Решение задач визуализации и поиска мотивов в электронной библиотеке фольклорных текстов

© Н. Д. Москин

Петрозаводский государственный университет
moskin@karelia.ru

Аннотация

В данной статье описаны методы исследования электронной коллекции фольклорных текстов с представлением их семантической структуры в виде теоретико-графовых моделей. В частности рассмотрены вопросы хранения и визуализации моделей, а также методика обнаружения схожих мотивов, основанная на структурном соответствии графов и подграфов.

1 Введение

В предыдущих докладах на конференцию RCDL [1, 3] автором была рассмотрена электронная коллекция фольклорных песен Заонежья конца XIX – начала XX века, основанная на представлении семантической структуры текстов в виде теоретико-графовых моделей. В дальнейшем эту коллекцию дополнил корпус Лужских песен Городенского хора, тексты духовных стихов о Голубиной книге, записи о народных святых Нижегородского края. В [2] был предложен проект специализированного Интернет-ресурса для представления и анализа фольклорных коллекций. Результаты анализа текстов, а также методику получения этих результатов можно записать в специальном формате, например, в формате RuleML, основанном на технологии XML [4]. Это позволит применить результаты исследования другими специалистами в рамках деятельности сетевых научных сообществ.

В данной работе рассмотрены вопросы хранения и визуализации теоретико-графовых моделей, а также методика обнаружения схожих мотивов, основанная на структурном соответствии графов и подграфов. Для хранения исходных фольклорных текстов коллекции и их теоретико-графовых моделей предусмотрен специальный формат TextGML (Textual Graph Modelling Language), разработанный на основе XML. Этот формат позволяет сохранить исходный текст и его характеристики, объекты и отношения в тексте, описать упорядоченность элементов семантической структуры, ее иерархическую организацию, выделить фольклорные мотивы и предоставить возможности для дальнейшего анализа.

При создании, редактировании и исследовании теоретико-графовых моделей необходимо использовать инструменты их визуализации. В третьей части описывается алгоритм отображения теоретико-графовых моделей фольклорных текстов, представляющий собой модификацию метода визуализации графов на основе физических аналогий. В четвертой части предложено решение проблемы поиска схожих мотивов на основе алгоритма поиска изоморфизма подграфу [17].

2 Описание теоретико-графовых моделей фольклорных текстов ня языке TextGML

В настоящее время разработано несколько общепризнанных стандартов описания графов и графовых моделей на основе технологии XML. Одним из предшественников таких форматов является язык GML (Graph Modelling Language) [16], который появился в результате работы, начатой на конференции «Graph Drawing-1995» в Пассау и завершенной на «Graph Drawing-1996» в Беркли. GML до сих пор поддерживается многими прикладными программами и библиотеками для работы с графами.

В 2000 году на 8-ом симпозиуме «Graph Drawing» в Вильямсбурге был предложен язык описания графов GraphXML [15]. На этом языке могут быть описаны как абстрактные графы, так и более сложные структуры: иерархии графов, динамические графы и т. д. В это же время на симпозиуме «Graph Drawing-2000» комитетом «Graph Drawing Steering Committee» был начат проект GraphML (Graph Markup Language) [11]. Рабочая встреча относительно формата файла была проведена накануне симпозиума, и на ней было согласовано создание группы, которая определила новый, основанный на языке XML, формат файла, который должен, в конечном счете, лечь в основу стандарта описания графов.

На базе GML был также создан другой язык XGMML (eXtensible Graph Markup and Modeling Language). XGMML использует все основные теги GML, а также несколько дополнительных тегов, поэтому перевод графов из одного формата в другой

Труды 11[й] Всероссийской научной конференции «Электронные библиотеки: перспективные методы и технологии, электронные коллекции» - RCDL'2009, Петрозаводск, Россия, 2009.

осуществляется очень просто. Описание этого языка можно найти на сайте [13]. Другой язык GXL (Graph eXchange Language) также задумывался как стандартный формат для обмена графами [12]. Формально GXL описывает помеченные, ориентированные и упорядоченные графы, которые могут быть расширены до гиперграфов и иерархических графов. GXL поглотил такие спецификации как GraX [14], GRAph eXchange, Tuple Attribute Language и др.

