STUDIA INFORMATICA 2008
Volume 29 Number 2A (76)

Marcin GORAWSKI, Kamil DOWLASZEWICZ
Politechnika Śląska, Instytut Informatyki

ANALIZA I ROZWINIĘCIE METOD OPTYMALIZACJI ZAPYTAŃ O PIERWSZYCH *K* PREFEROWANYCH LOKALIZACJI

Streszczenie. Artykuł przedstawia opis zapytań o pierwszych *k* preferowanych lokalizacji oraz przegląd opartych na R-drzewie istniejących metod ich realizacji. Przedstawiona zostaje także metoda optymalizacji procesu wykonywania zapytania oparta na poszerzonym opisie formalnym zapytań o pierwszych k preferowanych lokalizacji. Wszystkie opisane metody są następnie poddane analizie i przedstawiona jest ich charakterystyka. Artykuł wskazuje także algorytmy najbardziej efektywne w zależności od cech zapytania i konfiguracji danych.

Słowa kluczowe: zapytania o pierwszych *k* preferowanych lokalizacji, zapytania przestrzenne, metody optymalizacji

AN ANALYSIS AND EXTENSION OF TOP-K SPATIAL PREFERENCE QUERIES OPTIMIZATION METHODS

Summary. The paper presents a general description of top-k spatial preference queries and an overview of its existing, R-tree based, execution methods. It also introduces an optimization method based on a widened top-k spatial preference query description. All discussed techniques are then analyzed and their characteristics are presented together with the fields of their potential use.

Keywords: top-k spatial preference queries, spatial queries, optimization methods

1. Wstęp

Zapytania o pierwszych k preferowanych lokalizacji to interesująca klasa zapytań operujących na danych przestrzennych. Znajdują one obiekty o największej wartości względem wybranych cech innych obiektów znajdujących się w ich otoczeniu.

Zapytania takie mają szerokie pole zastosowań. Wykorzystywane być mogą w różnorodnych systemach wspierania podejmowania decyzji zarówno w dziedzinach takich jak rekomendowanie usług czy wyszukiwanie obiektów pod inwestycje, a także w dziedzinach takich jak zarządzanie kryzysowe.

Na zapytanie o pierwszych *k* preferowanych lokalizacji składa się wykonanie wielu typowych zapytań przestrzennych, takich jak wyszukiwanie obiektów znajdujących się w określonym regionie czy wyszukiwanie najbliższego sąsiada. Rozpatrywana klasa zapytań charakteryzuje się także wykorzystaniem wielu licznych zbiorów danych. Przedstawione powyżej cechy powodują, iż zapytania tego typu są wymagające zarówno pod względem liczby dostępów do indeksów, jak i obliczeniowo.

Kolejne rozdziały przedstawiają rozpatrywaną klasę zapytań oraz przegląd istniejących metod ich realizacji opartych na R-drzewie. Metody te wykorzystują techniki optymalizacji procesu wykonywania zapytań, mające na celu zmniejszenie liczby dostępów do indeksów przestrzennych i do danych, które mają znaczący wpływ na czas wykonania zapytania. Przedstawiony jest następnie zmodyfikowany opis formalny zapytania o pierwszych *k* preferowanych lokalizacji, biorący pod uwagę szersze spektrum zapytań. Na jego podstawie zaproponowana jest także kolejna technika umożliwiająca optymalizację procesu wykonywania rozpatrywanych zapytań. Wszystkie opisane metody są następnie poddane analizie.

2. Zapytania o pierwszych *k* preferowanych lokalizacji

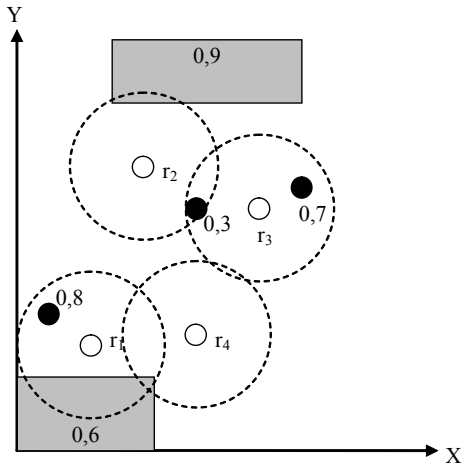
Zapytania o pierwszych *k* preferowanych lokalizacji to typ zapytań operujących na danych, które, oprócz atrybutów określających ich lokalizację, opisane są także atrybutami opisującymi ich cechy [1]. Zapytania tej klasy wyszukują w określonym zbiorze obiektów te elementy, które mają największą wartość względem wybranych cech innych obiektów znajdujących się w ich otoczeniu.

Definicja zapytania wskazuje nie tylko zbiór obiektów poddawany wartościowaniu, ale także zbiór cech opisujących inne obiekty, które decydują o wartości obiektów wyszukiwanych. Obiekty poddawane wartościowaniu w dalszej części artykułu nazywane będą obiektami wartościowanymi, a obiekty na podstawie cech, których wartość ta będzie wyznaczana, obiektami wartościującymi. Definicja zapytania określa także metodę wyboru obiektów wartościujących. Mogą być one zdefiniowane na przykład jako obiekty położone najbliżej obiektów wartościowanych lub też jako obiekty leżące w pewnej od nich odległości. W drugim z przedstawionych przypadków możliwa jest dalsza parametryzacja zapytania przez określenie metody wyznaczającej wartość wyszukiwanych obiektów, na podstawie obiektów wartoś-

ciujących znalezionych w rozpatrywanym regionie. Wartość obiektów jest więc subiektywna i może być różna dla różnych zapytań [1].

Przykładowym zapytaniem o pierwszych *k* preferowanych lokalizacji jest polecenie wyszukania budynków mieszkalnych, w pobliżu których znajdują się obszary leśne o wysokim zagrożeniu pożarowym oraz magazyny przechowujące materiały łatwopalne.

Rysunek 1 poglądowo ilustruje realizację takiego zapytania, w przypadku gdy metoda znajdująca obiekty mające wpływ na wartość obiektów wyszukiwanych, klasyfikuje do tego zbioru obiekty wartościujące znajdujące się w pewnej odległości od obiektów wartościowanych. Metoda ta w dalszej części artykułu nazywana będzie metodą zasięgu.



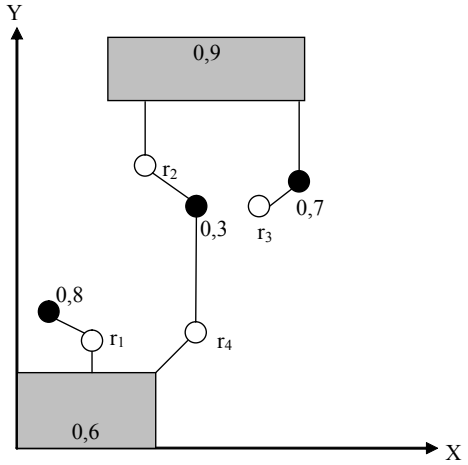
Rys. 1. Realizacja zapytania wykorzystującego metodę zasięgu
Fig. 1. Query execution using range search method

Białe okręgi na powyższym rysunku reprezentują obiekty wartościowane, czyli budynki mieszkalne. Okręgi czarne przedstawiają magazyny, a obszary szare reprezentują tereny zalesione. Na podstawie wybranych cech tych obiektów wyznaczana będzie wartość obiektów wyszukiwanych. Okręgi zaznaczone przerywaną linią przedstawiają otoczenie wartościowanych obiektów. Na wartość każdego z tych obiektów wpływają tylko te obiekty wartościujące, które należą do jego otoczenia.

Zakładając, iż spośród obiektów należących do otoczenia obiektu wartościowanego, wybierany jest ten, którego cecha określona w zapytaniu ma największą wartość, w wyniku wykonania zapytania wybrany zostanie obiekt r_1. Wynika to z faktu, iż jego sumaryczna wartość jest największa i wynosi $0,8 + 0,6 = 1,4$, podczas gdy wartość obiektu r_2 wynosi $0,3 + 0,9 = 1,2$, wartość obiektu r_3 wynosi $0,7 + 0 = 0,7$, a wartość obiektu r_4 wynosi $0 + 0,6 = 0,6$.

Rysunek 2 poglądowo ilustruje realizację opisanego wcześniej zapytania, w przypadku gdy metoda znajdująca obiekty, mające wpływ na wartość obiektów wyszukiwanych, klasyfikuje do tego zbioru obiekty wartościujące znajdujące się najbliżej obiektów wartościo-

wanych. Metoda ta w dalszej części artykułu nazywana będzie zapytaniem najbliższego sąsiada (zapytanie NN).



Rys. 2. Realizacja zapytania wykorzystującego metodę najbliższego sąsiada
Fig. 2. Query execution using nearest neighbor search method

Na powyższym rysunku obiekty wartościowane połączone są linią z najbliższymi im obiektami wpływającymi na ich wartość. W tym przypadku wynikiem zapytania będzie obiekt r_3, gdyż jego wartość wynosi 0,7 + 0,9 = 1,6, podczas gdy wartość obiektu r_1 wynosi 0,8 + 0,6 = 1,4, wartość obiektu r_1 wynosi 0,3 + 0,9 = 1,2, a wartość obiektu r_4 wynosi 0,3 + 0,6 = 0,9.

Przedstawione przykłady zapytań o pierwszych *k* preferowanych lokalizacji pokazują, iż wpływ na wynik zapytania mają nie tylko dane, ale także wybrany sposób wartościowania obiektów, który dobrany powinien być tak, aby zapytanie realizowane było w sposób odpowiedni dla danego zagadnienia.

Innym przykładem zapytań o pierwszych *k* preferowanych lokalizacji, związanym bardziej z zagadnieniami inwestycyjnymi, jest zapytanie znajdujące nieruchomości, w otoczeniu których istnieją najwyższej klasy biurowce oraz wysoko oceniane restauracje. Zapytania tego typu mogą też znajdować na przykład mieszkania, zlokalizowane w pobliżu restauracji, w których serwowane jest wegetariańskie jedzenie, w bliskiej odległości od przystanków komunikacji publicznej obsługiwanych przez wiele linii.

