УДК 519.6

ВИЗУАЛИЗАЦИЯ КООПЕРАТИВНЫХ СХЕМ: МЕТОД ПРОРИСОВКИ ГИПЕРРЕБЕР ИЕРАРХИЧЕСКОГО МНОГОСЛОЙНОГО ГИПЕРГРАФА

Васильев Юрий Михайлович, асп.
Фридман Григорий Морицович, д-р техн. наук, проф.

Санкт-Петербургский государственный экономический университет, Садовая ул., 21, Санкт-Петербург, Россия, 191023; e-mail: vas_yu_m@mail.ru; grifri@finec.ru

Цель: решение задачи прорисовки гиперребер иерархического гиперграфа при выполнении специфических требований по его укладке. *Обсуждение*: прорисовка гиперребер – заключительный этап метода Сугиямы для укладки иерархического гиперграфа. При этом необходимо оптимизировать значения всех метрик эстетичности укладки, что крайне затруднительно достичь в рамках универсального подхода. Предложено разбить общую задачу на последовательность подзадач, для каждой из которых предложены точные и эвристические алгоритмы решения. *Результаты*: сформулирован пошаговый метод прорисовки гиперребер иерархического гиперграфа, эффективность которого подтверждена результатами проведенных массовых расчетов на сгенерированной тестовой выборке графов.

Ключевые слова: направленный ациклический иерархический граф, гиперграф, гиперребро, прорисовка ортогональных гиперребер, укладка графа, метод ветвей и границ, эвристический алгоритм.

DOI: 10.17308/meps.2017.3/1628

1. Введение

Наиболее известный метод для размещения направленных ациклических иерархических графов – это метод Сугиямы [11], в соответствии с которым общая задача по укладке графа разбивается на несколько последовательных подзадач:

- разбиение множества вершин графа на подмножества (слои), при этом каждое подмножество располагается одно под другим, все вершины-предки расположены выше вершин-потомков;
- определение последовательности вершин на слоях с целью минимизации пересечения ребер;
- поиск координат вершин, учитывая набор заранее выбранных эстетических критериев;
- прорисовка ребер.

При прорисовке ребер вершины получают свои финальные координаты, которые были нестрого определены назначенным слоем и координатами вершин на предыдущих шагах метода Сугиямы.

Граф в диаграммах потоков данных и схем кооперации рассматривается как иерархический гиперграф, который отличается от обычного графа тем, что его гиперребра (которые, по сути, являются объединением нескольких обычных ребер) соединяют не две вершины различных слоев, а некоторое множество вершин-источников верхнего слоя с некоторым множеством вершин-стоков, расположенных в нижних слоях. При этом гиперребра прорисовывают вертикальными и горизонтальными сегментами [8].

В математических терминах – слойный гиперграф $H = (V,\ E_H, \lambda)$ – это граф, где каждой вершине $u \in V$ оператор λ ставит в соответствие натуральное число (номер слоя) $\lambda(u)$, $1 \le \lambda(u) \le k$, а E_H – множество гиперребер, при этом каждое гиперребро $e = (S, T)$ инцидентно множеству вершин-предков $S \subset V$ и множеству вершин-потомков $T \subset V$. Если S состоит только из одной вершины, то такое гиперребро называется «одноисточным» [7]. На этапе прорисовки гиперребер для каждой вершины – слойного гиперграфа известны координаты $(x(u), y(u))$, где $u \in V$. В данной статье рассматривается случай, когда гиперребра соединяют только вершины соседних слоев.

Сформулированы следующие требования к прорисовке гиперребер, учитывающие особенности диаграмм потоков данных и схем кооперации (подробно они описаны в [13]):

1. вершины представлены в виде прямоугольников фиксированного размера шириной 2ρ и высотой ρ.

2. каждое гиперребро инцидентно одной вершине-источнику и смежным вершинам нижнего слоя.

3. каждое гиперребро $e \in E_H$ состоит из трех сегментов:

- первый сегмент $sV_1(e)$ представляет собой вертикальный отрезок, выходящий из вершины-источника;
- второй сегмент $sH(e)$ представляет собой горизонтальный отрезок, проходящий на одной ординате с ординатой нижнего конца сегмента $sV_1(e)$ и имеющий с ним общую точку;
- третий сегмент $sV_2(e)$ представляет собой набор вертикальных отрезков, начинающихся на сегменте $sH(e)$ и входящих в вершины-стоки.

4. наложение (наличие бесконечного числа общих точек) двух гиперребер e_m и e_n возможно только для их вертикальных сегментов $sV_2(e_m)$ и $sV_2(e_n)$.

5. если два гиперребра имеют общую вершину-сток, и при этом для одного гиперребра она является крайней левой среди всех инцидентных этому ребру вершин-стоков, а для второго гиперребра – крайне правой, то

горизонтальные сегменты гиперребер могут иметь одинаковую ординату.

6. сегмент $sV_1(e)$ каждого гиперребра e выходит строго из середины вершины-источника.

7. все вертикальные отрезки сегмента $sV_2(e)$ каждого гиперребра e входят строго в середину каждой инцидентной ему вершины-стока.

Для упрощения задачи прорисовки ребер исходного гиперграфа разобьем его на упорядоченное множество подграфов, которые последовательно составляются из двух его соседних слоев. Тогда прорисовку гиперребер рационально проводить на каждом двухслойном подграфе отдельно и последовательно сверху вниз.