Однако данные форматы предназначены для описания произвольных графов и графовых моделей, не привязанных к тексту. Для формального описания, хранения и изучения теоретико-графовых моделей текстов мы предлагаем использовать язык разметки TextGML (Textual Graph Modelling Language), разработанный на основе XML. DTD-описание данного языка изложено в работе [8]. В его основе лежат следующие элементы (теги):

- *tgml* – корневой элемент.
- *text* – элемент, определяющий границы текста. Элемент *text* имеет два атрибута: *name* – название текста и *type* – тип текста (например, «стихотворение», «басня», «статья», «эссе» и т. д.).
- *text_parameter* – характеристики текста (например, автор, год и место издания), которые определяются в виде элементов *parameter*. Каждому параметру соответствует два атрибута: *id* – идентификатор параметра и *name* – название параметра.
- *graph* – граф, соответствующий тексту. Каждый граф задается набором вершин (*node*) и ребер (*link*), соединяющих эти вершины. У элемента *graph* четыре атрибута: *id* – идентификатор графа, *name* – название графа (например, «дерево зависимостей первого предложения»), *type* – тип графа и *directed* – индикатор, указывающий, является ли граф ориентированным.
- *node* – структурные единицы текста. У этого элемента пять атрибутов: *id* – идентификатор вершины, *name* – название вершины (например, «основная форма слова»), *type* – тип вершины, *order* – порядок вершины в графе и *id_graph* – ссылка на идентификатор графа-потомка. Последний параметр позволяет организовать в тексте иерархию уровней графа, где граф низшего уровня является вершиной графа более высокого уровня.
- *link* – отношения между единицами текста. У данного элемента семь параметров: *id* – идентификатор ребра, *name* – название ребра, *source* и *target* – ссылки на идентификаторы вершины-источника и вершины-приемника, *type* – тип ребра (например, «однородность слов»), *cost* – сила связи и *order* - порядок ребра в графе.

Рассмотрим фрагмент бесёдной песни «Как назябло, навеяло лицо» из сборника В. Д. Лысанова «Досюльная свадьба, песни, игры и танцы в Заонежье Олонецкой губернии» (запись 1916 года, г. Петрозаводск) [6]:

Красна девица во тереме сидит, да
Жемчужное ожерельицо садит; да
Разсыпалось ожерельицо, да
По всему высоку терему. Да
Не собрать, не собрать жемчуга, да
Что ль ни батюшку, ни матушки, да
Что ль ни братцам, ни ясным соколам, да
Ни сестрицам, белым лебедям, да
А собрать соберет жемчужок, да
Разудалый, добрый молодец.

Теоретико-графовая модель этого текста выглядит следующим образом:

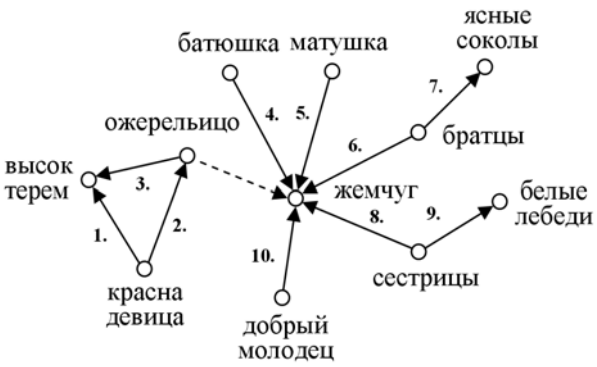


Рисунок 1. Теоретико-графовая модель фрагмента бесёдной песни «Как назябло, навеяло лицо»

