УДК 004.9

В.И. Виноградов, В.Г. Мазнев

МЕТОД ПОИСКА ЧАСТО ПОВТОРЯЮЩИХСЯ МАРШРУТОВ В ПРОСТРАНСТВЕННО-ВРЕМЕННЫХ ДАННЫХ

Предложено рассматривать применение анализа для решения задач трафика в целях получения данных о наиболее частых маршрутах. Разработана модель периодического шаблона.

E-mail: iu5vmch@rambler.ru

Ключевые слова: *модель периодического шаблона, пространственно-временные данные.*

Введение. Во многих областях нашей жизни движения объектов периодически повторяются, т. е. объекты следуют теми же маршрутами (приблизительно) относительно некоторых постоянных интервалов времени. К таким объектам относятся транспортные средства (ТС) (автобусы, лодки, самолеты, поезда и т. д.), пользователи мобильных телефонов, оснащенных системой навигации и пр. Благодаря широкому распространению портативных приборов с возможностью определения местоположения (GPS- и ГЛОНАСС-приемники) и накоплению большого объема исторических данных о перемещении, стало возможным анализировать эти данные и получать знания о закономерностях перемещений [1]. Решение такой задачи может иметь различные области применения:

— анализ трафика (выявление наиболее частых маршрутов заданной периодичности для учета при реконструкции дорожной сети и ее развитии);

— предоставление услуг на основе местоположения (улучшение качества услуг за счет использования текущего местоположения объекта, его поведения и типичных маршрутов) [2];

— в военных системах слежения (выявление типичного поведения и маршрутов патрулирования объектов);

— шаблоны движения метеорологических объектов для использования при прогнозировании;

— пространственно-временные шаблоны в произвольном многомерном пространстве свойств.

Далее предложено рассматривать применение анализа для решения задач трафика в целях получения данных о наиболее частых маршрутах. В заданном регионе существует большой парк ТС, оборудованных приборами определения местоположения; данные о местоположении этих ТС регулярно передаются и сохраняются в единую базу данных. Существует задача выявления наиболее частых периодически повторяющихся (ежедневно) маршрутов с заданной периодичностью для передачи этих знаний группе экспертов дорожного строительства. Похожие данные о загруженности дорог собираются такими системами, как Яндекс.

Пробки, Google, Rambler и др. В системе Яндекс.Пробки накапливаются статистические данные о загруженности конкретных участков дорог, затем эти данные используются при планировании маршрутов. Однако ни в этой системе, ни в аналогичных системах не анализируются сами маршруты, вызывающие подобные затруднения. Такие знания необходимо получать с применением более сложных методов анализа [3]. Для этого требуется выработать методы решения и алгоритмы, а также реализовать на их основе программный комплекс для установления закономерностей движения объектов на базе данных положений этих объектов. Под закономерностями понимается часто повторяющиеся передвижения объекта с определенной периодичностью, так называемые периодические шаблоны. Периодический шаблон представляют как не непрерывные последовательности положений объекта, которые периодически появляются в истории движения, так как не предполагается, что объект посещает точно те же места в то же время в рамках периода. Таким образом, шаблоны несколько размытые. Сформулированную задачу можно отнести к задаче поиска последовательных шаблонов.

Модель периодического шаблона. Траекторией объекта S называют последовательность длиной n пространственно-временных координат:

$$S = \{(l_0; t_0), (l_1; t_1), ..., (l_{n-1}; t_{n-1})\},\ n = |S|, \qquad (1)$$

где l_i — положение объекта в момент времени t_i, выраженное в пространственных координатах.

Если разность временных меток одинакова, то последовательность (1) представляют как $S = \{l_0, l_1, ..., l_{n-1}\}$.

Пусть $S = \{l_0, l_1, ..., l_{n-1}\}$ — движение объекта; T — период (например, день, неделя, месяц), $T \ll n$. Периодический сегмент s определяется подпоследовательностью $l_i, l_{i+1}, ..., l_{i+T-1}$ из последовательности S, причем остаток от деления i на T равен нулю. Таким образом, сегменты начинаются в позициях 0, T, $\left(\lfloor n/T \rfloor - 1\right)T$ и имеется $m = \lfloor n/T \rfloor$ периодических сегментов в последовательности S (если n не кратно T, то все последние $n \bmod T$ положений отбрасываются, и длина n последовательности S уменьшается). Пусть s^j — сегмент, начиная с положения l_{jT} для S, $0 \le j < m$, и $s_i^j = l_{jT+i}$ для i, $0 \le i < T$.

