STUDIA INFORMATICA 2009
Volume 30 Number 2A (83)

Marcin GORAWSKI, Przemysław TALAGA
Politechnika Śląska, Instytut Informatyki

STRUKTURY INDEKSUJĄCE ZINTEGROWANE Z MATERIALIZOWĄ LISTĄ AGREGATÓW DLA TRAJEKTORYJNEJ HURTOWNI DANYCH

Streszczenie. Artykuł przedstawia opis rozwiązań indeksujących zintegrowanych z Materializowaną Listą Agregatów (MAL) użytych w projektowanym inteligentnym systemie transportowym (ITS). System ITS bazuje na trajektoryjnej hurtowni danych (TrDW). W artykule przedstawiona zostaje Traw, ze szczególnym uwzględnieniem indeksacji trajektorii oraz użytych rozwiązań MAL.

Słowa kluczowe: obiekty mobilne, trajektoryjne hurtownie danych, indeksowanie danych, TB-drzewa

INDEX STRUCTURES INTEGRATED WITH MATERIALIZED AGGREGATE LIST FOR TRAJECTORY DATA WAREHOUSE

Summary. The paper presents a description of indexing techniques integrated with Materialized Aggregate List used in designed trajectory data warehouse system TrDW. It also describes the TrDW system and shows a need and means to index trajectories. Then conclusions on the future work are given.

Keywords: moving objects, trajectory data warehouses, data indexing, TB-trees

1. Wstęp

Zarządzanie transportem publicznym i organizacja przejazdów jest zadaniem trudnym i wymagającym dużych nakładów, w szczególności gdy dotyczy dużego obszaru i dużej liczby mieszkańców. Jednym z celów polityki transportowej jest integracja transportu. Zasadnicze znaczenie ma ona w sytuacji, kiedy występuje kilka rodzajów środków transportu oraz wielu przewoźników. Wprowadzenie informatycznego systemu zarządzania transportem pozwala połączyć w całość poszczególne etapy transportu, zatem system taki staje się podstawo-

wym narzędziem integracji w transporcie. Równocześnie wdrożenie inteligentnych systemów transportowych ITS (ang. *Intelligent Transportation System*) pozwala poprawić wykorzystanie infrastruktury, zmniejszyć koszty działalności, zwiększyć komfort pasażerów – ogólnie poprawić zarządzanie transportem [1]. Podstawowe funkcje, które takie systemy mogą spełniać to między innymi: lokalizacja pojazdów (co jednocześnie pozwala badać realizację rozkładu jazdy i obliczać odchylenia) lub zarządzanie taborem i pracownikami. Przez transport publiczny rozumie się tutaj publiczne środki transportu (autobusy, tramwaje, pociągi) oraz taksówki. Trajektoryjne hurtownie danych służą do obsługi ogromnej liczby danych przestrzenno-czasowych w postaci trajektorii. Trajektoria jest rozumiana jako lista punktów przestrzenno-czasowych uporządkowanych czasowo i przyporzadkowanych jakiemuś obiektowi mobilnemu. Aktualizowanie trajektorii na bieżąco może spowodować, że przy dużej liczbie obsługiwanych jednocześnie obiektów mobilnych, w systemie ITS będą gromadzone dane bez możliwości czasowej ich analizy.

Jednym z podsystemów ITS jest projektowana w Zespole Algorytmów, Programowania i Systemów Autonomicznych (APAS) w Zakładzie Teorii Informatyki Instytutu Informatyki Politechniki Śląskiej Trajektoryjna Hurtownia Danych (ang. *Trajectory Data Warehouse*, TrDW) [2]. W literaturze do tej pory został zaprezentowany tylko jeden podobny system TrDW o bardzo ograniczonej funkcjonalności [3].

System ITS bazujący na TrDW ma służyć analizie transportu publicznego w celu zapewnienia lepszej jego organizacji. Kluczowym problemem jest przechowywanie i korzystanie z historycznych danych opisujących trajektorie, m.in. z użyciem metod indeksowania. Indeksów obsługi trajektorii istnieje kilkanaście. Wkład niniejszego artykułu polega na połączeniu kilku indeksów w jednym działającym TrDW oraz połączeniu tych indeksów z Materializowaną Listą Agregatów (MAL) [4] przechowującą agregaty.

