Московский Государственный Университет
им. М.В. Ломоносова
Российская Академия Наук
Академия Технологических Наук России
Российская Академия Естественных Наук

Интеллектуальные Системы.

Теория и приложения

ТОМ **21** ВЫПУСК **4** * **2017**

МОСКВА

ISSN 2411–4448

УДК 519.95; 007:159.955
ББК 32.81

Издается с 1996 г.*

В журнале «Интеллектуальные системы. Теория и приложения» публикуются научные достижения в области теории и приложений интеллектуальных систем, новых информационных технологий и компьютерных наук.

Издание журнала осуществляется под эгидой МГУ им. М. В. Ломоносова, Научного Совета по комплексной проблеме «Кибернетика» РАН, Отделения «Математическое моделирование технологических процессов» АТН РФ, Секции «Информатики и кибернетики» РАЕН.

Учредитель журнала: ООО «Интеллектуальные системы».

Журнал входит в список изданий, включенных ВАК РФ в реестр публикаций материалов по кандидатским и докторским диссертациям по математике и механике.

Спонсором издания является:

ООО «Два Облака»
Разработка корпоративных информационных систем
http://www.dvaoblaka.ru

Индекс подписки на журнал: 64559 в каталоге НТИ «Роспечать».

Адрес редакции: 119899, Россия, Москва, Воробьевы Горы, МГУ, ГЗ, механико-математический факультет, комн. 12-01.
Адрес издателя: 115230, Россия, Москва, Хлебозаводский проезд, д. 7, стр. 9, офис 9. Тел. +7 (495) 939-46-37, e-mail: mail@intsysjournal.org

*) Прежнее название журнала: «Интеллектуальные системы».

ОГЛАВЛЕНИЕ

Часть 1.
Общие проблемы теории интеллектуальных систем

Формальная постановка Тезиса М

Куриленко Н.В.

Рассматривается Тезис М и освещаются основные проблемы с формальным доказательством корректности или ложности данного тезиса. Предложена математическая постановка вопроса и рассмотрены простейшие свойства введённой модели. Описан пример одномерной непрерывной однородной стриктуры, в которой может быть реализована машина с оракулом.

Ключевые слова: тезис Тьюринга-Чёрча, Тезис М, клеточные автоматы, непрерывные однородные структуры.

1. Введение

Каковы пределы возможностей вычислительных машин, и какие задачи мы можем решить с их помощью? Поиски ответов на эти вопросы десятилетиями влияли на направление научно-технического прогресса, начиная с изобретения простейших механических машин и заканчивая производством интегральных схем и устройством современных квантовых компьютеров. За почти столетнюю историю существования этого вопроса было получено много результатов, направленных на понимание этих пределов. К примеру, для роста производительности вычислительных систем были получены такие ограничения как закон Мура [1], асимптотическая сложность схем функциональных элементов [2] и свойства классов вычислительной сложности [3]. В данной статье мы рассмотрим первое и самое фундаментальное ограничение, устанавливающее границы для вычислительных устройств, – введённый в 1936 году тезис Тьюринга-Чёрча[4], [5]. В сильно упрощённой форме тезис гласит:

What is effectively calculable is computable (англ.).

Это утверждение посвящено тому, на какие умственные достижения способен человек, вооружённый только ручкой и бумагой («with pen and paper») и внимательно следующий заранее оговорённым инструкциям.

Очевидно, тезис не ставит фундаментальные ограничения на возможности человека, поскольку ручка и бумага – лишь простейшие механические инструменты, не претендующие на использование полного потенциала известных физических законов. Дабы обобщить тезис на произвольные инструменты, которые способен сконструировать человек, Робин Ганди[6] в 1980 году ввёл понятие «Тезиса М». Обобщение формулируется следующим образом:

What can be calculated by a machine is computable (англ.).

Оно посвящено фундаментальным возможностям человека, которые ограничены только физическими законами нашей вселенной.

Стоит сразу оговориться, что вероятнее всего данное утверждение нельзя ни доказать, ни опровергнуть. Если утверждение корректно, это будет означать, что человеку не под силу сконструировать прибор, способный выполнять вычисления, невоспроизводимые с помощью механических устройств. Полное доказательство корректности подразумевает точное понимания принципов работы физических законов вселенной (то есть достижения, так называемой окончательной теории[7]), что в свою очередь является недостижимой задачей: даже если физическая модель реальности, корректно объясняющая все когда-либо проведённые эксперименты, будет найдена, она всё равно не будет считаться окончательной – мы всё равно не сможем быть уверены, что провели все возможные эксперименты, и что поняли природу реальности как с бесконечно малой погрешностью, так и в бесконечно большом масштабе. Если же данное утверждение ложно, значит такой прибор может быть построен (хотя бы теоретически), однако достоверно убедиться в правильности его работы мы не сможем. Причиной этого является тот факт, что подобное устройство хотя бы в какой-то его части будет недискретно (иначе его работа может быть симулирована на компьютере), а поскольку сознание людей дискретно (подробнее это утверждение будет рассмотрено в §3), мы не сможем верифицировать исправную работу прибора, даже при помощи специальных устройств. Из-за этого в научной литературе вычисления за пределами класса вычислимых по тьюрингу функций часто именуют мифом [8].

Тем не менее, несмотря на недоказуемость данных утверждений, мы всё ещё можем рассмотреть вопрос о том, возможны ли в принципе сверхтьюринговые вычисления при тех или иных физических законах (пусть даже мы и не сможем сконструировать соответствующий при-

бор). В данной статье мы рассмотрим математическую постановку этой задачи и разберём несколько примеров её решения.