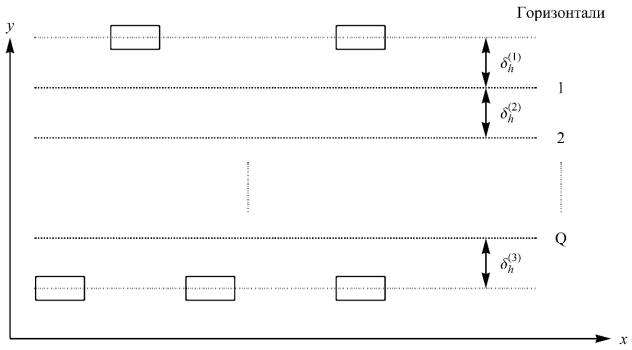

Рис. 1. Шаблон прорисовки гиперребер для двухслойного гиперграфа, горизонтали изображены пунктирными линиями.

Введем термин «горизонталь», под которым будем понимать одно из допустимых положений горизонтальных сегментов гиперребер двухслойного гиперграфа, [13]. Для прорисовки гиперребер двухслойного гиперграфа, в котором расположены Q горизонталей ($Q \geq 2$), будем использовать три вида расстояний, см. рис. 1:

- $\delta_h^{(1)}$ – расстояние между верхним слоем вершин и верхней (первой) горизонталью;
- $\delta_h^{(2)}$ – расстояние между соседними горизонталями;
- $\delta_h^{(3)}$ – расстояние между нижней (последней) горизонталью и нижним слоем вершин гиперграфа.

При этом $\delta_{h,min}^{(i)} \leq \delta_h^{(i)} \leq \delta_{h,max}^{(i)}$, $i = 1,2,3$, где $\delta_{h,min}^{(i)}$ и $\delta_{h,max}^{(i)}$ – заданные минимальное и максимальное возможные расстояния. Величины $\delta_h^{(1)}$, $\delta_h^{(2)}$ и $\delta_h^{(3)}$ рассчитываются для каждого двухслойного подграфа по отдельности.

При выполнении прорисовки гиперребер будем руководствоваться следующими метриками эстетичности:

- число пересечений гиперребер ($M1$);
- номер самой нижней занятой горизонтали ($M2$);
- сбалансированное (центрированное) положение горизонтальных сегментов гиперребер ($M3$);

- отклонение относительного удлинения (отношение ширины рисунка к его высоте) укладки от общепринятых значений ($M4$).

Чем меньше значения метрик эстетичности, тем укладка более удобна для восприятия.

На данный момент не существует метода [2, 3, 7-10], который позволяет одновременно оптимизировать все четыре метрики эстетичности для сформулированных требований к укладке гиперграфа. В статье представлены алгоритмы, каждый из которых воздействует лишь на одну из метрик. В соответствии с этим разделим задачу прорисовки гиперребер на четыре этапа:

- минимизация числа пересечений гиперребер;
- минимизация числа занятых горизонталей в укладке гиперграфа;
- поиск сбалансированного положения горизонтальных участков гиперребер;
- определение относительного удлинения укладки гиперграфа.

2. Набор алгоритмов для прорисовки гиперребер

Дан двухслойный гиперграф $H_2 = \left(V_1, V_2, E_{H_2}\right)$, где $V_1 = \left\{u_1, \ldots, u_{N_1}\right\}$ и $V_2 = \left\{v_1, \ldots, v_{N_2}\right\}$ – множества вершин верхнего и нижнего слоя соответственно, а E_{H_2} – множество одноисточных гиперребер. Для каждой вершины $v \in V_1 \cup V_2$ двухслойного гиперграфа известны координаты $\left(x(v), y(v)\right)$.

2.1. Алгоритмы минимизации числа пересечений гиперребер

2.1.1. Обобщенная постановка задачи минимизации пересечений гиперребер

В статье [13] сформулирована точная математическая постановка задачи минимизации пересечений гиперребер для представленных требований к прорисовке двухслойного гиперграфа. Числовое решение этой задачи целочисленного программирования, определяющее горизонтальные сдвиги вершин (для устранения недопустимых наложений) и соответствие горизонтальных сегментов гиперребер горизонталям ($h(e), \forall e \in E_{H_2}$), можно получить по методу ветвей и границ [16]. В результате каждая вершина гиперграфа $v \in V_1 \cup V_2$ получает новую абсциссу $x'(v)$.

Предложенный алгоритм дает минимальное значение метрики $M1$ для двухслойного гиперграфа, но при этом не оптимизирует значения метрик $M2$, $M3$ и $M4$. Для многослойного гиперграфа алгоритм должен быть последовательно применен к двухслойным подграфам при фиксированном верхнем слое вершин в каждом из них.

Выполненные числовые расчеты для модельных данных (относительно небольшой размерности) показали, что вследствие вычислительной сложности задачи получение точного решения на полномасштабных реальных данных требует значительных временных ресурсов и неприменимо в случае жестких временных рамок. Таким образом, возникает необходимость в декомпозиции задачи с возможностью использования быстрых эвристических алгоритмов.

2.1.2. Формулирование задачи минимизации пересечений гиперребер двухслойного подграфа как задачи устранения циклов в направленном графе

Одним из подходов к решению задачи минимизации числа пересечений гиперребер является ее формулировка в виде задачи устранения циклов в некотором взвешенном направленном графе. Каждая вершина этого графа соответствует горизонтальному сегменту гиперребра, и каждая пара вершин соединена двумя противонаправленными взвешенными дугами. Направление дуги показывает, какая вершина (т.е. соответствующий ей горизонтальный сегмент) расположена выше, а ее вес определяется взвешенной суммой числа пересечений и недопустимых наложений гиперребер, соответствующих вершинам.

