Neural net (Burke i Damany, 1992)	5,00	4,31	–	–	–
Simulated annealing (Voudouris i Tsang, 1999)	0,42	–	0,80	–	–
Simulated annealing (Malek i in., 1989)	0,09	–	–	–	–
Tabu search (Voudouris i Tsang, 1999)	0	–	0,25	–	–
Tabu search (Malek i in., 1989)	0,33	–	–	–	–
Tabu search & 3opt (Tsubakitani i Evans, 1998)	1,37	–	–	–	–
IWO, inver-over, średnia	0	0,005	0,246	0,508	0,202
IWO, inver-over, minimum	0	0	0,096	0	0
IWO, odwracanie, średnia	0	0,007	0,239	0,488	0,194
IWO, odwracanie, minimum	0	0	0	0,169	0

Celem niniejszej serii badań było porównanie jakości rozwiązań otrzymywanych za pomocą exIWO z rezultatami uzyskiwanymi przy użyciu innych algorytmów heurystycznych.

Źródłem danych do analizy porównawczej stała się praca [7]. Zawiera ona zestawienie wyników uzyskanych dzięki metaheurystyce losowego przeszukiwania priorytetowego (Meta-RaPS). Wybór padł na metodę Meta-RaPS, ponieważ osiąga ona bardzo dobre wyniki dla przedstawionych w pracy [7] zbiorów testowych.

Przedstawione w tabeli 1 wyniki stanowią porównanie IWO z innymi strategiami optymalizacyjnymi i są procentowym odchyleniem od wartości optymalnych. Rezultaty uzyskane dzięki exIWO zostały przedstawione w czterech ostatnich wierszach. Zawierają one wartości minimalne i średnie dla 100 uruchomień algorytmu przy zastosowaniu dwóch operatorów transformacji. Wszystkie wartości liczbowe dla pozostałych algorytmów (poza IWO) zostały zaczerpnięte z pracy [7]. W zestawieniu użyto oryginalnych nazw. Należy zwrócić uwagę na fakt, że zaprezentowane w tabeli 1 dane, pochodzące z pracy [7], zawierają wartości minimalne uzyskane przy użyciu wymienionych algorytmów.

Pogrubioną czcionką zaznaczono te wartości, w których wartość minimalna uzyskana za pomocą algorytmu IWO okazała się w przypadku co najmniej jednego z operatorów gorsza od danej metody. W 9 na 91 przypadków minimum ze 100 uruchomień exIWO okazuje się gorsze od innych metod, w 12 przypadkach wartości są takie same (i wynoszą 0), natomiast w pozostałych 70 przypadkach – IWO okazuje się lepsze. Na uwagę zasługuje fakt, iż dla wszystkich pięciu zestawów danych osiągnięto wartość optymalną (wartość minimum). Zestawienie to jednoznacznie wskazuje, że zmodyfikowany algorytm IWO może z powodzeniem konkurować z innymi uznanymi strategiami ewolucyjnymi.

4.2.3. Porównanie exIWO ze strategią sieci samoorganizujących

Niniejsza seria badań miała na celu porównanie zmodyfikowanego IWO ze strategiami nienależącymi do rodziny algorytmów ewolucyjnych. Wybór padł na sieci samoorganizujące [22], gdyż wykazują one bardzo dużą skuteczność w rozwiązywaniu problemu komiwojażera. Źródłem danych porównawczych stała się praca [4]. Praca zawiera wyniki uzyskane za pomocą różnych metod opartych na idei sieci samoorganizujących.

W tabeli 2 zaprezentowano zestawienie wyników uzyskanych przy użyciu exIWO z dwoma różnymi operatorami transformacji oraz różnych sieci samoorganizujących. Wykorzystano następujące skróty: *PKN* – Pure Kohonen Network, *GN* – Guilty Net, *AVL* – strategia Angéniol, de la Croix Vaubois i Le Texier, *KL*, *KG* – warianty lokalny i globalny sieci Kohonena (Local/Global Kohonen Network Incorporating Explicit Statistics), *SETSP* – SOM Efficiently applied in the TSP, *MGSOM* – Modified Growing ring SOM approach for TSP. Dwie ostatnie kolumny zawierają wyniki zebrane dla algorytmu IWO. Przeprowadzono eksperymenty dla operatora *inver-over* (kolumna „IWO i-o”) oraz odwracania (kolumna „IWO od”). Poszczególne wartości w wierszu „średnia” są wartościami średnimi odpowiadających im kolumn.

Wartości oznaczone pogrubioną czcionką wskazują metodę optymalizacji, która osiągnęła wartości najbliższe globalnemu minimum dla poszczególnych zestawów testowych. Rezultaty

otrzymane dzięki operatorowi odwracania są odrobinę lepsze niż wartości otrzymane za pomocą *inver-over*.

Tabela 2

Porównanie rezultatów uzyskanych za pomocą IWO i sieci samoorganizujących

Zestaw	PKN	GN	AVL	KL	KG	SETSP	MGSOM	IWO i-o	IWO od
bier127	3,322	31,181	3,713	2,762	3,079	1,850	1,097	0,069	**0,061**
eil51	4,202	10,493	4,108	2,864	2,864	2,221	1,398	**0,099**	0,127
eil76	6,171	14,182	6,190	4,981	5,483	4,234	3,384	0,338	**0,292**
kroA200	5,311	34,058	5,540	2,836	3,667	3,119	1,972	0,757	**0,716**
lin105	6,921	7,584	6,487	1,985	1,291	1,301	**0,028**	0,033	0,070
pcb442	–	–	17,472	11,072	10,447	10,160	8,577	3,011	**2,580**
pr107	0,454	81,661	1,791	0,734	0,425	0,409	0,172	**0,000**	**0,000**
pr136	7,343	–	6,893	4,531	5,147	4,400	**2,154**	3,834	3,768
pr152	1,523	42,817	1,302	0,968	1,285	1,169	**0,741**	0,924	0,863
rat195	–	–	15,420	12,238	11,916	11,192	5,984	1,868	**1,721**
rd100	–	10,382	4,498	2,095	2,622	2,601	1,172	**0,105**	0,127
st70	2,637	11,956	2,711	1,511	2,326	1,600	1,183	0,594	**0,533**
średnia	–	–	6,344	4,048	4,213	3,688	2,322	0,969	**0,905**