2. Математическая модель

Как будет показано в §3, природа тезиса носит метафизический характер, и поэтому при анализе подобного утверждения проще определить близкую к нему формальную математическую задачу. Прежде всего, необходимо выбрать математическую модель, подходящую для описания физической среды и процесса вычислений в ней. Если мы возьмём все когда-либо серьёзно рассматриваемые физические модели, то обнаружим, что большинство из них обладает общими свойствами, такими как:

- конечное число измерений пространства и одно выделенное измерение времени
- непрерывность пространства и времени
- непрерывность или дискретность значений в каждой из точек пространства
- однородность физических законов

Такими свойствами, к примеру, обладает классическая и квантовая теория поля и общая теория относительности. Математическую модель, описывающую среды с подобными свойствами, называют моделью однородных структур. Нас будут интересовать её частные случаи, такие как клеточные автоматы и непрерывные однородные структуры, и условия, при которых внутри данных структур могут быть произведены сверхтьюринговые вычисления. Очевидно, что в дискретной однородной структуре невозможны сверхтьюринговые вычисления, поскольку любая такая среда моделируется с помощью тьюринговой машины. Но как только мы добавляем непрерывность в какой-либо части модели, вопрос тут же усложняется. Чтобы проиллюстрировать это на примере, рассмотрим непрерывную однородную структуру, в которой такие вычисления возможны.

2.1. Пример гипервычислений в непрерывной среде

Приведём теоретический аналог тезиса М для более простой среды. Для этого будем использовать модель однородных структур, посколь-

ку данная модель хорошо изучена и изначально была создана именно как общая упрощенная модель физических сред [9]. В частности, будем рассматривать дискретную среду с непрерывными значениями в каждой точке. Модель носит название continuous cellular automata (англ.) и наглядно рассмотрена в книге Стивена Вольфрама «A new kind of science»[10].

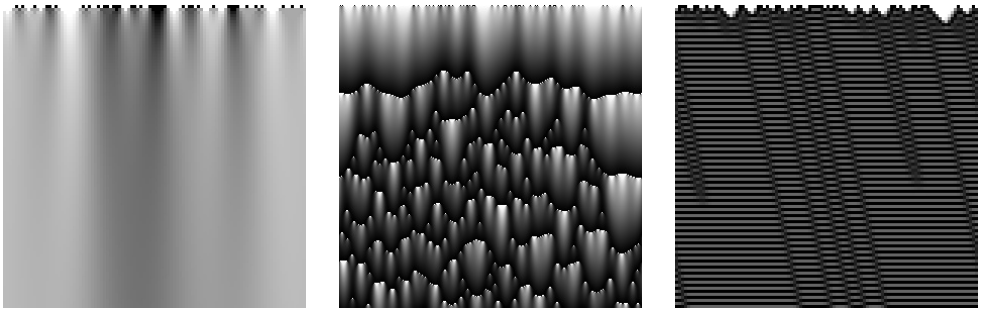
Одномерной непрерывной однородной структурой с окрестностью величины 2 называют счётное множество $\{c_i, i \in \mathbb{Z}\}$ элементов обладающее следующими свойствами:

1) В момент времени t_0 каждому элементу соответствует действительное число $c_i(t_0)$ (состояние элемента), заданное на отрезке $\mathbb{R}[0,1]$

2) В момент времени $t_0 + n$, $(n \in \mathbb{N})$ состояние элемента $c_i(t_0 + n)$ определено по правилу $c_i(t_0+n) = f(c_i(t_0+n-1), c_{i-1}(t_0+n-1))$, где $f : [0,1] \times [0,1] \to [0,1]$ – функция, фиксированная для данной структуры

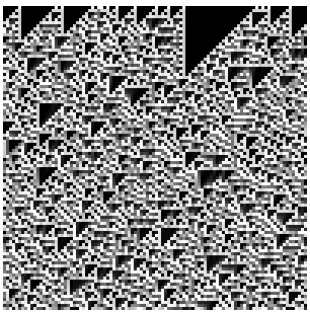
Элементы однородных структур часто называют клетками, а функции f - эволюционными правилами или законами изменения.

Поскольку f двуместна, её легко можно визуализировать в виде графика или тепловой карты (рис. 2b), а «участок» структуры за произвольный период [t, t + n] удобно визуализировать в виде окрашенного поля (рис. 2a). Примеры непрерывных однородных структур с окрестностью 3 изображены на рис. 1. Пример непрерывной однородной структуры с окрестностью 2 изображен на рисунке 2.

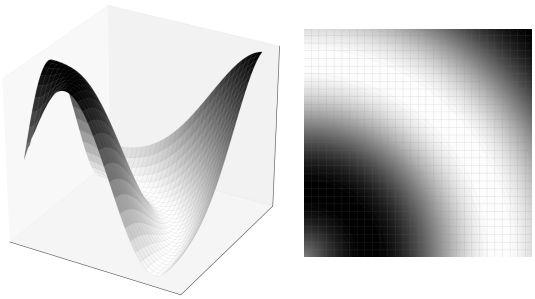


1a. $(x + y + z)/3$ 1b. $\{(x + y + z)/3 + .01\}$ 1c. $\sin(x + y + z/2)$

Рис. 1. Примеры структур с трёхместными функциями изменения.



2a. Поведение



2b. График функции изменения

Рис. 2. Структура с функцией изменения $f(x,y) \equiv \frac{1+\sin(5.5\|(x,y)\|_2)}{2}$.

