С. Н. Пупырев, А. В. Тихонов, Визуализация динамических графов для анализа сложных сетей,
Модел. и анализ информ. систем, 2010, том 17, номер 1, 117–135

Модел. и анализ информ. систем. Т. **17**, № **1** (2010) 117–135

УДК 519.6

Визуализация динамических графов для анализа сложных сетей

Пупырев С.Н., Тихонов А.В.

Уральский государственный университет,
ООО Corelogic

e-mail: spupyrev@gmail.com, altsoph@gmail.com

получена 13 января 2010

Ключевые слова: рисование графов, динамический граф, визуализация, анализ сложных сетей

Предложен метод визуализации динамических графов, позволяющий анализировать сложные сети. Наш метод основан на построении послойных укладок графов с сохранением ментальной карты. Для этого алгоритмы визуализации, основанные на физических аналогиях, обобщены и расширены на случай изменяющихся графов.

Предложенные алгоритмы применены для визуализации нескольких социальных сетей. Эксперименты показывают, что наши методы визуализации упрощают анализ реальных данных и помогают при решении задач, связанных с исследованием динамики социальных отношений.

1. Введение

Граф – фундаментальная структура данных, которая позволяет моделировать разнообразные объекты и взаимосвязи между ними. С ростом количества информации роль графов в науке сложно переоценить: они используются в физике для моделирования взаимодействия между телами, в химии для установления связей между молекулами и атомами; биологи при помощи графов моделируют поведение бактерий, социологи изучают связи в обществе и выявляют закономерности поведения, в программировании графы являются центральной структурой данных.

Визуализация графов активно развивается с начала 80-х годов, поскольку является удобным инструментом анализа сложных данных. Подтверждением актуальности темы является проведение ежегодных международных симпозиумов и конференций (крупнейшие – InfoVis и Graph Drawing). Задача визуализации состоит в том, чтобы представить граф (нарисовать, построить укладку) в простом и понятном виде, подчеркивающем его структуру и топологические свойства.

Данная статья посвящена визуализации динамических сетей (графов). Такие графы изменяются с течением времени: могут появиться (исчезнуть) некоторые вершины или ребра, могут быть изменены атрибуты вершин или ребер. Подобные сценарии часто встречаются в жизни. Например, если граф моделирует некоторую социальную сеть, то естественные изменения в графе – это появление новых знакомств или участников в сети. Основная сложность задачи рисования динамического графа состоит в том, чтобы наиболее ясно и точно подчеркнуть происходящие изменения.

Для рисования динамических сетей мы используем метод *послойной* визуализации (также в литературе используется термин 2.5D-визуализация), который состоит из следующих шагов:

- рассматриваемый период времени разбивается на непересекающиеся подпериоды. Далее строится последовательность графов, моделирующая исходные данные за каждый подпериод. Выбор количества и длины подпериодов зависит от предметной области. Например, в случае с социальными сетями можно анализировать изменения за неделю. В случае с графом ссылок в сети Интернет логично выбирать более крупные подпериоды;
- для каждого построенного графа строится укладка на плоскости;
- результирующая визуализация получается объединением всех укладок с соединением вершин, соответствующим одинаковым объектам в соседних графах (рис. 1(c)).

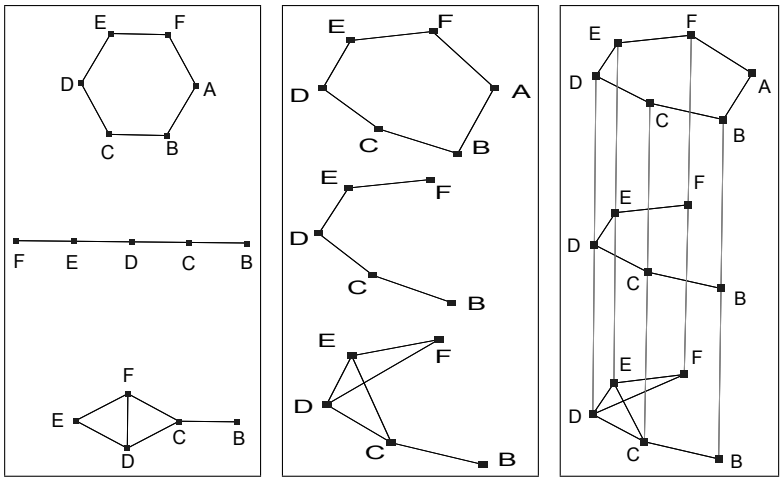


Рис. 1. Граф, состоящий из трех слоев. (a) Укладки без учета ментальной карты. (b) Сохранение ментальной карты. (c) Послойная визуализация.

При построении послойной визуализации важно учитывать два конкурирующих критерия. С одной стороны, каждый слой должен быть нарисован с учетом общепринятых критериев эстетичности графа: малое количество пересечений ребер,

симметрия, постоянная длина ребер [20]. С другой – последовательность укладок должна сохранять ментальную карту (*mental map*) [17]. Это означает, что графы должны быть нарисованы в унифицированном, похожем стиле. Такой подход заметно облегчает анализ данных и обеспечивает динамическую стабильность, подчеркивает изменения между графами. В данной статье мы считаем, что ментальная карта сохраняется, если вершины графа приближенно сохраняют свои координаты в последовательных укладках (рис. 1.).

Существует несколько способов построения укладок графов на плоскости. Самые популярные – методы, основанные на физических аналогиях – силовой (*force-directed*) и пружинный (*spring*). Для построения укладки строится специальная модель, в которой вершины и ребра графа соответствуют «реальным» физическим взаимодействующим объектам. Для этой системы вводится функция энергии таким образом, что конфигурации с меньшим уровнем энергии соответствуют лучшим укладкам. При этом задача поиска лучшей укладки графа сводится к поиску минимума энергии системы. Однако такие методы имеют ряд ограничений, и в классической формулировке не применимы к рисованию динамических графов. Мы построили модификации этих методов для укладки последовательности графов с сохранением ментальной карты. Резюмируя, вклад нашей работы состоит в следующем:

- силовой алгоритм сформулирован в обобщенном виде, в котором он может быть применен для более широкого класса задач, в том числе рисование взвешенных и несвязных графов, а также динамических сетей;
- пружинный метод (метод многомерного шкалирования, *multidimensional scaling*) адаптирован на случай динамических графов;
- создан прототип системы для интерактивного построения и визуализации укладок при помощи описанных алгоритмов;
- построены визуализации нескольких сложных динамических сетей, проведен анализ изменения и развития этих сетей.

В следующем разделе мы рассматриваем существующие подходы к решению задачи. Далее приводится детальное описание нашего алгоритма. Раздел 4. содержит экспериментальную часть и анализ сложных сетей. В заключении обсуждаются полученные результаты и формулируются направления дальнейших исследований.