4.2.4. Zestaw Mona Lisa

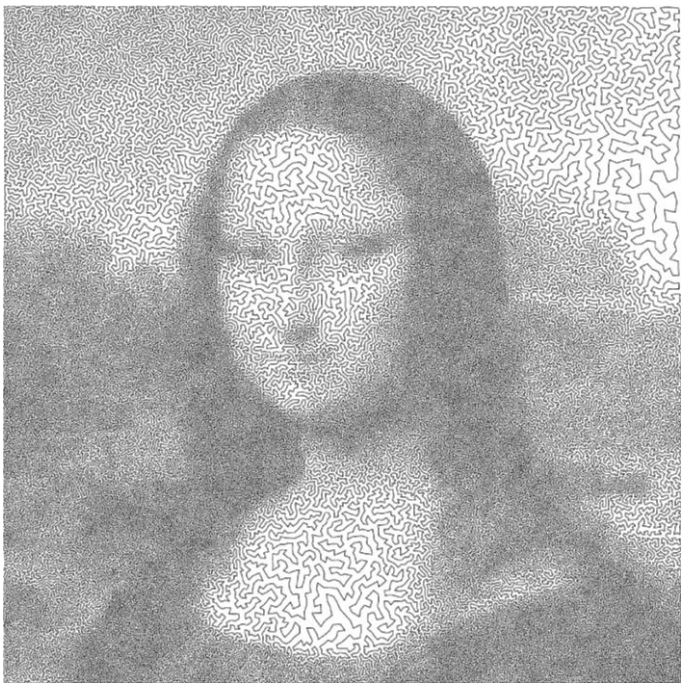
W lutym 2009 r. Robert Bosch stworzył zestaw danych testowych zawierający 100 000 miast. Jest on reprezentacją słynnego obrazu Mona Lisa autorstwa Leonarda da Vinci. Zestaw ten jest tematem konkursu, który ma na celu znalezienie jak najkrótszej drogi łączącej wszystkie punkty [39] (rys. 11).

Aktualnie znane najlepsze rozwiązanie wynoszące 5 757 191 podał Yuichi Nagata w marcu 2009 roku (czas wykonania optymalizacji nie jest podany, stan na dzień 3 stycznia 2014r.). Wynik ten dotychczas nie został poprawiony. Wartość dolnej granicy, wynoszącej 5 757 084 oznacza, że rezultat końcowy nie może być niższy od tej wartości. Granicę tę otrzymano w czasie 11,5 procesoro-lat.

Postanowiono sprawdzić jakość otrzymywanych dzięki exIWO rozwiązań dla tak dużego problemu optymalizacyjnego. Najlepszy wynik, jaki uzyskano przy zastosowaniu autorskiej zmodyfikowanej wersji algorytmu IWO, jest równy 5 919 404, czyli o 2,818% gorszy od najlepszego znanego. Rezultat ten otrzymano w czasie 19 dni 14 godzin i 22 minut (czyli ponad 200 razy krótszym niż wyznaczenie aktualnie obowiązującej granicy dolnej). Otrzymany wynik sugeruje, że opracowane metody pozwalają na optymalizację z dobrym skutkiem dużych zestawów danych. Niestety, trudno dokładnie ocenić wartość osiągniętego rozwiązania i przewagi czasowej, gdyż, według wiedzy autora niniejszej pracy, literatura naukowa dotycząca tego zestawu danych jest bardzo uboga.

Z przedstawionych czterech serii badań dotyczących problemu komiwojażera wynika, że autorska zmodyfikowana wersja algorytmu IWO doskonale sprawdza się w rozwiązywaniu niniejszego problemu. Może ona z bardzo dobrym skutkiem konkurować z wieloma algorytmami optymalizacyjnymi, należącymi zarówno do grupy algorytmów ewolucyjnych, jak i z metodami

opartymi na innych ideach. Radzi sobie także z bardzo dużymi problemami zawierającymi nawet 100 000 miast.



Rys. 11. Ilustracja zestawu testowego *mona-lisa*.

4.3. Problem określania kolejności złączeń w realizacji zapytań

Przeprowadzone eksperymenty numeryczne pozwalają twierdzić, że autorska wersja algorytmu IWO bardzo dobrze sprawdza się w optymalizacji klasycznych zadań. Uzyskane wyniki sugerują, iż zasadne jest wykorzystanie i przetestowanie opracowanej strategii na centralnym problemie optymalizacyjnym niniejszej pracy, jakim jest określanie kolejności złączeń w realizacji zapytań kierowanych do baz danych. Warto zaznaczyć, że opisywane zagadnienie jest znacznie bardziej złożone obliczeniowo niż poprzednio opisane. Jest to spowodowane znacznie większą ilością danych, które należy uwzględnić w prowadzonych badaniach i koniecznością obliczania dość skomplikowanej funkcji celu.

4.3.1. Dobór danych testowych i metodyka przeprowadzonych badań

Do przeprowadzenia kompleksowych badań wymagana jest odpowiednia liczba danych. Dane te powinny zawierać zapytania o różnych grafach złączeń, zarówno o charakterze gwiazdy, łańcucha, jak i nieregularnym. Zapytania testowe winny wykorzystywać jak największą liczbę tabel. Przypuszcza się, że im większa będzie liczba tabel, tym bardziej będą uwydatnione różnice w wynikach otrzymywanych przy użyciu różnych metod.

Niestety, pomimo wielu starań nie znaleziono wystarczającej liczby gotowych zestawów danych pozwalających na przeprowadzenie opisanych testów. Skonstruowano zatem generator danych oraz zapytań testowych. Wzorowano się przy tym na pracy [40].

Na potrzeby niniejszych badań utworzono 90 baz danych zawierających po 50 tabel każda.