В зависимости от шаблона соседства (окрестности) и законов изменения однородные структуры могут быть использованы для проведения тех же вычислений, на которые способна универсальная тьюринговая машина. Это утверждение очевидно, потому что саму универсальную машину тьюринга можно рассматривать как динамическую дискретную однородную структуру. Также очевидно, что и непрерывные однородные структуры могут быть полны по тьюрингу: для этого достаточно доопределить произвольным образом функцию изменения дискретной однородной структуры до функции изменения непрерывной однородной структуры. Построим пример структуры, обладающей свойством полноты по тьюрингу:

Рассмотрим дискретную однородную структуру, функция изменения которой выглядит следующим образом:

$$\begin{array}{llll} f(0,0,0)=0, & f(0,0,1)=1, & f(0,1,0)=1, & f(0,1,1)=1, \\ f(1,0,0)=0, & f(1,0,1)=1, & f(1,1,0)=1, & f(1,1,1)=0. \end{array}$$

Эта структура известна под именем Rule 110. В 2000 году Мэтью Кук показал, что данная однородная структура способна симулировать внутри себя машину тьюринга [11]. Для начала мы хотим, чтобы дискретная однородная структура с окрестностью 2 симулировала Rule 110. Для этого нам надо подобрать 4 дополнительных состояния (помимо 0 и 1) – a, b,

c и d:

$$\begin{array}{llll} f(0,0)=a, & f(0,1)=b, & f(1,0)=c, & f(1,1)=d, \\ f(a,a)=0, & f(a,b)=1, & f(b,c)=1, & f(b,d)=1, \\ f(c,a)=0, & f(c,b)=1, & f(d,c)=1, & f(d,d)=0. \end{array} \quad (1)$$

	0	a	b	c	d	1
0	a	0	0	0	0	c
a	0	0	0	0	0	0
b	0	1	0	1	0	0
c	0	0	1	0	1	0
d	0	0	1	0	0	0
1	b	0	0	0	0	d

Таблица 1.

Легко показать, что обе структуры полны по тьюрингу, а значит эквиваленты.

Теперь рассмотрим функцию, изображённую на рис. 3. Данная функ-

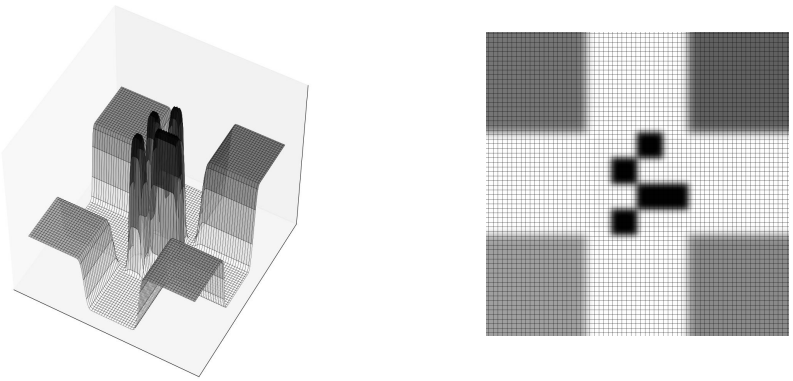


Рис. 3. Аналог Rule 110 в непрерывной среде.

ция изменения непрерывной среды построена на основе формул (1) при $a = \frac{9}{24}$, $b = \frac{11}{24}$, $c = \frac{13}{24}$, $d = \frac{15}{24}$. Предположим мы захотим использовать эту среду для проведения вычислений. В §3 будет показано, что нам стоит придерживаться сценария, в котором восприятие человека, также как и любые его действия в физическом смысле обладают некоторой неточностью. То есть мы можем записывать и считывать значения только с некоторой погрешностью ϵ (погрешностью в данном случае является

диаметр окрестности, в которую всегда попадает точка при записи или считывании). Если ϵ меньше, к примеру, 1/8, то из рисунка видно, что мы можем устанавлить в произвольной ячейке состояние 0 или 1, и лишь с некоторой вероятностью – состояния a, b, c и d. Поскольку (как показано на рис. 3) состояний 0 и 1 достаточно, чтобы симулировать Rule 110 в данной среде, это означает, что устройству записи и считывания, работающему с погрешностью $1/8$ будет доступен весь класс вычислимых по тьюрингу функций.

Далее мы хотим, используя схожий приём, выйти за класс тьюринговых вычислений. Это будет означать возможность симуляции хотя бы одной невычислимой функции, к примеру, фукнции, связанной с известной проблемы остановки [4]. Проблема остановки сводится к вычислению функции из $\mathbb{N}$ в $\{0,1\}$, поведение которой формируется с помощью понятия универсальной тьринговой машины [12]. Обозначим функцию через H. В сущности, мы хотим симулировать функцию H и Rule 110 в одной и той же среде. Как и в примере ранее, введём дополнительные состояния $\{a_i, i \in N\}$ и соответствующие правила вида $f(b_i, b_j) = b_k$ для $b_i, b_j, b_k \in \{0,1,2,3\} \cup \{a_i : i \in N\}$. Будем использовать следующие формулы:

$$
\begin{array}{cccc}
 & f(0,0)=a_1 & f(0,1)=a_2 & f(1,0)=a_3 & f(1,1)=a_4 \\
1) & f(a_1,a_1)=0 & f(a_1,a_2)=1 & f(a_2,a_3)=1 & f(a_2,a_4)=1 \\
 & f(a_3,a_1)=0 & f(a_3,a_2)=1 & f(a_4,a_3)=1 & f(a_4,a_4)=0
\end{array}
$$

2) для слова на входе из $\{2,3\}^*$ на выходе получаем один из двух специальных символом a' и a'', соответствующих остановке или бесконечной работе машины

3) $\forall a \in \{0,1\} : f(a,a') = f(a',a) = 0, f(a,a'') = f(a'',a) = 1$

4) $\forall a \in \{0,1\}, b \in \{2,3\} : f(a,b) = f(b,a) = a$

Здесь под "словом на входе"подразумевается записанное в некоторой области среды слово, а под "словом на выходе"подразумеваем значение в ячейке $c_i(t+n)$, если считать, что слово было записано в ячейках с $_i(t)$ по $c_{i+n}(t)$. Полученный закон изменения среды можно записать в матричном виде по аналогии с таблицей 1 (при этом в отличие от таблицы 1 новая таблица будет бесконечно большой):

	0	1	2	3	a_1	a_2	...
0	a_1	a_3	0	0	0	0	...
1	a_2	a_4	1	1	0	0	...
2	0	1	a_4	a_6	0	0	...
3	0	1	a_5	a_7	0	0	...
a_1	0	0	0	0	0	0	...
a_2	0	0	0	0	1	0	...
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$

Для того, чтобы превратить таблицу в график непрерывной функции $f : [0,1] \times [0,1] \to [0,1]$, необходимо, так же как и на рис. 3, сделать ширину столбцов и строк, соответствующую основным символам – 1, 2, 3, 4 – достаточно большой ($\geq 2\epsilon$), а ширину всех остальных строк и столбцов – достаточно маленькой, чтобы таблица уместилась в таблицу в отрезок $[0,1]$.