3. Stan aktualny

3.1. Opis formalny zapytania o pierwszych *k* preferowanych lokalizacji

Formalny opis zapytań o pierwszych *k* preferowanych lokalizacji przedstawiony w [1] mówi, iż zapytania te znajdują pierwszych *k* obiektów należących do zbioru obiektów war-

tościowanych D, których wartość jest największa. Dla każdego obiektu p ze zbioru D wartość względem zbiorów obiektów wartościujących $F_1,...,F_m$ zdefiniowana jest jako

$$t^{\theta}(p) = agg\left\{t_c^{\theta}(p) \,\middle|\, c \in [1,m]\right\} \tag{1}$$

We wzorze tym agg jest monotonicznym operatorem agregacji, a $t_c^{\theta}(p)$ oznacza wartość częściową obiektu p względem F_c, wyznaczoną na podstawie warunku sąsiedztwa θ. Typowym operatorem agregacji jest na przykład operator dodawania.

Dwiema podstawowymi metodami, które mogą być stosowane jako funkcja wartościująca $t_c^{\theta}(p)$, są:

A. Wartościowanie według zasięgu $t_c^{rng}(p)$.

B. Zwraca ono w zależności od wybranej metody wyznaczającej wartość otoczenia, największą, najmniejszą, lub średnią wartość cechy ω, obiektów s ze zbioru F_c, leżących w odległości mniejszej niż ε_c od obiektu p. W przypadku gdy w obszarze tym nie znajduje się żaden obiekt, zwracane jest 0.

C. Wartościowanie według najbliższego sąsiada $t_c^{nn}(p)$.

D. Zwraca ono wartość cechy ω, obiektu będącego najbliższym sąsiadem obiektu p, w zbiorze F_c.

3.2. Algorytmy

Realizacja zapytań o pierwszych *k* preferowanych lokalizacji wymaga znajdowania obiektów leżących w odpowiednio zdefiniowanym otoczeniu obiektów wartościowanych. Następnie na podstawie wybranej cechy znalezionych obiektów określana jest wartość obiektu wartościowanego.

Efektywne wyszukiwanie obiektów na podstawie ich lokalizacji możliwe jest przez wykorzystanie odpowiednich indeksów przestrzennych. W artykule tym uwaga skupiona została na metodach opartych na strukturze R-drzewa, która jest najpopularniejszą strukturą, wykorzystywaną do indeksowania danych przestrzennych.

R-drzewo jest dynamiczną, zrównoważoną strukturą drzewiastą, umożliwiającą efektywną realizację podstawowych typów zapytań przestrzennych, takich jak wyszukiwanie najbliższych sąsiadów czy znajdowanie obiektów leżących wewnątrz określonego regionu [3]. Umożliwia ona realizację zapytań zarówno na obiektach, których lokalizacja w przestrzeni określona jest jako punkt, jak i na obiektach, które zajmują pewien obszar. Struktura ta indeksuje tak zwane najmniejsze ograniczające prostokąty, które przybliżają obszar zajmowany przez dany obiekt. Są one następnie przechowywane na poziomie liści wraz z identyfikatorrem indeksowanego obiektu. Węzły niebędące liśćmi przechowują wskaźniki do węzłów będących ich dziećmi, oraz dla każdego z nich najmniejszy ograniczający prosto-

kąt, pokrywający wszystkie najmniejsze ograniczające prostokąty danego węzła dziecka. W ten sposób węzły na poziomach wyższych odpowiadają większym obszarom, a węzły na poziomach niższych obszarom mniejszym, bardziej uściślonym [4].

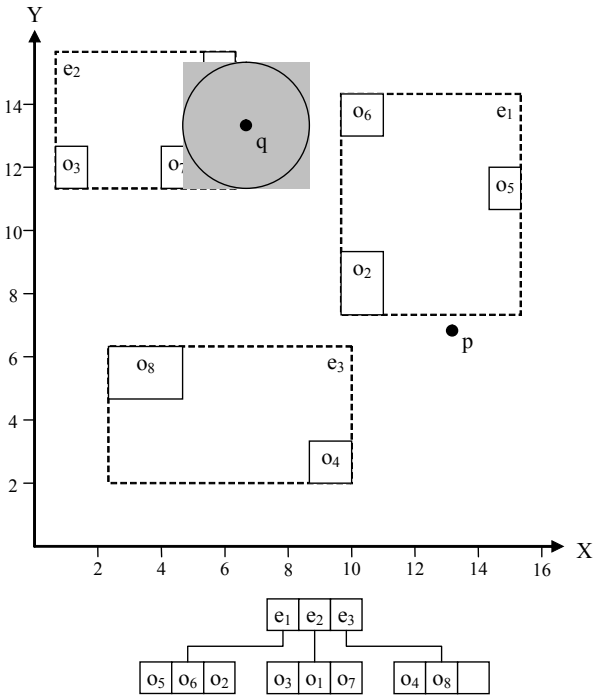
Podczas wyznaczania wartości częściowych obiektów wartościowanych, wykonywane być mogą oba, wspierane przez strukturę R-drzewa, podstawowe typy zapytań przestrzennych:

A. Zapytanie znajdujące obiekty należące do pewnego regionu

   Zapytanie tego typu znajduje wszystkie obiekty, mające chociaż jeden punkt wspólny z określonym regionem [3]. Przykładem takiego zapytania, przedstawionym na rysunku 3, jest polecenie wyszukania wszystkich obiektów znajdujących się w odległości mniejszej niż 4 od obiektu q. W tym przypadku regionem poszukiwań jest zaznaczony na szaro obszar wokół punktu q. Zapytanie takie uruchamiane jest na korzeniu drzewa, a następnie rekurencyjnie odwiedzane są te wpisy, których najmniejsze ograniczające prostokąty mają część wspólną z regionem zapytania. Wpis e_1 nie ma części wspólnej z rozpatrywanym regionem, jest więc odrzucany. Wpis e_2 ma część wspólną z regionem poszukiwań, więc algorytm zostaje ponownie wykonany na jego zawartości. Znalezione zostają obiekty o_1 i o_7 mające część wspólną z regionem poszukiwań. Wpis e_3, podobnie jak wpis e_1, nie ma części wspólnej z regionem poszukiwań, więc gałąź, którą zawiera, może zostać odrzucona. Wynikiem zapytania są obiekty o_1 i o_7.

B. Zapytanie znajdujące obiekty leżące najbliżej określonego obiektu

   Zapytanie tego typu znajduje wszystkie obiekty, o najmniejszej odległości od danego obiektu. Przykładem takiego zapytania, przedstawionym na rysunku 3, jest polecenie wyszukania obiektów leżących najbliżej obiektu p. W celu zrealizowania takiego zapytania wykorzystać można algorytm typu best-first [2]. Wykorzystuje on kolejkę priorytetową, która przechowuje wpisy uporządkowane rosnąco według ich odległości od obiektu p. Algorytm rozpoczyna działanie od korzenia i wstawia do kolejki priorytetowej znajdujące się w nim wpisy e_1, e_2 oraz e_3. Następnie z kolejki pobierany jest wpis o najmniejszej odległości od obiektu p. W rozpatrywanym przypadku jest to wpis e_1. Do kolejki wstawiane są zawierane przez niego wpisy o_5, o_6 i o_2, reprezentujące dane. Kolejne pobranie najbliższego wpisu z kolejki skutkuje znalezieniem obiektu o_2, będącego najbliższym sąsiadem obiektu p.



Rys. 3. Ilustracja realizacji zapytań przestrzennych z wykorzystanie R-drzewa
Fig. 3. An example of executing spatial queries on R-tree

W rozdziałach 3.2.1-3.2.3 pokrótce przedstawione są algorytmy, realizujące zapytania o pierwszych *k* preferowanych lokalizacji z wykorzystaniem R-drzewa.

3.2.1. Algorytm typu brute-force

Najprostszą metodą znalezienia obiektów o największej wartości jest wyznaczenie jej dla każdego z wartościowanych obiektów. Następnie jako wynik zwracanych jest *k* obiektów o największej wartości. Algorytm 1 przedstawia pseudokod algorytmu realizującego zapytania o pierwszych *k* preferowanych lokalizacji stosując opisane podejście. Wykorzystuje on kopiec W_k o porządku malejącym, służący do przechowywania *k* obiektów o największych wartościach. Po zakończeniu działania algorytmu kopiec ten zawiera obiekty będące wynikami zapytania. Jest to wersja algorytmu SP (ang. *Simple Probing Algorithm* SP) przedstawionego w [1], pozbawiona mechanizmu odrzucania obiektów.

Zasadniczą wadą tego algorytmu jest konieczność wyznaczania wartości wszystkich obiektów, a więc wykonywania zapytań przestrzennych dla każdego zbioru wartościującego, dla każdego obiektu wartościowanego.

Algorytm 1. Algorytm realizujący zapytania o pierwszych k preferowanych lokalizacji metodą brute-force

```
BRF(węzeł N)
1. Dla każdego e ∈ N:
2.     Jeżeli N jest liściem:
3.         Dla każdego F_c:
4.             Wyznacz t_c(e), korzystając z drzewa F_c
5.         Wyznacz t(e), na podstawie wartości częściowych t_c(e)
6.         Jeśli wartość t(e) jest większa od wartości najgorszego obiektu w W_k:
7.             Wstaw e do W_k
8.     W przeciwnym przypadku:
9.         Wykonaj BRF(e)
```

3.2.2. Algorytm SP

Algorytm SP przedstawiony w [1] oparty jest na założeniu, iż wartość względem każdej preferencji przedstawiona może zostać zakresie [0,1]. W przypadku takim zdefiniować można maksymalną możliwą wartość obiektu, dla którego nie są znane wszystkie wartości częściowe. Wykorzystując tę wielkość, możliwa jest optymalizacja przedstawionego wcześniej algorytmu, polegająca na przerwaniu obliczania wartości częściowych obiektów, których wartość nie może być większa od wartości najgorszego obiektu klasyfikowanego w danym momencie do zbioru wyników. Rozwiązanie to zmniejsza liczbę wykonywanych zapytań przestrzennych, a więc liczbę odwiedzanych węzłów i danych. Maksymalna możliwa wartość obiektu zdefiniowana została w [1] jako

$$t_+^{\theta}(p) = agg_{c=1}^{m} \begin{cases} t_c^{\theta}(p) \; jesli \; t_c^{\theta}(p) \; jest \; znane \\ 1 \; w \; przeciwnym \; przypadku \end{cases} \tag{2}$$

Algorytm 2 to pseudokod algorytmu SP, stosującego mechanizm, wykorzystujący opisaną własność, zwany dalej odrzucaniem.

