находить множество отметок всех путей из начальной вершины во все финальные.

Литература.

1. Ахо А. Построение и анализ вычислительных алгоритмов / А. Ахо, Дж. Хопкрофт, Дж. Ульман. – М. : Мир, 1979. – 536 с.

2. Хопкрофт Дж. Введение в теорию автоматов, языков и вычислений. 2-е издание / Дж. Хопкрофт, Р. Мотвани, Дж. Ульман. – М. : Издательский дом "Вильямс", 2002. – 528 с.

3. Ногина Н.В. Анализ языков, порожденных помеченными графами. / Н.В. Ногина, И.С. Грунский // Тезисы докладов международной научно-технической конференции «Системный анализ и информационные технологии» (SAIT 2012). – Киев, 2012. – в печати.

Савельев О.О.

Науч. руководитель чл.-кор. НАН Украины, д.т.н., профессор Шевченко А.И.

Институт информатики и искусственного интеллекта ДонНТУ

Построение динамического социального графа по транзакционным данным трафиков телефонных сетей

Трафики телефонных сетей (ТТС) – метаданные, содержащие информацию о времени, типе и длительности событий, идентификаторы абонентов, но не содержащие самих сообщений. ТТС можно рассматривать как транзакционные данные, генерируемые абонентами, множество которых представляет социальную сеть. Задачи анализа ТТС: поиск связей между объектами; профилирование объектов; обнаружение аномальных

ситуаций, могут решаться методами из области анализа социальных сетей. В данной работе сделана попытка формализовать ТТС как модель, описывающую состояние социальной сети.

Социальная сеть – структура, образованная множеством акторов, и отношений между ними [1]. Социальную сеть можно рассматривать как граф. Классически методы социально-сетевого анализа применялись к конечным статическим графам ввиду того, что информация о сети собиралась социологическими методами. Для ТТС характерно точное и полное отражение взаимодействия объектов с детализацией времени и продолжительности. Поэтому генерация динамического социального графа абонентов является актуальной задачей.

Динамический граф можно представить как

$$DG = \{ G^{t_1}, ..., G^{t_T} \}; G^{t_i} = \{ V^{t_i}, E^{t_i} \}, \quad (1)$$

где t_i – время окончания отсчета (время дискретно)

$$t_i \in \{ t_0, t_1, ..., t_n^T \}; \quad (2)$$

$G^{t_1}, ..., G^{t_n^T}$ – статические графы – последовательность состояний графа (1) в промежутки времени (2), каждый задается своим множеством вершин V^{t_i} и ребер E^{t_i}. Ребро определяется парой вершин и имеет вес

$$e = \{ v_i, v_j, w \}. \quad (3)$$

В работе [2] рассмотрена реляционная модель хранилища данных, в которое загружаются «сырые» ТТС из файлов. На рисунке 1 представлен фрагмент модели.

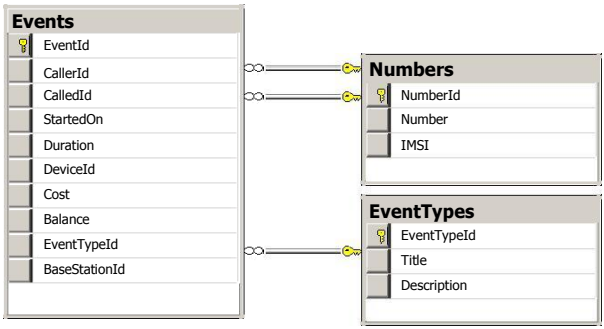

Рисунок 1 – Фрагмент реляционной модели хранилища ТТС

Для данной модели возможна автоматическая генерация моделей слоя доступа к данным в терминах классов ООП с помощью ORM фреймворков, пример диаграммы классов (использован фреймворк LINQ2SQL) показан на рисунке 2. Сгенерированные классы сущностей предметной области предоставляют набор CRUD операций, что практически исключает необходимость написания SQL запросов.

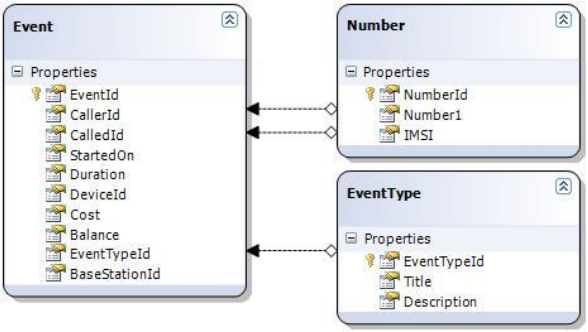

Рисунок 2 – Диаграмма классов объектно-реляционного отображения

Рассмотрим следующий алгоритм построения динамического графа. Исходные данные: классы сущностей предметной области, т.е. множества событий

Events, телефонных номеров *Numbers*, типов событий *EventTypes*, величина периода квантования $\Delta t = t_{i+1} - t_i$.

Выходные данные: динамический граф (1).

Шаг 1. Отбор событий. Для каждого события *Event* из множества *Events*, у которого $StartedOn \in \left[t_1 , t_{nT} \right]$.

Шаг 2. Определение состояния. Для события определяется $t_i : t_{i-1} < StartedOn < t_i$. Проверяется существование графа G^{t_i}, при необходимости он определяется с пустыми множествами V^{t_i}, E^{t_i}.

Шаг 3. Генерация ребер. По событию определяется ребро $e = \{ CallerId, CalledId, w \}$ (в случае наличия ребра во множестве E^{t_i}, используется существующее), у которого вес w увеличивается на величину $a(e, t_i)$ – функция приращения веса, определяющего его в зависимости от типа события и длительности (в простейшем случае – это длительность для звонка и фиксированная величина для текстового сообщения). При необходимости ребро e заносится во множество E^{t_i}.

Шаг 4. Генерация вершин. Для каждого ребра e из множества E^{t_i}, путем сопоставления идентификаторов *CallerId,CalledId* с идентификатором *NumberId* элементов множества *Numbers* определяются две вершины и заносятся во множество V^{t_i}, если их там еще нет.

Шаг 5. Переход на шаг 2 для следующего события, либо конец алгоритма, если все события просмотрены.

Полученные состояния динамического графа могут использоваться в качестве исходных данных для дальнейшего кластерного и секвенциального анализа сети.

Литература.

1. Wasserman S. Social Network Analysis: Methods and Applications / Stanley Wasserman, Katherine Faust – New York : Cambridge University Press, 1994. – 857 p.

2. Савельев О.О. Особенности разработки подсистемы хранения информации для системы поддержки принятия решений в области анализа телекоммуникационных данных / О.О. Савельев, А.И. Шевченко // Материалы VI международной научно-технической конференции студентов, аспирантов и молодых ученых «Информатика и компьютерные технологии». – Донецк : ДонНТУ. – 2010. С. 343-349.