AUTOMATYKA • 2011 • Tom 15 • Zeszyt 2

Jarosław Koźlak*, Anna Zygmunt*

Identyfikacja wzorców w ruchu drogowym metodą częstych sekwencji**

1. Wprowadzenie

Celem prac jest identyfikacja wzorców w ruchu drogowym opisujących zależności pomiędzy natężeniem ruchu na poszczególnych odcinkach dróg podczas zadanego przedziału czasowego. W rezultacie można wykryć odpowiednio wcześniej zmiany stanu ruchu na poszczególnych drogach oraz formowanie się korków, co pozwala osiągnąć korzyści na poziomie globalnym oraz lokalnym. Na poziomie globalnym, system sterowania ruchem może podjąć działanie mające na celu zapobieganie powstawaniu korków lub ograniczenie ich skutków. Efekty te można uzyskać przez odpowiednie sterowanie sygnalizacją świetlną lub też przez dostarczanie kierowcom odpowiednich informacji na temat stanu ruchu. Na poziomie lokalnym, poszczególni kierowcy mogą uwzględnić ostrzeżenia o możliwości powstawania korków oraz odpowiednio wcześniej zmodyfikować swoje planowane trasy podróży.

Wykorzystywanie podejścia agentowego do modelowania i optymalizacji ruchu drogowego zaawocowało stworzeniem systemów wykorzystujących różne rodzaje modeli oraz algorytmów sterujących [3]. Zrealizowany przez nas symulatora wieloagentowego do modelowania i optymalizacji ruchu drogowego [8] wykorzystuje model ruchu oparty na automatach komórkowych oraz wykorzystuje różne algorytmy zarządzania światłami i uwzględnienia wzorców ruchu podczas planowania tras podróży. Agenci-pojazdy generowane w zdefiniowanych węzłach grafu reprezentującego sieć drogową mają zadane punkty docelowe podróży. Trasy są przez nie wyliczane w oparciu o algorytmy najkrótszej odległości w grafie, co oznacza najkrótszy czas przejazdu, natomiast w razie uzyskania informacji o zmianach stanu ruchu agenci mogą zmodyfikować swoją trasę.

W celu konfiguracji analizowanych w niniejszej pracy scenariuszy eksperymentalnych wykorzystywane są dane wygenerowane przez symulator. Analiza wzorców ma na celu wyodrębnienie zależności między stanami ruchu na poszczególnych drogach oraz zachodzącymi

* AGH Akademia Górniczo-Hutnicza, Wydział Elektrotechniki, Automatyki, Informatyki i Elektroniki, Katedra Informatyki

** Prace finansowano z grantu Ministerstwa Nauki i Szkolnictwa Wyższego nr N N516 366236

w nich zmianami w rozpatrywanym przedziale czasowym. Opracowane algorytmy identyfikacji wzorców mogą być następnie wbudowane w symulator w celu weryfikacji stopnia wpływu ich uwzględnienia na poprawę stanu ruchu.

Prowadzone przez nas badania dotyczące analizy wzorców ruchu obejmują następujące kierunki:

- Określenie wpływu stan ruchu na poszczególnych drogach na ruch na innych drogach oraz siły tego wpływu. W rezultacie możliwe jest stworzenie wzorców decyzyjnych, które mogą stanowić podpowiedzi dla kierowców odnośnie wyboru tras lub dla służb kontroli ruchu sterujących fazami świateł na skrzyżowaniach.
- Wyodrębnienie zbiorów dróg, które są najbardziej ze sobą powiązane i dla których można zauważyć najsilniejsze regularności w zależnościach poziomów stanu ruchu.

W niniejszym artykule skupiamy się szczególnie na tym drugim zagadnieniu.

2. Przegląd dziedziny badań

Do analizy prawidłowości w rozkładach ruchu są wykorzystywane różne algorytmy rozpoznawania wzorców (szeroki przegląd algorytmów rozpoznawania wzorców jest przedstawiony w [11]). Analiza ruchu drogowego obejmuje różne rodzaje problemów, najszerzej badaną dziedzinę jest przewidywanie czasów podróży, czemu często towarzyszy analiza stanu ruchu.