Эту модель образуют шесть объектов группы «люди» (H), два объекта - «животный мир» (A), два объекта - «одежда, украшения» (CL), один объект - «постройки» (B). Единственная глобальная связь – отношение принадлежности – существует между объектами «ожерельицо» и «жемчуг», остальные связи локальные. Формат TextGML позволяет хранить сам текст, его характеристики, объекты и связи теоретико-графовой модели, их свойства: тип, упорядоченность, внутреннюю иерархию:

```
<tgml>
<text name="Как назябло, навеяло лицо" type=
"фольклорная песня">
<text_parameter>
<parameter id="p1" name="gubernia"> Олонецкая
</parameter>
<parameter id="p2" name="place_zap">1880-е годы
</parameter>
<parameter id="p3" name="sobiratel">В. Д. Лысанов
</parameter>
……
<parameter id="p8" name="movies">при игре парами
</parameter>
</text_parameter>
……
<!-- Мотив 1 -->
<graph id="g1" name="мотив 1" directed="true">
<node id="n1" type="H">Красна девица</node> во
<node id="n2" type="B">тереме</node> сидит, да
<node id="n3" type="CL">Жемчужное ожерельицо
</node> садит; да
```

```
<link id="l1" source="n1" target="n2" type="local"
order="1"/>
<link id="l2" source="n1" target="n3" type="local"
order="2"/>
</graph>

<!-- Мотив 2 -->
<graph id="g2" name="мотив 2" directed="true">
Разсыпалось <node id="n3" type="CL">
ожерельицо</node>, да
По всему <node id="n2" type="B">высоку тере-
му</node>. Да
Не собрать, не собрать <node id="n4" type="CL">
жемчуга</node>, да
Что ль ни <node id="n5" type="H">батюшку
</node>, ни <node id="n6" type="H">матушки
</node>, да
Что ль ни <node id="n7" type="H">братцам
</node>, ни <node id="n8" type="A">ясным соко-
лам</node>, да
Ни <node id="n9" type="H">сестрицам</node>,
<node id="n10" type="A">белым лебедям</node>,
да
<link id="l3" source="n3" target="n2" type="local"
order="3"/>
<link id="l4" source="n5" target="n4" type="local"
order="4"/>
<link id="l5" source="n6" target="n4" type="local"
order="5"/>
<link id="l6" source="n7" target="n4" type="local"
order="6"/>
<link id="l7" source="n7" target="n8" type="local"
order="7"/>
<link id="l8" source="n9" target="n4" type="local"
order="8"/>
<link id="l9" source="n9" target="n10" type="local"
order="9"/>
</graph>

<!-- Мотив 3 -->
<graph id="g3" name="мотив 3" directed="true">
А собрать соберет <node id="n4" type="CL"> жем-
чужок</node>, да
<node id="n11" type="H">Разудалый, добрый моло-
дец </node>.
<link id="l10" source="n11" target="n4" type="local"
order="10"/>
</graph>

<!-- Структура мотивов песни -->
<graph id="g4" name="граф мотивов" directed=
"true">
<!-- Мотивы песни -->
<node id="n12" name="мотив 1" type="motive"
id_graph="g1"/>
<node id="n13" name="мотив 2" type="motive"
id_graph="g2"/>
<node id="n14" name="мотив 3" type="motive"
id_graph="g3"/>
<node id="n15" name="песня" type="song"/>
<!-- Глобальные связи между вершинами -->
<link id="l11" source="n3" target="n4" type=
"global"/>
<!-- Глобальные связи между мотивами -->
<link id="l12" source="n15" target="n12" type=
"global"/>
<link id="l13" source="n15" target="n13" type=
"global"/>
<link id="l14" source="n15" target="n14" type=
"global"/>
</graph>
</text>
</tgml>
```

3. Визуализация теоретико-графовых моделей фольклорных текстов

При создании, редактировании и исследовании теоретико-графовых моделей фольклорных текстов необходимо иметь инструменты их визуализации. При этом алгоритм визуализации должен учитывать особенности построения фольклорного текста, и, следовательно, особенности теоретико-графовой модели. В методе, основанном на физических аналогиях, [5] граф рассматривается как система объектов с силами, взаимодействующими между этими объектами, где, например, вершины графа считаются телами, а ребра – пружинами. Рассмотрим модификацию этого метода для нашего случая [8]. Определим силу, приложенную к вершине u, по следующей формуле:

$$F(u) = \sum_{e=(u,v)\in E} f_e + \sum_{(u,v)\in V^2} g_{(u,v)} + \sum_{e=(u,v)\in E} h_e$$

где f_e – сила растяжения, действующая на вершину u из-за пружины (u,v). Здесь x-я координата силы f_e вычисляется по формуле:

$$f_e^x = k_e^{(1)}(d(u,v) - \lambda_1\left|p(u) - p(v)\right| - l_{\min})\frac{x_u - x_v}{d(u,v)},$$

где $d_{(u,v)}$ обозначает расстояние между u и v, $k_e^{(1)}$ – коэффициент жесткости (упругости) пружины между u и v. Чем он больше, тем сильнее пружина стремится установить расстояние между u и v, равным $l_e = l_{\min} + \lambda_1 \cdot \left|p(u) - p(v)\right|$. Подобное выражение для естественной длины пружины l_e, где $p(u)$ и $p(v)$ – номера слов в тексте, соответствующих объектам u и v, $l_{\min}$ – минимальная длина пружины, позволяет ближе расположить те объекты, которые в тексте находятся недалеко друг от друга. Коэффициент $\lambda_1 \geq 0$ характеризует значимость данного критерия.