Легко заметить, что правил (1), (2), (3) и (4) достаточно, чтобы симулировать в полученнной непрерывной среде Rule 110 (а значит и любое механическое устройство) и работу функции H: с помощью символов 2 и 3 мы записываем на ленту слова, которые хотим направить в функцию H, а с помощью символов 0 и 1 обрабатываем полученные результаты. Аналогиченым образом манипулируя функцией изменения среды, мы можем, взяв реализацию тьюринговой машины в структуре, превратить её в машину с оракулом.

2.2. Формальная постановка вопроса

В общем виде задачу проверки возможности проведения гипервычислений внутри той или иной однородной структуры можно сформулировать в виде поиска значений двуместной функции $F(.,.)$, которая принимает на вход параметры среды и закон её изменения и на выходе выдаёт наличие или отсутствие сверхтьюринговых вычислений. Рассмотрим более подробно, что представляют из себя аргументы F.

Первый аргумент отводится на параметры однородной структуры, такие как дискретность или непрерывность пространства и времени. Как было отмечено в начале раздела, мы придерживаемся только физических сред, обладающих фиксированным набором свойств. Ещё раз перечислим эти свойства:

- непрерывность и конечномерность пространства
- непрерывность и однонаправленность времени

- непрерывность и конечномерность значений в каждой точке

Кроме того, мы рассматриваем среды со стандартной топологией действивильной оси $\mathbb{R}$ в неперывном случае или топологией $\mathbb{Z}$ – в дискретном случае. В результате, описание типов сред, которые мы можем рассматривать всего лишь 5 кардинальных чисел:

- ω_1 и ω_2 – количество и размерность элентов пространства
- ω_3 – количество моментов времени
- ω_4 и ω_5 – количество и размерность значений в каждой точке

Отметим также, что все структуры, которые мы можем рассматривать, обладают бесконечным пространством и временем, иначе даже тьюринговые вычисления не будут возможны. Таким образом, ω_1 и ω_3 принадлежат $\{\aleph_0, \aleph_1\}$, ω_2 и ω_5 конечны, ω_4 конечно или принадлежит $\{\aleph_0, \aleph_1\}$. Второй аргумент функции F являет собой произвольную функцию изменения среды (вместе с механизмом действия фукнции в окрестности каждой точки), определённой с помощью ω_1, ω_2, ω_3, ω_4 и ω_5. В итоге функция принимает на вход 5 кардинальных числе и функцию f построенную в соответствие с данными числами.

Дальнейшей работой над этой задачей является выделение различных свойств функции F (данная работа выходит за рамки статьи). К примеру,

$$F(\aleph_0, 1, \aleph_0, 2, 1, f') = 1,$$

где f' – функция изменения для Rule 110. В §2.1 было показано, что

$$\exists f : F(\aleph_0, 1, \aleph_0, \aleph_1, 1, f) = 1.$$

Также легко показывается, что

$$\forall f, \omega', \omega'', \omega''', \omega'''' : F(\omega', \omega'', \aleph_0, \omega''', \omega'''', f) = F(\omega', \omega'', \aleph_1, \omega''', \omega'''', f).$$

3. Недискретные вычислительные машины

До этого момента мы рассматривали достижение сверхтьюринговых вычислений только через некоторое гипотетическое устройство, имеющее дискретный вход и выход и использующее недискретность среды для вычислений, не доступных тьюринговым машинам. Мы сознательно ввели

понятие погрешности вычислений, чтобы избежать моделей с недискретным входом и/или выходом. В этой главе мы попытаемся ответить на вопрос, почему данное понятие столь необходимо для рассматриваемой нами задачи.

Устройства с действительным входом или выходом является предметом споров в научном сообществе. Причина заключается в том, что при попытке спроектировать подобное устройство мы рано или поздно сталкиваемся с вопросом о том, (1) как устроены физические законы нашего мира, и (2) как работает человеческое сознания, и как уже было замечено в §1, абсолютно точный ответ на вопрос (1) получить невозможно. В результате, все рассуждения по данной тематике спекулятивный характер.

Покажем, каким образом понятие погрешности связано с вопросами (1) и (2).

Пускай мы создали устройство с действительным выходом, которое выдало результат – некоторое действительное число. Теперь нужно считать выход и узнать, какое же это число. Но каким образом в принципе возможно считать действительное число? По определению это число занимает счётный объём бит информации, и не может быть сохранено на переносной носитель. По той же причине (а также потому что человеческая жизнь конечна) это число не может быть проиграно в виде аудио или видео записи. Рассуждая дальше, заметим, что человеческие органы восприятия неточны по своей природе, и не смогли бы пропустить через себя это число за конечное время без искажений. Единственным выходом из данной ситуации является использование самого сознания в качестве устройства для гипервычислений и надеяться, что сознание (через интуицию или каким-либо другим пока непознанным образом) сможет воспринять это число. Существуют работы, которых, к примеру, приведены аргументы в пользу того, что люди при общении производят сверхтьюринговые вычисления [13], однако, несмотря на наличие подобных статей, самой разумной на сегодняшний день точкой зрения является дискретность сознания. Пока в вопросе о природе сознания не будет достигнут какой-либо сущесвтенный прогресс, существование устройства с дейсвтиельным выходом не может быть доказано в рамках формальной теории. Дополнительные аргументы в пользу описанного выше параграфа можно найти, к примеру, в [8] и [14].