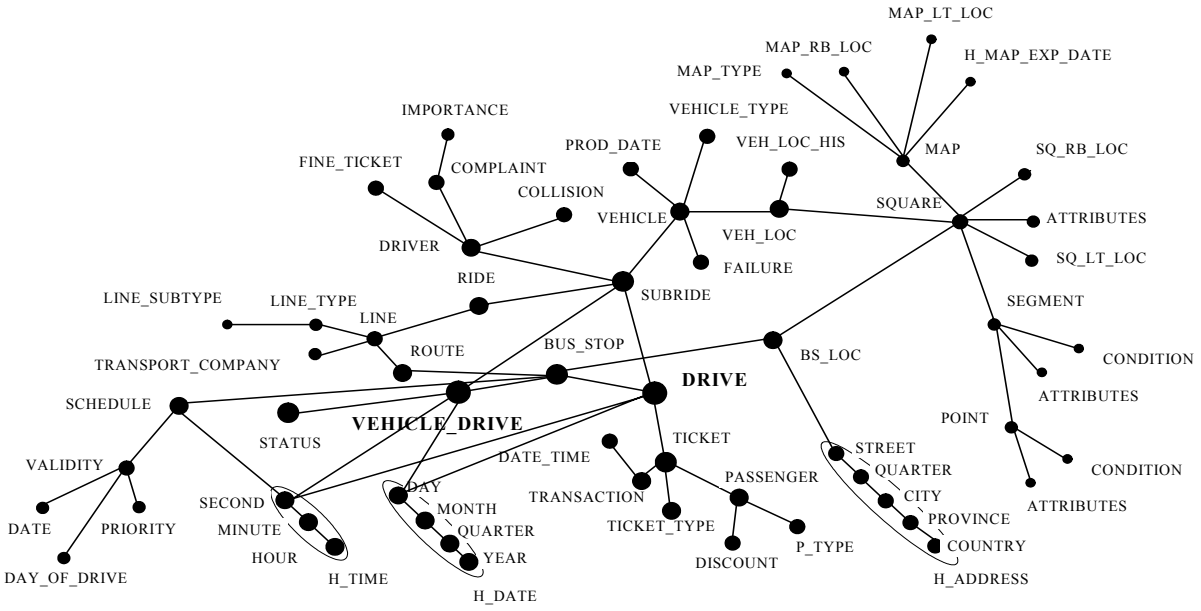
Kolejne rozdziały przedstawiają ogólnie system TrDW (rozdz. 2) oraz opis zastosowanej listy MAL (rozdz. 3). Następnie w rozdz. 4 przedstawione są indeksy użyte w TrDW. Na końcu znajduje się krótkie podsumowanie.

2. Trajektoryjna hurtownia danych

Projektowany system ITS bazujący na TrDW pozwala na usprawnienie zarządzania obsługiwanymi pojazdami oraz zatrudnionym personelem. Podstawowym jego zadaniem jest wykonywanie analiz dotyczących funkcjonowania transportu i dzięki temu wspomaganie osób zarządzających procesem decyzyjnym.

2.1. Schemat logiczny TrDW

Rysunek 1 przedstawia schemat logiczny TrDW. Schemat ten odpowiada schematowi konstelacji faktów połączonemu ze schematem gwiazdy kaskadowej – dwie gwiazdy kaskadowe współdzielą kilka wymiarów. Schemat ten można rozłożyć na dwa oddzielne schematy. Przedstawiają one rozszerzoną gwiazdę kaskadową III stopnia ($ES^{C}_{\{III-t\}}$), którą można przedstawić jako gwiazdę kaskadową II stopnia, w której istnieją co najmniej dwie tabele tworzące schemat konstelacji faktów [5].



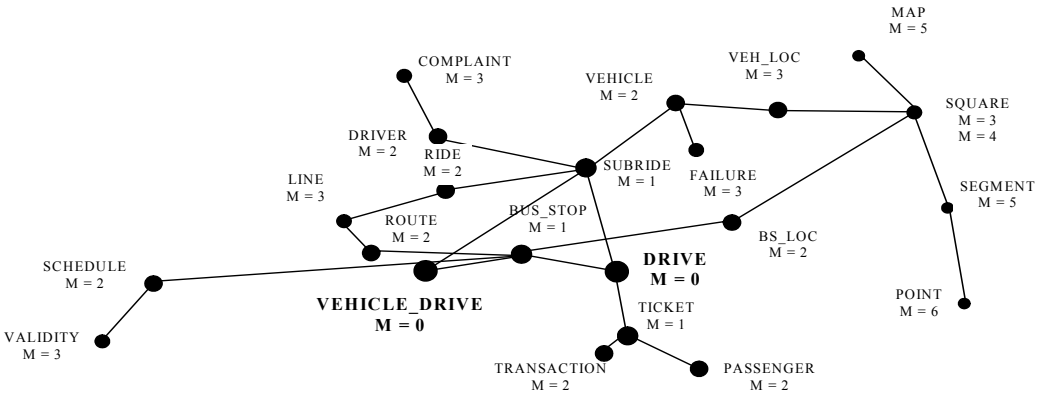
Rys. 1. Schemat logiczny TrDW
Fig. 1. TrDW logical schema