Algorytm 2. Algorytm SP realizujący zapytania o pierwszych k preferowanych lokalizacji stosując odrzucanie

```
SP(węzeł N)
1. Dla każdego e ∈ N:
2.     Jeżeli N jest liściem:
3.         Dla każdego F_c:
4.             Jeśli t_+(e) > wartości γ najgorszego obiektu w W_k:
5.                 Wyznacz t_c(e), korzystając z drzewa F_c
6.         Jeśli wartość t(e) jest większa od wartości najgorszego obiektu w W_k:
7.             Wstaw e do W_k
8.     W przeciwnym przypadku:
9.         Wykonaj SP(e)
```

3.2.3. Algorytm GP

Wykorzystując specyfikę R-drzewa, w którym dane przechowywane są na poziomie liści, możliwa jest dalsza optymalizacja procesu wykonywania zapytań o pierwszych *k* preferowanych lokalizacji. Względna odległość pomiędzy obiektami należącymi do jednego liścia R-drzewa, w stosunku do odległości pomiędzy obiektami w całym drzewie, jest stosunkowo niewielka, gdyż wszystkie te obiekty należą do określonego regionu w przestrzeni. Jedno-

czesne wyznaczanie wartości takich obiektów powinno wymagać mniejszej liczby dostępów do indeksu ze względu na fakt, iż obiekty wartościujące z ich otoczenia także należą do tych samych lub sąsiednich liści indeksującego je R-drzewa. Algorytm GP (*Group Probing*), omówiony w [1], wykorzystuje opisaną właściwość. Jego pseudokod przedstawiony jest jako Algorytm 3.

Algorytm 3. Algorytm GP realizujący zapytania o pierwszych k preferowanych lokalizacji, wyznaczając wartości obiektów z jednego liścia jednocześnie

```
GP(węzeł N)
1.  Jeżeli N jest liściem
2.      Dla każdego e ∈ N:
3.          Wstaw e do zbioru V
4.      Dla każdego F_c:
5.          Wyznacz t_c(e) obiektów ∈ V, korzystając z drzewa F_c
6.          Dla każdego e ∈ V:
7.              Jeśli t+(e) < wartości γ najgorszego obiektu w W_k:
8.                  Usuń e ze zbioru V
9.      Dla każdego e ∈ V:
10.         Jeśli wartość t(e) jest większa od wartości najgorszego obiektu w W_k:
11.             Wstaw e do W_k
12. W przeciwnym przypadku:
13.     Dla każdego e ∈ N:
14.     Wykonaj GP(e)
```

4. Proponowane modyfikacje

4.1. Modyfikacja opisu formalnego zapytania o pierwszych *k* preferowanych lokalizacji

Definicja wartości obiektu wartościowanego, przedstawiona w [1], opisana w rozdziale 3.1 i zdefiniowana wzorem w równaniu 1 sugeruje, iż każda z wartości częściowych wyznaczana jest względem innego zbioru obiektów wartościujących. Wynika to z faktu określenia wartości względem zbioru obiektów a nie cechy obiektów tego zbioru. Zapytanie zdefiniowane może być na podstawie więcej niż jednej cechy obiektów należących do jednego zbioru, a wartości częściowe obiektów, wyznaczane względem tych cech, mogą i najczęściej będą się różnić. Celem przedstawionego poniżej opisu zapytania jest przejrzyste przedstawienie możliwości definiowania wszystkich zapytań omawianej kategorii.

Zapytania o pierwszych *k* preferowanych lokalizacji znajdują pierwszych *k* obiektów należących do zbioru obiektów wartościowanych D, których wartość jest największa. Dla każdego obiektu p należącego do wartościowanego zbioru D wartość względem atrybutów $c_1,c_2,c_3..,c_n$ obiektów zbiorów $F_1,...,F_m$, na których oparte są preferencje $f_1,...,f_n$, zdefiniowana jest jako

$$t^{\theta}(p) = agg\left\{t_c^{\theta}(p) \,\middle|\, c \in [1,n]\right\},\ gdzie\ n \geq m \tag{3}$$

W dalszej części artykułu zapytaniami o pierwszych *k* preferowanych lokalizacji nazywane będą zapytania zgodne z powyższym opisem. Wszystkie implementacje algorytmów realizujących te zapytania, wykorzystywane podczas tworzenia tego artykułu, umożliwiały wykonywanie zapytań o przedstawionej charakterystyce.

4.2. Optymalizacja oparta na podobnych preferencjach

Modyfikacja opisu zapytania o pierwszych *k* preferowanych lokalizacji, zakładająca możliwość definiowania wielu preferencji opartych na jednym zbiorze danych, wpłynęła na opracowanie metody, która, w przypadku gdy preferencje takie spełniają pewne warunki, umożliwia optymalizację procesu wykonywania zapytania. Warunki te to oparcie na tym samym zbiorze obiektów wartościujących oraz wykorzystanie tej samej metody wyznaczającej otoczenie obiektu. Dodatkowo, w przypadku gdy metodą wyznaczania otoczenia jest metoda zasięgu, konieczne jest, aby jej parametr określający maksymalną odległość sąsiedniego obiektu wartościującego od obiektu wartościowanego posiadał tę samą wartość. Preferencje spełniające opisane warunki nazwane są preferencjami podobnymi.

Wartości obiektów względem preferencji podobnych, ze względu na fakt, iż spełniają opisane warunki, wyznaczone mogą być podczas jednego przejścia struktury indeksującej obiekty wartościujące. Umożliwia to zmniejszenie liczby indeksów i danych, jakie muszą zostać odwiedzone podczas realizacji zapytania.

Dodatkowo, gdy algorytm stosuje odrzucanie, wykonywana jest dodatkowa operacja. Wartości względem grup podobnych preferencji obliczane są przed wartościami względem pojedynczych preferencji. Co więcej, pierwszeństwo mają najbardziej liczne grupy podobnych preferencji. W ten sposób najpierw obliczane są składniki stanowiące największą część wartości obiektu. Umożliwia to wcześniejsze odrzucanie obiektów przy wykonaniu mniejszej liczby dostępów do indeksu i danych.

5. Implementacja

Implementacja stworzonej aplikacji, realizującej zapytania o pierwszych *k* preferowanych lokalizacji bazuje na bibliotece *Spatial Index* Mariosa Hadjieleftheriou [11], która udostępniona na licencji LGPL służy do tworzenia indeksów przestrzennych. Wykorzystana wersja biblioteki określa podstawowe interfejsy oraz zawiera implementację mechanizmów przechowywania danych w pamięci i na dysku. Zawiera ona także szczegółową implementację R*-drzewa, wykorzystywanego podczas realizacji zapytań o pierwszych *k* preferowanych lokalizacji.

5.1. Algorytmy

Algorytmy realizujące zapytania o pierwszych *k* preferowanych lokalizacji wymagają wyznaczenia wartości obiektów względem określonych cech innych obiektów, znajdujących się w ich otoczeniu. Algorytmy realizujące zapytania o pierwszych *k* preferowanych lokalizacji przez wyznaczanie wartości obiektów pojedynczo korzystają, w tym celu, w zależności od metody określającej otoczenie obiektu z algorytmów NNValue i RNGValue, przedstawionych jako algorytmy 4 i 5. Z kolei, algorytmy realizujące zapytania o pierwszych *k* preferowanych lokalizacji przez jednoczesne wyznaczanie wartości grup obiektów korzystają z algorytmów GroupNNValue, oraz GroupRNGValue, przedstawionych jako algorytmy 6 i 7. Poniższe pseudokody przedstawiają implementacje algorytmów wyznaczających wartości częściowe obiektów, oparte na ogólnym opisie ich działania przedstawionym w [1]. Dysponują one dodatkowo możliwością jednoczesnego wyznaczania wartości względem podobnych preferencji.

Algorytm 4. Algorytm NNValue wyznaczania wartości częściowych pojedynczych obiektów dla metody najbliższego sąsiada

```
NNValue(Korzeń drzewa obiektów wartościujących R, obiekt wartościowany p, zbiór
preferencji podobnych S)
1. Stwórz kolejkę priorytetową Q, przechowującą wpisy w porządku opartym na ich
   odległości od p
2. Wstaw R do Q
3. Dopóki nie znaleziono najbliższego sąsiada, lub kolejka Q jest pusta:
4.     Pobierz pierwszy element e z kolejki Q
5.     Jeżeli e jest obiektem:
6          Dla każdej f_c ∈ S: [jednoczesne wyznaczanie wartości częściowych
           obiektu opartych na preferencjach podobnych]
7              Wyznacz t_c(p) na podstawie cechy c obiektu e, określonej przez f_c,
               oraz aktualnej wartości t_c(p)
8.         Jeśli aktualnie pierwszy wpis w Q jest bardziej odległy od p niż e:
9.             Oznacz fakt znalezienia najbliższego sąsiada
10.    W przeciwnym przypadku, jeżeli e jest liściem:
11.        Do Q wstaw obiekt wskazywany przez e
12.    W przeciwnym przypadku, jeżeli e jest węzłem wewnętrznym:
13.        Dla każdego g ∈ e:
14.            Do Q wstaw g
```

Algorytm 5. Algorytm RNGValue wyznaczania wartości częściowych pojedynczych obiektów dla metody zasięgu

```
RNGValue(Węzeł drzewa obiektów wartościujących N, obiekt wartościowany p,
Odległość d, zbiór preferencji podobnych S)
1. Dla każdego e ∈ N:
2.     Jeśli N jest liściem:
3.         Jeśli odległość najmniejszego ograniczającego prostokąta e od p ≤ d:
4.             Pobierz obiekt g wskazywany przez e
5.             Jeśli odległość g od p ≤ d:
6.                 Dla każdej f_c ∈ S: [jednoczesne wyznaczanie wartości
                   częściowych obiektu opartych na preferencjach podobnych]
7.                     Wyznacz t_c(p) na podstawie cechy c obiektu e, określonej
                       przez f_c, oraz aktualnej wartości t_c(p)
8.     W przeciwnym przypadku:
9.         Jeśli odległość e od p ≤ d:
10.            Wykonaj RNGValue(e, p, d, S)
```