Пусть мы создали устройство с действительным входом, и нам необходимо записать ан входе какое-либо действительное число. Каким образом мы может этого достичь? Вероятно, последовательности действий,

приводящей к такому результату, попросту не существует. Прежде всего человек не может это совершить без специального устройства, поскольку, очевидно, человеческий организм не способен на бесконечно точные действия. Значит, нужно построить устройство, действующее с бесконечной точностью. Здесь мы сталкиваемся с фундаментальным утверждением, согласно которому неточные действия приводят к неточному результату. В отличие от рассуждений о природе сознания при известных физических законах данное утверждение является формально доказуемым. Это свойство физических систем хорошо изучено в теории хаоса; здесь мы приведём идею доказательства, основываясь на задаче трёх тел.

В задаче трёх тел (в одной из её форм) мы рассматриваем динамическую систему из трёх точек, меняющуюся по правилу:

$$\begin{aligned}
\ddot{r_1} &= -Gm_2\frac{r_1-r_2}{\|r_1-r_2\|^3} - Gm_3\frac{r_1-r_3}{\|r_1-r_3\|^3} \\
\ddot{r_2} &= -Gm_3\frac{r_2-r_3}{\|r_2-r_3\|^3} - Gm_1\frac{r_2-r_1}{\|r_2-r_1\|^3} \\
\ddot{r_3} &= -Gm_1\frac{r_3-r_1}{\|r_3-r_1\|^3} - Gm_2\frac{r_3-r_2}{\|r_3-r_2\|^3}
\end{aligned}$$

где r_i – радиус-векторы в двумерной системе координат, m_i – массы тел, G – гравитационная постоянная. Предположим, мы можем устанавливать систему в произвольное начальное положение с некоторой точностью. В результате неточности мы не знаем, в каком конкретно положении будут находиться точки после установки, но тем не менее хотим, чтобы спустя некоторое фиксированное время две (к примеру) из трёх точек всегда (несмотря на неточность установки в начальное положение)описывали одну и ту же траекторию. Сразу становится ясно, что траектории двух точек в каждом отдельном случае всё равно будут разные, из-за стороннего влияния третьей точки, поэтому стоит рассматривать задачу не для двух, а сразу для всех точек. Отбрасывая тривиальные случаи, когда, к примеру, все три точки сталкиваются, мы понимаем, что обеспечить такую гарантию не представляется возможным: состояние системы описывается 18 числами (координатами, скоростью и ускорением по оси OX и OY для каждого из трёх тел), а функция изменения этих чисел вне области частных случаев является бесконечно гладкой. То есть существуют два состояния системы, которые переходят в одно и тоже состояние спустя некоторое время, то разность этих состояний будет вектором из нулей, изменяющимся с помощью функции, являющейся тождественным нулём. Это означает, что если система при любом из двух состояний спустя какое-либо время переходит в одно и то же состояние, то два начальных состояния также являются одинаковыми состояниями.

Большинство физических теорий, описывающих нашу реальность, как и в задаче трёх тел, имеют дело только с бесконечно гладкими функциями, и поэтому утверждение о том, что неточные действия приводят к неточному результату, можно распространить и на них. Однако до тех пор, пока нам не известны в точности все физические законы вселенной, это утверждение остаётся всё равно недоказанной гипотезой. Отметим также, что в дискретных однородных структура утверждение неверно: к примеру, в среде Game of Life [15] существуют различные конфигурации, которые спустя несколько итераций становятся неотличимыми.

Таким образом, пока не описаны физические законы вселенной и природа сознания, возможность конструирования устройств, более точных нежели дискретные, фундаментально недосказываема, и, более того, согласно текущему пониманию этих вопросов, это скорее всего невозможно.

Список литературы

[1] Moore G., Cramming more components onto integrated circuits // Electronics — 1965. — Vol. 38, No. 8. — P. 114–117.

[2] Лупанов О. Б. О синтезе некоторых классов управляющих систем // Проблемы кибернетики, М., Физматгиз — 1963. — Вып. 10. — С. 63–97.

[3] Sanjeev A., Boaz B, Computational Complexity: A Modern Approach. — Cambridge University, 2009.

[4] Turing A., On computable numbers with an application to the Entscheidungsproblem //Proceedings of the London Mathematic Society — 1937. — Series 2, Vol. 42. — P. 230–265.

[5] Church A., An Unsolvable Problem of Elementary Number Theory //American Journal of Mathematics — 1936. — Vol. 58. — P. 345–363.

[6] Gandy R., Church's Thesis and Principles for Mechanisms //The Kleene Symposium — North-Holland Publishing Company, Amsterdam, 1980. — P. 123–148.

[7] Weinberg S., Dreams of a Final Theory. —New York: Pantheon Books, 1992.

[8] Davis M., The myth of hypercomputation //Alan Turing: Life and Legacy of a Great Thinker — Springer, Berlin, 2004. — P. 195-–212.

[9] Toffoli T., Margolus N., Cellular Automata Machines: A New Environment for Modeling — MIT Press, Cambridge, Massachusetts, 1987.

[10] Wolfram S., A New Kind of Science — Wolfram Media Inc., Champaign, 2002.

[11] Cook M., Universality in Elementary Cellular Automata // Complex Systems — 2004. — Vol. 15 — P. 1–40.

[12] Davis M., The Universal Computer: The Road from Leibniz to Turing — W.W. Norton, New York, London, 2000.

[13] Spivey M., Grosjean M., Knoblich G., Continuous attraction toward phonological competitors // Proceedings of the National Academy of Sciences — 2005. — Vol. 102 — P. 10393–10398.

[14] Davis M., Why there is no such discipline as hypercomputation // Applied Mathematics and Computation — 2006. — Vol. 178 — P. 4–7.

[15] Gardner M., The fantastic combinations of John Conway's new solitaire game "life" // Scientific American — 1970. — Vol. 223 — P. 120-–123.

Formal definition of Thesis M

Kurilenko N.V.

Thesis M is introduced and described in terms of its basic problems with confirming or disproving it. A mathematical formulation of the problem is proposed and the simplest properties of the model are discovered. An example of a one-dimensional continuous cellular stricture in which an oracle machine can be implemented is constructed.