Schemat logiczny TrDW bazuje na 2 niezależnych podschematach, a mianowicie:

- Obsługi trajektorii poszczególnych pasażerów – założeniem jest tutaj posiadanie danych o podróżach wszystkich pasażerów. Główną tabelą faktów dla tego podschematu jest tabela DRIVE, która jest pomocna przy analizach dotyczących: tras podróży pasażerów, popularności danych tras, wzorców zachowań pasażerów.
- Obsługi przejazdów poszczególnych pojazdów. Główną tabelą tutaj jest VEHICLE-_DRIVE, dzięki której jest możliwość analizowania tłoku na przystankach, opóźnień na liniach, obłożenia przejazdów pasażerami.

Rysunek 2 przedstawia poziomy zagłębienia (współczynnik M [6]) poszczególnych wymiarów. Maksymalny poziom zagłębienia wynosi 6, co oznacza, że maksymalna odległość pomiędzy centralną tabelą faktów a najbardziej zagnieżdżonym schematem (schematem przechowującym punkty zainteresowania na mapie) wynosi 6. Schemat ten zawiera dwie centralne tabele faktów (o poziomie zagłębienia równym 0). Wymiar SQUARE posiada dwie war-

tości współczynnika M, ponieważ jest on dzielony przez dwa wymiary i można się do niego dostać na dwa sposoby.



Rys. 2. Poziom zagłębienia poszczególnych wymiarów
Fig. 2. Depth level of each dimension

2.2. Agregacja

Do tworzenia agregatów z surowych danych służą funkcje agregujące, które można podzielić na trzy typy [7]. Podział ten wskazuje typy agregatów, które mogą być wykorzystane jako częściowe agregaty – służące do agregacji danych na różnych poziomach szczegółowości. Częściowe agregaty potrzebne są do budowy wielopoziomowej struktury agregatów, na przykład drzewa agregatów. Drzewo takie w sposób naturalny pokazuje hierarchię, w jaką zorganizowane są dane. Rozróżniamy rodzaje funkcji agregujących – dystrybutywne, algebraiczne i holistyczne. Funkcje dystrybutywne są łatwo stosowalne w wielopoziomowej strukturze agregatów (również w rozproszonej strukturze) – do ich obliczenia na wyższym poziomie wystarczą same agregaty z poziomów niższych (np.: minimum, suma). Funkcje algebraiczne również są łatwe do zastosowania w wielopoziomowej strukturze – aby obliczyć te funkcje na wyższym poziomie, musimy zapisywać potrzebne do obliczenia danej funkcji wartości i korzystając z nich obliczamy wartość funkcji (np.: średnia i odchylenie standardowe). Funkcje holistyczne stanowią największy problem w przypadku wielopoziomowych lub rozproszonych struktur, ponieważ do ich obliczenia wymagane są wszystkie dane i nie można skorzystać z obliczeń na niższych poziomach (np.: mediana i ranking).

W TrDW użyto dwa sposoby agregacji – ze względu na czas oraz ze względu na trajektorie. Najniższym poziomem w hierarchii agregacji czasu są poszczególne godziny dnia. Tak szczegółowe dane z konkretnego dnia są mało przydatne (analiza tak małej liczby danych nie da nam żadnych ogólnych wniosków), ale w poszczególnych godzinach przejazdy mogą się znacząco różnić od siebie. Dane z całego miesiąca (np. przejazdy w każdym dniu miesiąca od godziny 12.00 do 13.00) są już przydatne. Drugą hierarchią jest hierarchia trajektorii. Trajektoria składa się z określonej liczby podtrajektorii, w szczególnym przypadku z jednej podtra-

jektorii, czyli samej siebie. Taki przypadek występuje, jeśli danej trasy nie można (lub się nie opłaca) zdekomponować. Dana podtrajektoria może pokrywać trasę dowolnej liczby linii, ale również może nie pokrywać żadnej. W takim przypadku składa się ona z określonej liczby postojów. Linia również składa się z określonej liczby postojów. Różnica między linią a trajektorią polega na tym, że pojazd danej linii porusza się na określonej trajektorii, ale na danej trajektorii może istnieć więcej niż jedna linia.