Algorytm 6. Algorytm GroupNNValue jednoczesnego wyznaczania wartości częściowych wielu obiektów dla metody najbliższego sąsiada

```
GroupNNValue(Korzeń drzewa obiektów wartościujących R, Zbiór obiektów
   wartościowanych V, zbiór preferencji podobnych S)
1. Na podstawie lokalizacji obiektów ze zbioru V, wyznacz centroid C
2. Stwórz zbiór T przechowujący obiekty, dla których nie znaleziono dotychczas
   najbliższego sąsiada i wstaw do niego wszystkie obiekty p ∈ V
3. Stwórz kolejkę priorytetową Q, przechowującą wpisy w porządku opartym na ich
   odległościach od C
4. Do Q wstaw R
5. Dopóki T = Ø lub Q = Ø:
6.     Pobierz pierwszy element e z Q
7.     Jeżeli e jest obiektem:
8.         Dla każdego p ∈ T: [jednoczesne wyznaczanie wartości częściowych
           obiektów z jednego liścia]
9.             Jeżeli dotychczas znaleziona najmniejsza odległość p od obiektu
               wartościującego ≥ odległość p od e:
10.                Przypisz najmniejszej dotychczas znalezionej odległości p od
                   obiektu wartościującego wartość odległości p od e.
11.                Dla każdej f_c ∈ S: [jednoczesne wyznaczanie wartości
                   częściowych obiektu opartych na preferencjach podobnych]
12.                    Wyznacz t_c(p) na podstawie cechy c obiektu e, określonej
                       przez f_c, oraz aktualnej wartości t_c(p)
13.            Jeżeli odległość aktualnie pierwszego elementu w Q od C > od sumy
               odległości p od C i aktualnie znanej najmniejszej odległości p od
               obiektu wartościującego:
14.                Usuń p z T.
15.    W innym przypadku, jeśli e jest liściem:
16.        Do Q wstaw obiekt wskazywany przez e
17.    W innym przypadku, jeśli e jest węzłem wewnętrznym:
18.        Dla każdego g ∈ e:
19.            Wstaw g do Q.
```

Algorytm 7. Algorytm GroupRNGValue jednoczesnego wyznaczania wartości częściowych wielu obiektów dla metody zasięgu

```
GroupRNGValue(Węzeł drzewa obiektów wartościujących N, Zbiór obiektów
wartościowanych V, Odległość d, zbiór preferencji podobnych S)
1. Dla każdego e ∈ N:
2.     Jeśli N jest liściem:
3.         Dla każdego p ze zbioru V wykonaj: [jednoczesne wyznaczanie wartości
           częściowych obiektów z jednego liścia]
4.             Jeśli odległość najmniejszego ograniczającego prostokąta e od p
               ≤ d:
5.                 Pobierz obiekt g wskazywany przez e:
6.                     Jeśli odległość g od p ≤ d:
7.                         Dla każdej f_c ∈ S: [jednoczesne wyznaczanie wartości
                           częściowych obiektu opartych na preferencjach podobnych]
8.                             Wyznacz t_c(p) na podstawie cechy c obiektu g,
                               określonej przez f_c, oraz aktualnej wartości t_c(p)
9.     W przeciwnym przypadku:
10.        Dla każdego p ∈ V:
11.        Jeśli odległość najmniejszego ograniczającego prostokąta p od e ≤ d:
12.            Wykonaj GroupRNGValue(e, V, d, S)
13.            Wyjdź z najbardziej wewnętrznej pętli
```

Zestawienie wszystkich zaimplementowanych algorytmów realizujących zapytania o pierwszych *k* preferowanych lokalizacji, wraz z wykorzystywanymi przez nie mechanizmami optymalizacji przedstawia tabela 1. Algorytm 8 przedstawia pseudokod algorytmu wykorzystującego wszystkie techniki optymalizacji. Wykorzystanie algorytmów wyznaczających wartości częściowe zaznaczone zostało pogrubioną czcionką.

Tabela 1

Zestawienie algorytmów i wykorzystywanych przez nie mechanizmów optymalizacji

Nazwa	Odrzucanie obiektów	Wyznaczanie wartości obiektów z jednego liścia jednocześnie	Wyznaczanie wartości względem podobnych preferencji jednocześnie
L0P0S0	nie	nie	nie
L0P1S0	tak	nie	nie
L0P0S1	nie	nie	tak
L0P1S1	tak	nie	tak
L1P0S0	nie	Tak	nie
L1P1S0	tak	Tak	nie
L1P0S1	nie	Tak	tak
L1P1S1	tak	Tak	tak

Algorytm 8. Algorytm L1P1S1

```
L1P1S1 (korzeń drzewa obiektów wartościowanych R, zbiór preferencji zapytania P,
   wartość parametru k)
1. Stwórz zbiór G zawierający zbiory S preferencji podobnych
2. Na bazie zbioru G stwórz listę I zawierającą zbiory S w porządku malejącym
   według ich liczności
3. Wartości odrzucania x przypisz 0
4. Stwórz kopiec o porządku malejącym D służący do przechowywania obiektów
   uporządkowanych według ich wartości
5. Dla każdego liścia L drzewa R:
6.     Stwórz zbiór V i wstaw do niego obiekty p z liścia L
7.     Jeśli V ≠ Ø, dla każdego kolejnego S z listy I:
8.        Jeśli preferencje ∈ S wyznaczają otoczenie metodą zasięgu:
9.           Wyznacz wartości częściowe t_d(p) obiektów p ∈ V wykonując
             GroupRNGValue(korzeń drzewa danych wykorzystywanych przez
             preferencje zbioru S, V, parametr odległości metody zasięgu
             preferencji zbioru S, S) [jednoczesne wyznaczanie wartości
             częściowych obiektów z jednego liścia, jednoczesne wyznaczanie
             wartości częściowych obiektu opartych na preferencjach podobnych]
10.       W innym przypadku, jeśli preferencje ∈ S wyznaczają otoczenie metodą
          najbliższego sąsiada:
11.          Wyznacz wartości częściowe t_d(p) obiektów p ∈ V wykonując
             GroupNNValue(korzeń drzewa danych wykorzystywanych przez
             preferencje zbioru S, V, S) [jednoczesne wyznaczanie wartości
             częściowych obiektów z jednego liścia, jednoczesne wyznaczanie
             wartości częściowych obiektu opartych na preferencjach podobnych]
12.       Dla każdego obiektu p ze zbioru V wykonaj:
13.          Aktualizuj maksymalną możliwą wartość obiektu p, odejmując
             maksymalną możliwą wartość względem liczby preferencji zbioru S i
             dodając wyznaczoną wartość t_d(p)
14.          Jeżeli t+(p) ≤ x:
15.             Usuń p z V [odrzucanie obiektów]
16.    Dla każdego obiektu p ∈ V:
17.       Jeśli t(p) > x:
18.          Wstaw p do D
19.          Jeśli D zawiera więcej niż k elementów:
20.             Usuń pierwszy element kopca D
21.             Przypisz x wartość pierwszego obiektu w kopcu D, a więc
                najgorszego obiektu klasyfikującego się aktualnie do zbioru
                wyników
```

6. Wyniki

6.1. Wpływ charakteru zapytania

Celem tego testu było poznanie wpływu charakteru zapytania na proces jego wykonywania za pomocą wybranego algorytmu. Wykorzystane zostały algorytmy korzystające z kombinacji wszystkich opisanych technik optymalizacji, przedstawione w tabeli 1. W celu zbadania ich właściwości wykonano za ich pomocą osiem zapytań o różnej specyfice. Tabela 2 przedstawia charakter wykonywanych zapytań.

Każdy ze zbiorów obiektów składał się z 10000 losowo rozmieszczonych obiektów w przestrzeni o rozmiarze 1000 × 1000. Komputer wykorzystany podczas tego testu posiadał procesor Intel Pentium M 1,7GHz oraz 1,5GB pamięci RAM.

Tabela 2

Charakter wykonywanych zapytań

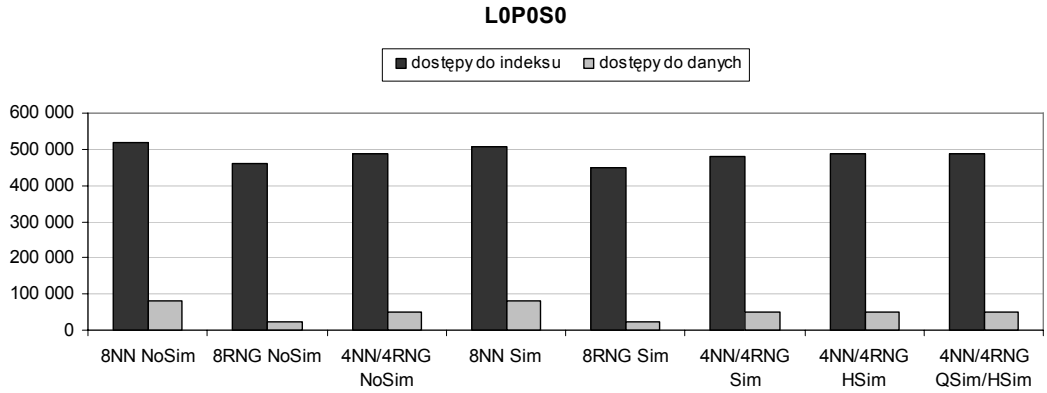
Nr	Nazwa zapytania	Liczba pref. NN	Liczba pref. RNG	Liczba podobieństw pref. NN	Liczba podobieństw pref. RNG
1	8NN NoSim	8	0	0	0
2	8RNG NoSim	0	8	0	0
3	4NN/4RNG NoSim	4	4	0	0
4	8NN Sim	8	0	8	0
5	8RNG Sim	0	8	0	8
6	4NN/4RNG Sim	4	4	4	4
7	4NN/4RNG HSim	4	4	2	2
8	4NN/4RNG QSim/HSim	4	4	1	2

Zapytania różnią się przede wszystkim wykorzystywanymi metodami wyznaczania otoczenia obiektów. Zapytania numer 1 i 4 składają się tylko z preferencji typu NN, określających otoczenie obiektu za pomocą metody najbliższego sąsiada. Zapytania numer 2 i 5 składają się tylko z preferencji typu RNG, określających otoczenie obiektu za pomocą metody zasięgu. Pozostałe zapytania zawierają preferencje obu typów.

Kolejną różnicą pomiędzy zapytaniami jest podobieństwo ich preferencji. Zapytania numer 1, 2 i 3 nie zawierają preferencji uznawanych za podobne. Zapytania numer 4,5 i 6 składają się tylko z preferencji, wzajemnie do siebie podobnych, a preferencje zapytań numer 7 i 8 charakteryzują się tym, iż niektóre spośród nich są do siebie podobne.