Для решения задачи устранения циклов во взвешенном направленном графе известна как точная постановка, так и набор эвристических алгоритмов различной эффективности и быстродействия [5]. В результате каждому горизонтальному сегменту гиперребра e исходного гиперграфа ставится в соответствие уникальная горизонталь $h(e)$. При этом, однако, возможно появление недопустимых наложений гиперребер. В таком случае необходимо выполнять дополнительную постобработку результата, которая устраняет эти нарушения к требованиям по прорисовке.

2.1.3. Алгоритм постобработки для устранения недопустимых наложений

Задача устранения недопустимых наложений может быть сформулирована в виде задачи целочисленного программирования как частный случай постановки, представленной в статье [13], без учета переменных и ограничений, связанных с минимизацией числа пересечений гиперребер. Недопустимые наложения должны быть устранены за счет сдвигов вершин нижнего и/или верхнего слоя на определенное целое число «единичных сдвигов» длины Δ.

Введем в рассмотрение 5 видов целочисленных переменных Z', Z'', $AZ^{'}$, $AZ^{''}$ и BV.

Переменные $Z_i^{'}$ и $Z_j^{''}$ характеризуют количество единичных сдвигов вершин верхнего $u_i \in V_1$ и нижнего слоя $v_j \in V_2$ соответственно, а их знаки определяют направления сдвигов. Тогда новые (вследствие проведенных сдвигов) абсциссы вершин $u_i \in V_1$ и $v_j \in V_2$ будут равны

$$x'(u_i) = x(u_i) + \Delta \times Z_i^{'},$$
$$x'(v_j) = x(v_j) + \Delta \times Z_j^{''}.$$

Переменные $AZ_i^{'}$ и $AZ_j^{''}$ – определяют число единичных сдвигов вершин u_i и v_j, соответственно, т.е. $AZ_i^{'} = \left|Z_i^{'}\right|$, а $AZ_j^{''} = \left|Z_j^{''}\right|$.

Переменные $BV_{\mu,j}$ – это булевы фиктивные переменные, которые вводятся для каждого гиперребра $e_m = (u_\mu, T_\mu)$ и множества вершин $v_j \in V_2^m$, где V_2^m – объединение вершин нижнего слоя, инцидентных гиперребрам, чьи горизонтальные сегменты находятся выше горизонтального сегмента гиперребра e_m.

Целевая функция задачи целочисленного программирования минимизирует общее число сдвигов вершин верхнего и нижнего слоя:

$$L\left(Z',Z'',\ AZ',AZ'',BV\right)=\sum_{i=1}^{|V_1|}AZ_i'+\sum_{j=1}^{|V_2|}AZ_j''\to\min. \quad (1)$$

Система ограничений задачи целочисленного программирования имеет вид:

$$x\left(v_j\right)+\Delta Z_j''\left(v_j\right)\ \le x\left(u_\mu\right)+\Delta Z_\mu'+K\ BV_{\mu,j}-\varepsilon,$$

$$\forall e_m=\left(u_i,T_i\right),\quad v_j\in V_2^m \quad (2)$$

$$x\left(u_\mu\right)+\Delta Z_\mu'\le x\left(v_j\right)+\Delta Z_j''\left(v_j\right)+K\left(1-\ BV_{\mu,j}\right)-\varepsilon, \quad (3)$$

$$\forall e_m=\left(u_i,T_i\right),\quad v_j\in V_2^m$$

$$AZ_i'\ge Z_i',\quad AZ_i'\ge -Z_i',\quad \forall\ u_i\in V_1 \quad (4)$$

$$AZ_j''\ge Z_j'',\quad AZ_j''\ge -Z_j'',\quad \forall\ v_j\in V_2 \quad (5)$$

$$x\left(u_{i+1}\right)+\Delta Z_{i+1}'-\left(x\left(u_i\right)+\Delta Z_i'\right)\ge\delta_w,\quad i=1,\ldots,|V_1|-1 \quad (6)$$

$$x\left(v_{j+1}\right)+\Delta Z_{j+1}''-\left(x\left(v_j\right)+\Delta Z_j''\right)\ge\delta_w,\quad j=1,\ldots,|V_2|-1\ , \quad (7)$$

где K – произвольное большое положительное число, а ε – малая положительная величина. В приведенных ниже расчетах было принято, что $K=1.3\max\left(x\left(u_{|V_1|}\right)-x\left(u_1\right),x\left(v_{|V_2|}\right)-x\left(v_1\right)\right)$ и $\varepsilon=0.05\rho$.

Ограничения (2) и (3) устраняют недопустимые наложения, обеспечивая выполнение условия о том, что для каждого гиперребра $e_m=\left(u_\mu,T_\mu\right)$ ни одна из вершин из V_2^m не должна лежать строго под вершиной u_μ. Ограничения (4) и (5) связаны с необходимостью линеаризации условий $AZ_i'=\left|Z_i'\right|$ и $AZ_j''=\left|Z_j''\right|$, которые определяют для всех вершин $u_i\in V_1$ и $v_j\in V_2$ число их единичных сдвигов. Условия (6) и (7) гарантируют, что расстояние между соседними вершинами одного слоя не меньше заранее определенной величины δ_w.

Точная постановка (1) – (7) может быть использована при прорисовке многослойного гиперграфа за счет последовательного ее решения для каждой пары соседних слоев при фиксированном верхнем слое в каждой паре.