Opracowane zestawy można podzielić pod względem wykorzystanego grafu złączeń na trzy grupy:

- zestawy oparte na grafach o charakterze łańcucha (20 zestawów);
- zestawy oparte na grafach o charakterze gwiazdy (30 zestawów); liczba zestawów jest zwiększona w stosunku do zestawów opartych na grafach o charakterze łańcucha ze względu na dodatkowe kryterium, jakim jest liczba wierszy w tabeli faktów;
- zestawy oparte na grafach o charakterze nieregularnym (40 zestawów); zdecydowano się na podwojenie liczby zestawów testowych z grafem o charakterze nieregularnym, w stosunku do danych z grafem łańcuchowym, z dwóch przyczyn: tego typu zapytania występują najczęściej; optymalizacja kolejności realizacji złączeń jest bardziej skomplikowana z powodu większej liczby możliwych do zrealizowania złączeń.

Dodatkowo każdą z wymienionych grup można podzielić na dwie równoliczne podgrupy pod względem liczby wierszy, jakie znajdują się w tabelach. Zestawy z tabelami zawierającymi od 10 do 5 000 wierszy nazywane są skrótowo *bazami* z *małymi tabelami*, natomiast zestawy z tabelami zawierającymi od 100 do 50 000 wierszy – *bazami* z *dużymi tabelami*.

Liczba wierszy dla poszczególnych tabel jest losowana z różnym prawdopodobieństwem:

- liczba wierszy dla baz z małymi tabelami zawiera się w przedziale:
  - od 10 do 99 z prawdopodobieństwem 0,2,
  - od 100 do 999 z prawdopodobieństwem 0,6,
  - od 1 000 do 5 000 z prawdopodobieństwem 0,2,
- liczba wierszy dla baz z dużymi tabelami zawiera się w przedziale:
  - od 100 do 999 z prawdopodobieństwem 0,2,
  - od 1 000 do 9 999 z prawdopodobieństwem 0,6,
  - od 10 000 do 50 000 z prawdopodobieństwem 0,2.

Do każdej tabeli dodano kolumnę, której długość ma losowy rozmiar w granicach od 50 do 500 znaków. Kolumna ta reprezentuje dane zawarte w każdym wierszu, a nie będące kluczem głównym lub obcym. Dodatkowa długość wiersza pozwala na symulację obciążenia systemu bazodanowego związanego z odczytem danych z dysku twardego. Wspomniany odczyt ma duże znaczenie przy wyznaczaniu wartości funkcji celu.

Do każdego zestawu danych przyporządkowane jest jedno zapytanie pobierające wszystkie informacje ze wszystkich tabel (SELECT * FROM…). Dane testowe skonstruowano w taki sposób, aby nie było sytuacji wymuszającej realizację iloczynu kartezjańskiego, ani żeby nie było możliwości utworzenia zamkniętych cykli. Wszystkie zapytania zwracały jako wynik końcowy od 100 000 do 1 000 000 wierszy.

Celem przeprowadzanych eksperymentów było porównanie czasów wykonania zapytania uzyskanych w systemie Microsoft SQL Server 2008. Dla każdego zapytania realizowano plan wyznaczony przez moduł optymalizacyjny systemu zarządzania bazą danych oraz plany skonstruowane przez algorytm IWO. Wykonanie planów utworzonych przy użyciu opracowanej strategii było możliwe dzięki zestawowi *wskazówek* pozwalających użytkownikowi na wymuszenie określonych zachowań modułu optymalizacyjnego.

Podobnie jak w przypadku określenia minimum funkcji wielowymiarowej oraz problemu komiwojażera wszystkie badania przeprowadzono dwuetapowo. Fazą wstępną było określenie współczynników algorytmu dla każdego z typów grafu złączeń z osobna. Jest to możliwe ze względu na łatwą metodę sprawdzenia typu grafu. Należy podkreślić, że w tej fazie nie wykorzystywano danych testowych.

Badania właściwe przeprowadzono w następujących krokach:

1. Określenie planu realizacji zapytania testowego przy użyciu algorytmu IWO. Przyjęto, że minimalny czas obliczeń wynosi 5 sekund, zaś chwilę ich zakończenia wyznacza utworzenie pierwszej populacji, którą skonstruowano już po upływie tego czasu. Wartość interwału czasowego wybrano arbitralnie. Ze względu na niedeterministyczny charakteru algorytmu IWO obliczenia powtarzano dziesięciokrotnie, otrzymując w ten sposób 10 różnych planów realizacji zapytania.
2. Wymuszenie realizacji zapytania testowego w systemie Microsoft SQL Server 2008 według planu wyznaczonego w punkcie 1. Zastosowanie wskazówki OPTION(FORCE ORDER) [49] pozwoliło na narzucenie takiego porządku złączeń, jaki został wyznaczony przez algorytm IWO. Ponadto dzięki wskazówce OPTION(LOOP JOIN) wymuszono realizację złączeń zgodnie z algorytmem pętli zagnieżdżonych [9]. Użycie tej metody zakłada aktualna wersja algorytmu IWO. Ponieważ dla każdego wariantu planu otrzymanego dla danego zapytania wykonywano 10 uruchomień, rejestrowano w ten sposób 100 pomiarów czasu realizacji.
3. Określenie planu realizacji zapytania testowego za pomocą metody wbudowanej w systemie Microsoft SQL Server 2008. Optymalizator dysponował histogramami opisującymi aktualne dane zawarte w tabelach.
4. Następnie wymuszano wykonanie zapytania według planu określonego w punkcie 3. Dla zapewnienia identycznych warunków realizacji zapytania złączenia były wykonywane przy użyciu metody pętli zagnieżdżonych, gdyż tylko ta metoda jest wspierana aktualnie przez algorytm IWO. Zapytania wykonywano 10-krotnie otrzymując 10 pomiarów czasu realizacji zapytania.

Na wszystkich kolumnach z kluczami głównymi oraz obcymi utworzono indeksy typowe dla tego SZBD. Dla uniknięcia wpływu wcześniej wykonywanych operacji na rezultat kolejnej każdorazowo opróżniano bufory pamięci podręcznej (*cache*) wykonując zestaw poleceń: CHECKPOINT oraz DBCC DROPCLEANBUFFERS.