2. Обзор существующих методов

Алгоритмы визуализации графов разрабатываются с начала 60-х годов [21]. Наиболее активно область рисования графов начала развиваться с появлением методов, основанных на физических аналогиях. Классическими работами считаются пружинный алгоритм Eades [7] и алгоритм Kamada и Kawai [14]. Силовая модель предложена Fruchterman и Reingold [9]. Позже были предложены многочисленные модификации и оптимизации этих алгоритмов [13, 18].

Задача динамической визуализации графов активно изучается последнее десятилетие, когда стала доступна обработка больших объемов информации. Хороший обзор по теме дан в [3]. Большинство существующих алгоритмов предназначено для рисования ограниченного класса графов или не применимо для графов большого размера [5]. Способ визуализации динамических графов в трехмерном пространстве, при котором каждый граф последовательности располагается на отдельной плоскости, рассматривается в [6]. Понятие ментальной карты впервые обсуждается в работе [17].

Метод многомерного шкалирования используется для визуализации графов с 1980 г. [16]. Широкое распространение он получил после работы Gansner и др. [11], и сейчас является одним из самых популярных методов рисования графов [19]. В классической формулировке многомерное шкалирование не применимо для рисования последовательности графов, в [4] упоминается соответствующая модификация.

Визуализация графов часто используется для анализа сложных сетей. Например, в [2] изучается граф ссылок в сети Интернет, а в [10] методы рисования графов применены для исследования архива электронной почты. Среди работ, использующих послойную визуализацию для анализа данных, стоит отметить также [8] и [12]. В каждой из этих работ метод визуализации графов «переизобретается». Насколько нам известно, не существует единого подхода для рисования произвольных динамических графов.

3. Алгоритм визуализации динамических графов

В классической формулировке силовой [9] и пружинный [14] алгоритмы предназначены для рисования графов, состоящих из одной компоненты связности. Кроме того, эти методы строят укладки без учета сохранения ментальной карты, т.е. не применимы для рисования динамических графов. Ниже мы показываем, как модифицировать оба алгоритма для визуализации произвольных взвешенных динамических графов.

1. Определения

Приведем необходимые определения. Граф G состоит из множества вершин V и множества ребер $E \subseteq V \times V$. Количество вершин в графе обозначается через $n = |V|$, количество ребер – $m = |E|$. Вершины графа $V = \{v_1, \ldots, v_n\}$ также будем обозначать индексами $V = \{1, \ldots, n\}$. Для вершины v вводится ее вес w_v, аналогично для ребра (u, v) вес обозначается через w_{uv}. В данной статье рассматриваются обыкновенные (неориентированные, без петель и кратных ребер) графы.

Под укладкой L графа $G = (V, E)$ мы понимаем отображение $L : V \to R^d$ вершин графа в множество точек d-мерного пространства. *Нарисовать* граф означает указать координаты его вершин, а ребра графа изобразить прямыми отрезками между вершинами. Позицию вершины v_i будем обозначать через p_{v_i} или p_i, ее координаты – $p_i^1, \ldots, p_i^d$. Под расстоянием между вершинами v и u будем понимать евклидово расстояние между соответствующими точками и обозначать через $||p_v - p_u||$.

В задаче динамической визуализации требуется нарисовать последовательность графов $G_1, G_2, \ldots, G_T$, описывающих некоторые данные за последовательные промежутки времени. С каждой вершиной $v \in G_i$ ассоциирована метка l_v. Будем считать, что $l_v = l_u$ тогда и только тогда, когда вершины v и u соответствуют одному объекту исходных данных.

2. Силовая модель

В силовой модели вершины графа соответствуют заряженным частицам, между которыми действуют силы притяжения и отталкивания. Силы отталкивания действуют между всеми парами вершин и обратно пропорциональны расстоянию между вершинами. Эти силы «растаскивают» близко находящиеся вершины. Таким образом, в хорошей укладке не будет слипшихся вершин. Силы притяжения действуют между вершинами, соединенными ребром, и обеспечивают, что в итоговой укладке такие вершины будут находиться рядом. Обычно эти силы пропорциональны расстоянию между вершинами. В общем виде энергию системы можно выразить следующим образом:

$$U_{attr} + U_{repu} = \sum_{(u,v)\in E} f_a w_{uv} ||p_u - p_v||^a + \sum_{(u,v):u\neq v} f_r w_u w_v ||p_u - p_v||^r.$$

Первое слагаемое – энергия притяжения, второе – энергия отталкивания между вершинами. Показатели степени $a, r \in R$ и коэффициенты $f_a, f_r \in R$ определяют степень влияния каждого компонента на итоговую укладку. Общепринятыми ограничениями являются: $f_a > 0, f_r > 0, a \geq 0, r \leq 0$, т.е. силы притяжения растут, а отталкивания убывают с увеличением расстояния.

Наиболее известными являются модель Fruchterman и Reingold с параметрами $a = 3$ и $r = 0$, модель Davidson и Harel с $a = 2$ и $r = -2$, а также LinLog модель ($a = 1$, $r = 0$). Параметры f_a, f_r и веса w_v и w_{uv} выбирают исходя из особенностей структуры графа и предметной области.

1. Силовой алгоритм для несвязных графов

В такой формулировке алгоритм не применим для визуализации несвязных графов, т.к. минимум энергии будет соответствовать укладке, в которой разные компоненты связности находятся на бесконечном расстоянии. Поэтому мы вводим дополнительную гравитационную силу, притягивающую все вершины к фиксированной точке $p_{cen} = (0, \ldots, 0) \in R^d$:

$$U_{grav} = \sum_{v\in V} f_g ||p_v - p_{cen}||^g.$$

Гравитационная сила – сила притяжения, поэтому должны выполняться ограничения $f_g > 0$ и $g \geq 0$.

2. Силовой алгоритм для динамических графов

В задаче динамической визуализации необходимо обеспечить «похожесть» укладок для последовательности графов $G_1, \ldots, G_T$. Под похожестью мы понимаем прибли-

женное сохранение позиций вершин с одинаковыми метками в разных графах. Для этого мы строим новый граф, являющийся объединением всех графов последовательности с добавлением дополнительных ребер: каждая пара вершин с одинаковыми метками соединяется ребром. Граф-объединение $G_s = (V_s, E_s)$, где $V_s = \cup_{i=1}^{T} V^i$ и $E_s = \cup_{i=1}^{T} E^i \cup \{(u, v) \subseteq V_s \times V_s : l_u = l_v\}$. Дополнительные ребра между различными графами последовательности будут обеспечивать силы притяжения между вершинами с одинаковыми метками. Благодаря этому, исходные объекты будут стремиться занять одинаковые позиции во всех слоях результирующей послойной укладки, что соответствует сохранению ментальной карты. Дополнительная составляющая энергии для работы с динамическими графами выражается следующим образом:

$$U_{mental} = \sum_{(u,v)\in E_s} f_m ||p_u - p_v||^m, \text{ где } f_m > 0 \text{ и } m \geq 0.$$