2.2. Алгоритмы для сокращения числа горизонталей

Решение задачи о минимизации числа пересечений дает возможность минимизировать метрику $M1$ в результате сопоставления горизонтальному сегменту каждого гиперребра e уникальной горизонтали $h(e)$. Оптимизация метрики $M2$ (номер самой нижней занятой горизонтали) может быть выполнена за счет расположения на одной горизонтали нескольких горизонтальных сегментов, т.е. за счет сокращения числа горизонталей, при этом, однако, число пересечений гиперребер не должно увеличиться.

Описанные ниже алгоритмы размещают горизонтальный сегмент каждого гиперребра на самой высокой доступной ему горизонтали.

Простейшим вариантом алгоритма для сокращения числа горизонталей является последовательный подъем горизонтальных сегментов гиперребер (уменьшение номера соответствующей ему горизонтали), начиная с

верхнего. При этом подъем горизонтального сегмента $sH(e_m)$ на каждую более высокую горизонталь осуществляется, только если его проекция на ось абсцисс не пересекается ни с одной из проекций горизонтальных сегментов $sH(e_{n_1}),\ldots,sH(e_{n_2})$, уже расположенных на этой горизонтали:

$$pr_x\, sH(e_m) \cap pr_x\, sH(e_n) = \varnothing \quad \forall n \in \{n_1,\ldots,n_2\},$$

где $pr_x\, sH(e)$ – проекция горизонтального сегмента гиперребра e на ось абсцисс. Все горизонтали, на которых в результате выполнения такого алгоритма не осталось горизонтальных участков гиперребер, исключаются, и, соответственно, уменьшается номер самой низкой используемой горизонтали.

Более сложный, обобщенный, алгоритм также последовательно поднимает горизонтальные сегменты гиперребер, не рассматривая, однако, некоторые варианты их наложения на одной горизонтали как препятствие для подъема. На рис. 2 показаны все возможные варианты допустимого (не препятствующего подъему) расположения горизонтальных сегментов гиперребер на одной горизонтали в случае их наложения.

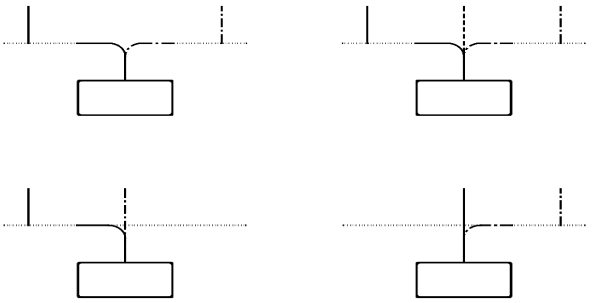

Рис. 2. Возможные варианты допустимого расположения горизонтальных сегментов гиперребер на одной горизонтали в случае их наложения

Если гиперграф содержит более одного гиперребра, то на первом шаге алгоритма формируется трехмерный список $D = (d_{m,j,\alpha})$, $m = 1,\ldots,|E_{H_2}|$, $j = 1,\ldots,|V_2|$, $\alpha = 1,2$. Пара элементов $d_{m,\ j,1}$ и $d_{m,\ j,2}$ этого списка содержит полную информацию о связи гиперребра $e_m = (u_i, T_i) \in E_{H_2}$ с вершиной нижнего слоя $v_j \in V_2$, причем

$$d_{m,\ j,1} = \begin{cases} m, & \text{если } x'(v_j) \in pr_x\, sH(e_m), \\ 0, & \text{в противном случае,} \end{cases}$$

а параметр $d_{m,\ j,2}$ принимает одно возможных значений «*l*», «*r*», «*c*», «*o*» в зависимости от взаимного расположения горизонтального сегмента $sH(e_m)$ гиперребра $e_m = (u_\mu, T_\mu)$, его вершины-источника u_μ и вершины v_j нижнего слоя, см. табл. 1.

Таблица 1

Возможные значения параметра $d_{m,\ j,2}$

$d_{m,\ j,2}$	Описание
«*l*»	$x'(v_j) \in pr_x\, sH(e_m),\ x'(v_j) = \min x'(v) \ \mid\ v \in T_i,\ x'(v_j) < x'(u_i)$
«*r*»	$x'(v_j) \in pr_x\, sH(e_m),\ x'(v_j) = \max x'(v) \ \mid\ v \in T_i,\ x'(v_j) > x'(u_i)$
«*c*»	$\mid T_i \mid = 1,\ v_j \in T_i,\ x'(v_j) = x'(u_i)$
«*o*»	не выполнено ни одно из сформулированных выше условий.

Алгоритм по назначению горизонтальным сегментам гиперребер новых горизонталей представлен ниже в виде псевдокода 1.

Псевдокод 1.
Input: список отсортированных гиперребер $sort^{sH}$, трехмерная матрица D
Output: номера горизонталей гиперребер $h_{high}(e),\ \forall e \in E_{H_2}$.