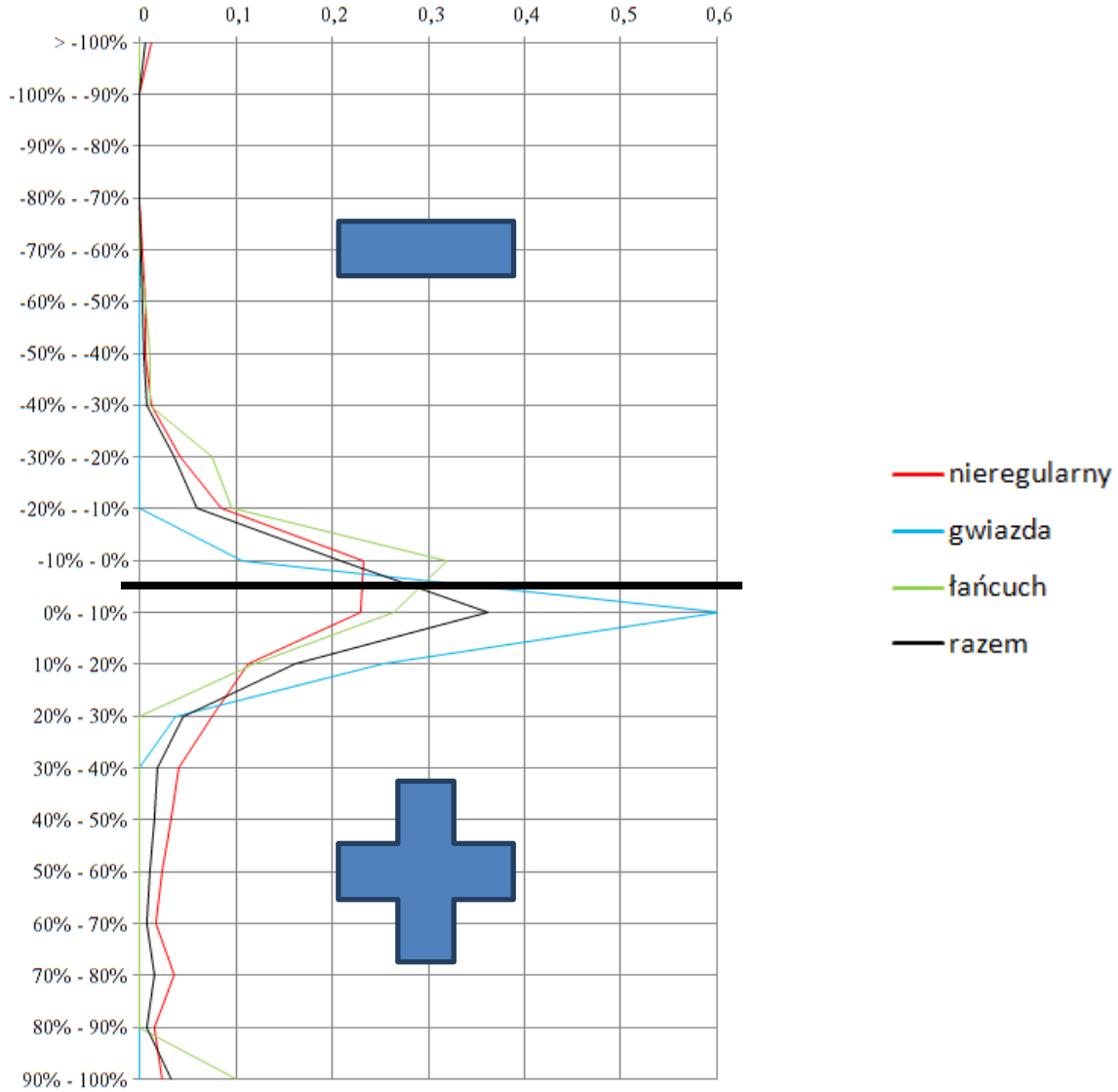
4.3.2. Porównanie czasów wykonania zapytań w systemie SQL Server 2008

Rysunek 12 ilustruje wyniki przeprowadzonych badań porównawczych. Wartości zaprezentowane na osi pionowej są zakresami odchylenia pomiędzy średnim czasem wykonania zapytania na podstawie planu wygenerowanego przez metodę zawartą w systemie SQL Server 2008, a planu zbudowanego przez exIWO. Można to zapisać następującą relacją:

$$\text{względna_jakość_rozwiązań} = \frac{\text{śr_czas_SQLServera} - \text{śr_czas_IWO}}{\text{śr_czas_SQLServera}} \cdot 100\% \qquad (15)$$

Wartość dodatnia oznacza, że zapytanie zrealizowane przy użyciu exIWO zostało wykonane szybciej niż na podstawie metody wbudowanej. W przypadku, gdy czas realizacji zapytania przy użyciu zmodyfikowanego algorytmu IWO jest dłuższy niż dwukrotna wartość czasu realizacji zapytania przy użyciu metody wbudowanej w optymalizator SQL Server 2008, względna jakość rozwiązań jest niższa niż -100%.

Dane zostały przeskalowane w taki sposób, aby suma wartości dla każdego typu grafu była równa jedności. Oś pozioma stanowi prawdopodobieństwo wystąpienia czasu wykonania zapytania, zrealizowanego przy użyciu algorytmu IWO, w danym przedziale odchylenia w stosunku do czasu realizacji zapytania zrealizowanego na podstawie metody wbudowanej w system Microsoft SQL Server 2008.



Rys. 12. Wykres ilustrujący zestawienie rezultatów badań porównawczych.

Z informacji zawartych na rysunku 12 można wywnioskować, że w przypadku zapytań o grafie łańcuchowym algorytm IWO generuje plany trochę gorsze niż metoda wbudowana, natomiast w przypadku zapytań z grafem o charakterze gwiazdy oraz, w mniejszym stopniu, nieregularnym zaobserwować można sytuację odwrotną. Na uwagę zasługuje fakt, iż w wielu

przypadkach (wiersz, dla którego względna jakość rozwiązań wynosi 90% – 100 %) algorytm IWO wygenerował plan realizacji zapytania, którego czas wykonania był kilka-kilkanaście razy krótszy niż w przypadku metody wbudowanej w system SQL Server 2008.