Przewidywanie stanu ruchu jest wykonywane przy użyciu różnych rodzajów technik: stosowane są między innymi metody regresji, estymacje szeregów czasowych. Używane są rozwiązania oparte na elementach sztucznej inteligencji np. na sieciach neuronowych [5, 13], popularnym podejściem jest także wykorzystanie filtru Kalmana.

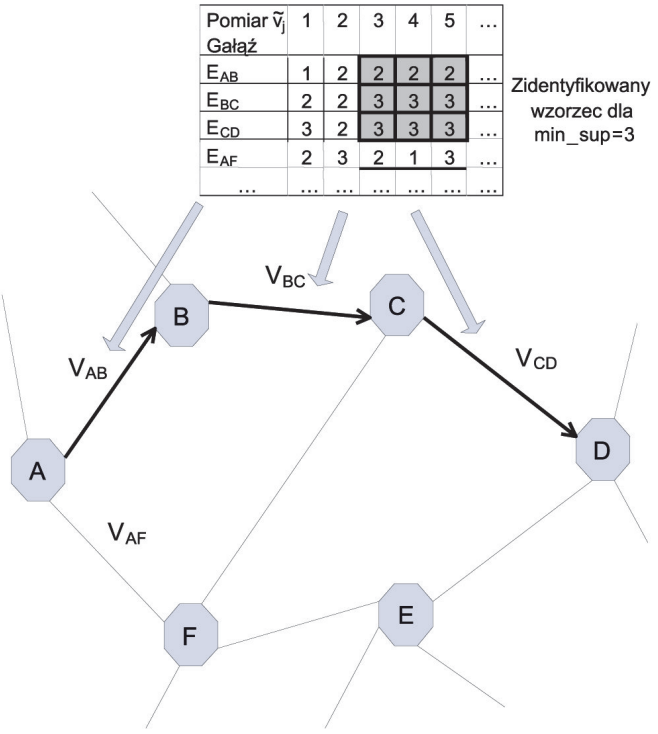
W [4] stosowana jest analiza wzorców ruchu, polegająca na wyszukiwaniu wzorców ruchu podobnych do rozpatrywanej aktualnie sytuacji, do grupowania podobnych rozkładów są stosowane algorytmy klastrowania, np. SLR (*small large ratio*).

3. Model ruchu drogowego

Rozpatrywany model ruchu drogowego obejmuje następujące elementy:

- graf (N, E) reprezentujący sieć drogową, w którym węzły (N) reprezentują skrzyżowania lub punkty charakterystyczne (reprezentujące wjazd lub wyjazd ruchu przelotowego), zaś gałęzie (E) – odpowiednie połączenia drogowe;
- wektory wag v_j oraz n_j związane z poszczególnymi gałęziami E_j reprezentujące średnie prędkości lub liczbę pojazdów w rozpatrywanych kolejnych przedziałach czasu; dla poszczególnego przedziału czasu δ_k opisującego okres $[T_k, T_k + \Delta t]$, odpowiednie elementy wektorów wag są opisywane przez $v_j^{(k)}$ oraz $n_j^{(k)}$.

W celu przygotowania uzyskanych danych do analiz za pomocą metody częstych sekwencji przeprowadzono kwantyzację prędkości na poszczególnych gałęziach w rozpatrywanych przedziałach, przypisując im jedną z n różnych wartości prędkości, gdzie n zmieniała się w poszczególnych badaniach eksperymentalnych. Takie skwantyzowane wartości są oznaczone jako $\tilde{v}_j^{(k)}$. Analizowana sekwencja częsta polegała na tym, że na n kolejnych adjacentnych gałęziach wielokrotnie powtarzały się te same poziomy natężenia ruchu. Koncepcja identyfikacji częstych sekwencji w danych na temat ruchu jest przedstawiona na rysunku 1.



Rys. 1. Koncepcja identyfikacji sekwencji częstych w danych na temat ruchu

4. Identyfikacja wzorców sekwencji

4.1. Opis problemu

Sekwencja jest uporządkowaną listą zdarzeń oznaczaną jako $< e_1\ e_2 \ldots e_l >$, gdzie zdarzenie e_1 występuje przed zdarzeniem e_2, które z kolei występuje przed e_3 i tak dalej. Zdarzenie e_j jest nazywane elementem sekwencji s. Mając dwie sekwencje $\alpha = < a_1\ a_2 \ldots a_n >$ i $\beta = < b_1\ b_2 \ldots b_m >$, α jest nazywane podsekwencją β, co jest oznaczane jako $\alpha \subseteq \beta$, jeżeli istnieją liczby całkowite $1 \le j_1 < j_2 < \ldots < j_n \le m$ takie, że $a_1 \subseteq b_{j1}$, $a_2 \subseteq b_{j2}$, …, $a_n \subseteq b_{jn}$.