Периодический шаблон P — последовательность вида $r_0, r_1, ..., r_{T-1}$ длиной T, r_i — пространственная область или вся пространственная вселенная. Длина шаблона P равна числу небесконечных регионов.

Сегмент s^j удовлетворяет шаблону P, если для всех $r_i \in P, r_i = *$ или s_i^j находится в пространственной области r_i.

Поддержкой $|P|$ шаблона P называется число периодических сегментов s из последовательности S, удовлетворяющих данному шаблону.

Пусть минимальная поддержка min_sup будет действительным числом из диапазона значений (0; 1]. Тогда шаблон P называется ***частым шаблоном***, если его поддержка $|P|$ больше минимальной m min_sup.

Каждая область r_i в шаблоне P является ***правильной***, если множество положений $R_i^P = \{s_i^j \mid s^j \in S^P\}$ формирует ***плотный кластер*** [4]. Шаблон P называется правильным, если все его не небесконечные регионы правильны.

Задача поиска периодически повторяющихся шаблонов заключается в поиске всех правильных периодических шаблонов P в последовательности S, которые являются частыми и неизбыточными по отношению к минимальной поддержке min_sup.

В качестве общего метода решения этой задачи предложено использовать:

1) дискретизацию точек в пространственно-временной системе координат с заданным интервалом времени. Таким образом, для каждой точки имеем последовательность координат;

2) разделение последовательности движения каждого объекта на множество последовательностей с длиной, равной периоду, для которого находятся шаблоны;

3) кластеризация точек в пространстве для выделения значимых регионов и нивелирования колебаний при определении координат;

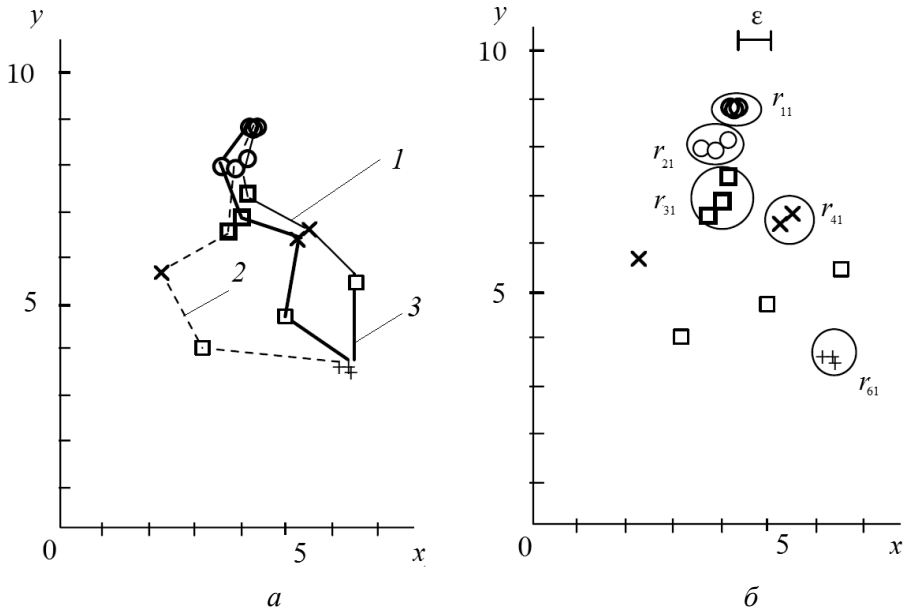
4) поиск периодически повторяющихся последовательностей и их валидация, если это необходимо;

5) визуализация результата.

Рассмотрим нахождение частых шаблонов единичной длины, затем — существующих алгоритмов поиска периодических шаблонов, в которых в качестве параметра принимается набор частых шаблонов единичной длины.

Для поиска шаблонов единичной длины можно применять следующую методологию. Последовательность S разбивается на T пространственных наборов данных, по одному на каждый отступ в период T (часть *а* рисунка). Другими словами, положения $\{l_i, l_{i+T}, ..., l_{i+(m-1)T}\}$ образуют множество R_i (часть *б* рисунка) для каждого $0 \le i < T$. Каждому положению присваивается идентификатор $j \in [0; m-1]$ сегмента, содержащего его.

Следует отметить, что плотный кластер r в наборе данных R_i (часть *б* рисунка) соответствует частому шаблону, у которого на i-м месте r, а на остальных символ «∗». Чтобы определить плотные кластеры, необходимо использовать алгоритм кластеризации по плотности, например DBSCAN [5]. Кластеры с числом точек меньшим $\alpha (\alpha = m \min\_\sup)$ отбрасываются как нечастые. Оригинальный алгоритм DBSCAN имеет квадратичную вычислительную стоимость относительно числа точек для кластеризации, поэтому алгоритм, основанный на хешировании, используется для уменьшения стоимости [6].