Dane mogą być agregowane zgodnie z hierarchią czasu lub trajektorii. Agregaty wyższych poziomów hierarchii możemy obliczyć korzystając z agregatów niższego poziomu (nie stosujemy funkcji agregujących holistycznych). Liczba możliwych do obliczenia i przechowywania agregatów jest duża i zależy tylko od wymagań końcowego użytkownika. W przypadku agregatów będących wartością średnią wymagane jest zapisywanie dodatkowych danych, tzn. sumy wartości i liczby tych wartości. Do przechowywania i obsługi agregatów używana jest Materializowana Lista Agregatów [4].

3. Materializowana Lista Agregatów

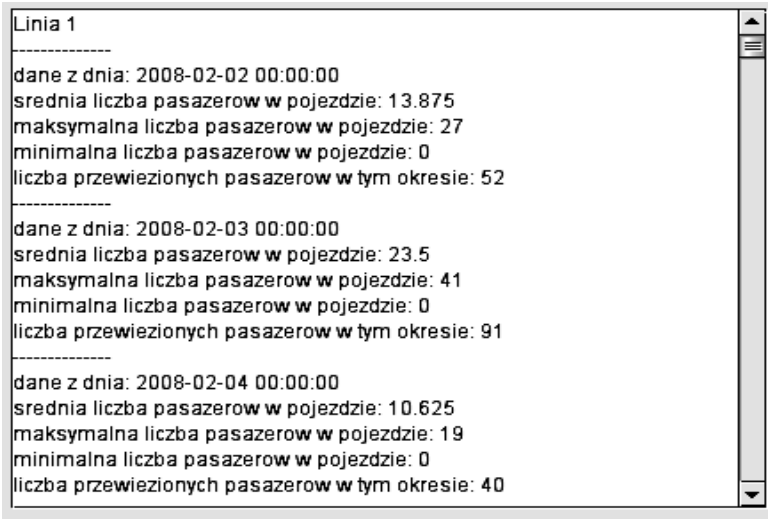
W Zespole APAS opracowano nowatorski sposób składowania i materializacji agregatów nazwany Materializowaną Listą Agregatów (MAL) [4]. Rozwiązanie to bazuje na wzorcu projektowym listy dostępnym w języku Java (*java.util.List*). Zasadniczą różnicą w stosunku do zwykłej listy jest źródło danych listy – elementy listy nie są wstawiane przez klienta, ale pobierane są ze źródła danych, którym może być baza danych lub struktura indeksująca. Klient nie ma możliwości wstawiania danych do listy, ponadto może on komunikować się z listą tylko za pośrednictwem iteratora. W przeciwieństwie do zwykłej struktury obiekt listy nie przechowuje żadnych danych (agregatów) – jest on tylko pośrednikiem między iteratorem (wykorzystywanym przez klienta) a źródłem danych. Klient za pomocą iteratora ma możliwość sekwencyjnego przeglądania listy i pobierania z niej agregatów. Operacja usuwania nie jest dostępna zgodnie z ideą hurtowni danych – dobór danych, które znajdą się w hurtowni odbywa się w czasie procesu ETL (ekstrakcji, transformacji i ładowania). W celu poprawy wydajności działania stosuje się mechanizm materializacji obliczonych agregatów. W przypadku rozwiązania opartego na bazie danych materializacja polega na zapisie do specjalnie przygotowanej tabeli obliczonych agregatów. Zmaterializowane dane zapisywane są do bazy z wykorzystaniem strumieni danych.

W podstawowej wersji listy MAL agregaty wykorzystywały jedynie funkcję sumy, tzn. przechowywana była tylko suma danych dla jakiegoś okresu. W przypadku TrDW wykorzystywane są również inne funkcje dystrybutywne, np. MIN czy MAX. Dodatkowo konieczne jest przechowywanie wartości średnich, a więc wykorzystanie funkcji algebraicznej. Typy

wartości stanowiących agregat oprócz samych danych przechowują również informację o rodzaju funkcji agregującej, którą dany typ reprezentuje (dla funkcji dystrybutywnych). Dla funkcji AVG typ wartości stanowią dwie osobne liczby – suma wartości oraz liczba wartości, z których obliczana jest wartość średnia. Wymagane jest to przy obliczaniu agregatów wyższych poziomów z wykorzystaniem agregatów niższych poziomów.