Baza danych sekwencji S jest zbiorem krotek < SID, s>, gdzie SID jest numerem sekwencji sequence_ID, a s jest sekwencją. Krotka < SID, s> zawiera sekwencję α wtedy gdy α jest podsekwencją s. Poparcie (*support*) sekwencji α w bazie danych sekwencji S określa liczbę krotek w bazie zawierających α. Ze względu na potrzebę ograniczenia liczby znalezionych sekwencji do tych, które są wystarczająco interesujące, wprowadza się próg minimalnego poparcia oznaczany jako min_sup (*minimum support threshold*), który jest dodatnią liczbą całkowitą i w tym kontekście dana sekwencja α jest częsta w sekwencyjnej bazie S, jeżeli występuje przynajmniej min_sup razy w S: $support_S(\alpha) \geq min\_sup$. Taka częsta sekwencja jest nazywana wzorcem sekwencji [7].

4.2. Przegląd algorytmów

Zadanie znajdowania wzorców częstych sekwencji można zdefiniować następująco: mając na wejściu zbiór sekwencji zdarzeń, chcemy znaleźć często występujące podsekwencje.

Algorytm powinien móc znaleźć możliwie kompletny zbiór wzorców spełniających próg minimalnego poparcia w sposób maksymalnie efektywny i skalowalny przy minimalnej liczbie przeglądnięć bazy danych. Pożądana byłaby możliwość włączania dodatkowych warunków ograniczających liczbę szukanych wzorców w oparciu o wiedzę dziedzinową użytkownika.

Zaproponowane algorytmy można podzielić na dwie kategorie: oparte o zasadę Apriori (*Apriori-based approaches*) oraz o rozrost wzorców (*pattern-growth-based approaches*). Do pierwszej grupy można zaliczyć algorytm GSP (*Generalized Sequential Patterns*) i SPADE (*Sequential Pattern Discovery using Equivalent Class*), do drugiej zaś FreeSpan (*Frequent pattern-projected Sequential pattern mining*) [6] i PrefixSpan (*Prefix-projected Sequential pattern mining*) [12].

Pojęcia związane z eksploracją wzorców sekwencji oraz pierwszy – oparty o zasadę Apriori – algorytm zaproponowany został w [1], rozwinięty w GSP [10]. Na podobnej zasadzie oparty jest algorytm SPADE [15] – wprowadzono wertykalny format bazy. Zasada Apriori w odniesieniu do eksploracji sekwencji mówi, iż każda niepusta podsekwencja częstej sekwencji jest również częsta. Własność Apriori jest antymonotoniczna w tym sensie, że jeżeli dana sekwencja nie jest częsta, to żadna z sekwencji, która ją zawiera, nie jest również częsta. Prowadzi to do znacznego ograniczenia przestrzeni przeszukiwań. Zasada działania algorytmów tej grupy jest podobna: początkowo każdy element bazy jest traktowany jako kandydat o długości 1. Następnie dla każdego poziomu (czyli sekwencji o długości k) przeglądana jest baza danych w celu ustalenia poparcia każdej kandydackiej sekwencji. Te sekwencje, których poparcie przekracza próg minimalnego poparcia, służą do generowania kandydackich sekwencji o długości k+1 przy wykorzystaniu algorytmu Apriori do momentu, aż nie nie będzie już ani kandydackich, ani częstych sekwencji.

Mimo że Apriori ogranicza znacząco przestrzeń przeszukiwań, bo średnio aż o 44,57% kandydatów, nie jest wydajny ze względu na konieczność wielokrotnego przeglądania bazy danych w celu wyliczenia poparcia kandydatów oraz generowanie ogromnej liczby kandydackich sekwencji (w szczególności o długości 2).