Силу отталкивания $g_{(u,v)}$, существующую между вершинами u и v, определим по следующей формуле:

$$g_{(u,v)}^x = \frac{\lambda_2}{\Delta(u) + \Delta(v)} \frac{x_u - x_v}{(d(u,v))^3},$$

где $\Delta(u)$ – число ребер, инцидентных вершине u, λ_2 – коэффициент отталкивания, постоянный для всех вершин. Чем больше будет у вершины инцидентных ребер, тем меньше будет коэффициент отталкивания, а, следовательно, и сила $g_{(u,v)}$ для всех вершин v. Это позволит расположить вершины с большой степенью (основные персонажи фольклорного текста) в центре экрана, а вершины с меньшей степенью ближе к его границам.

Чтобы учитывать порядок появления связей в сюжете фольклорного текста, для каждого ребра $e=(u,v)$ введем дополнительную силу h_e. Эта сила будет стремиться расположить ребра графа как можно ближе к установленным заранее упорядоченным точкам $q_{(u,v)}$. Точки $q_{(u,v)}$ следует расположить последовательно на одинаковом расстоянии друг от друга по окружности (или полуокружности) с центром в середине экрана:

$$h^x_{e=(u,v)} = k^{(3)}_{(u,v)} \cdot d(q_{(u,v)}, c_{(u,v)}) \cdot (x_q - x_c),$$

где $c_{(u,v)}$ – центральная точка ребра (центр ребра), координаты которой вычисляются как среднее арифметическое координат вершин u и v, а $k^{(3)}_{(u,v)}$ – коэффициент силы притяжения между $q_{(u,v)}$ и $c_{(u,v)}$. Чем он больше, тем сильнее ребро $e=(u,v)$ стремится к точке $q_{(u,v)}$.

Алгоритм визуализации работает в два этапа: вначале вершины размещаются на плоскости случайным образом, затем выполняется последовательность итераций до стабилизации, на каждой из которых для всех вершин u вычисляется сила $F(u)$ и происходит перемещение вершины в направлении этой силы на расстояние, пропорциональное модулю силы [5]. Применение данного алгоритма позволит упорядочить элементы графа по мере их появления в сюжете фольклорного текста, сгруппировать вершины и ребра согласно структуре мотивов и их функциональному весу.

Обычно при анализе результатов визуализации графов используют ряд эстетических критериев, среди которых можно выделить следующие: минимизация пересечений ребер, минимизация области размещения, минимизация и унификация длин ребер, минимизация и унификация сгибов, максимальная симметричность и др. [5]. В нашем случае дополнительным критерием является упорядоченность ребер, отражающая развитие сюжета по времени.

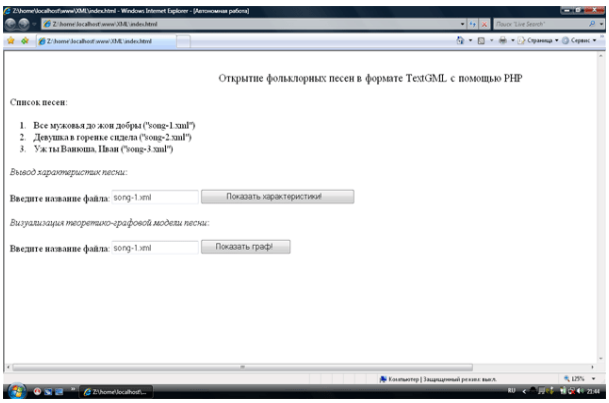


Рисунок 2. Процедура выбора фольклорного текста

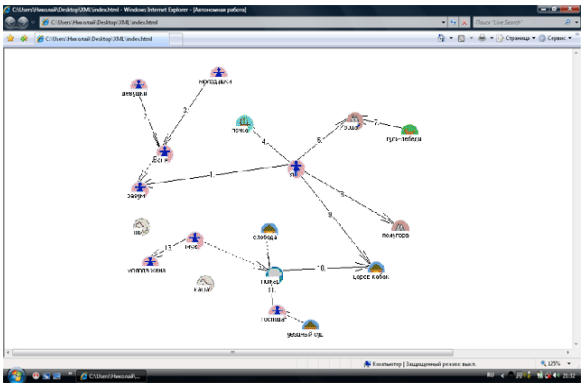


Рисунок 3. Отображение теоретико-графовой модели в окне браузера

С помощью рассмотренных выше коэффициентов можно задать значимость того или иного критерия, например:

- λ_1 и $k^{(3)}_{(u,v)}$ определяют упорядоченность вершин и ребер графа;
- $k^{(1)}_e$ минимизирует и унифицирует длины ребер;
- λ_2 влияет на минимизацию области размещения и число пересечений ребер.

В результате настройки данных коэффициентов алгоритм показал достаточно хорошие результаты: пересечения ребер появляются тогда, когда их нельзя избежать, ребра графа упорядочены по мере появления связей в сюжете, длины ребер соответствуют расстоянию между объектами в фольклорном тексте. Недостатком алгоритма является то, что результат визуализации достаточно сильно зависит от начального (случайного) распределения вершин на плоскости. Поэтому для получения более хорошего изображения следует либо использовать несколько (до десяти) случайных разбиений, а затем выбрать лучший вариант, либо вершины графа изначально расположить на плоскости в соответствии с некоторым порядком (например, по мере появления объектов в тексте).

На рисунках 2 и 3 представлены процедура выбора фольклорного текста, который хранится в формате TextGML, и визуализация его теоретико-графовой модели, выполненные на языке PHP 5.

4. Поиск мотивов в электронной коллекции фольклорных текстов

Другой важной задачей, возникающей при работе с фольклорной коллекцией, является проблема обнаружении в текстах схожих мотивов и их сравнительный анализ. Мотивы – это композиционные фрагменты, которые повторяются в других песнях (не всегда в одной и той же последовательности) и служат исходными элементами для построения новых текстов. По выражению известного фольклориста Б. Н. Путилова мотив является «узловой категорией художественной организации произведения фольклора».

Рассмотрим фрагмент беседной песни, записанной Ф. Студитским в 1841 году [8]:

На матушке на Неве
Гуси, лебеди сидели,
Гуси, лебеди сидели,
Серы утки налетели,
Серы утки налетели,
Свежу воду помутили.

Допустим, перед исследователем стоит задача: обнаружить подобный мотив в других песнях (возможно в скрытой форме). Самый простой способ – это поиск по ключевым словам: *гуси, лебеди, Нева, серы утки, вода*. Однако это решение будет недостаточным по следующим причинам. Во-первых, автор, исполняя произведение, мог заменить существительные, прилагательные и глаголы синонимами или близкими по звучанию словами. Во-вторых, наличие в тексте ключевых слов еще не говорит об их семантической связности, тем более о наличие схожего мотива. Например, объекты «девушка» или «парень» встречаются почти во всех текстах, образуя совершенно разные сюжеты.

Другое решение основано на использовании графов. В этом случае задача поиска мотива в коллекции сводится к задаче поиска изоморфного подграфа (например, для данного фрагмента текста искомый граф изображен на рисунке 4).

1) гуси, лебеди *сидели* на Неве;
2) серы утки *налетели* к Неве;
3) серы утки *помутили* свежу воду;
4) вода *есть* (*принадлежит*) в Неве.

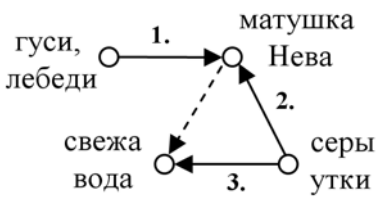
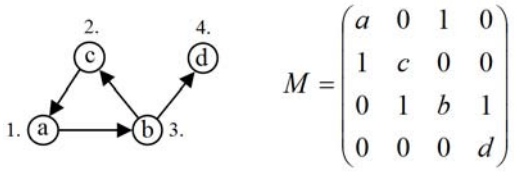