Rysunek 4 przedstawia liczbę dostępów do indeksu i danych koniecznych do wykonania zapytań o różnej charakterystyce z wykorzystaniem algorytmu L0P0S0. Pomimo iż algorytm

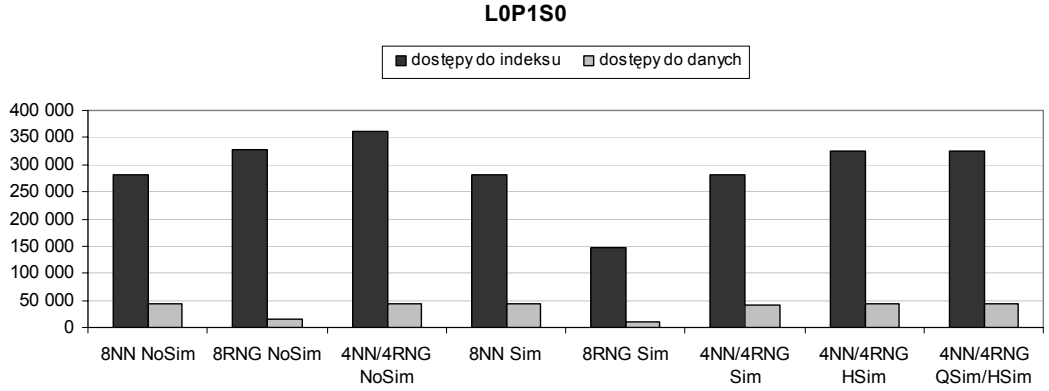
ten nie wykorzystuje żadnej z możliwych opcji optymalizacji procesu wykonywania zapytania, widoczny jest wpływ definicji zapytania na liczbę dostępów do indeksu i danych. Różnice występują głównie pomiędzy zapytaniami, których preferencje definiują otoczenie obiektu w inny sposób. Ze względu na fakt, iż wykorzystywana w zapytaniach odległość metody zasięgu jest stosunkowo mała, zapytania, których preferencje oparte są na tej metodzie, wymagały mniejszej liczby dostępów niż preferencje oparte na metodzie najbliższego sąsiada, która zawsze wymaga znalezienia jednego obiektu wartościującego dla każdego obiektu wartościowanego.



Rys. 4. Liczba dostępów do indeksu i danych dla algorytmu L0P0S0
Fig. 4. Number of index and data accesses for L0P0S0 algorithm

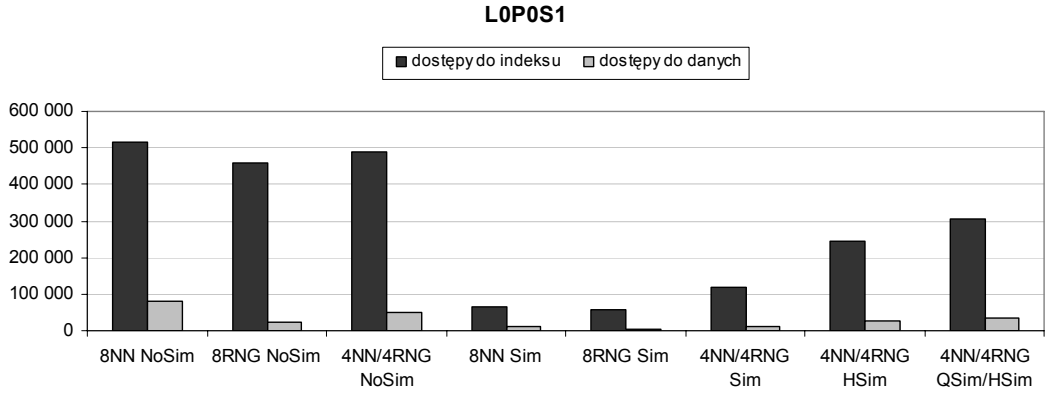
Rysunek 5 przedstawia liczbę dostępów do indeksu i do danych koniecznych do wykonania zapytań o różnej charakterystyce z wykorzystaniem algorytmu L0P1S0. Algorytm ten wykorzystuje odrzucanie. W porównaniu do wyników uzyskanych przez algorytm L0P0S0 widoczne jest ograniczenie liczby dostępów wykonywanych podczas realizacji zapytań. Zauważyć można także znacznie mniejszą liczbę dostępów koniecznych do wykonania zapytania 8RNG Sim. Wszystkie jego preferencje są preferencjami podobnymi, czyli zdefiniowanymi na podstawie tego samego zbioru danych. Wpływa to w pewnym stopniu na wykorzystywaną przez rozpatrywany algorytm technikę odrzucania. Staje się ona nieco bardziej efektywna, gdyż rozpatrywana jest stała grupa lokalizacji. Przykładem może być obiekt, który w swoim otoczeniu nie ma względem innego zbioru danych żadnego obiektu wartościującego. W takim przypadku jego wartość względem każdej preferencji podobnej równa będzie zero, co spowoduje szybkie jego odrzucenie. Z kolei obiekty, w których otoczeniu znajduje się więcej obiektów wartościujących, z większym prawdopodobieństwem będą miały większą wartość. Wynika to z faktu, iż względem każdej preferencji ich wartość wynosi, dla rozpatrywanych zapytań, maksimum spośród cech obiektów należących do ich otoczenia. Zapytania zawierające podobne preferencje określające otoczenie metodą najbliższego sąsiada nie powodują analogicznego, dodatkowego spadku liczby dostępów,

gdyż wymaga ona bezwarunkowego znalezienia jednego najbliższego sąsiada dla każdego obiektu wartościowanego.



Rys. 5. Liczba dostępów do indeksu i danych dla algorytmu L0P1S0
Fig. 5. Number of index and data accesses for L0P1S0 algorithm

Rysunek 6 przedstawia liczbę dostępów do indeksu i do danych koniecznych do wykonania zapytań o różnej charakterystyce z wykorzystaniem algorytmu L0P0S1. Algorytm ten oblicza wartości podobnych preferencji jednocześnie. Na wykresie widać wyraźnie, iż w przypadku, gdy zapytanie zawiera podobne preferencje, liczba dostępów do indeksu i danych wyraźnie spada. Zapytanie złożone z ośmiu podobnych preferencji, zgodnie z przewidywaniami, wymaga w przybliżeniu osiem razy mniej dostępów niż zapytania złożone z ośmiu różnych preferencji. Istotny jest jednak fakt, iż spadek liczby dostępów dotyczy jedynie zapytań o określonym charakterze. Zapytania oparte na różnych preferencjach wymagają takiej samej liczby dostępów, jaka jest potrzeba podczas wykonywania ich za pomocą algorytmu L0P0S0.

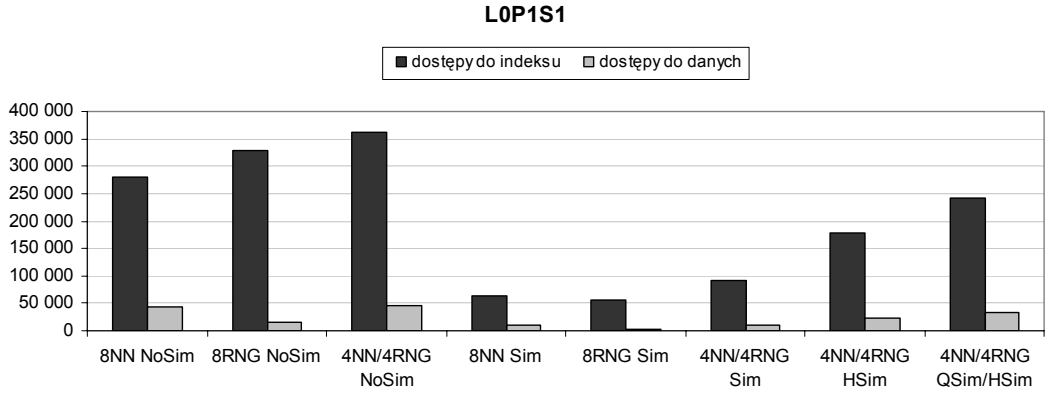


Rys. 6. Liczba dostępów do indeksu i danych dla algorytmu L0P0S1
Fig. 6. Number of index and data accesses for L0P0S1 algorithm

Rysunek 7 przedstawia liczbę dostępów do indeksu i do danych koniecznych do wykonania zapytań o różnej charakterystyce z wykorzystaniem algorytmu L0P1S1. Algorytm ten wykorzystuje zarówno odrzucanie, jak i jednoczesne wyznaczanie wartości częściowych

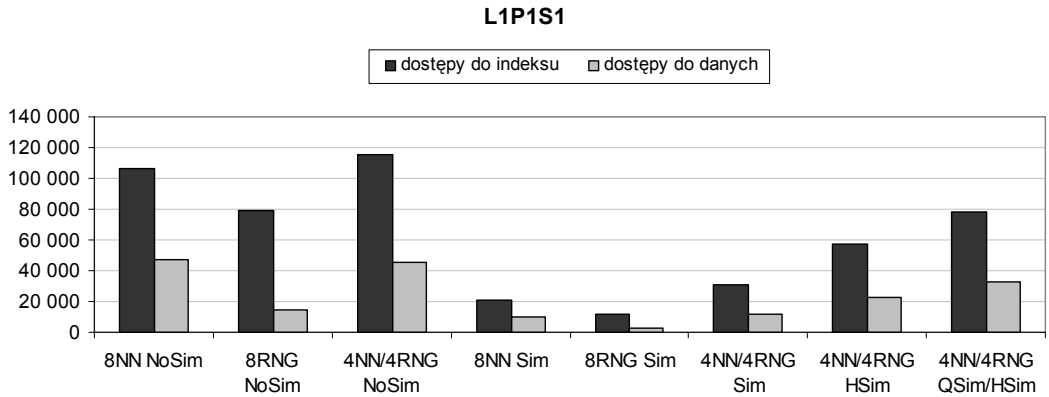
podobnych preferencji. Widać na nim, iż możliwe jest wykorzystanie obu metod optymalizacji jednocześnie i prowadzi to do uzyskania najlepszych wyników.

Algorytm ten umożliwia zarówno wyraźne zmniejszenie liczby dostępów dla zapytań opartych na podobnych preferencjach, jak i zmniejszenie liczby dostępów dla zapytań, które nie mogą skorzystać z tej opcji.



Rys. 7. Liczba dostępów do indeksu i danych dla algorytmu L0P1S1
Fig. 7. Number of index and data accesses for L0P1S1 algorithm

Kolejną metodą zmniejszającą liczbę dostępów do indeksu i danych podczas realizowania zapytań o pierwszych *k* preferowanych lokalizacji jest jednoczesne wyznaczanie wartości obiektów pochodzących z jednego liścia R-drzewa. Rysunek 8 przedstawia wyniki uzyskane przez wykorzystujący tę metodę algorytm L1P1S1. Wykorzystuje on wszystkie metody zmniejszania liczby dostępów, czyli także odrzucanie, oraz wyznaczanie wartości podobnych preferencji jednocześnie. Widać, iż jednoczesne obliczanie wartości obiektów pochodzących z jednego liścia pozwala zmniejszyć liczbę dostępów do indeksu w przypadku wykonywania zapytania o dowolnym charakterze.