Pierwszy z wyżej wymienionych problemów próbowano rozwiązać w algorytmie SPADE, wprowadzając wertykalny format bazy danych sekwencji, w którym każda krotka reprezentuje sekwencję oraz listę zdarzeń, w której elementy sekwencji występują. W podejściu tym wymagane jest jednokrotne przeglądnięcie bazy w celu przekształcenia jej do wertykalnego formatu, a wyliczenie poparcia sekwencji o długości k odbywa się przez połączenie list dla dwóch podsekwencji o długości k–1. W dalszym ciągu jednak problemem było generowanie ogromnej liczby sekwencji kandydatów, co wyjątkowo nieefektywne zwłaszcza przy długich sekwencjach, gdyż powstają one z wykładniczej liczby krótkich sekwencji kandydatów. Próbę stworzenia algorytmu bez potrzeby generowania sekwencji kandydatów zaproponowano w algorytmie FreeSpan, a następnie zoptymalizowano w PrefixSpan. Etap generowania sekwencji kandydujących i sprawdzania, czy jej poparcie spełnia wymóg minimalnego poparcia, zastąpiony jest w tych algorytmach etapem polegającym na budowie i analizie prefiksów sekwencji należących do bazy danych sekwencji. Algorytm PrefixSpan generuje rekurencyjnie wzorce sekwencji o rozmiarze k w oparciu o wzorce sekwencji o rozmiarze k–1.

Mając dane dwie sekwencje $\alpha = < a_1\ a_2 \ldots a_n >$ i $\beta = < b_1\ b_2 \ldots b_m >$, $m < n$, sekwencja β jest prefiksem α wtedy i tylko wtedy gdy:

- $b_i = a_i$ dla $i \leq m–1$;
- $b_m \subseteq a_m$;
- wszystkie elementy w $(a_m–b_m)$ są w alfabetycznym porządku po tych z b_m.

Główną ideą tych dwóch algorytmów jest wykorzystanie częstych elementów do rekurencyjnej projekcji bazy sekwencji na mniejsze części – projekcyjne bazy danych (partycje) i rozrost sekwencji w każdej takiej partycji. W rezultacie algorytm składa się z trzech głównych kroków:

1. Znajdowanie częstych sekwencji o długości 1, co wymaga jednokrotnego przeglądnięcia bazy danych S.
2. Dzielenie przestrzeni poszukiwań na partycje. Liczba partycji zgodna z liczbą częstych jednoelementowych sekwencji znalezionych w kroku 1, gdyż dowolny wzorzec sekwencji może rozpoczynać się tylko od częstego elementu.
3. Analiza podzbiorów sekwencji przechowywanych w poszczególnych partycjach. W każdej z nich wyszukiwane są podsekwencje częstych sekwencji, tworzone są kolejne projekcyjne bazy danych dla znalezionych podsekwencji częstych itd.

5. Koncepcja środowiska do analiz ruchu

5.1. Charakterystyka środowiska

Dane zostały wygenerowane systemem agentowym Kraksim służącym do symulacji ruchu miejskiego. Symulacja odbywa się przy wykorzystaniu automatów komórkowych. Sterowanie ruchem poszczególnych pojazdów odbywa się przez niezależne agenty, analizujące posiadane dane na temat stanu modelu. Model ma postać grafu, którego wierzchołki

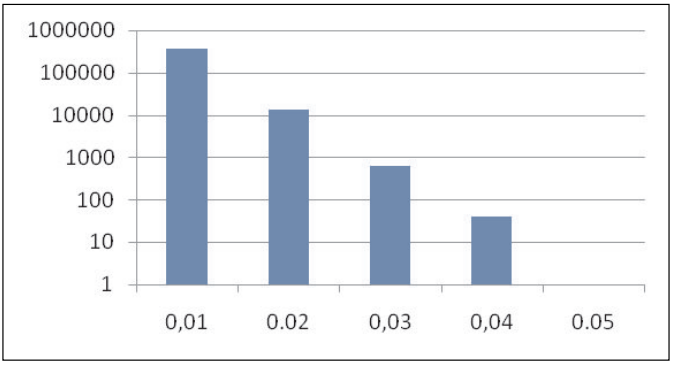
stanowią skrzyżowania i bramy, zaś krawędziami są ulice łączące poszczególne skrzyżowania i bramy. Każdemu ze skrzyżowań przypisane są fazy, które kolejno się zmieniają. Dla każdej z faz zdefiniowane są te ulice wchodzące do skrzyżowania, z którego można przejechać przez skrzyżowanie bez groźby kolizji z innymi pojazdami. Za zmianę faz skrzyżowań również odpowiedzialne są autonomiczne agenty. W systemie Kraksim zaimplementowane zostały również zarówno statyczne, jak i dynamiczne mechanizmy zmiany fazy świateł na skrzyżowaniach – pierwsze z nich zakładają zmiany fazy w określonych odstępach czasowych, niezależnie od natężeń ruchu, drugie natomiast analizują natężenia na ulicach dochodzących do skrzyżowania i dostosowują długość trwania poszczególnych faz do aktualnego zapotrzebowania. W celu zbliżenia modelu symulacyjnego do rzeczywistości dodano możliwość tworzenia ulic składających się z wielu pasów oraz równoważenia obciążenia na pasach. System został rozbudowany o edytor map pozwalający na nanoszenie wygenerowanych danych na mapę centrum Krakowa.