3. Минимизация энергии

На общую энергию системы для последовательности графов

$$U = U_{attr} + U_{repu} + U_{grav} + U_{mental}$$

можно смотреть, как на функцию $n \cdot d$ переменных (n вершин, d координат). Для нахождения ее минимума (часто локального) применим метод *градиентного спуска*. Изначально каждой вершине графа присваиваются случайные координаты. На каждом шаге алгоритма мы предполагаем, что все вершины, кроме одной, фиксированы. Сдвигаем эту вершину в направлении антиградиента, выбирая длину сдвига так, чтобы максимально уменьшить энергию системы. Этот шаг итеративно применяется для всех вершин графа. Алгоритм заканчивает работу, когда энергия системы достигает своего локального минимума (позиции вершин не изменяются). Ниже приведено формальное описание алгоритма.

```
ForceDirectedLayout(G)
   //присвоить вершинам случайные координаты
   for i = 1 to n do
      p_i ← random_position(R^d)

   //пока позиции вершин изменяются
   while (change in any p_i)> eps do
      for i = 1 to n do
         for k = 1 to d do
            //выбор длины оптимального шага
            γ ← argmin_γ U(p_i^k − γ∇U(p_i^k))
            //сдвиг в направлении антиградиента
            p_i^k ← p_i^k − γ∇U(p_i^k)

   end while
```

3. Пружинная модель

Другой популярной физической моделью для рисования графов является пружинная модель. В этой модели ребра графа заменяют пружинами, при растяжении и

сжатии которых возникают силы упругости, действующие по закону Гука и стремящиеся вернуть пружине свою первоначальную длину. Укладка графа при помощи пружинной модели эквивалентна методу многомерного шкалирования (MDS, multidimensional scaling), поэтому перейдем к его описанию.

Метод MDS позволяет расположить объекты в пространстве некоторой небольшой размерности таким образом, чтобы достаточно точно воспроизвести заданные расстояния между ними. Задача многомерного шкалирования ставится следующим образом.

Задача 1. *По заданной матрице расстояний* $D = (d_{ij}) \in R^{n \times n}$ *между объектами найти их позиции* $p_1, \ldots, p_n \in R^d$ *так, чтобы выполнялось* $||p_i - p_j|| = d_{ij}$ *для всех* $i, j \in [1, n]$. *На матрицу расстояний* D *чаще всего накладывают ограничения: она симметричная* $d_{ij} = d_{ji}$, *имеет неотрицательные элементы* $d_{ij} \geq 0$ *и нули на главной диагонали* $d_{ii} = 0$.

На практике задача 1 не всегда имеет решение, поэтому разумно искать позиции объектов, приближенно сохраняющие заданные расстояния, т.е. такие, что $||p_i - p_j|| \approx d_{ij}$. Мерой качества приближения является функция, называемая стресс: $Stress = \sum_{i=1}^{n} \sum_{j=1}^{n} (d_{ij} - ||p_i - p_j||)^2$. Если разрешить варьировать вклад каждого слагаемого в общую сумму, то получится следующая взвешенная задача многомерного шкалирования.

Задача 2. *По матрицам расстояний* $D = (d_{ij}) \in R^{n \times n}$ *и весов* $W = (w_{ij}) \in R^{n \times n}$ *найти позиции* $p_1, \ldots, p_n \in R^d$ *объектов, минимизирующих стресс:*

$$\min_{p_1, \ldots, p_n \in R^d} \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} (d_{ij} - ||p_i - p_j||)^2.$$

При использовании метода MDS для визуализации графов в качестве D чаще всего берут матрицу кратчайших расстояний между вершинами. В результате вершины, соединенные ребром, будут расположены рядом, а вершины, имеющие большое теоретико-графовое расстояние, будут отдалены друг от друга. Вес w_{ij} обычно обратно пропорционален расстоянию между вершинами ($w_{ij} = d_{ij}^q, q < 0$). В такой формулировке MDS эквивалентен классической пружинной модели, впервые предложенной в работе Kamada и Kawai [14], в которой d_{ij} – минимальное расстояние в графе между вершинами v_i и v_j, и $w_{ij} = {d_{ij}}^{-2}$.

1. MDS для несвязных графов

Для вершин v_i и v_j из разных компонент связности соответствующий элемент d_{ij} не определен. Естественной модификацией является исключение элемента из функции стресса: $w_{ij} = 0$. Однако при таком подходе результирующие укладки разных компонент связности могут пересекаться. Для разрешения этой проблемы введем дополнительные «силы отталкивания» между вершинами v_i и v_j из разных компонент связности. Пусть $w_{ij} = {d_{ij}}^{-1}$. Тогда

$$Stress(i, j) = w_{ij}(||p_i - p_j|| - d_{ij})^2 = \frac{||p_i - p_j||^2}{d_{ij}} - 2||p_i - p_j|| + d_{ij}.$$

Заметим, что последнее слагаемое не зависит от позиций вершин и не влияет на решение задачи 2. Поэтому вклад стресса вершин v_i и v_j из разных компонент связности можно считать равным $\frac{||p_i-p_j||^2}{d_{ij}} - 2||p_i - p_j||$. Если положить $d_{ij} \to +\infty$, то стресс будет иметь вид $-2||p_i - p_j||$, что соответствует «силе отталкивания» между этими вершинами. Стоит добавить, что на практике можно выбирать $d_{ij} \approx 10^6$ и $w_{ij} = t_{ij}/(2d_{ij})$. Легко убедиться, что при этом стресс имеет вид $-t_{ij}||p_i - p_j||$. Параметр t_{ij} позволяет варьировать степень влияния силы отталкивания на результирующую укладку.

Для окончательного решения проблемы несвязных графов осталось ввести аналог силы гравитации для метода MDS. Это делается введением дополнительной вершины v_{cen} с координатами $p_{cen} = (0, \ldots, 0) \in R^d$ и добавлением ребер нулевой длины ($d_{i,cen} = 0$), соединяющих ее со всеми остальными вершинами графа. Влияние этой вершины на общий стресс будет равен $w_{i,cen}||p_i||^2$, что соответствует «силе притяжения» каждой вершины к центру системы. Степень влияния гравитации на укладку определяется параметром $w_{i,cen}$.

2. MDS для динамических графов

Поступаем аналогично силовой модели: строится новый граф, являющийся объединением всех графов последовательности с добавлением ребер между всеми парами вершин, имеющих одинаковые метки. Мы хотим, чтобы такие вершины располагались рядом, поэтому устанавливаем соответствующие расстояния: $d_{ij} = 0$ для всех v_i, v_j для которых $l_{v_i} = l_{v_j}$; $w_{ij} = 0$ для тех v_i, v_j, для которых $l_{v_i} \neq l_{v_j}$.