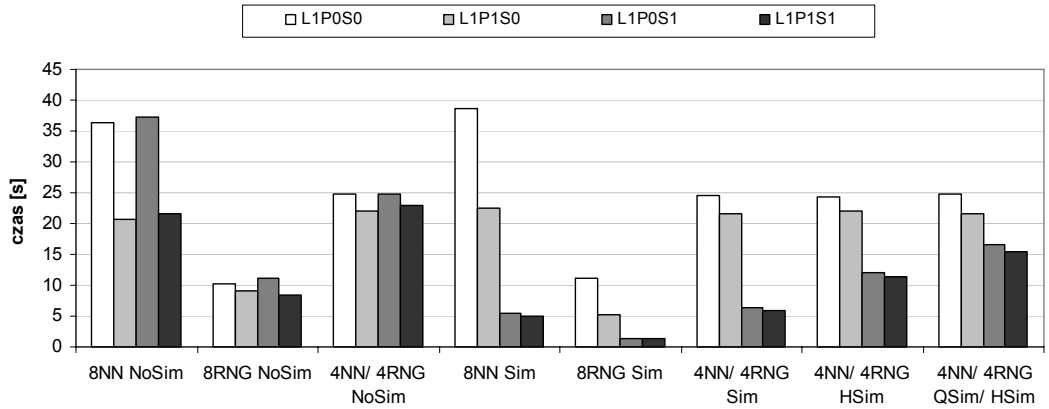


Rys. 8. Liczba dostępów do indeksu i danych dla algorytmu L1P1S1
Fig. 8. Number of index and data accesses for L1P1S1 algorithm

Przedstawione wykresy pozwalają stwierdzić, iż charakter zapytania ma znaczny wpływ na proces jego wykonywania. Liczba dostępów podczas wykonywania dwóch pozornie podobnych zapytań za pomocą tego samego algorytmu może się znacznie różnić. Cechami,

które mają największy wpływ na proces realizacji zapytań, są metoda wyznaczania otoczenia obiektów wartościowanych oraz wzajemne podobieństwo preferencji.

Wyniki uzyskane podczas prowadzonych testów potwierdzają, iż wprowadzony mechanizm optymalizacji można wykorzystywać łącznie z istniejącymi metodami zwiększającymi efektywność realizacji zapytań o pierwszych *k* preferowanych lokalizacji, a realizacja taka umożliwia dalszą optymalizację procesu wykonywania zapytania. Wniosek ten potwierdza rysunek 9, przedstawiający czas wykonywania zapytań za pomocą algorytmów jednocześnie wyznaczających wartości obiektów pochodzących z jednego liścia R-drzewa. Widać na nim, iż algorytm stosujący połączenie wszystkich metod optymalizacji realizuje każde z zapytań w czasie porównywalnym z najbardziej optymalną metodą dla danego zapytania.



Rys. 9. Czas wykonywania zapytań za pomocą różnych algorytmów
Fig. 9. Time required to execute queries using different algorithms

6.2. Wpływ charakteru danych

Celem tego eksperymentu było poznanie wpływu charakteru danych na zapytanie wykonywane za pomocą różnych algorytmów. Pozwoli to zweryfikować, czy algorytm zachowuje się podobnie dla zbiorów danych o różnej charakterystyce.

Podczas testu, dla różnych konfiguracji danych, uruchamiane było zapytanie 4NN-/4RNG Qsim/HSim, przedstawione w rozdziale 6.1. Ze względu na fakt, iż jedynie niektóre jego preferencje uznane mogą być za podobne, oraz korzystają one zarówno z metody zasięgu, jak i metody najbliższego sąsiada, charakteryzuje się ono umiarkowaną podatnością na wszystkie metody optymalizacji. Każde z zapytań wykonane zostało przez osiem algorytmów znajdujących pierwszych *k* obiektów o największej wartości względem preferencji zdefiniowanych w zapytaniu. Komputer wykorzystany podczas tego testu posiadał procesor Pentium M 1,7 GHz oraz 1,5 GB pamięci RAM.

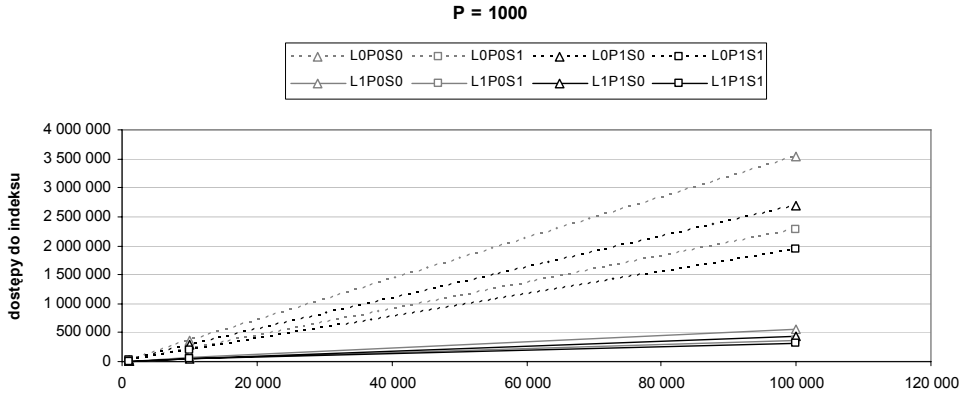
Konfiguracje danych wykorzystane podczas tego testu przedstawione są w tabeli 3.

Tabela 3

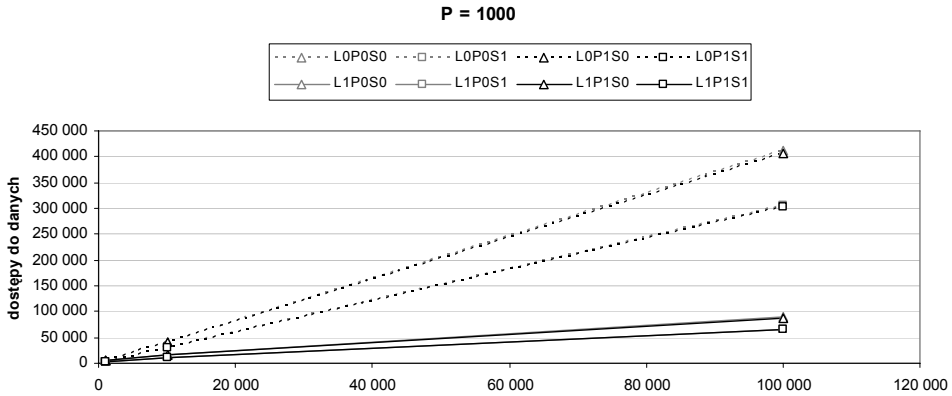
Wykorzystane konfiguracje danych

Nazwa konfiguracji	Liczność zbioru danych wartościowanych	Liczność zbiorów danych wartościujących	Powierzchnia
TMIN PMIN	1 000	1 000	1000 x 1000
TMED PMIN	10 000	1 000	1000 x 1000
TMAX PMIN	100 000	1 000	1000 x 1000
TMIN PMED	1 000	10 000	1000 x 1000
TMED PMED	10 000	10 000	1000 x 1000
TMAX PMED	100 000	10 000	1000 x 1000
TMIN PMAX	1 000	100 000	1000 x 1000
TMED PMAX	10 000	100 000	1000 x 1000
TMAX PMAX	100 000	100 000	1000 x 1000

Rysunki 10 i 11 przedstawiają odpowiednio liczbę dostępów do indeksu i do danych w przypadku wykonywania zapytań opartych na zbiorach danych wartościujących o liczności P = 1000. W przypadku tym, wraz ze wzrostem liczby obiektów wartościowanych, liczba dostępów do indeksu i danych rosną dużo szybciej dla algorytmów niekorzystających z metody jednoczesnego obliczania wartości obiektów pochodzących z jednego liścia. Zauważyć należy, iż liczba dostępów do węzłów i danych w przypadku zastosowania metody jednoczesnego wyznaczania wartości obiektów z jednego liścia rośnie logarytmicznie, podczas gdy dla algorytmów niestosujących tej metody wzrost liczby odwiedzonych węzłów jest liniowy. Wynika to z faktu, iż przy stałej powierzchni oraz stałej pojemności liści R-drzewa wzrost liczby obiektów wartościujących powoduje, iż obiekty należące do jednego liścia są obiektami bliższymi sobie w przestrzeni, co z kolei wpływa na zmniejszenie liczby węzłów, które muszą zostać odwiedzone podczas wyznaczania ich wartości.



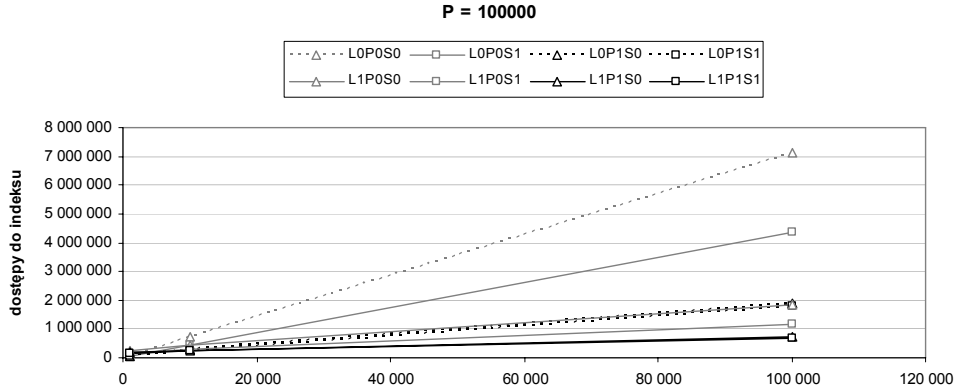
Rys. 10. Dostępy do indeksu przy stałej liczbie obiektów wartościujących P = 1000
Fig. 10. Number of node accesses when the number of feature objects equals P = 1000