Dodatkowo, przy materializacji agregatów funkcji dystrybutywnych zapisywane są nie tylko wartości danych, lecz również typ funkcji (w postaci liczby) – w tej kolejności, najpierw wartość, a następnie liczba identyfikująca typ funkcji agregującej. Dla agregatu będącego wartością średnią materializowane są wartości sumy i liczby – sama wartość średnia jest z nich wyliczana i nie jest ona zapisywana w bazie. Przy materializacji ważna jest również kolejność zapisu – najpierw zapisujemy znacznik czasowy, następnie listę wartości funkcji dystrybucyjnych, a na końcu listę wartości średnich.

Przykładowe agregaty przedstawia rys. 3.



Rys. 3. Przykładowe agregaty wyliczone za pomocą MAL
Fig. 3. Example aggregates computed with MAL

4. Indeksacja trajektorii

W przypadku TrDW wyróżniamy 3 rodzaje indeksowania – indeksacja danych historycznych (dla analizy długookresowej), indeksacja aktualnych danych (przy bieżącej obsłudze obiektów mobilnych) oraz indeksacja bliskiej przyszłości (dane dotyczą tego, jak zmieni się położenie obiektów). W niniejszej pracy rozważania dotyczą 2 rodzajów indeksowania: bieżącej obsługi oraz analizy historycznej.

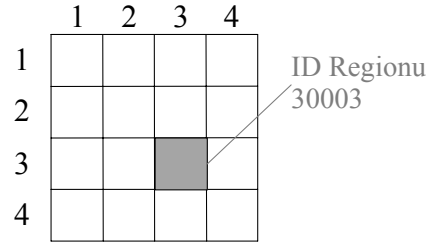
W pierwszym przypadku obiekty mobilne cały czas mogą zmieniać swoje położenie (również niektóre atrybuty, jak na przykład prędkość). Mogą też przestać przesyłać informa-

cje o położeniu (np. obiekt ten wyszedł poza obszar naszego zainteresowania). W związku z tym indeks musi umożliwiać nie tylko szybkie wyszukiwanie, ale również szybkie uaktualnianie danych oraz usuwanie obiektów. W drugim przypadku mamy tylko historyczne położenia obiektów i nie zmieniają się one cały czas (czas uaktualniania danych nie musi być tak znaczący, a usuwania danych w ogóle się nie rozważa. Pozostaje tylko zapewnienie odpowiedniej wydajności wyszukiwania. W obu przypadkach liczba danych może być olbrzymia i ich indeksowanie jest nieodzowne, aby wydajność była zadowalająca.

4.1. Indeksacja bieżących położeń obiektów

Podstawowym problemem przy bieżącym indeksowaniu obiektów mobilnych jest częste uaktualnianie ich pozycji. W standardowym rozwiązaniu w każdej chwili czasu należy uaktualnić pozycję wszystkich obiektów, co może zająć znaczący (często nieakceptowalny) okres czasu. W systemach ITS z TrDW zwykle do analizy decyzyjnej wystarczą przybliżone położenia pojazdów. Przykładowo, pojazd znajduje się na Placu Piastów w Gliwicach – dokładne położenie na nim już nie ma dla nas znaczenia.

Stosujemy tutaj rozwiązanie oparte na Technice Haszującej (ang. *Hashing Technique*) [8]. Dzielimy obsługiwaną przestrzeń na regiony o stałej i niezmiennej wielkości (regiony nie mogą być zbyt duże, ale równocześnie nie może ich być też zbyt wiele – optymalny rozmiar regionów zależy od wymagań konkretnej aplikacji) i wykorzystujemy informację o przynależności danego obiektu do regionu. Identyfikator regionu jest tworzony automatycznie na podstawie jego pozycji na mapie – na podstawie numeru wiersza i kolumny, w której znajduje się dany region. Wzór na obliczenie identyfikatora jest następujący: (10000 * wiersz + kolumna). Założeniem jest tutaj istnienie maksymalnie 10000 wierszy i kolumn, ale w razie potrzeby można zwiększyć te wartości. Użycie takiego wzoru powoduje, że identyfikator od razu umożliwia odczytanie położenia danego regionu na mapie. Rysunek 4 przedstawia przykład.