Do przechowywania wybranych danych wygenerowanych przez KrakSim'a, na których przeprowadzono eksperyment wyszukiwania częstych sekwencji zaprojektowano i zaimplementowano schemat bazy danych (zaimplementowany w systemie bazodanowym Oracle 10g[1]).

5.2. Wybrane eksperymenty

Celem przeprowadzonych eksperymentów było znalezienie sekwencji dróg w centrum Krakowa najczęściej wybieranych przez kierowców. Każda droga charakteryzuje się punktem początkowym, punktem końcowym oraz średnią prędkością jej przejazdu.

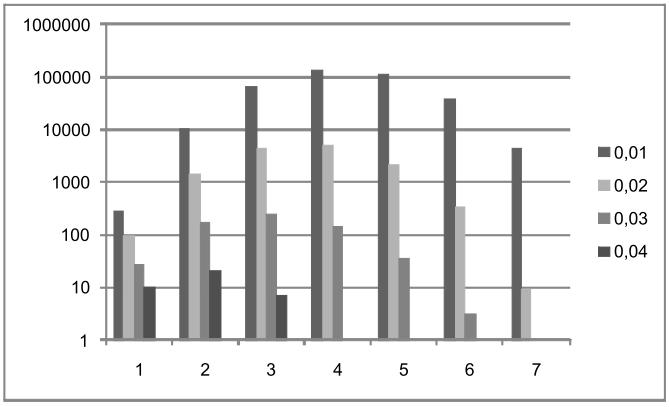
W pierwszym eksperymencie próbowano znaleźć zależność między liczbą znajdowanych sekwencji a wartością współczynnika minimalnego poparcia. Znaleziono 385 618 częstych sekwencji, ich liczba znacząco malała wraz z obniżaniem współczynnika minimalnego poparcia (rys. 2).



Rys. 2. Liczba częstych sekwencji w zależności od min_sup

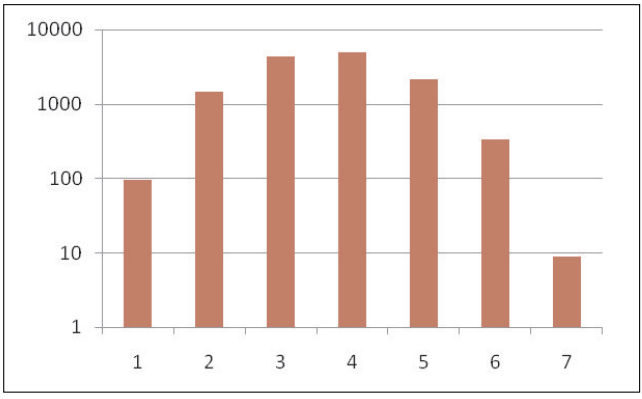
[1] http://www.oracle.com/technetwork/database/express-edition/overview/index.html

Dodatkowo wraz ze wzrostem współczynnika minimalnego poparcia malała długość sekwencji, co widać na rysunku 3.



Rys. 3. Liczba częstych sekwencji określonej długości w zależności od min_sup

W ramach innego eksperymentu próbowano znaleźć najbardziej wpływowe drogi w mieście. Parametr minimalnego poparcia ustawiono na 0,02 a maksymalną długość sekwencji na 8 (założono, że cykle nie mogą występować), a zbiór danych wejściowych liczył 9240 dróg. Uzyskano 2 072 804 sekwencje, spośród których 13 357 było częstych. Zależność liczby sekwencji od jej długości przedstawiono na rysunku 4.