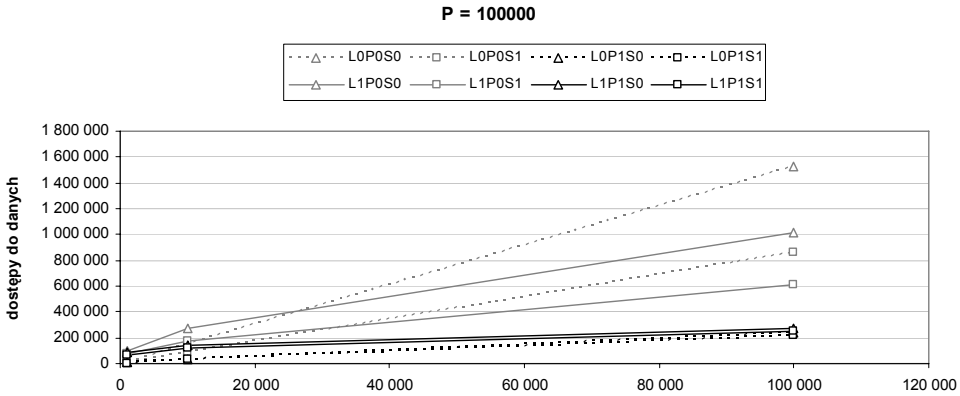
Rys. 11. Dostępy do danych przy stałej liczbie obiektów wartościujących P = 1000
Fig. 11. Number of data accesses when the number of feature objects equals P = 1000

Rysunki 12 i 13 przedstawiają liczbę dostępów do indeksu i do danych w przypadku, gdy zbiory danych wartościujących składają się z 100000 obiektów. Na tych wykresach wyraźnie widać, iż w przypadku, gdy zbiory danych wartościujących są bardziej liczne od zbioru danych wartościowanych, metody jednocześnie wyznaczające wartości wielu obiektów stają się mniej efektywne niż metody obliczające wartość każdego obiektu z osobna. Powodem tej sytuacji jest fakt analogiczny do tego, który powoduje, iż metody te są lepsze, gdy zbiory danych wartościowanych są bardziej liczne od zbiorów danych wartościujących. W sytuacji gdy powierzchnia, na której rozmieszczone są zarówno obiekty wartościowane, jak i obiekty wartościujące, jest taka sama, oraz pojemność liścia R-drzew jest porównywalna, obiekty wartościowane należące do jednego liścia są od siebie w takim przypadku bardziej oddalone niż obiekty znajdujące się w jednym liściu drzewa, zawierającego obiekty wartościowane. Obszary określone jako otoczenie obiektów wartościowanych mają punkty wspólne z większą liczbą węzłów i obiektów. Podobnie w przypadku, gdy obszar wyznaczany jest jako najbliższy sąsiad obiektu wartościowanego, centroid reprezentujący punkt centralny pomiędzy obiektami z jednego liścia jest od nich bardziej oddalony, a pomiędzy nim i wartościowanymi obiektami znajduje się wiele obiektów wartościujących.

Opisana zależność jest prawdziwa, przy założeniu iż obiekty rozłożone są na tej samej powierzchni, a liście R-drzew mają porównywalną pojemność. W przypadku gdyby liście R-drzewa przechowującego obiekty wartościowane miały mniejszą pojemność lub liście drzewa przechowującego obiekty wartościujące miały większą pojemność, algorytmy stosujące grupowanie mogłyby realizować zapytania, oparte na danych o takiej specyfice, bardziej wydajnie. Zasadniczą cechą jest tu więc odległość pomiędzy obiektami w liściach R-drzew.

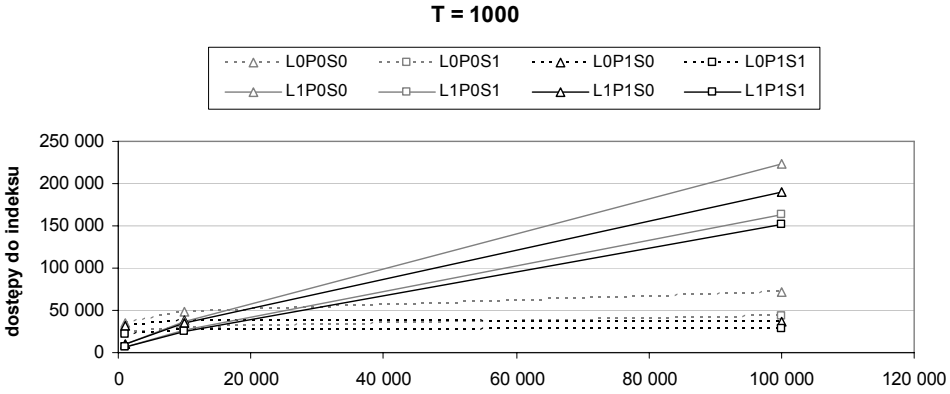


Rys. 12. Dostępy do indeksu przy stałej liczbie obiektów wartościujących P = 100000
Fig. 12. Number of index accesses when the number of feature objects equals P = 100000

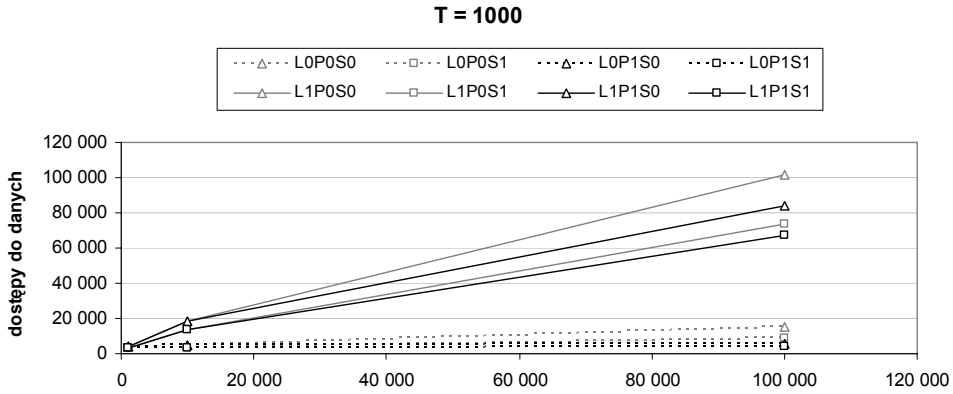


Rys. 13. Dostępy do danych przy stałej liczbie obiektów wartościujących P = 100000
Fig. 13. Number of data accesses when the number of feature objects equals P = 100000

Rysunki 14 i 15 przedstawiające liczbę dostępów potwierdzają, iż przy małej liczbie obiektów wartościowanych T = 1000, a wysokiej liczbie obiektów wartościujących metody wyznaczające wartości pojedynczych obiektów realizować mogą zapytania efektywniej.



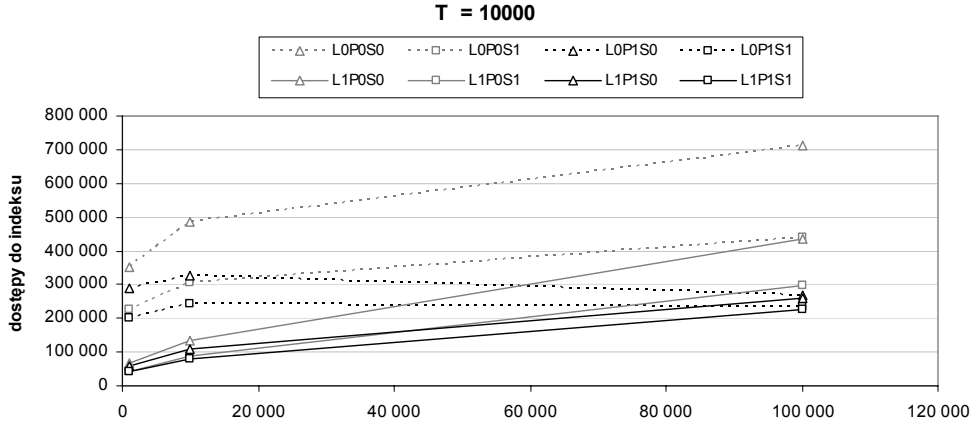
Rys. 14. Dostępy do indeksu przy stałej liczbie obiektów wartościowanych T = 1000
Fig. 14. Number of index accesses when the number of objects equals P = 1000



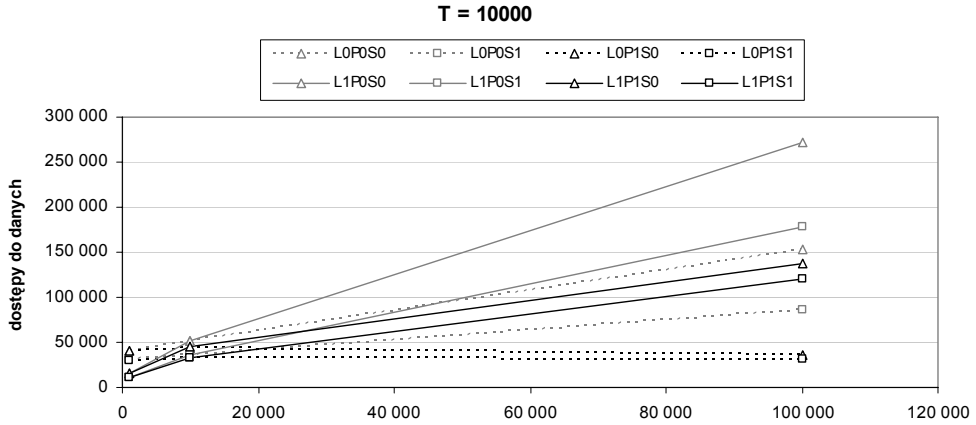
Rys. 15. Dostępy do danych przy stałej liczbie obiektów wartościowanych T = 1000
Fig. 15. Number of data accesses when the number of objects equals P = 1000

Rysunki 16 i 17 przedstawiają analogiczne serie danych dla przypadku, gdy zbiór obiektów wartościujących składa się z T = 10000 obiektów. Widać na nich zmianę na miejscu algorytmów realizujących zapytanie z wykorzystaniem mniejszej liczby dostępów do indeksu i do danych. Algorytmy realizujące zapytania, wyznaczając wartości obiektów pochodzących z jednego liścia jednocześnie, w okolicach punktu, w którym liczba obiektów wartościujących staje się większa od liczby obiektów wartościowanych, zaczynają wymagać większej liczby dostępów do danych. Następnie stają się także mniej efektywne pod względem liczby dostępów do indeksu.

Kolejną tendencją, jaką można zauważyć, jest fakt, iż algorytmy pojedynczo wyznaczające wartości obiektów, stosujące odrzucanie, wraz ze wzrostem liczby obiektów wartościujących realizują zapytania przy mniejszej liczbie dostępów. Odrzucanie staje się bardziej efektywne, gdyż większa gęstość obiektów wartościujących umożliwia, w przypadku wyznaczania otoczenia obiektu za pomocą metody zasięgu, szybsze znalezienie obiektów o wysokich wartościach i efektywne odrzucanie obiektów, wartościowanych później.