3. Метод решения MDS

Задача 2 решается итеративно при помощи техники, известной как мажорирование стресса (stress majorization). На каждом шаге алгоритма вычисляются новые координаты объектов с меньшим стрессом. На практике для нахождения локального минимума функции достаточно константного количества шагов. Подробное описание и детали реализации алгоритма мажорирования стресса можно найти в [11, 19].

StressMajorization$(D = (d_{ij}), W = (w_{ij}))$
  //присвоить объектам случайные координаты
  for $i = 1$ **to** n **do**
    $p_i \leftarrow random\_position(R^d)$

  //пока координаты объектов изменяются
  while (change in any p_i)> eps **do**
    for $i = 1$ **to** n **do**
      for $k = 1$ **to** d **do**

$$p_i^k \leftarrow \frac{\sum_{j \neq i} w_{ij} \left(p_j^k + d_{ij} \frac{p_i^k - p_j^k}{||p_i - p_j||} \right)}{\sum_{j \neq i} w_{ij}}$$

  end while

4. Сравнение моделей

Важный вопрос, который мы хотели решить в ходе экспериментов: какой из предложенных двух алгоритмов лучше подходит для визуализации динамических графов?

При проведении экспериментов мы выявили их основные преимущества и недостатки. Силовой алгоритм имеет большое количество параметров, что делает его очень гибким. Например, можно регулировать степень влияния каждой компоненты (притяжения, отталкивания, гравитации и сохранения ментальной карты) на итоговую укладку. Увеличение параметров f_m и m влечет за собой более точное сохранение ментальной карты для последовательности графов, но при этом качество каждой отдельной укладки может ухудшаться. Напротив, уменьшение значимости этой компоненты влечет повышение эстетичности каждого слоя. Интересной является конфигурация с параметрами $a = 1$ и $r = 0$, которая известна как LinLog модель [18]. В этой модели вершины графа группируются таким образом, что проявляются скрытые кластеры исходных данных. При помощи нашего динамического силового алгоритма можно просматривать динамику этих групп. Основным недостатком силового алгоритма является его скорость: минимизация энергии требует порядка $O(n)$ итераций, каждая из которых занимает $O(n^2 + m)$ времени. Таким образом, с помощью этого алгоритма можно обрабатывать графы только сравнительно малых размеров. В свою очередь, метод MDS достаточно быстр – он работает за время $O(n^2)$. При этом он менее гибок и не позволяет существенно настраивать влияние разных компонент.

В итоге у нас нет ответа, какой из алгоритмов лучше. Каждый из них может применяться в зависимости от исходных данных и требований: MDS более быстрый, силовой алгоритм строит более качественные укладки. Нам видится перспективным следующий сценарий визуализации динамического графа: сначала строится приближенная укладка графа при помощи быстрого MDS, а затем она «дорабатывается» до нужного результата силовым алгоритмом. Такая схема базируется на том, что если силовой алгоритм начинать не со случайной укладки, а с той, которая приближенно похожа на финальную, то ему для сходимости требуется гораздо меньшее число шагов.

4. Эксперименты

Мы применили описанные методы для анализа сложных динамических графов. В качестве входных данных нами были выбраны трассы (архивы истории) взаимодействия пользователей различных социальных сетей. Алгоритмы визуализации были реализованы в системе GraphVis [22] с использованием библиотек Java3D и Processing. Тестирование проводилось на машине с процессором Intel Core-2 Duo 1.83GHz и 2GB оперативной памяти.

1. Использованные наборы данных

В рамках нашей работы мы использовали следующие наборы данных:

- VNC-dataset – архив личных сообщений производственного чата российского промышленного предприятия средних размеров, предоставленный для наших исследований службой безопасности этого предприятия;

- Enron-dataset – архив сообщений электронной почты, отправляемых и получаемых сотрудниками компании Enron в течение нескольких лет (архив был официально опубликован в свободном доступе [15] в рамках расследования по делу о банкротстве компании Enron и в настоящее время является одним из самых популярных банков данных такого рода в академической среде);

- Twitter-dataset – архив некоторого подмножества сообщений за период около полугода системы Twitter, выгруженных из системы Twitter через её открытое API.

Каждый набор данных – это множество сообщений (событий коммуникаций) между участниками сети. Для каждого сообщения известен его отправитель, получатель и время отправления. В данной работе мы не учитывали текст сообщения для анализа. Основные характеристики использованных наборов данных представлены в таблице 1.

	VNC	Enron	Twitter
Число пользователей	235	3112	551
Число событий коммуникации	131879	873963	4220
Период времени, дней	113	1285	187
Среднее число событий в сутки	1167	680	22.5
Показатель степенного распределения числа исходящих сообщений у пользователя	3.18	1.85	3.11
Показатель степенного распределения числа входящих сообщений у пользователя	2.83	3.5	3.5

Таблица 1. Сравнительные характеристики использованных наборов данных.

2. Подготовка данных для анализа

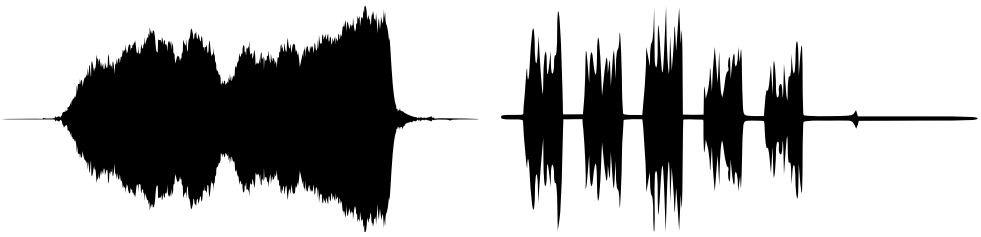
1. Разбиение данных на подпериоды

Применение предлагаемого подхода к изучению динамики связей внутри сообщества предполагает разбиение изучаемого периода времени на непересекающиеся подпериоды. Каждому из этих подпериодов соответствует отдельный слой визуализации, содержащий укладку графа, сопоставленного этому подпериоду. При выборе разбиения мы учитывали следующие факторы:

- представляется удобным выбор разбиения, при котором все подпериоды имеют равную продолжительность, т.к. такое разбиение позволяет наглядно визуализировать процессы роста, убывания и стабилизации активности взаимодействий в сообществе в течение всего исследуемого периода;

- общее число подпериодов не должно быть слишком большим, т.к. это, с одной стороны, существенно замедляет расчеты, а с другой стороны, уменьшает наглядность результирующей визуализации;

- длительность подпериода удобно выбирать кратной периоду некоторого внутреннего (присущего рассматриваемому сообществу) колебания интенсивности общения. Таких «сезонных составляющих» интенсивности общения может быть несколько: часто прослеживаются суточная (рис. 2(a)), недельная (рис. 2(b)), месячная и годовая периодичности. Наличие той или иной периодичности зависит от конкретного сообщества, так, например, суточная периодичность наблюдается в случаях, когда все участники сообщества или основная их масса принадлежит одному часовому поясу, а месячная периодичность обычно зашумлена некратной ей недельной, но может проявляться в общении, связанном, например, с производственным циклом.