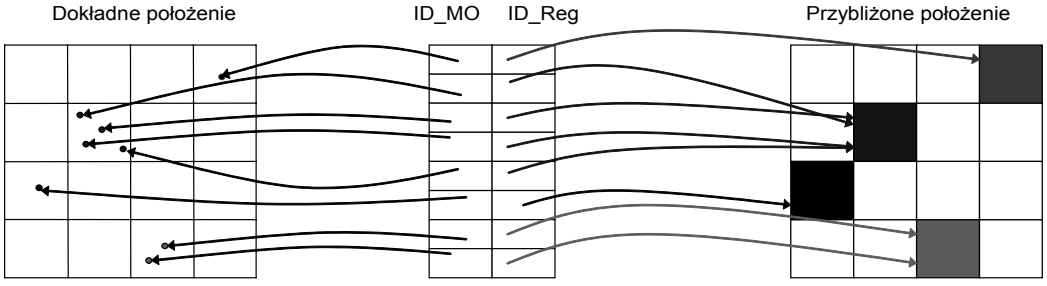
Rys. 4. Mapa w postaci regionów
Fig. 4. Map in the form of regions

Aktualizacja położenia obiektu w bazie danych występuje tylko przy zmianie regionu – kiedy obiekt porusza się wewnątrz jednego regionu, zmiany te nie są przenoszone do bazy

danych. Dzięki temu nie ma potrzeby ciągłej aktualizacji. Pseudokod operacji aktualizacji jest następujący:

```
Jeśli obiekt nie zmienił regionu
{
  Dodaj aktualne położenie do listy historycznych położeń
  Jeśli osiągnięto założony rozmiar bufora
  {
    Zapisz wszystkie historie w bazie danych
    Jeśli poprawnie zapisano
    Wyczyść bufor
  }
}
W przeciwnym wypadku – obiekt zmienił region
  Uaktualnij położenie obiektu w bazie danych
Aktualizacja położenia w obiekcie.
Aktualizacja znacznika czasowego w obiekcie.
```

Obiekt przesyła do systemu informacje z GPS o swoim położeniu, a tutaj zostaje ona przekształcona na informacje o regionie. Każdy region jako atrybut posiada informację o liczbie obiektów znajdujących się w nim. Ponadto, przechowuje się informacje o bardziej szczegółowym położeniu obiektów, istnieje lista zawierająca historyczne dokładne położenia (informacje te mogą być przydatne przy historycznej analizie). Po przekroczeniu pewnej liczby zapisanych obiektów historii następuje przeniesienie tej informacji do bazy danych – bufor pozwala zmniejszyć liczbę dostępów do bazy danych i zastosować ładowanie wsadowe. Obsługiwanych pojazdów jest określona liczba – są to rzędy tysięcy, a więc nie są to wielkie liczby. Pozwala nam to przechowywać w pamięci listę obiektów mobilnych – każdy obiekt ma w sobie zapisaną lokalizację. Przechowywanie jej w pamięci pozwala na natychmiastowe operacje wyszukiwania regionu (jak również uaktualniania), w którym znajduje się obiekt. Rysunek 5 przedstawia poglądowo to rozwiązanie.

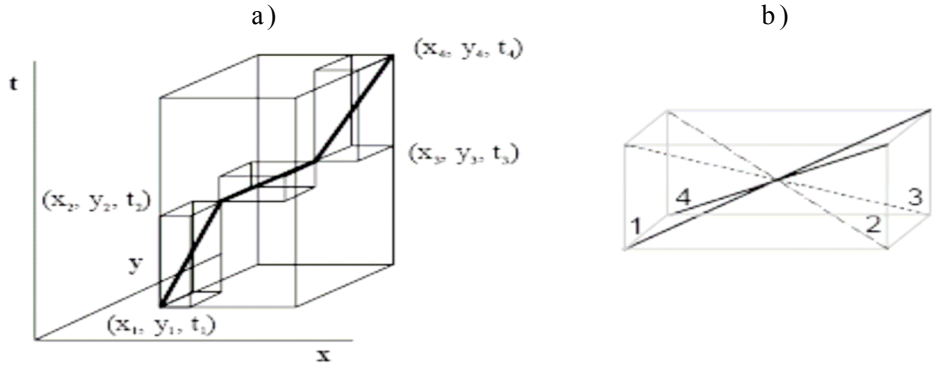


Rys. 5. Indeksacja bieżąca obiektów
Fig. 5. Indexing of current objects' position