Keywords: Church-Turing thesis, Thesis M, cellular automata, continuous cellular automata.

Рекомендательные системы: обзор основных постановок и результатов

Кутянин А.Р.

В работе описываются основные алгоритмы рекомендательных систем. Приводится содержательная постановка задачи нахождения N лучших рекомендаций, описываются преимущества и недостатки каждого из названных методов. Приводится обзор работ и результатов, начиная с возникновения первых рекомендательных систем и до настоящего времени.

Ключевые слова: Рекомендательная система, рекомендации, коллаборативная фильтрация, item-based, user-based, фильтрация на основе содержания, фильтрация на основе знаний, гибридные рекомендательные системы.

1. Введение

Мы живем в мире материального и информационного изобилия. Поэтому каждый день мы делаем выбор и задаемся следующими вопросами:

Какую камеру, книгу, футболку мне бы купить? Какое образование выбрать для своих детей? Куда поехать в отпуск? Какое кино посмотреть? И т.д.

Каждый из нас с легкостью может дополнить этот список. Во время принятия решения о покупке того или иного продукта, либо о том, куда сходить сегодня вечером, мы часто обращаемся за помощью: спрашиваем совета у друзей, ищем информацию в интернете, строим деревья решений, нанимаем команду экспертов, доверяемся инстинктам или просто следуем за толпой. При этом каждый сталкивался с ситуацией, когда советы друзей оказывались бесполезными, либо, просидев несколько часов в интернете, мы не могли найти нужную информацию. В подобных ситуациях нам трудно было заполучить нужный совет, либо на это требовалось много сил и времени.

Как результат, все мы, наверняка, задумывались о такой программе, которая могла бы давать нам индивидуальные рекомендации, способные помочь в принятие решений.

Создание алгоритма для такой программы является главной задачей области, которая занимается разработкой рекомендательных систем. А уже цель самой системы - это обеспечить большую группу пользователей доступной информацией и высококачественными рекомендациями.

В связи с повсеместным внедрением интернета и наличием большого объема статистических данных о предпочтениях пользователей ученые всего мира стали очень активно заниматься разработкой таких алгоритмов в последние двадцать-тридцать лет. А вследствие широкой применимости такие программы оказались очень действенными. Например, интернет-магазин Amazon.com делает рекомендации даже относительно покупки автомобильных двигателей.

Конечно, можно возразить, что такие системы могут быть полезны только тем, кто не может или не хочет позволить себе высококачественные советы экспертов в данной области. Но это верно лишь частично и только в некоторых областях, таких как финансы или медицина. В то же время в других сферах может и не быть возможности обратиться к квалифицированному эксперту, либо мнение других пользователей, воспользовавшихся данных продуктом или услугой, может больше влиять на наше решение. Таким образом, возникает вопрос, можем ли мы, имея большую базу данных о предпочтениях пользователей, разработать программу, которая будет давать более качественные рекомендации, чем человек.

Вопрос предоставления доступных, индивидуальных, а главное высококачественных рекомендаций является основным в области рекомендательных систем и ставит перед нами много интересных задач, как с технической точки зрения, так и с точки зрения психологии. С технической точки зрения мы пытаемся разработать алгоритм, который поможет наиболее точно и логично спрогнозировать пожелания пользователей на основе имеющихся данных. А психологические факторы мы должны учесть, когда будем продумывать то, каким образом программа будет взаимодействовать с конечным потребителем. Коммуникация между потребителем и системой должна быть настроена таким образом, чтобы потребитель доверял рекомендациям, которые она дает, а также своим решениям, принятым на основе этих рекомендаций. Также сложность заключается в том, что пользователи не всегда точно знают, что они хотят. Поэтому при создании рекомендательного алгоритма важно

учитывать то, каким образом программа будет собирать данные о потребностях пользователей, в каком виде представлять обратную связь, а также как все это будет влиять на решения пользователей.

Несмотря на все эти трудности, рекомендательные системы активно развиваются с момента их первого появления двадцать-тридцать лет назад. Область применения таких алгоритмов очень широкая. На сегодняшний день их можно встретить в областях занимающихся недвижимостью, финансами, электроникой, фильмами, книгами, музыкой новостями и т.д. Каждый день мы все чаще начинаем встречать образцы таких программ, например, на сайтах Amazon.com, Last.fm, Ozon.ru и других.По статистике на Amazon.com за год совершается около 5 млрд. транзакций, а около 30% из них идет от рекомендаций. Таким образом, приблизительно 125 млн. покупок в месяц совершается через рекомендации, а сколько их выдается системой и представить сложно.

В данной статье рассказывается об основных рекомендательных алгоритмах, а также приводятся примеры самых свежих разработок в данной области.

Целью данной работы является обзор основных рекомендательных алгоритмов, а именно рассказать о принципах работы методов коллаборативной фильтрации, о системах фильтрации на основе содержания, а также на основе знаний о пользователях, познакомить читателям с преимуществами и недостатками каждого из методов. Знакомство с ними наиболее важно, так как их чаще всего используют, и, по большому счету, все новые разработки являются усовершенствованием данных подходов

2. Постановка задачи

Формализуем нашу задачу. Пусть у нас есть множество пользователей $U = \{u_1, u_2, \ldots, u_n\}$, множество объектов $P = \{p_1, p_2, \ldots, p_m\}$ и матрица рейтингов $R = (r_{i,j})$ размера $n \times m$, где $i \in 1 \ldots n$, $j \in 1 \ldots m$. Возможными значениями рейтингов могут быть числами от 1 (совсем не понравилось) до 5 (очень понравилось), либо значениями понравилось/не понравилось, выраженные 0 и 1, а также другие. Если пользователь i никак не оценил элемент j, то на соответствующем месте $r_{i,j}$ будет стоять пустое значение.