1. $h_{high}\left(sort_1^{sH}\right) = 1$
2. $D_1^{'} = D_1$
3. for $m = 2, \ldots, \left|E_{H_2}\right|$
4. $flag = True$
5. $n = length(D') + 1$
6. while $flag \wedge n > 1$
7. $n--$
8. if not $compare\left(D_m, D_n^{'}\right)$ then
9. $n++$
10. $flag = False$
11. end if
12. end while
13. if $n > length(D')$ then
14. $D_n^{'} = D_m$
15. else
16. $D_n^{'} = reform\left(D_m, D_n^{'}\right)$
17. end if
18. $h_{high}\left(sort_m^{sH}\right) = n$
19. end for
20. return h_{high}

Величины D_m и $D_n^{'}$ обозначают множества элементов $\left\{d_{m,j,\alpha}\right\}$ и $\left\{d_{n,j,\alpha}^{'}\right\}$, $j = 1, \ldots, \left|V_2\right|$, $\alpha = 1,2$, при фиксированных значениях m и n соответственно. Операция присваивания $D_n^{'} = D_m$ означает $d_{n,j,\alpha}^{'} = d_{m,j,\alpha}$, $\forall j = 1, \ldots, \left|V_2\right|$, $\forall \alpha = 1,2$. Оператор $length(D')$ вычисляет количество элементов сечения списка D' по первому индексу. Работа оператора $compare\left(D_m, D_n^{'}\right)$, сравнивающего величины D_m и $D_n^{'}$, показана ниже в псевдокоде 2.

Псевдокод 2.

Input: множества D_m и $D_n^{'}$.

Output: можно ли разместить горизонтальные сегменты на одной горизонтали, *Trueили False*.

1. $k = 1$
2. *isCommon*=*True*
3. while $isCommon \wedge j \le |V_2|$
4. if $d_{m,j,1} == 0 \vee d_{n,j,1}^{'} == 0 \vee featcomp\left(d_{m,j,2}, d_{n,j,2}^{'}\right)$ then
5. $j++$
6. else
7. *isCommon*=*False*
8. end if
9. end while
10. return *isCommon*

Оператор $featcomp(arg1, arg2)$ принимает значение *True* только для аргументов, показанных в табл. 2. Во всех остальных случаях он равен *False*.

Таблица 2

Возможные варианты значения аргументов $arg1$ и $arg2$, для которых $featcomp(arg1, arg2) = True$

$arg1$	«l»	«r»	«l»	«c»	«r»	«c»	«c»
$arg2$	«r»	«l»	«c»	«l»	«c»	«r»	«c»

Оператор преобразования множества $D_n^{'}$, $reform\left(D_m, D_n^{'}\right)$ описан ниже в псевдокоде 3.

Псевдокод 3.

Input: множества D_m и $D_n^{'}$.

Output: преобразованное множество $D_n^{'}$.

1. for $j = 1, \ldots, |V_2|$
2. if $d_{m,j,1} \neq 0$ then
3. if $d_{n,j,1}^{'} == 0$ then
4. $d_{n,j,2}^{'} = d_{m,j,2}$
5. else
6. switch$\left\{d_{n,j,2}^{'}, d_{m,j,2}\right\}$
7. case$\{«l», «r»\} \vee \{«r», «l»\}$
8. $d_{n,j,2}^{'} = «o»$
9. case$\{«l», «c»\} \vee \{«c», «l»\}$
10. $d_{n,j,2}^{'} = «l»$
11. case$\{«c», «r»\} \vee \{«r», «c»\}$
12. $d_{n,j,k}^{'} = «r»$
13. end if
14. end if
15. $d_{n,j,1}^{'} = d_{n,j,1}^{'} + d_{m,j,1}$
16. end for
17. return $D_n^{'}$

2.3. Алгоритм сбалансированного расположения горизонтальных сегментов гиперребер

Алгоритмы, оптимизирующие метрику $M2$, уменьшают число горизонталей, на которых расположены горизонтальные сегменты его гиперребер, до величины $Q \le \left|E_{H_2}\right|$.

Минимизация метрики $M3$ осуществляется алгоритмом, который определяет сбалансированное (усредненное) положение горизонтальных сегментов гиперребер. В соответствии с ним для каждого гиперребра e (последовательно, начиная с того, чей горизонтальный сегмент занимает горизонталь с наибольшим номером и слева направо, если горизонтальные сегменты гиперребер расположены на одной горизонтали) вычисляется самое нижнее допустимое положение его горизонтального сегмента $h_{low}(e)$, при котором не увеличивается число пересечений гиперребер. В результате определяется интервал допустимых номеров горизонталей, и горизонтальный сегмент помещается в «среднее» положение

$$h(e) = \frac{h_{high}(e) + h_{low}(e)}{2}.$$

2.4. Алгоритм минимизации отклонения относительного удлинения укладки от общепринятых значений

Минимизация метрики $M4$ выполняется за счет решения оптимизационной задачи для двухслойного гиперграфа, в результате чего относительное удлинение (отношение высоты к ширине) его укладки приближается к некоторому элементу ar из заранее определенного списка AR общепринятых значений [12].

Целевая функция задачи линейного программирования минимизирует величину dev, обозначающую отклонение относительного удлинения укладки двухслойного гиперграфа от ar, одного из элементов списка AR:

$$F_{ar}\left(\delta_h^{(1)}, \delta_h^{(2)}, \delta_h^{(3)}, dev\right) = dev \to \min. \quad (8)$$

Система ограничений задачи целочисленного программирования имеет вид:

$$dev \ge \frac{\delta_h^{(1)} + (Q-1)\times\delta_h^{(2)} + \delta_h^{(3)}}{ar \times width} - 1, \quad (9)$$

$$dev \ge -\frac{\delta_h^{(1)} + (Q-1)\times\delta_h^{(2)} + \delta_h^{(3)}}{ar \times width} + 1, \quad (10)$$

$$\delta_{h,min}^{(i)} \le \delta_h^{(i)} \le \delta_{h,max}^{(i)}, \quad i = 1,2,3, \quad (11)$$

где $width$ – ширина укладки двухслойного гиперграфа:

$$width = \max_{v \in V_1 \cup V_2} x(v) - \min_{v \in V_1 \cup V_2} x(v).$$

Ограничения (9) – (10) связаны с необходимостью линеаризации условия $dev = \left|\left(\delta_h^{(1)} + (Q-1)\times\delta_h^{(2)} + \delta_h^{(3)}\right) / (ar \times width) - 1\right|$, которое определяет значение отклонения относительного удлинения от величины ar. Условия (11) гарантируют, что значения переменных $\delta_h^{(1)}$, $\delta_h^{(2)}$ и $\delta_h^{(3)}$ находятся в выбранных диапазонах.