Траектория, разбитая на сегменты и регионы:

а — декомпозиция по периоду; *б* — плотные кластеры среди множества R_i, $i = 1, \ldots, 6$; *1*, *2*, *3* — первый, второй и третий дни соответственно

Рассмотрим поуровневый подход «снизу вверх», предложенный в алгоритме STPMine 1. Начиная с шаблонов единичной длины, в таком подходе применяется вариация алгоритма априори (apriori — TID) для нахождения более длинных последовательностей. Входными данными для алгоритма является набор частых однопоследовательностей [7, 8]. Пары $< P_1, P_2 >$ шаблонов длиной $k - 1$ из L_{k-1} с их первыми $k - 2$ небесконечными областями, находящимися в одинаковых позициях, и разными не небесконечными $k - 1$ позициями образуют кандидат в шаблон длиной k (строки 4—6). Для каждого кандидата P_{cand} осуществляется соединение множеств по идентификатору сегмента между P_1 и P_2. Если число сегментов, удовлетворяющих обоим шаблонам, равно минимальной поддержке, происходит проверка, являются ли регионы шаблона P_{cand} по-прежнему кластерами. После того, как все шаблоны длиной k найдены, определяются шаблоны следующего уровня до тех пор, пока еще есть шаблоны на данном уровне, и еще есть уровни.

Псевдокод алгоритма выглядит следующим образом:

1) k:=2
2) пока $(L_{k-1} \neq 0 \wedge k < T)$
3) $L_k := 0$;
4) для каждой пары $(P_1, P_2) \in L_{k-1}$
5) такой, что P_1, P_2 совпадают по первым $k - 2$

6) и отличаются в k–1-й небесконечной позиции

7) $P_{cand} :=$ сгенерировать_кандидат(P_1, P_2)

8) если $P_{cand} \neq \text{null}$, то

9) $P_{cand} := \text{P}_1 \triangleright\triangleleft_{P_1 sid = P_2 sid} P_2$

10) если $|P_{cand}| \geq m \min\_\text{sup}$, то

11) проверить_шаблон $(P_{cand}, L_k, \min\_\text{sup})$

12) $k := k + 1$

13) вернуть $P := \cup L_k, \forall 1 \leq k \leq T.$

Рассмотрим двухфазный подход «сверху вниз», реализованный в алгоритме STPMine 2. Первая фаза алгоритма STPMine 2 замещает каждое положение в последовательности S идентификатором кластера, к которому он принадлежит, или пустым значением (например, *), если не принадлежит никакому кластеру. Следовательно, изначальная последовательность S преобразуется в символьную последовательность S`.

Алгоритм, предложенный Ж. Ханом и Я. Йином [9], можно использовать для быстрого поиска всех частых шаблонов вида $r_0, r_1, ..., r_{T-1}$, где r_i — кластер в R_i или *. Тем не менее неизвестно, являются ли результаты алгоритма, основанного на последовательностях, реальными шаблонами, поскольку значение каждой небесконечной позиции могут и не формировать кластер. Шаблоны P`, генерируемые этим алгоритмом, называются псевдошаблонами, поскольку они могут быть ошибочными.

Для нахождения реальных шаблонов необходимо применить некоторые изменения к исходному алгоритму. Во время создания дерева максимальных подшаблонов каждый узел хранится с идентификаторами сегментов, которые соответствуют псевдошаблону узла. Таким образом, один идентификатор сегмента идет только в один узел дерева. Последовательность S может быть слишком большой, чтобы содержаться в памяти. Для решения этой проблемы во время сканирования последовательности S для каждого сегмента s выполняются следующие действия:

1) добавление сегмента в дерево максимальных подшаблонов, в результате увеличивается счетчик шаблона кандидата;

2) вставка записи вида $< P`.id, s.id >$ в файл $F$, где $P`.id$ — идентификатор узла, соответствующего данному псевдошаблону; $s.id$ — идентификатор сегмента. В конце файл F сортируется по $P`.id$, чтобы собрать вместе идентификаторы сегментов, удовлетворяющих одному и тому же (максимальному) псевдошаблону. Для каждого псевдошаблона с хотя бы одним сегментом вставляется указатель на положение в файле первого идентификатора сегмента.

Для каждого псевдошаблона с хотя бы $\alpha(\alpha = m\min\_\text{sup})$ сегментами вызывается функция проверки шаблона, чтобы получить потенциально правильные шаблоны.