W analizie otrzymanych wyników posłużono się dwoma wielkościami: szacunkowym kosztem realizacji planu oraz czasem realizacji zapytania. Szacunkowy koszt realizacji planu jest wielkością bezwymiarową. Liczba ta jest generowana przez optymalizator wbudowany w system Microsoft SQL Server 2008 i jest podstawą podejmowanych przez ten optymalizator decyzji. Odzwierciedla ona szacowany czas wykonania zapytania na podstawie dostarczonego planu. Należy zwrócić uwagę na fakt, iż liczba ta jest wyznaczana przed przystąpieniem do realizacji zapytania, w związku z czym może być obarczona dużym błędem. Czas realizacji zapytania jest z kolei rzeczywistym czasem, który upływa od chwili wysłania zapytania do systemu zarządzania bazą danych do momentu otrzymania wyniku końcowego.

Na podstawie uzyskanych wyników szczegółowych można wyciągnąć następujące wnioski:

1. W 59 na 90 przypadków (65,56%) szacunkowy koszt realizacji planu skonstruowanego przez system SQL Server 2008 jest niższy o ponad 25% od średniego szacunkowego kosztu realizacji planu wygenerowanego przez algorytm exIWO. W 17 na 59 przypadków relacja ta została zachowana dla czasów realizacji zapytania (czas wykonania zapytania na podstawie planu wygenerowanego przez SQL Server był krótszy niż czas wykonania zapytania dla planu wygenerowanego przez algorytm IWO). Jednak dysproporcja pomiędzy czasami realizacji zapytania była zdecydowanie mniejsza niż między szacowanymi kosztami realizacji planów. W pozostałych 42 sytuacjach czasy realizacji zapytania przy użyciu planu zbudowanego na podstawie algorytmu IWO okazały się krótsze niż na podstawie metody wbudowanej w system SQL Server.W 9 na 90 przypadków (10%) szacunkowy koszt realizacji planu skonstruowanego przez system SQL Server 2008 był nieco niższy (o mniej niż 25%) niż średni koszt realizacji planów wygenerowanych przez algorytm IWO. Jednak w 6 przypadkach okazało się, że relacja między czasami realizacji zapytania testowego jest odwrotna (czas wykonania zapytania na podstawie planu wyznaczonego przez algorytm IWO był krótszy niż czas wykonania zapytania przy użyciu planu pochodzącego z metody wbudowanej).
2. Szacunkowy koszt realizacji planu skonstruowanego przez system SQL Server 2008 był trochę wyższy w 5 na 90 przypadków (5,55%) niż średni szacowany koszt realizacji planów wygenerowanych przez algorytm IWO. Zostało to potwierdzone w 3 przypadkach przez porównanie czasów realizacji zapytania testowego według każdego z planów.
3. Szacunkowy koszt realizacji planu skonstruowanego przez system SQL Server 2008 był wyższy o ponad 25% niż średni szacunkowy koszt realizacji planów wygenerowanych przez algorytm IWO w 17 na 90 przypadków (18,89%). W 5 sytuacjach tak duża dysproporcja została potwierdzona czasami realizacji zapytań (czas realizacji zapytania na podstawie IWO okazał się znacznie krótszy), w kolejnych 2 przypadkach dysproporcja nie była tak znaczna, w pozostałych 9 – relacja między czasami realizacji zapytania testowego według każdego z planów okazała się odwrotna (wykonanie zapytania przy użyciu algorytmu IWO było wolniejsze w stosunku do realizacji zapytania na podstawie metody wbudowanej).

Kompleksowo przeprowadzone badania dotyczące konstruowania planów realizacji zapytań dowodzą, że w bardzo wielu przypadkach algorytm IWO generuje plany lepsze od optymalizatora wbudowanego w system Microsoft SQL Server 2008. W 58 na 90 przypadków czas realizacji zapytania opartego na planie stworzonym przez IWO jest krótszy.

Dla 27 z 30 różnych baz danych z grafem złączeń o charakterze gwiazdy otrzymano krótszy czas realizacji zapytania (268 na 300 eksperymentów) używając planów wyznaczonych przez autorski, zmodyfikowany algorytm IWO. Można wysnuć wniosek, iż dla baz danych z grafem złączeń o charakterze gwiazdy exIWO wyraźnie przewyższa metodę wbudowaną w optymalizator kosztowy MS SQL Server 2008.

Zastanawiający jest fakt bardzo dużej dysproporcji pomiędzy szacunkowymi kosztami realizacji planów: wyznaczonego przez system SQL Server, a wygenerowanego przez exIWO. W większości przypadków szacunkowy koszt realizacji planu jest niższy o ponad 25% dla planu skonstruowanego przez metodę wbudowaną w SQL Server 2008. Jednak ta tendencja nie została potwierdzona przez rzeczywiste czasy realizacji zapytań. Z badań wynika, iż wbudowany optymalizator niepoprawnie szacuje liczbę wierszy biorących udział w złączeniach. Prowadzi to do błędnego oszacowania kosztu realizacji zapytania. W tym też miejscu upatruje się możliwości znacznej poprawy optymalizatora systemu Microsoft SQL Server 2008.

5. Wnioski końcowe

Rozprawa dotyczy opracowania ogólnej metody optymalizacji, która może być wykorzystana do optymalizacji zapytań w bazach danych. Celem rozprawy było opracowanie metody wyznaczenia kolejności realizacji operacji złączenia, wykorzystującej zmodyfikowany algorytm *Invasive Weed Optimization* (exIWO).

Osiągnięcie tak postawionego celu wymagało zarówno dokładnego zapoznania się ze znanymi rozwiązaniami, jak i przeprowadzenia szczegółowej analizy algorytmu IWO. Czynniki te dały podstawę do opracowania metody autorskiej.