Оптимизационная задача в точной постановке (8) – (11) решается для $\forall ar \in AR$. При прорисовке гиперребер используются такие величины переменных $\delta_h^{(1)}$, $\delta_h^{(2)}$, $\delta_h^{(3)}$, при которых $dev = \left| \left(\delta_h^{(1)} + (Q-1) \times \delta_h^{(2)} + \delta_h^{(3)} \right) / \left(ar \times width \right) - 1 \right|$ принимает минимальное значение. Если этот минимум достигается при нескольких ar, то из них выбирается минимальное значение.

Данный алгоритм является завершающим этапом перед получением укладки гиперграфа.

3. Числовые расчеты

Используя некоторые последовательности из представленных алгоритмов, можно построить гибридный алгоритм [15], т.е. сформировать для любого заданного гиперграфа, в зависимости от его характеристик, эффективную последовательность алгоритмов, которая в результате применения даст наилучшие значения метрик эстетичности для укладки этого гиперграфа за приемлемое время счета.

Работоспособность предложенных в статье алгоритмов была проверена в рамках проведенных массовых расчетов для сгенерированной выборки тестовых графов. При генерировании всех исходных данных при вычислении матрицы ограничений и вектора коэффициентов целевой функции для решения задач в точной постановке, а также для разработки эвристических алгоритмов применена компьютерная математическая среда Mathematica 11 [13]; поиск числового решения выполнялся в оптимизаторе Gurobi Optimizer 7.0 [4] при помощи метода ветвей и границ.

Были выбраны следующие значения параметров прорисовки:

$\delta_{h,min}^{(1)} = \rho$, $\delta_{h,max}^{(1)} = 2.5\rho$,

$\delta_{h,min}^{(2)} = \rho$, $\delta_{h,max}^{(2)} = 5\rho$,

$\delta_{h,min}^{(3)} = \rho$, $\delta_{h,max}^{(3)} = 2.5\rho$,

$\Delta = 0.5\rho$, $\delta_w = 3\rho$,

$$AR = \left\{1, \frac{3}{4}, \frac{8}{11}, \frac{2}{1+\sqrt{5}}, \frac{1}{1.85}, \frac{1}{1+\sqrt{2}}, \frac{1}{2.76}, \frac{1}{4}\right\}.$$

В данном разделе представлены результаты работы следующих алгоритмов/последовательности алгоритмов:

- ТП – обобщенная точная постановка задачи минимизации пересечений гиперребер;
- ТА – точная постановка для решения задачи устранения циклов во взвешенном направленном графе с последующим применением алгоритма постобработки для устранения недопустимых наложений;
- ЭА – жадный алгоритм для решения задачи устранения циклов во взвешенном направленном графе [1] с последующим применением алгоритма постобработки для устранения недопустимых наложений;
- ПЭ – алгоритм последовательного подъема горизонтальных сегментов гиперребер с последующим балансированием их расположения;

- РЭ – обобщенный алгоритм подъема горизонтальных сегментов гиперребер с последующим балансированием их расположения.

Для примера применения алгоритмов при прорисовке гиперребер ниже представлены (рис. 3):

- укладка исходного графа относительного небольшого размера. При этом были решены задачи: разбиения множества вершин графа на подмножества (слои), определения последовательности вершин на слоях с целью минимизации пересечения ребер, поиска координат вершин;
- результат решения задачи в обобщенной постановке задачи минимизации пересечений гиперребер соответствующего гиперграфа;
- результат применения обобщенного алгоритма подъема горизонтальных сегментов гиперребер с последующим балансированием их расположения.

В табл. 3 приведены количественные характеристики полученных решений, а именно значения метрик эстетичности ($M1$, $M2$, $M3$ и $M4$) и величина T – время работы алгоритма для одного графа, взятого из сгенерированной тестовой выборки. При этом значение $M2$ равно сумме номеров самых нижних горизонталей каждого двухслойного подграфа, а величина $M3$ вычисляется по формуле:

$$M3 = \frac{1}{|E_H|} \sum_{e \in E_H} \left| 2h(e) - h'_{high}(e) - h'_{low}(e) \right|,$$

где $h'_{high}(e)$ ($h'_{low}(e)$) – номер самой высокой (низкой) горизонтали, для которой проекция $pr_x sH(e)$ с выколотыми крайними точками не имеет пересечений с проекциями на ось абсцисс горизонтальных сегментов гиперребер данной горизонтали