(a) (b)

Рис. 2. Интенсивность общения, рассчитанная для VNC-dataset: средняя частота общения (a) в течение суток, (b) в течение семи дней.

Исходя из приведенных выше соображений, в наших экспериментах мы использовали разбиения на подпериоды длительностью в 1, 7, 30, 60 и 365 суток, получая при этом приблизительно от 5 до 25 слоев визуализации.

2. Построение графа общения для каждого подпериода

После разбиения периода на непересекающиеся подпериоды необходимо преобразовать данные истории сообщений в граф общения, соответствующий каждому подпериоду. Для этого:

- производится поиск всех пользователей, которые в течение анализируемого подпериода фигурировали как отправители или как получатели сообщений;

- каждому такому пользователю сопоставляется узел с весом, равным общему числу сообщений, отправленных этим пользователем за подпериод;

- если в течение подпериода пользователь A отправил хотя бы одно сообщение пользователю B, в граф добавляется направленное ребро (A,B), с весом равным общему числу сообщений, отправленных от A к B за рассматриваемый подпериод.

Полученная последовательность взвешенных ориентированных графов содержит сводную статистику общения между пользователями в течение каждого подпериода.

3. Подготовка графов общения для визуализации

Стандартные трудности при анализе и визуализации социальных сообществ связаны с тем, что большинство таких сообществ имеют степенное распределение числа пользователей по объему их общения, а также проявляют свойство ассортативности по числу собеседников, т.е. имеется явная положительная корреляция между числом собеседников у двух общающихся пользователей. Эти особенности приводят к тому, что граф общения, соответствующий каждому слою, оказывается немасштабируемым и содержит гигантскую плохо разделимую компоненту с очень плотным ядром [1].

Визуализация такого графа «в лоб» практически нечитаема (рис. 3(a)). Кроме того, большое число ребер в плотном ядре немасштабируемого графа делает существующие алгоритмы укладки неэффективными [19]. В связи с этим, в наших экспериментах мы использовали специально «прореженные» графы общения, т.е. подграфы, построенные путем удаления малоинформативных узлов и ребер. Для построения таких подграфов нами использовались два алгоритма:

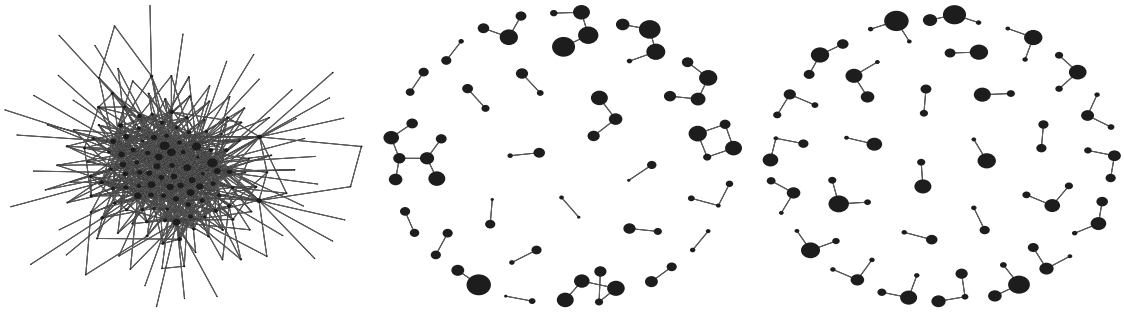
- «btw-алгоритм» последовательного разбиения графа на кластеры на основе напряженности ребер (edge betweenness) [1]. Напряженность ребра определяется как число проходящих через него кратчайших путей между всевозможными парами вершин графа. Алгоритм состоит в последовательном удалении ребер с максимальной напряженностью. Алгоритм останавливается, когда граф разбит на компоненты, не превышающие размером некоторое значение (параметр алгоритма), после чего осуществляется переход к неориентированному графу;

- алгоритм локальной нормировки и фильтрации исходящих ребер по их весам («sg-алгоритм»). Работа алгоритма состоит в удалении для каждого узла исходящих ребер с минимальными весами, до тех пор, пока общий вес удаленных ребер этого узла не составит заданную долю (параметр алгоритма) от первоначальной суммы весов исходящих ребер этого узла. После обработки всех узлов осуществляется переход к неориентированному графу.

Использование данных алгоритмов позволило в каждом подпериоде анализировать вместо исходного графа (рис. 3(a)) некоторый неориентированный подграф (рис. 3(b) и рис. 3(c)), содержащий множество небольших компонент (средний размер 2 – 5 вершин). При этом компоненты результирующего графа соответствуют явно выраженным группам общения, а его вершины и ребра – пользователям и наиболее значимым связям между ними.

3. Результаты экспериментов

В ходе работы нам удалось экспериментально показать возможность применения предложенных алгоритмов визуализации графов для решения некоторых задач ана-



(a) (b) (c)

Рис. 3. Граф общения в наборе данных VNC. Размер вершины пропорционален интенсивности общения пользователя. (a) Полный граф. Большое количество вершин и ребер делает укладку практически нечитаемой. (b) Граф, прореженный sg-алгоритмом, т.е. для каждой вершины оставлены соседи, с которыми идет наиболее интенсивный обмен информацией. (c) Пример использования btw-алгоритма, фильтрующего наименее «важные» ребра.

лиза реальных данных. Далее демонстрируется ряд примеров применения разработанных алгоритмов и системы визуализации.

1. Изучение динамики общения в группе

При изучении изменений группы общения выбранного пользователя (или небольшого их множества) во времени использовались визуализации подграфа, состоящего из узлов, соответствующих выбранному пользователю и его ближайшим соседям. Подобный подход позволяет определить состав основной группы общения пользователя, её размер и стабильность во времени. Например, явно различимы:

- Случай общающейся пары, стабильной во времени (рис. 4(a)).
- Группа «1+3» (рис. 4(b)): центральный пользователь регулярно, но с разной периодичностью общается с тремя другими пользователями.
- Случайное общение на фоне наличия одного постоянного собеседника (рис. 4(c)). Изучаемый пользователь имеет одного постоянного собеседника и еще одного, но различного в каждом подпериоде.
- Вариант с регулярной сменой постоянного собеседника пользователем (рис. 4(d)). В данном случае один подпериод был равен 60 дням, следовательно, смена партнера по общению в рамках данного сообщества происходила приблизительно раз в полгода.

Также нам удалось выявить и исследовать ситуации, соответствующие изменениям структуры общения в сообществе и полной или частичной смене пользователем группы общения, например:

- Взаимодействия между устойчивыми группами, напрямую (рис. 5(a)) или через посредников (рис. 5(b)).