Realizacja celu postawionego w pracy przebiegała w kilku etapach. W pierwszym szczególny nacisk położono na sformułowanie podstawowych pojęć związanych z optymalizacją zapytania oraz na analizę całego procesu optymalizacji w systemie zarządzania bazą danych. Na podstawie dokonanej analizy sformułowano problem wyznaczenia kolejności złączeń w realizacji zapytań bazodanowych, który jest jednym z podstawowych zagadnień zadania optymalizacji zapytania. Jest to zarazem główny problem badawczy rozprawy. Wzięto pod uwagę również dwa klasyczne problemy optymalizacyjne: znajdowanie minimum funkcji wielowymiarowej (problem o charakterze ciągłym) i problem komiwojażera (problem o charakterze dyskretnym). Celem, który przyświecał wyborowi dwóch dodatkowych zagadnień, było dogłębne zbadanie opracowanych metod i strategii optymalizacyjnych. Bardzo dobre wyniki badań, jakie otrzymano stosując algorytm *Invasive Weed Optimization* do rozwiązania różnych problemów optymalizacyjnych, stanowiły dla autora niniejszej pracy bodziec do podjęcia próby rozwiązania przy jego użyciu problemu wyznaczenia optymalnej kolejności złączeń. Efektem prowadzonych prac stała się zmodyfikowana wersja algorytmu IWO. Metodę tę przystosowano do rozwiązywania

trzech zadań optymalizacyjnych (problem wyznaczania kolejności realizacji złączeń, znajdowanie minimum funkcji wielowymiarowej oraz problem komiwojażera).

Opracowanie nowych strategii optymalizacyjnych pozwoliło na sformułowanie następujących wniosków:

1. Autorska wersja IWO pozwala na optymalizację planów realizacji złączeń zapytań bazodanowych.
2. Opracowana strategia optymalizacyjna pozwala na znalezienie bardzo dobrych wyników w przypadku baz danych z grafami złączeń o charakterze gwiazdy. W tym przypadku przygotowana przez autora metoda może konkurować z komercyjnym systemem zarządzania bazą danych.
3. Zmodyfikowany algorytm IWO znajduje bardzo dobre rozwiązania dla klasycznych problemów optymalizacyjnych (wyznaczanie minimum globalnego funkcji wielowymiarowej oraz symetrycznego i asymetrycznego problemu komiwojażera).
4. Złożoność obliczeniowa opracowanej strategii jest wyższa niż złożoność oryginalnej wersji IWO, co jest rekompensowane jakością znajdowanych rozwiązań. Dzięki temu algorytm exIWO często znajduje rozwiązania lepszej jakości niż jego oryginalny odpowiednik, co potwierdzono w podrozdziale 4.1.3.
5. Zmodyfikowana wersja algorytmu IWO może konkurować z innymi algorytmami ewolucyjnymi oraz innymi metodami optymalizacyjnymi (np. sieciami samoorganizującymi, wyniki takiego porównania zamieszczono w podrozdziale 4.2.3.).
6. ExIWO może znaleźć zastosowanie dla zestawów danych o ogromnej liczbie możliwych rozwiązań (dla symetrycznego zadania komiwojażera zestaw *mona-lisa* o 100 000 lokalizacji).

Należy nadmienić, iż część wyników uzyskanych przez autora dla różnych problemów optymalizacyjnych została opublikowana w następujących pracach:

- dla problemu określania kolejności złączeń [24, 25, 27, 28, 31, 32],
- dla znajdowania minimum funkcji wielowymiarowych [15, 29],
- dla problemu komiwojażera [23, 30, 33].

Uzyskane wyniki pozwalają na stwierdzenie, że cel pracy został osiągnięty, a teza pracy udowodniona.

Efektem przeprowadzonych badań było także stworzenie kilku programów komputerowych, co dało możliwość weryfikacji opracowanych metod i składowych algorytmu IWO. Z ich pomocą wyznaczono wartości numeryczne dla przygotowanych wcześniej scenariuszy testowych. Jest to pierwszy krok w planie dalszych badań autora, a zaimplementowane aplikacje stanowią bazę dla dalszych prac. Kolejne zamierzenia dotyczą następujących zagadnień:

1. Rozszerzenie exIWO w zakresie dodania kolejnych algorytmów złączeń. Planuje się dodanie metody z użyciem funkcji haszującej (ang. *hash join*) oraz strategii sortowania i złączenia (ang. *sort merge*).

2. Rozszerzenie możliwości algorytmu IWO o prowadzenie optymalizacji dla baz danych rozproszonych (wymaga to określenia innej reprezentacji rozwiązania). Warto nadmienić, że prace nad tym zagadnieniem zostały rozpoczęte, a osiągnięte rezultaty zaprezentowano w [32, 27].
3. Testowanie różnych modeli obliczeniowych służących do oszacowania czasu realizacji zadań złączenia i przesyłu danych. Wstępne prace dotyczące szacowania selektywności zapytań, w których autor pracy brał udział, zamieszczono w [3].
4. Wykorzystanie innych operatorów transformujących osobniki (np. MSFX).
5. Użycie różnych tzw. miękkich metod selekcji osobników. Wstępne prace i wyniki eksperymentów dotyczących zastosowania algorytmu symulowanego wyżarzania zamieszczono w [23].
6. Stworzenie algorytmu hybrydowego łączącego zalety algorytmu exIWO z algorytmem ewolucyjnym. Wcześniejsze badania dotyczące problemu określania kolejności realizacji złączeń przy użyciu algorytmu ewolucyjnego zostały przez autora niniejszej pracy zaprezentowane w [27, 26].
7. Zastosowanie zmodyfikowanego algorytmu IWO do selekcji cech. Technika ta polega na wyborze podzbioru cech z całego zbioru. Kryterium doboru składowych podzbioru jest skuteczność zadania klasyfikacji na podstawie wybranego podzbioru. Metoda ta posłuży rozwiązywaniu problemu rozpoznawania osób na podstawie ich sposobu poruszania się. Jedną z głównych przyczyn hamujących rozwój tej dziedziny jest duża ilość danych, które należy przeanalizować. Pierwsze wyniki badań prowadzonych w tej dziedzinie zamieszczono w [15, 16]. Autor brał także udział w innych badaniach dotyczących redukcji liczby cech na potrzeby klasyfikacji osób na podstawie chodu ([17, 19, 14, 18]).