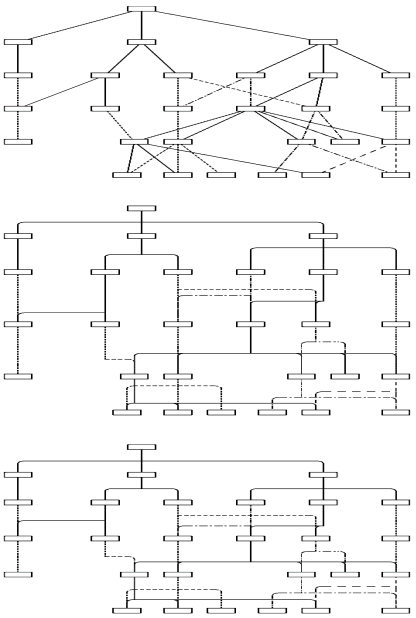

Рис. 3. Укладка исходного графа; применение алгоритма ТП для соответствующего гиперграфа; применение алгоритма РЭ

Расчет $M4$ выполняется следующим образом:

$$M4 = \sum_{i=1}^{k-1} \min_{ar \in AR} \left| \frac{height_i}{width_i} - ar \right| / ar \ ,$$

где $height_i$ и $width_i$ – высота и ширина укладки i-го по порядку двухслойного подграфа исходного k-слойного гиперграфа. В зависимости от поставленной задачи, от требований к качеству прорисовки и быстродействию алгоритмов укладки гиперграфа можно подобрать оптимальную последовательность алгоритмов [15].

Таблица 3

Значения метрик эстетичности и время работы алгоритмов

Алгоритм	Значение метрики	ТП	ТЭ	ЭА
Начальный этап	*M1*	38	38	40
	M2	36	36	36
	M3	4,8	4,4	3,6
	M4	0,38	0,38	0,38
	T, сек	2,2	5,8	2,7
ПЭ	*M1*	38	38	40
	M2	31	30	30
	M3	3,9	3,4	3
	M4	0,38	0,38	0,38
	T, сек	0,005	0,006	0,006
РЭ	*M1*	38	38	40
	M2	27	26	27
	M3	3,5	3,1	2,9
	M4	0,03	0,03	0,03
	T, сек	0,02	0,02	0,02

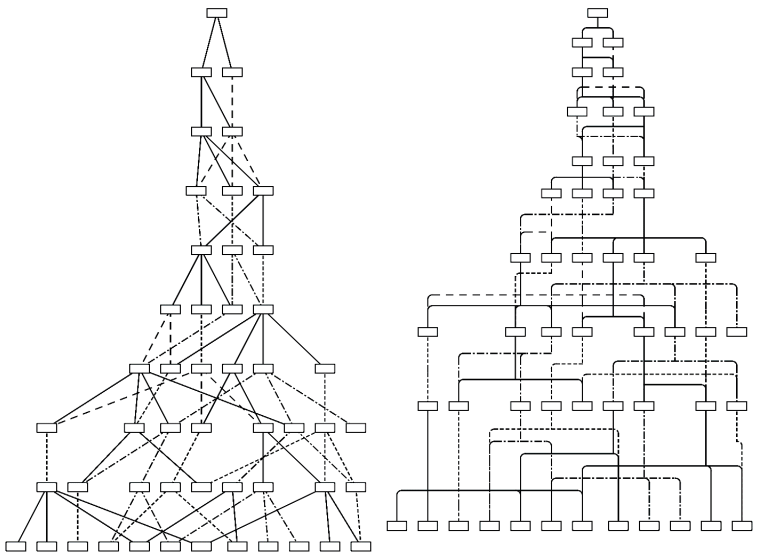

Рис. 4. Укладка иерархического многослойного графа (слева), укладка соответствующего гиперграфа (справа)

На рис. 4 изображена укладка иерархического многослойного графа (слева), для которого были выполнены расчеты из табл. 3, и укладка соот-

ветствующему ему гиперграфа (справа). При этом укладка графа выполнена со 166 пересечениями ребер, а укладка гиперграфа с 38.

Заключение

В статье описан новый метод пошаговой укладки гиперребер иерархического гиперграфа, позволяющий выполнять все требования к прорисовке и оптимизировать сформулированные метрики эстетичности. Метод предполагает разбиение общей задачи прорисовки на последовательность подзадач, для каждой из которых предложены точные и эвристические алгоритмы решения. Выполнены массовые расчеты на сгенерированной тестовой выборке графов, продемонстрировавшие эффективность метода.

Список источников

1. Alaa A.K. Ismaeel. Dynamic Drawing of Hierarchical Graphs // *PHD thesis*, Karlsruher Instituts Technologie, 2012.

2. Eichelberger H. Aesthetics and Automatic Layout of UML Class Diagrams // *PHD Thesis*, Bayerischen Julius-Maximilians-Universität Würzburg, 2005.

3. Eschbach T., Gunther W., Becker B. Orthogonal hypergraph drawing for improved visibility // *Journal of Graph Algorithms and Applications*, vol. 10, no. 2, 2006, pp. 141-157.

4. Gurobi Optimizer. Available at: http://www.gurobi.com (accessed: 21.01.17).

5. Healy P., Nikolov N.S. Hierarchical drawing algorithms // *Handbook of Graph Drawing and Visualization*, vol. 1, chapter 13, 2005, pp. 409-454.

6. Sander G. A fast heuristic for hierarchical Manhattan layout // *Graph Drawing*, vol. 1027 of the series Lecture Notes in Computer Science, 2005, pp. 447-458.

7. Sander G. Layout of directed hypergraphs with orthogonal hyperedges // *Graph Drawing*, vol. 2912 of the series Lecture Notes in Computer Science, 2003, pp. 381-386.

8. Schulze C.D. Optimizing Automatic Layout for Data Flow Diagrams // *Diploma Thesis*, Christian-Albrechts-Universität zu Kiel, 2011.