Рисунок 4. Граф мотива песни

Для решения данной задачи применим классический алгоритм поиска изоморфизма подграфу, предложенный Ульманом [17]. Пусть дан граф $G = (V, E, \alpha, \beta, L_v, L_e)$ с множеством вершин $V = \{v_1, v_2, \ldots, v_n\}$ и множеством ребер $E \subseteq V \times V$. Функции $\alpha : V \to L_V$ и $\beta : E \to L_E$ задают метки вершинам и ребрам G соответственно. Граф можно представить с помощью матрицы смежности следующего вида: $M = \{m_{ij}\}_{i,j=1}^{n}$, где $m_{ii} = \alpha(v_i)$ и $m_{ij} = \beta((v_i, v_j))$ для $i \neq j$ (см. пример на рис. 5).



$$M = \begin{pmatrix} a & 0 & 1 & 0 \\ 1 & c & 0 & 0 \\ 0 & 1 & b & 1 \\ 0 & 0 & 0 & d \end{pmatrix}$$

Рисунок 5. Граф G и его матрица смежности

Очевидно, что матрица M является не единственным представлением графа G. Если в ней определенным образом поменять строки и столбцы, то полученная матрица тоже будет однозначно определять G.

Определение 1. Матрица $P = \{p_{ij}\}_{i,j=1}^{n}$ называется матрицей перестановок, если выполняются следующие условия:

1. $p_{ij} \in \{0,1\}$ для $i, j = 1, \ldots, n$;
2. $\sum_{i=1}^{n} p_{ij} = 1$ для $j = 1, \ldots, n$;
3. $\sum_{j=1}^{n} p_{ij} = 1$ для $i = 1, \ldots, n$.

Если граф G представлен с помощью матрицы смежности M размерности $n \times n$ и P - это матрица перестановок размерности $n \times n$, то тогда матрица

$$M' = PMP^T,$$

где P^T - транспонированная матрица P, также является матрицей смежности для графа G. При этом если $p_{ij} = 1$, то j-я вершина в M становится i-й вершиной в M'. Например, на рисунке 6 приведена матрица перестановок, которая первую вершину делает второй ($p_{21} = 1$), вторую вершину – четвертой ($p_{42} = 1$), четвертую вершину – первой ($p_{14} = 1$), а третью оставляет без изменения ($p_{33} = 1$).

$$P = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix} \quad M' = \begin{pmatrix} d & 0 & 0 & 0 \\ 0 & a & 1 & 0 \\ 1 & 0 & b & 1 \\ 0 & 1 & 0 & c \end{pmatrix}$$

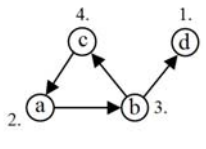


Рисунок 6. Матрица перестановки P и новая матрица смежности M'

Определение 2. Обозначим матрицу $S_{k,m}(M)$ размерности $k \times m$, которая получена из M путем удаления строк с номерами $k+1, \ldots, n$ и столбцов $m+1, \ldots, n$, где $k, m \leq n$. Таким образом, $S_{k,m}(M)$

совпадает с M только при $i=1,\ldots,k$ и $j=1,\ldots,m$. Например, для матрицы смежности M на рисунке 5, $S_{2,3}(M)$ принимает следующий вид:

$$S_{2,3}(M)=\begin{pmatrix} a & 0 & 1 \\ 1 & c & 0 \end{pmatrix}$$

Теперь сформулируем понятие изоморфизма подграфу в терминах матриц смежности и перестановок. Пусть G_1 и G_2 - графы с матрицами смежности M_1 и M_2 размерности $m\times m$ и $n\times n$ соответственно, где $m\le n$. Тогда изоморфизм графа G_1 подграфу в G_2 существует тогда, когда существует матрица перестановок P размерности $n\times n$, такая что

$$M_1=S_{m,m}(PM_2P^T).$$

Рассмотрим рекурсивную процедуру Backtrack (на входе счетчик $k=1$, $G=(V,E,\alpha,\beta,L_v,L_e)$, $n=|V|$, $G_I=(V_I,E_I,\alpha_I,\beta_I,L_v,L_e)$, $m=|V_I|$, M и M_I - матрицы смежности для G и G_I соответственно; $P=(p_{ij})$ - матрица перестановок $n\times n$;).

Procedure Backtrack:
1. **if** $k>m$ **then**
2. // P определяет изоморфизм G_I подграфу G
3. Print (P);
4. **return**;
5. **end if**;
6. **for** i :=1 to n **do**
7. $p_{ki}:=1$;
8. **for** $j\ne i$ **do**
9. $p_{kj}:=0$;
10. **end for**;
11. **if** $S_{k,k}(M_I)=S_{k,n}(P)M(S_{k,n}(P))^T$ **then**
12. $Backtrack(M,M_I,P,k+1)$;
13. **end if**;
14. **end for**;
15. **return**.