6. Bibliografia

1. Ackley D. H.: A connectionist machine for genetic hillclimbing. The Kluwer International Series in Engineering and Computer Science, Vol. 28, Kluwer Academic Publishers, Boston, MA, USA 1987.
2. Applegate D. L., Bixby R. E., Chvátal V., Cook W. J.: The travelling salesman problem: A Computational Study. Princeton University Press, Princeton, New Jersey, USA 2006.
3. Augustyn D. R., **Kostrzewa D.**: Szacowanie selektywności zapytań oparte na transformacie Hougha i metodzie PCA. Studia Informatica, Vol. 32, No. 2A(105), Wydawnictwo Politechniki Śląskiej, Gliwice 2012, str. 211-227.
4. Bai Y., Zhang W., Jin Z.: An self-organizing maps strategy for solving the travelling salesman problem. Chaos, Solitons and Fractals, Vol. 28, Elsevier, 2006, s. 1082-1089.
5. Chen Y.: A systematic metod for query evaluation in distributed heterogeneous databases. Journal of Information Science and Engineering, Vol. 16, 2000, s. 463-497.
6. De Jong K. A.: Analysis of the behavior of a class of genetic adaptive systems, PhD dissertation. University of Michigan, USA 1975.

7. DePuy G. W., Moraga R. J., Whitehouse G. E.: Meta-RaPS: a simple and effective approach for solving the travelling salesman problem. Transportation Research, Part E, Vol. 41, Elsevier, 2005, s. 115-130.
8. Dong H., Liang Y.: Genetic algorithms for large join query optimization. Proceedings of the 9th Annual Conference on Genetic and Evolutionary Computation GECCO ’07, ACM, New York, NY, USA 2007, s. 1211-1218.
9. Garcia-Molina H., Ullman J. D., Widom J.: Systemy baz danych pełny wykład. Wydawnictwa Naukowo-Techniczne, Warszawa 2006.
10. Ghalenoei M. R., Hajimirsadeghi H., Lucas C.: Discrete invasive weed optimization algorithm: Application to cooperative multiple task assignment of UAVs. IEEE Conference on Decision and Control, IEEE, Shanghai 2009, s. 1665-1670.
11. Giri R., Chowdhury A., Ghosh A., Das S., Abraham A., Snasel V.: A modified invasive weed optimization algorithm for training of feed-forward neural networks. IEEE International Conference on Systems Man and Cybernetics 2010, IEEE, Istanbul 2010, s. 3166-3173.
12. Griewank A. O.: Generalized descent for global optimization. Journal of Optimization Theory and Applications, Vol. 34, No. 1, Plenum Publishing Corporation, 1981, s. 11-39.
13. Ioannidis Y. E., Kang Y. C.: Randomized algorithms for optimizing large join queries. Proceedings of the ACM SIGMOD Conference on Management Data, Atlantic City, NJ, USA 1990, s. 312-321.
14. Josiński H., **Kostrzewa D.**, Michalczuk A., Świtoński A.: The exIWO metaheuristic for solving continuous and discrete optimization problems. The Scientific World Journal (**Impact Factor 1.730**), Vol. 2014, Article ID 831691, Hindawi Publishing Corporation, 2014.
15. Josiński H., **Kostrzewa D.**, Michalczuk A., Świtoński A., Wojciechowski K.: Feature Extraction and HMM-Based Classification of Gait Video Sequences for the Purpose of Human Indentification. Vision Based Systems for UAV Applications, SCI 481, Springer, Switzerland 2013.
16. Josiński H., Michalczuk A., **Kostrzewa D.**, Świtoński A., Wojciechowski K.: Heuristic method of feature selection for person reindentification based on gait motion capture data. Intelligent Information and Database Systems, Lecture Notes in Computer Science, Vol. 8398, Springer, 2014, s. 585-594.
17. Josiński H., Świtoński A., Jędrasiak K., **Kostrzewa D.**: Human Identification Based on Gait Motion Capture Data. Lecture Notes in Engineering and Computer Science, Vol. 2195, Hong Kong 2012, str. 507-510.
18. Josiński H., Świtoński A., Jędrasiak K., **Kostrzewa D.**: Human Identification Based on Tensor Representation of the Gait Motion Capture Data. IAENG Transactions on Electrical Engineering, Vol. 1 – Special Issue of the International MultiConference of Engineers and Computer Scientists 2012, World Scientific Publishing 2013, s. 111-122.
19. Josiński H., Świtoński A., Michalczuk A., **Kostrzewa D.**, Wojciechowski K.: Human Identification Based on Gait Video Sequences. International Conference on Computer Science and Engineering (ICCSE 2013), World Academy of Science, Engineering and Technology (WASET), Kuala Lumpur, 14-15.02.2013, s. 312-317.