4.2. Indeksacja historycznych położeń obiektów

W celu ograniczenia liczby przechowywanych danych dla pojazdów poruszających się po stałych ustalonych trasach (autobusów, pociągów itp.) przechowujemy tylko informacje o czasie na poszczególnych przystankach – jeśli trasa jest znana, to nie ma sensu przechowywać bardziej dokładnych danych. Dla taksówek przechowujemy całość danych na temat pod-

róży. Dane te mają formę: (położenie X, położenie Y, czas). Co najmniej 2 takie punkty uporządkowane zgodnie z wartością czasu tworzą segment trajektorii. Składając razem kolejne segmenty, otrzymujemy całkowitą trajektorię. Podział na segmenty jest konieczny ze względu na sposób zapisu – przybliżanie za pomocą MBB: Minimalne Pudełko Otaczające (ang. *Minimal Bounding Box*) wprowadza dużą martwą przestrzeń, tym większą, im dłuższa jest trajektoria (rys. 6a [9]). Segmenty te zapisujemy w postaci trójwymiarowej (2 wymiary przestrzenne i czas). Indeks umożliwia również automatyczne rzutowanie do postaci dwuwymiarowej (dwóch wymiarów przestrzennych) w celu szybszej realizacji zapytań przestrzennych bez uwzględniania czasu.



Rys. 6. Różnice w MBB dla segmentów: a) całej trajektorii, b) linii
Fig. 6. Differences in MBB for segments: a) whole trajectory, b) line

Położenie samej trajektorii w MBB danego segmentu jest nazywane orientacją (ang. *orientation*). Orientacja przyjmuje wartości ze zbioru {1, 2, 3, 4} – jak pokazuje rys. 6b są tylko 4 sposoby, na które segment może być zawarty w MBB.

Zapytania dla historycznych danych obiektów mobilnych ogólnie można podzielić na przestrzenne (zakresowe, przedziałowe, czasowe, kNN) oraz trajektoryjne. W [9] zostało porównanych ze sobą kilka rozwiązań, a wnioski z tego porównania w stosunku do zastosowań przedstawia tabela 1.

Tabela 1

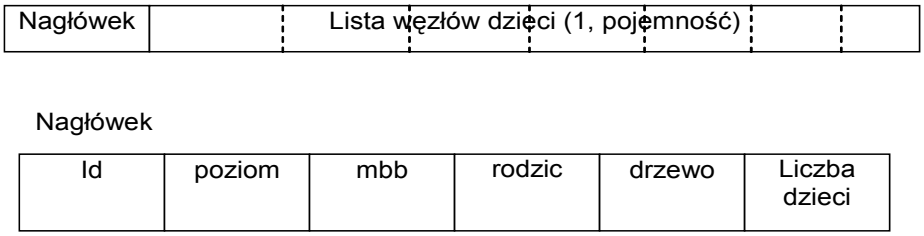
Indeksy pod względem klas zapytań

Klasy zapytań	Indeksy stosunkowo wydajne dla danej klasy
Trajektoryjne	TB-tree
Zwykłe zakresowe	Grid
Uaktualnienia	LUR-tree, R*tree,
Wyszukiwanie	Quad-tree
Historyczne	MV3R-tree, BB^x-tree
Predykcyjne	B^x-tree, BB^x-tree
Czasowe	STR-tree, HR-tree
Przedziałowe	MV3R-tree

TB-drzewa są przeznaczone do obsługi zapytań trajektoryjnych i osiągają w nich bardzo dobrą wydajność. Nie wykazują one jednak podobnie dużej efektywności dla zapytań zakre-

sowych. Trajektorie nakładają się na siebie, co powoduje nakładanie się MBB pośrednich węzłów oraz dużą przestrzeń martwych stref. Ponadto, pomimo przestrzennej bliskości segmentów różnych trajektorii przechowywane są one w różnych węzłach. Dlatego jako pomocniczy indeks dla tego rodzaju zapytań użyto R*-drzewo [10].

Wpis danych dla R*-drzewa ma postać (ID, trajectoryID, nodeID, MBB, orientacja). TrajectoryID identyfikuje trajektorię, do której należy dany segment, nodeID zawiera identyfikator liścia, który zawiera ten wpis, MBB i orientacja to kształt segmentu trajektorii. W przypadku TB-drzewa identyfikator trajektorii zapisywany jest w liściu (ponieważ liście przechowują segmenty tylko jednej trajektorii). Postać węzła pośredniego przedstawia rys. 7.



Rys. 7. Postać węzła pośredniego R*-drzewa
Fig. 7. Structure of a middle node in R*-tree

Węzeł składa się z nagłówka oraz z listy węzłów podrzędnych (dzieci), które mogą być albo kolejnymi węzłami pośrednimi, albo liśćmi. Liście mają taką samą postać z tą różnicą, że zamiast dzieci są wpisy danych. Nagłówek składa się z: identyfikatora danego węzła, liczby określającej poziom danego węzła (poziom 0 to poziom liści), MBB zawierającego wszystkie dzieci węzła, referencji do rodzica, referencji do drzewa, do którego należy ten węzeł oraz liczby dzieci.