Алгоритм основан на идее поиска всех изоморфизмов подграфу с помощью последовательного определения строка за строкой матрицы перестановок P. Из определения 1 следует, что каждая строка k матрицы P содержит в точности одно ненулевое число $p_{ki}=1$ (остальные элементы p_{kj} строки k при $j\ne i$ равны 0). Рекурсивная процедура *Backtrack* начинается установкой первого элемента $p_{11}=1$, а всех остальных элементов первой строки 0. Если $S_{1,n}(P)$ - неполное соответствие, которое представляет изоморфизм подграфу, тогда процедура *Backtrack* рекурсивно вызывается еще раз и вторая строка предварительно устанавливается. Процесс продолжается до тех пор, пока m строк P не будут успешно установлены и найден изоморфизм подграфу или условие на шаге 11 не будет выполнено. В любом случае процедура возвращается на предыдущий уровень и проверяет другие значения p_{ki}. В данный алгоритм можно ввести дополнительные ограничения на матрицу перехода P, например, учитывающие порядок появления связей в графе. Это позволит сократить количество переборных вариантов и приведет к ускорению работы алгоритма.

В настоящее время процедура поиска схожих мотивов реализована в локальной версии системы, написанной на языке Delphi 7.0. Искомый мотив можно задать двумя способами: либо пользователь самостоятельно определяет объекты и связи, либо выделяет в фольклорном тексте границы мотива и программа автоматически строит граф (см. рис. 7).

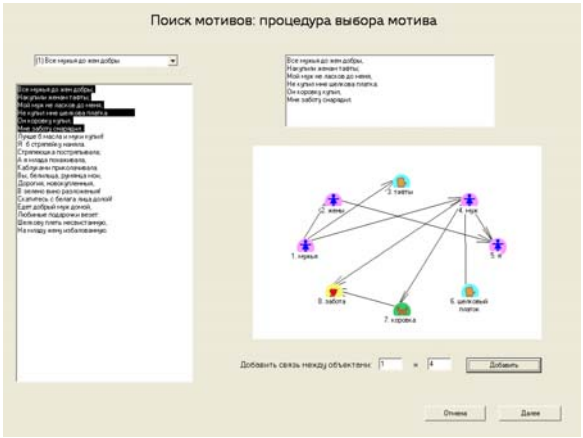


Рисунок 7. Процедура поиска мотивов

Поскольку алгоритм ищет точное соответствие графов и подграфов, наилучшие результаты получаются при поиске мотивов с 4-6 объектами. Особенно хорошие результаты программа выдает при сравнении фольклорных текстов с одинаковыми сюжетами, записанными разными собирателями (например, «Как назябло, навеяло лицо», «Позябло, позябло лицо» и «Разсыпалось ожерельице»). В остальных случаях получается либо много лишних вариантов (при небольшом числе объектов), либо их отсутствие (при числе объектов больших 10).

При проведении подобного анализа программа способна обнаружить «скрытые» мотивы, которые сложно обнаружить традиционными методами. Например, в бесёдной песне «Девушка в горенке сидела» объект «коса» встречается как в начале текста (девушка *плела* русу косу), так и в конце (коса *повысушила* парня, *с ног сронила*), образую два мотива. С другой стороны, эти мотивы взаимосвязаны и их можно рассматривать как единый мотив, разбитый в тексте на несколько частей. При традиционном поиске обнаружить данный фрагмент будет достаточно сложно. При использовании теоретико-графовых моделей мотив представлен как связанный подграф, который структурно не отличается от остальных подграфов. Этот эффект особенно проявляется при исследовании больших текстов.

Для того чтобы данный метод давал лучшие результаты, его необходимо дополнить поиском по ключевым словам. Также можно ввести ограничения на принадлежность объектов к определенной группе, на тип и порядок появления связей в тексте.

5. Заключение

В данной работе были рассмотрены метод визуализации теоретико-графовых моделей и алгоритм поиска схожих мотивов, основанный на структурном соответствии графов и подграфов. Кроме того, в статье описывается способ хранения теоретико-графовых моделей на языке TextGML.