Описанную матрицу рейтингов можно представить в виде таблице, показанной на рисунке. Обозначим через $\hat{r}_{i,j}$ наш прогноз относительно того, какую оценку пользователь i поставит продукту j. Наша задача наилучшим образом предсказать, какие оценки $r_{i,j}$ должны стоять на

Продукты	Продукт 1	Продукт 2	Продукт 3	Продукт 4	Продукт 5	Продукт 6
Пользователь 1	2	4	?	3	4	?
Пользователь 2	1	?	1	4	5	?
Пользователь 3	3	4	?	1	4	?
Пользователь 4	?	?	4	2	?	2
Пользователь 5	?	4	5	3	?	?

месте пропусков в матрице рейтингов, то есть рассчитать $\hat{r}_{i,j}$. Затем для каждого пользователя u на основе спрогнозированных оценок $\hat{r}_{i,j}$, нам нужно сформировать список из N продуктов, которые наиболее точно удовлетворяют предпочтениям пользователя, и, которые еще не были им оценены. Список этих N продуктов обозначим через N-мерный вектор $(p_{i_1}, p_{i_2}, \ldots, p_{i_N})$. Таким образом, математическая задача звучит так:

Дано:

$U = \{u_1, u_2, \ldots, u_n\}$ - множество пользователей,

$P = \{p_1, p_2, \ldots, p_m\}$ - множество продуктов,

$R = (r_{i,j})$ - матрица рейтингов размера $n \times m$, где на месте $r_{i,j}$ будет стоять некоторое число, если пользователь u_i оценил продукт p_j и пусто в противном случае.

N - требуемое число рекомендаций, которые хотим получить от системы.

Требуется найти:

Для данного пользователя u_i найти N-мерный вектор $(p_{i_1}, p_{i_2}, \ldots, p_{i_N})$, где продукты $p_{i_k}, k \in N$ еще не оценены этим пользователем, то есть в матрице рейтингов $R = (r_{i,j})$ стоит пусто на месте r_{i,i_k}, а также, чтобы эти продукты наиболее точно удовлетворяли предпочтениям пользователя, то есть прогнозные рейтинги $\hat{r}_{i,i_k}$ были наибольшими.

Алгоритмы, которые это делают, могут быть очень разными, и использовать различные входные данные. Одни из них формируют рекомендации, основываясь только на данных об известных рейтингах. Другие используют дополнительные характеристики продуктов, на основе рейтингов определяют, какие из этих характеристик наиболее точно удовлетворяют предпочтениям пользователя, а затем подбирают продукты с такими характеристиками.

А теперь более подробно поговорим о каждом из этих алгоритмов, проблемах, с которыми сталкивается исследователь в процессе их использования, преимущества и недостатках каждого метода.

3. Введение в основные рекомендательные алгоритмы

3.1. Системы коллаборативной фильтрации

Основная идея таких систем состоит в том, что если пользователи имели одинаковые интересы в прошлом, то в будущем их предпочтения также будут совпадать. В качестве примера рассмотрим книжный интернет-магазин. Пусть в прошлом истории покупок пользователей А и Б в данном магазине сильно пересекались. При появлении пользователя А на сайте мы хотим предложить ему новую книгу, которую он еще не читал, а пользователь Б, как раз, недавно приобрел книгу, которую А еще не видел. В такой ситуации будет разумно предложить пользователю А прочитать ее.

Выше описан принцип работы класса алгоритмов коллаборативной фильтрации, которые в англоязычной литературе называются user-based, то есть основанные на статистике о пользователях. Как видно из примера, мы сравниваем похожесть между пользователями, основываясь на рейтингах оцененных объектов. В то же время, почему бы нам не использовать ту же статистику для сравнения продуктов между собой, а потом сопоставить результаты двух подходов.

Такой метод, основанный на сравнение схожести объектов, называется item-based. Здесь основная идея состоит в том, что если пользователям, которые оценили два продукта, понравились оба, то пользователям, которые попробовали только один, можно предлагать второй, который вероятнее всего, им понравится. То есть, если в примере с книжным магазином мы заметили, что пользователи, которые покупали книгу А, также покупали книгу Б, то тем потребителям, которые уже купили А, но еще не обратили внимание на Б разумно будет ее предложить.

Первой исследовательской работой по рекомендательным системам считается работа Lee Giles "An autonomous Web Agent for Automatic Retrieval and Identification of Interesting Publications"1998 года. Однако первой печатной правильнее назвать работу 1992 года David Goldberg, David Nichols "Using collaborative filtering to weave an information Tapestry"

В данной работе описан принцип работы экспериментальной почтовой системы Taperstry. Разработчики Tapestry первыми использовали термин "коллаборативная фильтрация"как метод сбора качественных данных. Данная система была разработана в Xerox PARC как способ

обработки большого количества сообщений электронной почты и сообщений, отправляемых в группы новостей. Особенностью данной системы было то, что система собирала и анализировала данные о реакции людей на прочитанные ими документы, в следствие чего процесс фильтрации стал более эффективным.

Одновременно с Taperstry развивались и другие рекомендательные системы на основе коллаборативной фильтрации:

- В 1995-1996 годах были разработана сразу три системы для рекомендации музыки: Helpful Online Music Recommendations, Ringo, Firefly.
- Также в то время активно развивались системы рекомендаций наиболее интересных и популярных страниц в интернете: Point's Top 5%, PHOAKS (People Helping One Another Know Stuff), Webdoggie, Alexa Internet.

Метод Item-based был изобретен и использован Amazon.com в 1998 году. Впервые представлен публике на научной конференции в 2001, а его авторы в 2016 получили награду Test of Time.

Данный алгоритм помог справиться с некоторыми из проблем, имевшимися у методов, основанных на схожести пользователей:

- системы работали плохо, когда у них было много продуктов, но сравнительно немного оценок
- трудоемко вычислить сходства между всеми парами пользователей
- профили пользователей быстро менялись, и всю модель необходимо было пересчитывать

Но вопросов, с которыми приходится столкнуться разработчику в процессе создания такой системы остается еще много. Вот одни из них:

1) Как для данного пользователя, для которого мы хотим сделать рекомендацию, определить пользователей, которые имеют схожие предпочтения?
2) Как измерять схожесть между пользователями?
3) Что делать если у нас имеется мало данных о рейтингах?