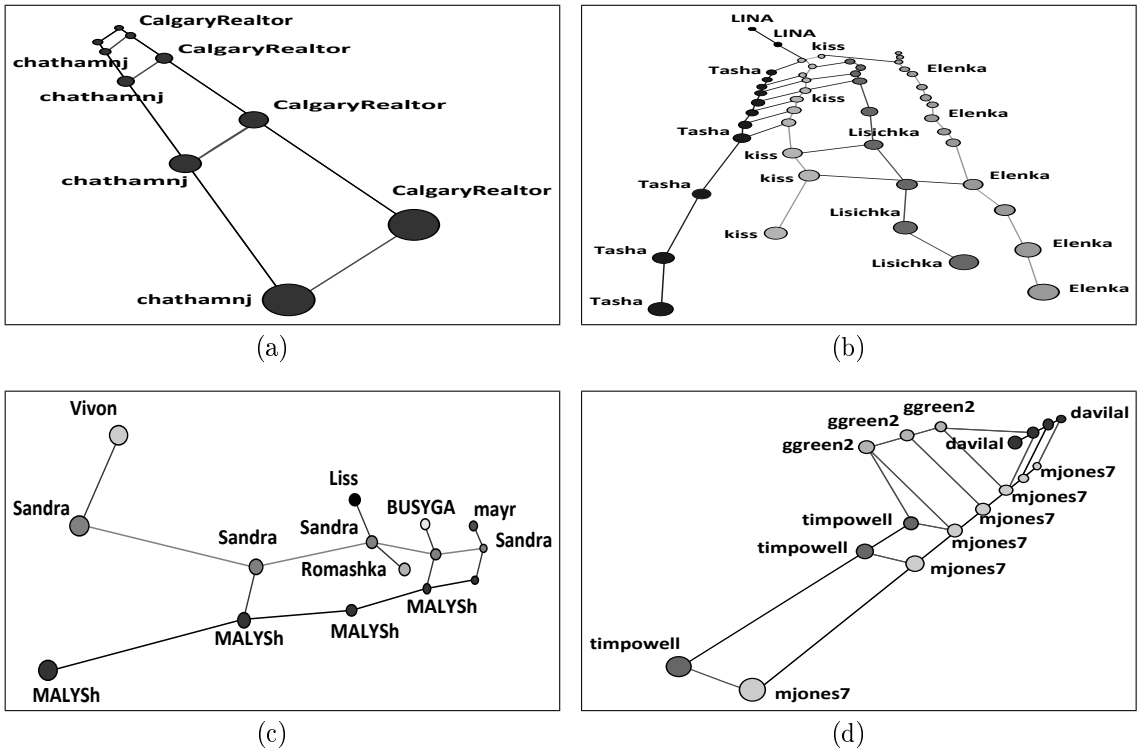


Рис. 4. Примеры различной динамики группы общения. (a) Устойчивая пара. (b) Устойчивая группа «1+3». (c) Один постоянный собеседник плюс один случайный. (d) Регулярная смена постоянного собеседника.

- Слияния или распады (рис. 5(c) и рис. 5(d)) устойчивых групп.

2. Сопоставление различных идентификаторов одному реальному пользователю

Для социальных сетей типична ситуация, в которой несколько различных идентификаторов пользователей принадлежат одному и тому же человеку. Подобное «размножение» идентификаторов может происходить как вынужденно или случайно (например, в связи с изменениями адреса электронной почты, опечатками и т.п.), так и по сознательному желанию пользователя.

Наличие таких «кратных» пользователей может существенно искажать результаты исследования социальной сети, поэтому одной из рассматриваемых нами прикладных задач стала задача по выявлению подобных ситуаций или некоторого их подмножества. Предлагаемый метод призван обнаруживать случаи, когда участник сообщества после использования одного идентификатора по каким-то причинам оставляет его и меняет его на другой, новый идентификатор.

Идея метода состоит в сопоставлении круга общения обоих пользователей и в сравнительном анализе периодов их присутствия в сообществе:

- Периоды действия идентификаторов, соответствующих описанной ситуации, не должны существенно пересекаться (рис. 6(a),6(c),6(b));

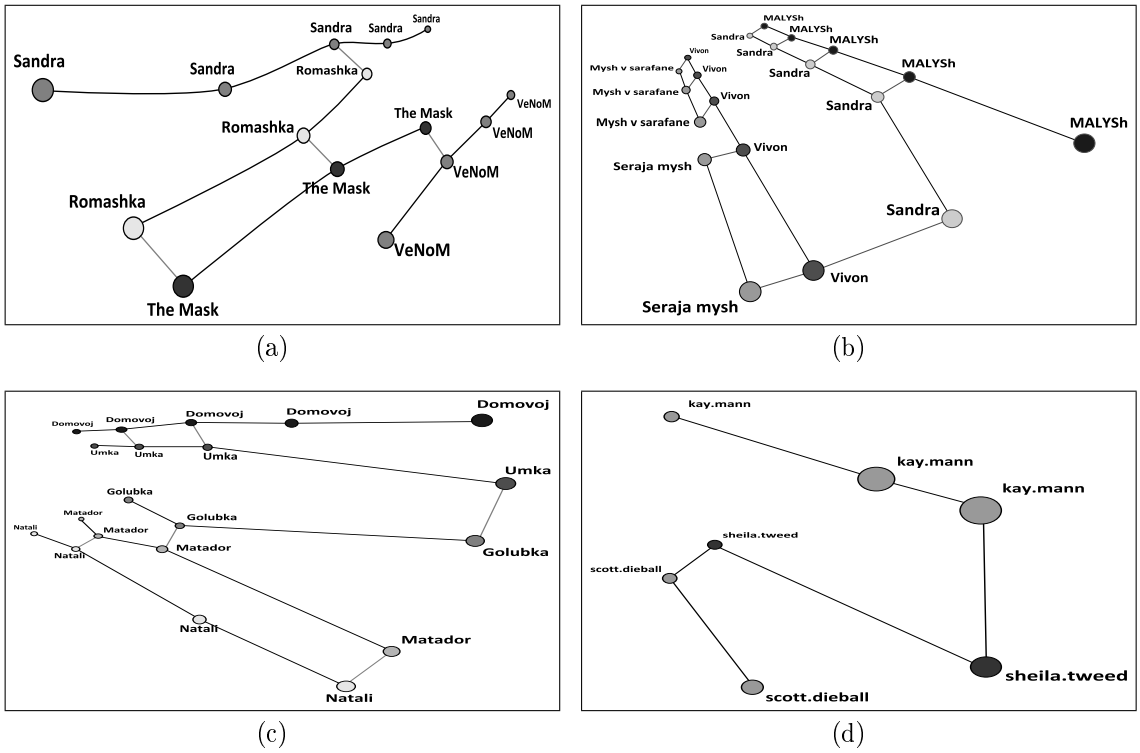


Рис. 5. Примеры изменений структуры общения. (a) Общение между двумя устойчивыми парами. (b) Общение между двумя устойчивыми парами через посредника. (c) Распад устойчивой пары. (d) Смена основного собеседника.