Псевдокод алгоритма выглядит следующим образом:

1) построить дерево максимальных подшаблонов *Tr* и файл шаблонов *F*
2) отсортировать *F* по $P`.id$ и соединить с узлами *Tr*
3) для каждого $k:=T$ уменьшая до 2
4) для каждого шаблона $P`$ на уровне *k* дерева *Tr*,
5) $|P`|:= P`.\text{счетчик} + \sum_{P``\supset P`, \text{длина}(P``)=k+1} |P``|$
6) если $|P`| \geq m \min\_\sup$ то
7) $P_{cand} := \cup_{P``\supseteq P`} P``.\text{идентификаторы}$
8) проверить_шаблон $(P_{cand}, L, \min\_\sup)$
9) если P изменился, то
10) удалить из $P`$ все идентификаторы,
11) входящие в новые шаблоны P
12) если нераспределенных идентификаторов <m min_sup то
13) вернуть P
14) вернуть P

Существует алгоритм STPMine 2-V2, который аналогичен алгоритму STPMine 2, за исключением отсутствия в данном алгоритме проверки шаблонов и переиндексации сегментов (строки 8—12). Вследствие этого алгоритм менее точен и может содержать неверные шаблоны, регионы шаблонов совпадают с регионами частых одношаблонов, полученных на первом шаге. Однако алгоритм STPMine 2-V2 работает значительно быстрее описанных выше двух алгоритмов.

Результаты сравнения существующих алгоритмов. Единственным алгоритмом, который находит все шаблоны, является STPMine 1, но он обладает очень низкой скоростью работы и высокой ресурсоемкостью. Алгоритм STPMine 2 более быстрый алгоритм, однако он определяет только максимальные шаблоны, тем самым менее частые, но не менее полезные знания (шаблоны) могут быть пропущены. Алгоритм STPMine 2-V2 применим там, где важна скорость работы, чем качество полученных результатов. Для описанной задачи ни один из данных алгоритмов не подходит. Кроме того, рассмотренные алгоритмы не устойчивы к сдвигам и неточностям во времени. Например, в следующей последовательности этот же шаблон может повторяться с небольшим сдвигом по времени как для всего шаблона, так и отдельные его регионы. Такие шаблоны не смогут быть найдены. Необходимо проводить доработку модели поиска периодических шаблонов и разрабатывать новый метод, использование которого позволит соответствовать требованиям по производительности и качеству поиска.

Заключение. В статье были рассмотрены проблема выявления наиболее часто используемых маршрутов с заданной периодичностью, а также существующие программные комплексы. Описана модель периодически повторяющихся шаблонов в пространственно-временных данных, предложено общее решение и сформирована задача нахождения частых периодически повторяющихся шаблонов. Проанализированы существующие алгоритмы, выявлены их недостатки и обоснована необходимость разработки нового решения.

СПИСОК ЛИТЕРАТУРЫ

1. Hadjieleftheriou M., Kollios G., Tsotras V.J., Gunopulos D. Efficient Indexing of Spatiotemporal Objects // In Proc. of Extending Database Technology Conference. 2002. — P. 251—268.
2. Pfoser D, Jensen C. S. and Theodoridis Y. Novel Approaches in Query Processing for Moving Object Trajectories // In Proc. of Very Large Data Bases Conf. 2000. P. 395—406.
3. Yavas G., Katsaros D., Ulusoy O., Manolopoulos Y. A Data Mining Approach for Location Prediction in Mobile Environments // Data and Knowledge Engineering. 54(2). 2005. P. 121—146.
4. Чубукова И.А. Data Mining. — М.: ИУИТ, 2008.
5. Барсегян А.А., Куприянов М.С. Технологии анализа данных. Data Mining, Visual Mining, Text Mining, OLAP. — СПб.: БХВ-Петербург, 2007.
6. Ester M., Kriegel H.P., Sander J., Xu X. A Density-based Algorithm for Discovering Clusters in Large Spatial Databases with Noise // In Proc. of International Conference on Knowledge Discovery and Data Mining. 1996. — P. 226—231.
7. Cao H., Mamoulis N., Cheung D.W. Discovery of Periodic Patterns in Spatiotemporal Sequences // IEEE Trans. Knowl. Data Eng. 19(4). 2007. — P. 453—467.
8. Agrawal R., Srikant R. Fast Algorithms for Mining Association Rules // In Proc. Of Very Large Data Bases Conference. 1994. — P. 487–499.
9. Han J., Dong G., Yin Y. Efficient Mining of Partial Periodic Patterns in Time Series Database // In Proc. of International Conference on Data Engineering. 1999. — P. 106—115.

Статья поступила в редакцию 4.07.2012