5. Podsumowanie

Przedstawiono rozwiązania indeksujące użyte w trajektoryjnej hurtowni danych TrDW, na której bazuje system ITS. Do celów indeksacji historycznych położeń zostały wybrane TB-drzewa oraz R*-drzewa. Głównym rodzajem zapytań w systemie będą zapytania trajektoryjne, dla których zaprojektowane są TB-drzewa. R*-drzewo zostało wybrane jako indeks wspomagający wykonywanie zapytań zakresowych, z którymi nie najlepiej radzi sobie TB-drzewo.

Dalsze prace dotyczą modelowania wielostopniowej struktury indeksów, w której na różnych poziomach istniałyby osobne listy MAL przechowujące odpowiednie agregaty. W wersji rozproszonej dodatkowo będę istniały lustrzane struktury, które przechowywałyby różne rodzaje agregatów.

Kolejnym kierunkiem rozwoju TrDW jest opracowanie specjalnego języka zapytań do obsługi trajektorii. Za pomocą SQLa niektóre zapytania są trudne do zrealizowania i specjalny język byłby znacznym ułatwieniem dla użytkowników. Język taki powinien uwzględniać specyfikę trajektorii.

BIBLIOGRAFIA

1. Gorawski M., Gorawski M. J.: Narzędzia informatyczne związane z gromadzeniem dużej liczby danych, ich analizą w trybie on-line. Inteligentny system zarządzania transportem publicznym. Zespół Automatyki w Transporcie. Wydział Transportu. Politechnika Śląska, 2008, s. 20÷55.
2. Gorawski M.: Lokalizacja obiektów i trajektoryjne hurtownie danych. Inteligentny system transportowy dla Śląska. Konferencja, 2008, s. 1÷14.
3. Braz F. J.: Trajectory Data Warehouses: Proposal of Design and Application to Exploit Data. IX Brazilian Symposium on GeoInformatics, 2007, INPE, s. 61÷72.
4. Gorawski M., Malczok R.: On Efficient Storing and Processing of Long Aggregate Lists. 7th International Conference Data Warehousing and Knowledge Discovery, DaWak 2005. Copenhagen, LNCS 3589, s. 190÷199.
5. Gorawski M.: Definiowanie schematów rozszerzonej gwiazdy kaskadowej. BDAS`07, Konferencja – Bazy Danych: Aplikacje i Systemy, 05.2007, s. 28÷31.
6. Yu S., Atluri V., Nabil R. A.: „Cascaded star: A hyper-dimensional model for a data warehouse". 17th International Conference on Database and Expert Systems Applications, DEXA 2006, Vol. 4080 of Lecture Notes in Computer Science, 2006, s. 439÷448.
7. Gray J., Chaudhuri S., Bosworth A., Layman A., Reichart D., Venkatrao M., Pellow F., Pirahesh H.: Data Cube: A Relational Aggregation Operator Generalizing Group-By, Cross-Tab, and Sub-Totals. DataMining and Knowledge Discovery 1, 1997, s. 29÷53.
8. Song Z., Roussopoulos N.: Hashing Moving Objects. Proc. Second Int'l Conf. Mobile Data Management, 2001, s. 161÷172.
9. Pfoser D., Jensen C. S., Theodoridis Y.: Novel Approaches to the Indexing of Moving Object Trajectories. In Proc.VLDB, 2000, s. 395÷406.
10. Beckmann N., Kriegel H.P., Schneider R., Seeger B.: The R*-tree: An efficient and robust access method for points and rectangles. In Proceedings of ACM SIGMOD International Conference on Management of Data, 1990, s. 322÷331.
11. Hu W., Sha L., Sperber M.: Moving Object Spatial Queries and Indexing Techniques. http://www.spatial.cs.umn.edu/Courses/Fall07/8715/sample_project_report_2.doc.

Recenzent: Dr hab. inż. Robert Wrembel

Wpłynęło do Redakcji 4 lutego 2009 r.

Abstract

The paper describes indexing techniques used in designed trajectory data warehouse system TrDW. First general information about this system and MAL are given. Next, indices used in the system were introduced. TB-trees and R*-trees were chosen to index historical data of moving objects. Trajectory queries will be main type of queries in the system, and TB-trees are designed specifically for those queries. R*-tree was chosen as supporting index for executing range queries, because TB-trees are not so good in that type of queries. Method of indexing objects' current locations was described next. Presented solutions were implemented and tested. Tests shows that designed system works correctly, and constitute a good base for further development.

Adresy

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