9. Schulze C.D., Spönemann M., R. von Hanxleden. Drawing Layered Graphs with Port Constraints // *Journal of Visual Languages & Computing*, 2013, pp. 1-34.

10. Siebenhaller M. Orthogonal Graph Drawing with Constraints: Algorithms and Applications // *PHD Thesis*, Eberhard-Karls-Universitat, 2009.

11. Sugiyama K., Tagawa S., Toda M. Methods for visual undertanding of hierarchical systems // *IEEE Transactions on Systems, Man, and Cybernatics*, vol. SMC-11, no. 2, 1981, pp. 109-125.

12. Taylor M., Rodgers P. Applying Graphical Design Techniques to Graph Visualisation // *Proceedings of the Ninth International Conference on Information Visualisation*, 2005, pp. 651-656.

13. Wolfram|One. Available at: http://www.wolfram.com (accessed: 21.01.17).

14. Васильев Ю.М. Точная математическая постановка для прорисовки гиперребер двухслойного гиперграфа в схемах // *Современная экономика: проблемы и решения*, 2017, no. 1.

15. Васильев Ю.М., Фридман Г.М. Визуализация кооперативных схем: гибридный эвристический алгоритм для минимизации количества пересечений ребер при укладке графа // *Известия Санкт-Петербургского государственного экономического университета*, 2017, no. 1.

16. Мину М. *Математическое программирование. Теория и алгоритмы*. Москва, Наука, 1990.

COOPERATION SCHEME VISUALIZATION: HYPEREDGE ROUTING METHOD FOR HIERARCHICAL MULTILAYER HYPERGRAPH

Vasiliev Yuriy Mihailovich, graduate student
Fridman Gregory Moricovich, Dr. Sc. (Tech.), Prof.

Saint-Petersburg State University of Economics, Sadovaya str., 21, St. Petersburg, Russia, 191023; e-mail: vas_yu_m@mail.ru; grifri@finec.ru

Purpose: solving the problem of drawing hyper-edges of a hierarchical hypergraph while fulfilling specific requirements for its packing. *Discussion*: drawing hyperedges – the final stage of the method for laying Sugiyama hierarchical hypergraph. At the same time, it is necessary to optimize the values of all the metrics of aesthetic styling, which is extremely difficult to achieve in the framework of the universal approach. It is proposed to break the general problem into a sequence of subproblems, for each of which exact and heuristic decision algorithms are proposed. *Results*: a step-by-step method for drawing hyperreframs of a hierarchical hypergraph is formulated, the effectiveness of which is confirmed by the results of the mass calculations performed on the generated test sample of graphs.

Keywords: directed acyclic hierarchical graph, hypergraph, hyperedge, orthogonal hyperedge, graph packing, branch and boundary method, heuristic algorithm.

References

1. Alaa A.K. Ismaeel. Dynamic Drawing of Hierarchical Graphs. *PHD thesis*, Karlsruher Instituts Technologie, 2012.

2. Eichelberger H. Aesthetics and Automatic Layout of UML Class Diagrams. *PHD Thesis,* Bayerischen Julius-Maximilians-Universität Würzburg, 2005.

3. Eschbach T., Gunther W., Becker B. Orthogonal hypergraph drawing for improved visibility. *Journal of Graph Algorithms and Applications*, vol. 10, no. 2, 2006, pp. 141-157.

4. Gurobi Optimizer. Available at: http://www.gurobi.com (accessed: 21.01.17).

5. Healy P., Nikolov N.S. Hierarchical drawing algorithms. *Handbook of Graph Drawing and Visualization*, vol. 1, no. 13, 2005, pp. 409-454.

6. Sander G. A fast heuristic for hierarchical Manhattan layout. *Graph Drawing*, vol. 1027 of the series Lecture Notes in Computer Science, 2005, pp. 447-458.

7. Sander G. Layout of directed hypergraphs with orthogonal hyperedges. *Graph Drawing*, vol. 2912 of the series Lecture Notes in Computer Science, 2003, pp. 381-386.

8. Schulze C. D. Optimizing Automatic Layout for Data Flow Diagrams. *Diploma Thesis*, Christian-Albrechts-Universität zu Kiel, 2011.

9. Schulze C.D., Spönemann M., R. von Hanxleden. Drawing Layered Graphs with Port Constraints. *Journal of Visual Languages & Computing*, 2013, pp. 1–34.

10. Siebenhaller M. Orthogonal Graph Drawing with Constraints: Algorithms and Applications. *PHD Thesis*, Eberhard-Karls-Universitat, 2009.

11. Sugiyama K., Tagawa S., Toda M. Methods for visual undertanding of hierarchical systems. *IEEE Transactions on*

Systems, Man, and Cybernatics, vol. SMC-11, no. 2, 1981, pp. 109-125.

12. Taylor M., Rodgers P. Applying Graphical Design Techniques to Graph Visualisation. *Proceedings of the Ninth International Conference on Information Visualisation*, 2005, pp. 651-656.

13. Wolfram|One. Available at: http://www.wolfram.com (accessed: 21.01.17).

14. Vasiliev Yu.M. Exact mathematical formulation for hyperedge routing in coopertion scheme. *Sovremennaia ekonomika: problemy i resheniia*, 2017, no. 1. (In Russ).

15. Vasiliev Yu.M., Fridman G.M. Cooperation scheme visualization: a new hybrid algorithm for edge crossing minimization in graph drawing problem. *Izvestiya UNECON*, 2017, no. 1. (In Russ.).

16. Minoux M. *Mathematical programming. Theory and algorithms*. Moscow, Nauka, 1990. (In Russ.)