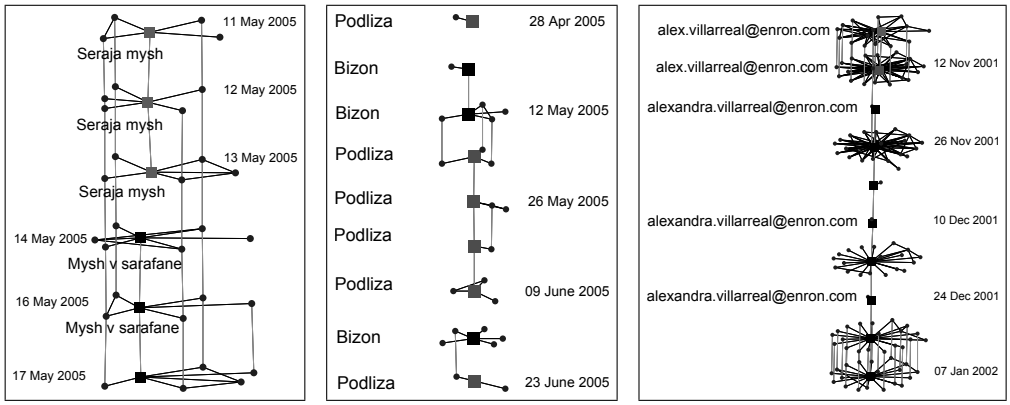
- Группы основного общения для таких идентификаторов должны совпадать (рис. 6(a) и рис. 6(b)).

Использование предложенных методов укладки автоматически располагает подобных пользователей рядом, что делает визуальный поиск таких случаев более удобным.

3. Исследование динамики интенсивности общения в сообществе

Помимо детального изучения структур общения на уровне пользователя, полезным представляется и общий анализ интенсивности общения в сообществе, в т.ч. её изменений с течением времени. Изучение общей визуализации (рис. 7) позволяет выявить характер развития сообщества и выделить некоторые периоды, соответствующие определенным фазам этого развития.

Так, при изучении общей визуализации для Enron-dataset (рис. 7(b)) можно выделить этап роста компании, этап стабильного существования, кризис, сопровождающийся бурным ростом коммуникаций, и быстрый спад общения на нет, связанный с гибелью компании. Напротив, VNC-dataset (рис. 7(a)) демонстрирует стабильный и относительно неизменный уровень общения в течение всего периода. Анализ участка подневной визуализации Twitter-dataset (рис. 7(c)) дает возможность обна-



(a) (b) (c)

Рис. 6. Сопоставление периодов присутствия и групп общения пользователей. (a) Идентификаторы «Seraja mysh» и «Mysh v sarafane» принадлежат одному пользователю. (b) Идентификаторы «Podliza» и «Bizon» принадлежат разным пользователям. (c) Идентификаторы «alex.villarreal@enron.com» и «alexandra.villarreal@enron.com» принадлежат одному пользователю.

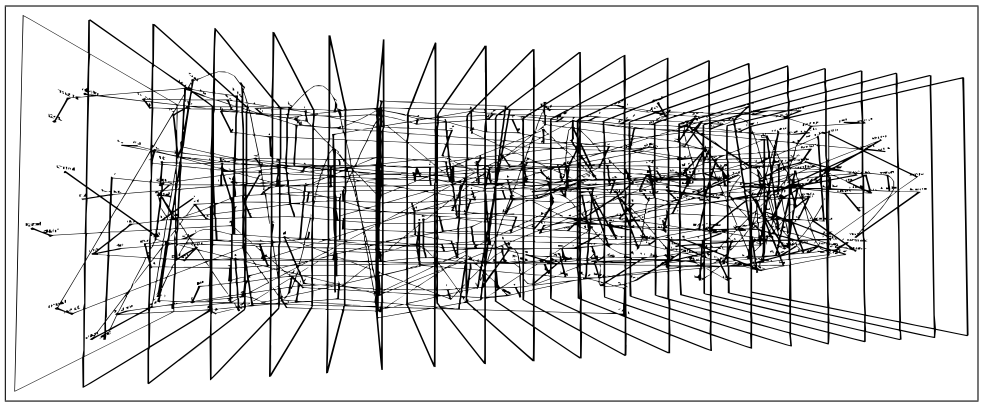
ружить периодические всплески активности (раз в 7 дней) при общей тенденции к росту объема общения.

5. Заключение

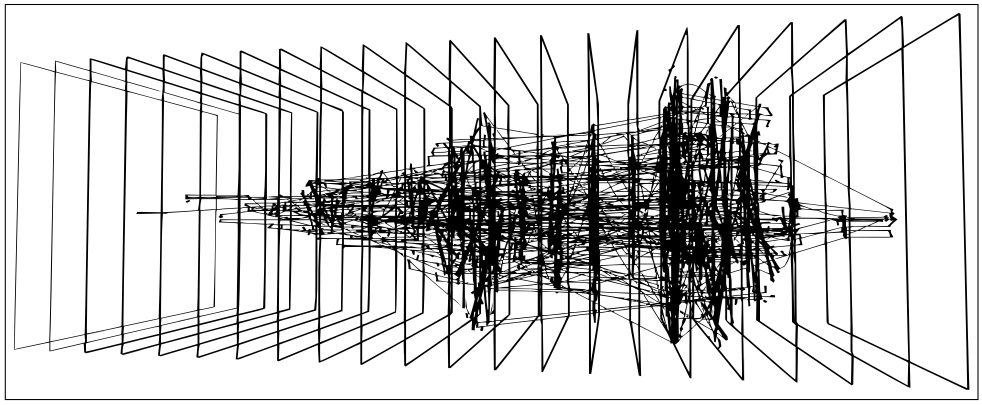
Мы предложили метод рисования изменяющихся графов с использованием послойной визуализации. Для этого существующие алгоритмы укладок графов, основанные на физических аналогиях (силовой и пружинный), были адаптированы для работы с более широким классом графов (взвешенные, несвязные, динамические). Мы применили разработанные алгоритмы для визуализации и анализа динамики трех реальных социальных сетей. В результате экспериментов нам удалось выявить некоторые скрытые связи, шаблоны общения и характер изменений в социальных отношениях. Таким образом, можно утверждать, что предложенный подход к визуализации графов применим для анализа сложных реальных сетей.