Предложенные методы могут найти свое применение при составлении тематических указателей, указателей фольклорных мотивов и формул. Также их можно использовать при исследовании других видов семантических сетей, построенных на основе текстов.

Литература

[1] Варфоломеев А. Г., Кравцов И. В., Москин Н.Д. Информационная система по фольклорным песням Заонежья как инструмент формализации и классификации песен // Электронные библиотеки: перспективные методы и технологии, электронные коллекции: Труды IV Всероссийской научной конференции RCDL'2002 – Дубна, 2002. – Т. 2. – С. 143-147.

[2] Варфоломеев А. Г., Кравцов И. В., Москин Н.Д. Проект специализированного Интернет-ресурса для представления и анализа фольклорных песен // Электронные библиотеки: перспективные методы и технологии, электронные коллекции: Труды V Всероссийской научной конференции RCDL'2003. – Санкт-Петербург, 2003. – С. 339-343.

[3] Варфоломеев А. Г., Москин Н. Д. Об электронной коллекции фольклорных песен с теоретико-графовой формализацией текстов // Электронные библиотеки: перспективные методы и технологии, электронные коллекции: Сборник аннотаций стендовых докладов III Всероссийской научной конференции RCDL'2001. – Петрозаводск, 2001. – С. 20.

[4] Каргинова Н. В., Кравцов И. В., Москин Н. Д., Варфоломеев А. Г. Проект электронной библиотеки методик и результатов исследований текстовых коллекций для системы «Источник» // Электронные библиотеки: перспективные методы и технологии, электронные коллекции: Труды X Всероссийской конференции RCDL'2008. – Дубна: ОИЯИ, 2008. – С. 239-245.

[5] Касьянов В. Н., Евстигнеев В. А. Графы в программировании: обработка, визуализация и применение. – СПб.: БХВ-Петербург, 2003. – 1104 с.

[6] Лысанов В. Д. Досюльная свадьба, песни, игры и танцы в Заонежье Олонецкой губернии. – Петрозаводск: Северная скоропечатня Р. Г. Каца, 1916. – 119, 30 с.

[7] Москин Н. Д. Теоретико-графовые модели структуры фольклорных песен и методы их анализа // Круг идей: Междисциплинарные подходы в исторической информатике. Труды X конференции Ассоциации «История и компьютер» / Под ред. Л. И. Бородкина, И. М. Гарсковой. – М.: Изд-во МГУ, 2008. – С. 280-300.

[8] Москин Н. Д. Теоретико-графовые модели структуры фольклорных текстов, алгоритмы поиска закономерностей и их программная реализация // Дисс. на соиск. уч. ст. к.т.н. – Петрозаводск, 2006.

[9] Новиков А. И. Семантика текста и ее формализация. М: Наука, 1983. – 215 с.

[10] Скороходько Э. Ф. Семантические сети и автоматическая обработка текста. – Киев: Наукова думка, 1983. – 218 с.

[11] Спецификация GraphML. http://graphml.graphdrawing.org/index.html

[12] Спецификация GXL. http://www.gupro.de/GXL/

[13] Спецификация XGMML. http://www.cs.rpi.edu/~puninj/XGMML/

[14] Ebert J., Kullbach B., Winter A. GraX: Graph Exchange Format // In Proceedings of the Workshop on Standard Exchange Formats (WoSEF) at ICSE'00, 2000. http://www.gupro.de/winter/Papers/ebert+2000.pdf

[15] Herman I., Marshall M. S. GraphXML – An XML-based graph description format // Proceedings of Graph Drawing 2000 (Lecture Notes in Computer Science, vol. 1984). – Springer: Berlin, 2001. – P. 52-62. http://www.euronet.nl/users/scott/public_html/publications/GraphXMLShort.pdf

[16] Himsolt M. GML: A portable Graph File Format // Technical report, University at Passau, 1997. http://www.infosun.fim.uni-passau.de/Graphlet/GML/gml-tr.html

[17] Ullmann J. R. An algorithm for subgraph isomorphism // Journal of the Association for Computing Machinery. – 1976. – Vol. 23, No. 1. – P. 31-42.

The solution of visualization and motive search problems in the digital library of folklore texts

N. D. Moskin

This paper describes research methods for the digital library of folklore texts with graph representation of their semantic structures. In particular the questions of storing and visualization graph models, motive search algorithm based on graph and subgraph matching are considered.