20. Josiński H.: Model realizacji zapytania dla danych rozproszonych. Wysokowydajne sieci komputerowe, Tom 1, WKiŁ, Warszawa 2005.
21. Kang R.-G., Jung C.-Y.: The optimal solution of TSP using the new mixture initialization and sequential transformation method in genetic algorithm. Lecture Notes in Artificial Intelligence, Vol. 4099, Springer-Verlag, Berlin Heidelberg 2006, s. 1181-1185.
22. Kohonen T.: The self-organizing map. Neurocomputing, Vol. 21, Elsevier, 1998, s. 1-6.
23. **Kostrzewa D.**, Josiński H.: Evaluation of the exIWO Algorithm Based on the Travelling Salesman Problem. International Journal of Mathematical, Computational, Natural and Physical Engineering, Vol. 8, No. 9, WASET 2014, International Conference on Computing and Artificial Intelligence. Singapore 2014, s. 1175-1178.
24. **Kostrzewa D.**, Josiński H.: Metody przeszukiwania przestrzeni planów realizacji zapytań za pomocą algorytmu IWO. Studia Informatica, Vol. 31, No. 2A(89). Wydawnictwo Politechniki Śląskiej, Gliwice 2010.
25. **Kostrzewa D.**, Josiński H.: Ocena jakości strategii eksploracji przestrzeni poszukiwań dla problemu określenia kolejności realizacji złączeń. Studia Informatica, Vol.32, No. 2A (96). Wydawnictwo Politechniki Śląskiej, Gliwice, 2011, s. 37-46.
26. **Kostrzewa D.**, Josiński H.: Planowanie procesu scalania danych rozproszonych za pomocą algorytmu ewolucyjnego. Bazy danych: Rozwój metod i technologii, Tom 1, WKiŁ, Warszawa 2008.
27. **Kostrzewa D.**, Josiński H.: The comparison of an adapted evolutionary algorithm with the invasive weed optimization algorithm based on the problem of predetermining progress of distributed data merging process. Advances in Intelligent and Soft Computing: Man-Machine Interactions, Springer-Verlag, Berlin Heidelberg 2009.
28. **Kostrzewa D.**, Josiński H.: The exIWO metaheuristic – a recapitulation of the research on the join ordering problem. Communications in Computer and Information Science, Vol. 424, Springer, 2014, s. 10-19.
29. **Kostrzewa D.**, Josiński H.: The modified IWO algorithm for optimization of numerical functions. Lecture Notes in Computer Science, Vol. 7269, Springer-Verlag, Berlin Heidelberg 2012, s. 267-274.
30. **Kostrzewa D.**, Josiński H.: Using the Expanded IWO Algorithm to Solve the Traveling Salesman Problem. Proceedings of the 5th International Conference on Agents and Artificial Intelligence Barcelona 2013, Vol. 2. SCITEPRESS – Science and Technology Publications, Portugal 2013, str. 451-456.
31. **Kostrzewa D.**, Josiński H.: Verification of the search space exploration strategy based on the solutions of the join ordering problem. Advances in Intelligent and Soft Computing: Man-Machine Interactions 2, Springer-Verlag, Berlin Heidelberg 2011 s. 447-455.
32. **Kostrzewa D.**, Josiński H.: Zastosowanie algorytmu IWO do planowania procesu scalania danych rozproszonych. Studia Informatica, Vol. 30, No. 2A (83), Wydawnictwo Politechniki Śląskiej, Gliwice 2009.
33. **Kostrzewa D.**, Josiński H.: Zastosowanie algorytmu IWO do rozwiązania problemu komiwojażera. Technologie Przetwarzania Danych, Wydawnictwa Naukowo-Techniczne, Warszawa 2010.

34. Mehrabian A. R., Lucas C.: A novel numerical optimization algorithm inspired from weed colonization. Ecological Informatics, Vol. 1, Issue 4. Elsevier, 2006, s. 355-366.
35. Michalewicz Z.: Algorytmy genetyczne + struktury danych = programy ewolucyjne. WNT, Warszawa 2003.
36. Michalewicz Z., Fogel D. B.: Jak to rozwiązać, czyli nowoczesna heurystyka. WNT, Warszawa 2006.
37. Moerkotte G., Neumann T.: Analysis of two existing and one new dynamic programming algorithm for the generation of optimal bushy join trees without cross products. Proceedings of the 32nd VLDB '06 Conference, Seoul, Korea 2006, s. 930-941.
38. Molga M., Smutnicki C.: Test functions for optimization needs. http://www.zsd.ict.pwr.wroc.pl/files/docs/functions.pdf, 2005.
39. Mona Lisa TSP Challenge. http://www.tsp.gatech.edu/data/ml/monalisa.html (data ostatniej wizyty 10 sierpnia 2012r.).
40. Morzy T., Matysiak M., Salza S.: Tabu search optimization of large join queries. [w:] Jarke M. i in. (red.): Proceedings of the 4th International Conference on Extending Database Technology, Cambridge 1994, s. 309-322.
41. Rastrigin L. A.: External control by the method of random scanning. Automatika i Telemechanika, Vol. 21, No. 9, ZSRR 1960, s. 1264-1271.
42. Research Group Combinatorial Optimization, Ruprecht-Karls-Universität Heidelberg. http://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95/index.html (data ostatniej wizyty 5 sierpnia 2012r.).
43. Rosenbrock H. H.: An automatic method for finding the greatest or least value of a function. Computer Journal, Vol. 3, Issue 3, Oxford University Press, 1960, s. 175-184.
44. Sahraei-Ardakani M., Roshanaei M., Rahimi-Kian A., Lucas C.: A study of electricity market dynamics using invasive weed colonization optimization. IEEE Symposium on Computational Intelligence and Games 2008. IEEE, Perth, WA 2008, s. 276-282.
45. Scheufele W., Moerkotte G.: Optimal ordering of selections and joins in acyclic queries with expensive predicates. Interner Bericht, RWTH Aachen, Lehrstuhl für Informatik III, Aachen 1996.
46. Steinbrunn M., Moerkotte G., Kemper A.: Optimizing join orders. Technical Report MIP9307, Faculty of Mathematics, University of Passau, Germany 1993.
47. Swami A., Gupta A.: Optimization of large join queries. Proceedings of the 1988 ACM SIGMOD International Conference on Management of Data, ACM, New York, NY, USA 1988, s. 8-17.
48. Tao G., Michalewicz Z.: Inver-over operator for the TSP. Lecture Notes in Computer Science, Vol. 1498, Springer, Berlin Heidelberg 1998, s. 803-812.
49. Tow D.: SQL. Optymalizacja. Wydawnictwo Helion, Gliwice 2004.
50. Waas F., Pellenkoft A.: Join order selection – good enough s easy. Proceedings of the 17th British National Conference on Databases: Advances in Databases, BNCOD 17, Springer-Verlag, London, UK 2000, s. 51-67.
51. Xie X.-F., Zhang W.-J., Yang Z.-L.: Adaptive particle swarm optimization on individual level. 6th International Conference on Signal Processing, Vol. 2. IEEE, China 2002, s. 1215-1218.