Существует несколько направлений для дальнейших исследований. В ходе экспериментов мы заметили, что алгоритмы укладки графов, основанные на физических аналогиях (особенно пружинный метод), не всегда дают оптимальный для анализа сети результат. В ряде случаев некоторые вершины сохраняют свою позицию в соседних слоях только приблизительно, в результате чего могут возникать «закручивания» и искажения в позициях этих вершин, что снижает читаемость визуализации. Эта проблема может быть решена, например, модификацией энергии системы с введением дополнительных сил притяжения.

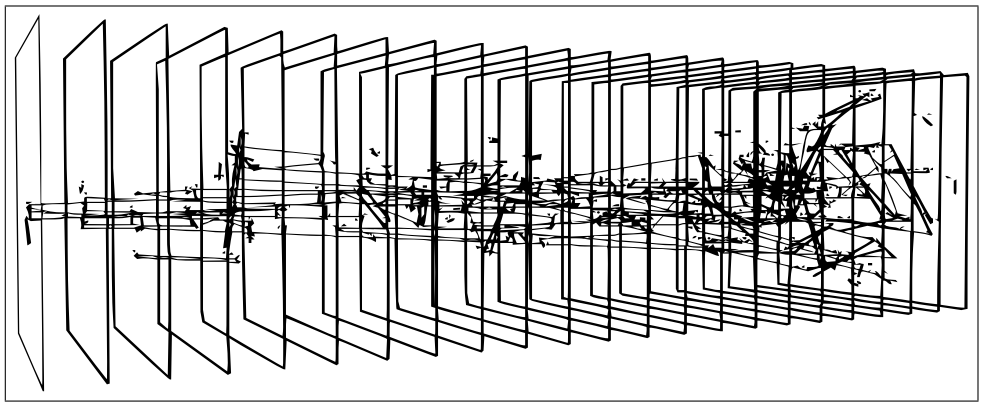
Важным направлением развития, на наш взгляд, является создание системы для автоматического анализа сложных сетей. Сейчас поиск постоянных групп общения, поиск шаблонных и аномальных структур в сообществе выполняется в ручном режиме. Было бы удобно иметь набор средств для автоматизированного поиска харак-



(a)



(b)



(c)

Рис. 7. Анализ динамики интенсивности общения. (a) VNC-dataset. (b) Enron-dataset. (c) Twitter-dataset.

терных шаблонов динамики общения на основе построенных укладок графа. Такая система могла бы, например, производить поиск и выделять вероятные ситуации «смена идентификатора пользователя».

Кроме того, мы планируем провести более детальный анализ динамики общения в сети, провести исследования устойчивости структур общения, построить класси-

фикацию шаблонов общения в этих структурах.

Список литературы

[1] R. Albert, A.-L. Barabasi, "Statistical mechanics of complex networks", *Reviews of Modern Physics*, **74** (2002), 47–97.

[2] K. Boitmanis, U. Brandes, C. Pich, "Visualizing internet evolution on the autonomous systems level", *Proc. 15th Intl. Symp. Graph Drawing*, 2007, 365–376.

[3] J. Branke, "Dynamic graph drawing", *Drawing Graphs, Springer Lecture Notes In Computer Science*, 2001, 228–246.

[4] L. Chen, A. Buja, "Local multidimensional scaling for nonlinear dimension reduction, graph layout and proximity analysis", *Journal of the American Statistical Association*, **104**:485 (2009), 209–219.

[5] S. Diehl, C. Görg, "Graphs, they are changing", *Proc. 10th Int. Symp. on Graph Drawing*, 2002, 23–30.

[6] T. Dwyer, "Two and a Half Dimensional Visualisation of Relational Networks", *PhD thesis*, The University of Sydney, 2005.

[7] P. Eades, "A heuristic for graph drawing", *Congressus Numerantium*, **42**, 1984, 149–160.

[8] C. Erten, P. J. Harding, S. G.Kobourov, K. Wampler, G. Yee, "Exploring the computing literature using temporal graph visualization", *Proc. of SPIE*, **5295** (2004), 45–56.

[9] T. Fruchterman, E. Reingold, "Graph drawing by force-directed placement", *Softw. Pract. Exp.*, **21**:11 (1991), 1129–1164.

[10] X. Fu, S.-H. Hong, N. S. Nikolov, X. Shen, Y. Wu, K. Xu, "Visualization and analysis of email networks", *Proc. of Asia-Pacific Symp. on Visualisation*, 2007, 1–8.

[11] E. R. Gansner, Y. Koren, S. C. North, "Graph drawing by stress majorization", *Proc. 12th Int. Symposium on Graph Drawing*, 2004, 239–250.

[12] H.Hanstein, G. Groh, "Interactive visualization of dynamic social networks", *GI Jahrestagung*, **2**, 2008, 929–936.

[13] D. Harel, Y. Koren, "A fast multi-scale method for drawing large graphs", *Journal of Graph Algorithms and Applications*, **6**:3 (2002), 282–285.

[14] T. Kamada, S. Kawai, "An algorithm for drawing general undirected graphs", *Inform. Process. Lett.*, **31** (1989), 7–15.

[15] B. Klimt, Y. Yang, "Introducing the enron corpus", *Proc. of First Conf. on Email and Anti-Spam*, 2004.

[16] J. B. Kruskal, J. B. Seery, "Designing network diagrams", *Proc. of the First General Conference on Social Graphics*, 1980, 22–50.

[17] K. Misue, P. Eades, W. Lai, K. Sugiyama, "Layout adjustment and the mental map", *Journal of Visual Languages and Computing*, **6**:2 (1995), 183–210.

[18] A. Noack, "An energy model for visual graph clustering", *Proc. 11th Int. Symp. on Graph Drawing*, 2003, 425–436.

[19] C. Pich, "Applications of Multidimensional Scaling to Graph Drawing", *PhD thesis*, Universitat Konstanz, 2009.

[20] H. Purchase, "Which aesthetic has the greatest effect on human understanding?", *Proc. 5th Int. Symp. on Graph Drawing*, 1998, 248–261.

[21] W. T. Tutte, "How to draw a graph", *Proc. London Math. Society*, **13**, 1963, 743–768.

[22] *Система визуализации графов GraphVis.*, http://graphvis.googlecode.com.

The Analysis of Complex Networks with Dynamic Graph Visualization

Pupyrev S.N., Tikhonov A.V.

Keywords: graph drawing, dynamic graph, visualization, complex network analysis

The work describes a technique for dynamic graph visualization that helps us to analyze complex networks. Our method builds layouts for a sequence of graphs, while preserving the mental map. We show how to modify force-directed algorithms for the dynamic graphs. The proposed method is applied for visualization of several social networks. The experiments demonstrate the effectiveness of dynamic graph drawing for analyzing real-world data and the evolution of social relationships changing over time.

Сведения об авторах:
Пупырев Сергей Николаевич,
г. Екатеринбург, Уральский государственный университет, аспирант;
Тихонов Алексей Викторович,
г. Москва, ООО Corelogic,
директор по информационным технологиям, ведущий аналитик.