Rys. 4. Liczba częstych sekwencji o określonej długości

W znalezionych częstych sekwencjach próbowano znaleźć drogi, które rozpoczynały najwięcej częstych sekwencji. Po naniesieniu na plan miasta okazało się, że najwięcej częstych sekwencji rozpoczynało się od dróg zlokalizowanych w okolicach ulicy Długiej (ponad połowa znalezionych częstych sekwencji)

6. Podsumowanie

W rezultacie przedstawionych prac dokonano analizy stanu ruchu drogowego wygenerowanego przy użyciu symulatora. Dalsze prace obejmują wykorzystanie opracowanych narzędzi od analizy danych na temat ruchu uzyskanych w oparciu o dane pozyskane z Zarządu Infrastruktury Komunalnej i Transportu w Krakowie.

Autorzy dziękują magistrantom i studentom Katedry Informatyki AGH, którzy uczestniczyli w realizacji symulatora oraz przeprowadzanych eksperymentów, a zwłaszcza panom J. Martynie i M. Skowronowi, którzy przygotowali i przetestowali narzędzia do analizy uzyskanych danych na temat ruchu drogowego, wykorzystując metodę krótkich sekwencji.

Literatura

[1] Agrawal R., Srikant R., *Mining sequential patterns*. ICDE'95. 1995.
[2] Bajwa S., Chung E., Kuwahara M., *Performance evaluation of an adaptive travel time prediction model*. Intelligent Transportation Systems, 2005. Proceedings. 2005 IEEE, vol., no., pp. 1000–1005, 13–15 Sept. 2005.
[3] Bazzan A.L.C., Klügl F., *Multi-agent systems for traffic and transportation engineering*. Information Science Reference, 2009.
[4] Chung E., Chung, E., *Classification of traffic pattern*. 10th World Congress on Intelligent Transport Systems, Madrid, Spain, 2003.
[5] De Fabritiis, C., Ragona, R., Valenti G., *Traffic Estimation And Prediction Based On Real Time Floating Car Data*. 2008 11th International IEEE Conference on Intelligent Transportation Systems, 2008, 197–203.
[6] Han J., Pei J., MortazaviI-Asl B., Chen Q., Dayal U., Hsu M.-C., *FreeSpan: Frequent Pattern-Projected Sequential Pattern Mining*. KDD'0), Boston, MA, August 2000.
[7] Han J., Kamber M., *Data Minining. Concepts and Techniques*. Morgan Kaufmann Publishers, 2006.
[8] Koźlak J., Dobrowolski G., Kisiel-Dorohinicki M., Nawarecki E., *Anti-crisis management of city traffic using agent-based approach*. Journal of Universal Computer Science, 2008, vol. 14, iss. 14, 2359–2380.
[9] Mark C.D., Sadek, A.W., Rizzo, D., *Predicting experienced travel time with neural networks: a PARAMICS simulation study*. Intelligent Transportation Systems, 2004. Proceedings. The 7th International IEEE Conference on , vol., no., pp. 906–911, 3–6 Oct. 2004.
[10] Srikant R., Agrawal R., *Mining sequential patterns: Generalizations and performance improvements*. EDBT'96, 1996. [GSP]
[11] Theodoridis S., Koutroumbas K., *Pattern Recognition. Fourth Edition*. Elsevier, 2009.
[12] Pei J., Han J, Pinto H., ChenQ., Dayal U., Hsu M.-C., *PrefixSpan: Mining Sequential Patterns Efficiently by Prefix-Projected Pattern Growth*. ICDE'01, Heidelberg, 2001.
[13] Wei-Hsun L., Shian-Shyong T., Sheng-Han T., *A knowledge based real-time travel time prediction system for urban network*. Expert Systems with Applications, vol. 36, iss. 3, Part 1, April 2009, 4239–4247.
[14] Yan X., Han J., Afshar R., *CloSpan: Mining Closed Sequential Patterns in Large Datasets*. SDM'03, 2003.
[15] Zaki M., *SPADE: An Efficient Algorithm for Mining Frequent Sequences*. Machine Learning, 2001.