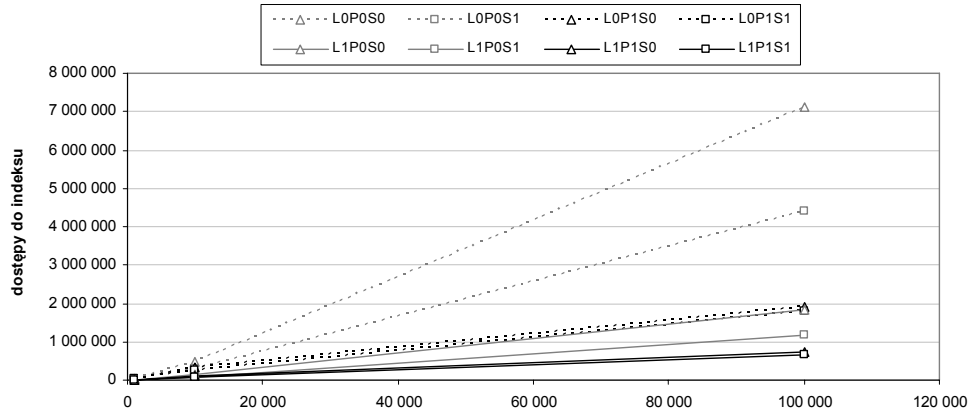
Rys. 16. Dostępy do indeksu przy stałej liczbie obiektów wartościowanych T = 10000
Fig. 16. Number of index accesses when the number of objects equals T = 10000



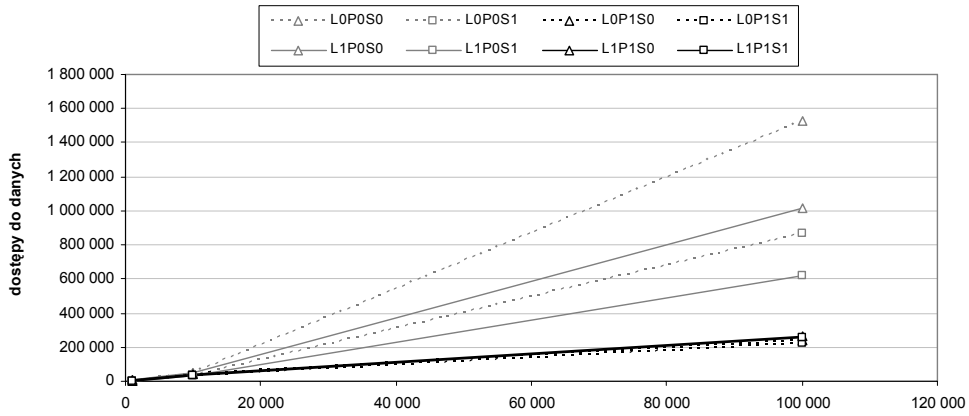
Rys. 17. Dostępy do danych przy stałej liczbie obiektów wartościowanych T = 10000
Fig. 17. Number of data accesses when the number of objects equals T = 10000

Jak pokazują powyższe wykresy, w przypadku gdy obiekty zlokalizowane są na tej samej powierzchni oraz pojemność liścia R-drzew jest porównywalna, algorytmy wyznaczające wartości obiektów pojedynczo okazują się efektywniejsze, gdy zbiory obiektów wartościujących są liczniejsze od zbioru obiektów wartościowanych. W przeciwnym przypadku sytuacja jest odwrotna. W przypadku gdy liczności zbiorów są porównywalne, porównywalna jest także efektywność tych dwóch typów algorytmów.

Rysunki 18 i 19 przedstawiają liczbę dostępów do indeksów i danych, gdy zwiększa się zarówno liczba obiektów wartościowanych, jak i liczba obiektów wartościujących. Widać na nich, iż wraz ze wzrostem liczności wszystkich zbiorów obiektów, na których oparte jest zapytanie, metody wyznaczające wartości wielu obiektów jednocześnie stają się jednak bardziej wydajne niż metody wyznaczające wartości obiektów pojedynczo.



Rys. 18. Dostępy do indeksu przy rosnącej liczności zbiorów danych
Fig. 18. Number of index accesses when the number of objects rises



Rys. 19. Dostępy do danych przy rosnącej liczności zbiorów danych
Fig. 19. Number of data accesses when the number of objects rises

Na podstawie wszystkich przedstawionych faktów stwierdzić można, iż charakter danych ma znaczny wpływ na proces wykonywania zapytania. Co więcej, określa on typ algorytmów realizujących najefektywniej zapytania o preferowane lokalizacje na bazie określonych danych. Algorytmy wykorzystujące metodę równoczesnego wyznaczania wartości wielu obiektów okazują się odrobinę bardziej uniwersalne. Zauważyć trzeba, iż okazują się one dużo bardziej efektywne dla przypadków, które wymagają wielu dostępów do indeksów i danych, które jednocześnie wykonywane są stosunkowo długo. Przypadki, w których lepsze są metody wyznaczające wartości obiektów pojedynczo, wymagają małej liczby dostępów do indeksów i danych. W tym przypadku czas wykonywania zapytania nie jest tak długi, a więc różnica pomiędzy dwiema omawianymi klasami algorytmów będzie mniejsza. Stosunek czasu zyskiwanego podczas wykonywania zapytań złożonych do czasu traconego podczas wykonywania zapytań prostszych będzie więc korzystniejszy dla algorytmów realizujących zapytania, wyznaczając wartości wielu obiektów jednocześnie.

7. Podsumowanie

W artykule przedstawiono interesującą klasę zapytań przestrzennych, jakimi są zapytania o pierwszych k preferowanych lokalizacji. Omówione zostały także istniejące metody optymalizacji wykonywania zapytań tego typu.

Przedstawiona została propozycja modyfikacji opisu formalnego zapytania o pierwszych k preferowanych lokalizacji, biorącą pod uwagę możliwość definiowania wielu preferencji opartych na jednym zbiorze obiektów wartościujących.

W artykule przedstawiono też metodę optymalizacji procesu wykonywania zapytań o pierwszych k preferowanych lokalizacji, umożliwiającą w określonych przypadkach na wyznaczenie wartości względem więcej niż jednej preferencji podczas jednego przejścia

R-drzewa przechowującego obiekty wartościujące. Jak pokazały przeprowadzone testy, metoda ta stosowana może być razem z istniejącymi metodami optymalizacji, umożliwiając dalszą redukcję liczby dostępów koniecznych do wykonania podczas wykonywania zapytań.

Przeprowadzono także testy, mające na celu zbadanie wpływu charakteru zapytania oraz charakteru danych na proces wykonywania zapytania za pomocą różnych algorytmów. W ich wyniku okazało się, iż oba te czynniki mają znaczny wpływ na proces wykonywania zapytania. Potwierdzone zostało przypuszczenie, iż równoczesne stosowanie metody odrzucania oraz opracowanej metody jednoczesnego wyznaczania wartości względem preferencji podobnych jest możliwe i przynosi najlepsze rezultaty. Bardziej zaskakujące okazało się spostrzeżenie, iż metoda optymalizacji wykonywania zapytania, polegająca na jednoczesnym wyznaczaniu wartości obiektów pochodzących z jednego liścia R-drzewa, nie zawsze jest lepszym rozwiązaniem od metody wyznaczającej wartości obiektów pojedynczo. Jest to uzależnione w rozpatrywanym przypadku licznością zbiorów obiektów wartościowanego i wartościujących. W ogólnym przypadku wybór efektywniejszej metody zależy także od obszaru występowania danego typu obiektów oraz pojemności liści R-drzewa będącego ich indeksem.

8. Możliwości rozwoju

Interesującym obszarem, w jakim można by prowadzić dalsze prace związane z przedstawionym zagadnieniem, jest konstrukcja odpowiedniego modelu kosztowego, wybierającego najbardziej optymalną metodę wykonywania zapytania opierając się na możliwej do pozyskania informacji o danych i indeksujących je strukturach.

W celu uzyskania wyników bardziej odpowiadających rzeczywistym zastosowaniom należałoby także wykonać opisane testy przy użyciu rzeczywistych danych.

LITERATURA

1. Yiu M. L., Dai X., Mamoulis N., Vaitis M.: Top-k Spatial Preference Queries. ICDE 2007, s. 1076÷1085.
2. Hjaltason G. R., Samet H.: Distance Browsing in Spatial Databases. ACM Trans. Database Syst. 24(2), s. 265÷318 (1999).
3. Guttman A.: R-Trees: A Dynamic Index Structure for Spatial Searching. SIGMOD Conference 1984, s. 47÷57.
4. Procopiuc O.: Data Structures for Spatial Database Systems. 1997.
5. Strona internetowa www.research.att.com/~marioh/spatialindex.

Recenzent: Dr inż. Bogumiła Hnatowska

Wpłynęło do Redakcji 18 grudnia 2007 r.

Abstract

The paper describes top-k spatial preference queries in general and presents an overview of its existing R-tree based optimization methods.

It also proposes a modification of top-k spatial preference query formal description, which takes into account the possibility of defining a query consisting of more than one preference based on one feature data set.

Moreover, the paper introduces a new optimization technique which is based on the widened definition of top-k spatial preference query. It makes use of similarity between preferences which allows to compute the values based on them simultaneously, resulting in lowering the number of accesses required to execute the query.

The paper also presents an analysis of top-k spatial preference query definition influence on the process of its execution using different algorithms. It confirms, that the introduced optimization can lead to significant increase in overall algorithm efficiency. It can also be used together with existing optimization techniques, further optimizing the process of executing top-k spatial preference queries.

An analysis of data characteristics influence on the process of top-k spatial preference query execution leads to interesting findings. The policy of computing values of objects stored in one R-tree leaf simultaneously doesn't always offer better performance then computing the values of each object. It depends on the characteristics of object and feature data sets. In cases when the number of objects in feature data sets is significantly larger, than the number of objects in object data sets, the optimization based on aforementioned policy leads to more index and data accesses, than required by an algorithm which doesn't use it.

Adresy

Marcin GORAWSKI: Politechnika Śląska, Instytut Informatyki, ul. Akademicka 16, 44-100 Gliwice, Polska, M.Gorawski@polsl.pl.

Kamil DOWLASZEWICZ: Politechnika Śląska, Instytut Informatyki, ul. Akademicka 16, 44-100 Gliwice, Polska, K.Dowlaszewicz@polsl.pl.