Преимущества метода:

1) Является достаточно универсальным подходом, поэтому часто дает высокие результаты.

2) Для работы данного метода не нужна детальная информация о продуктах. В примере с книжным магазином - автор, жанр, описание книги. Вместо этого используется как история оценок самого пользователя, так и других пользователей.

Недостатки метода:

1) Как работать с новыми пользователями, для которых еще нет истории покупок (задача холодного старта).

2) Что делать с новыми объектами, которые еще никто не оценил.

3) Ресурсоемкость вычислений, которая замедляет время работы системы.

4) Необходим большой объем данных для высокой точности предсказаний. Ниже в процессе более детального рассмотрения этого типа алгоритмов я приведу ответы на все эти вопросы, а также расскажу, как обходить проблемы данного метода.

3.2. Системы фильтрации на основе содержания

Этот тип систем основан на наличие информации об описании и профиле, состоящем из набора характеристик элемента. Если снова рассмотреть пример книжного магазина, то в качестве характеристик можно взять жанр, тему или автора книги. Затем для каждого пользователя создается профиль путем присвоения характеристик сходных с характеристиками элементов, исходя из анализа его поведения в прошлом, либо явно спрашивая о его предпочтениях. Далее пользователю рекомендуются объекты, похожие на те, которые этот пользователь уже употребил, либо указал как предпочтительные. Похожести оцениваются по признакам содержимого объектов.

В примере с книжным магазином система могла определить, что автору нравятся детективы и новеллы определенных авторов, и в последствие рекомендовать книги этих жанров или авторов.

В процессе изучения систем фильтрации на основе содержания также возникают интересные вопросы:

1) Как система может автоматически создать профиль пользователя и затем улучшать в процессе обновления данных?

2) Как определить какой элемент соответствует предпочтениям пользователя?

3) Как автоматически извлекать информацию о продукте, чтобы избежать ручного заполнения?

Преимущества метода:

1) Не требует большой группы пользователей для достижения высокой точности рекомендаций.

2) Новые элементы можно рекомендовать сразу, как только у них появляются заполненные характеристики.

Недостатки метода:

1) Сильная зависимость от предметной области, полезность рекомендаций ограничена.

2) Профиль пользователей и элементов должен состоять из одинакового набора характеристик, чтобы их можно было сравнивать.

3.3. Системы, основанные на знаниях

Рекомендации, основанные на знаниях, используются обычно в таких областях, как электроника, где покупатели совершают покупки раз в пару лет, так как в данной области мы не можем положиться на историю покупок, которая используется в качестве входных данных для методов коллаборативной фильтрации и методов, основанных на содержании.

Рассмотрим для примера рекомендательную систему, которая помогает пользователя выбрать фотокамеру. Обычный пользователей покупает новую камеру только один раз в несколько лет. Таким образом, рекомендательная система не может построить профиль пользователя или предложить камеры, которые понравились другим пользователям, так как в противном случае предлагаться будут только бестселлеры. Поэтому алгоритмы, основанные на знаниях, обычно используют дополнительные данные, как о пользователях, так и о самих продукта, для формирования списка рекомендаций. В области фотокамер такая система

может использовать детальную информацию о характеристиках камер, таких как разрешение, вес, цена.

Просто представлять продукт, удовлетворяющий выбранному пользователем набору характеристик, является недостаточным, поскольку в таком случае каждый пользователь получает одинаковые рекомендации с теми, кто выбрал такой же набор характеристик. Таким образом, данные системы должны не просто собирать информацию о желаемых характеристиках, а также формировать некоторый профиль пользователя.

Поэтому важным аспектом построения таких систем является настройка взаимодействия между пользователем и системой. Если вспомнить пример с книжным магазином и алгоритмом коллаборативной фильтрации, то можно заметить, что пользователь может взаимодействовать с программой ограниченным числом способов. Множество приложений допускает только возможность ставить рейтинги от 1 до 5. Возвращаясь к примеру с фотокамерой, когда у нас нет информации об истории покупок пользователя, нам необходимо настроить диалог между пользователем и системой, в процессе которого программа задаст вопрос о требованиях покупателя, таких как максимальная цена, минимальное разрешение и т.д.

Такой подход требует не только детального технического понимания характеристик продукта, но также строит приблизительный сценарий на основе выбранных характеристик. В такой ситуации ограничивающие факторы могут быть использованы для описания контекста, в котором определенные характеристики являются релевантными для покупателя. Например, камера высокого разрешения является более предпочтительной, если пользователь планирует печатать фотографии большого размера.

В целом при рассмотрении систем такого вида возникает достаточно много вопросов:

1) В каких областях может быть применим данный метод?

2) Как получить профиль пользователя в областях, где нет истории его покупок, и как учесть предпочтения пользователя?

3) Как настроить взаимодействие с пользователями?

4) Каким образом можно персонифицировать процесс взаимодействия, чтобы максимизировать точность процесса сбора информации о предпочтениях пользователей?

Преимущества метода:

1) Требования пользователей могут быть определены точнее, благодаря явному взаимодействию.

2) Метод дает хорошие результат в сфере, где нет достаточной информации об истории покупок.

Недостатки метода:

1) От пользователя требуются дополнительные действия, чтобы система могла собрать данные о его предпочтениях.

2) Данные о требованиях пользователя могут быть неправильно интерпретированы системой.

3.4. Гибридные системы

Каждый из вышеописанных методов имеет свои преимущества и недостатки в зависимости от поставленной задачи. Достаточно очевидным решением всех этих проблем является объединение различных подходов для того, чтобы обеспечить большую точность рекомендаций. О том, как измерять точность рекомендаций, мы поговорим в разделе о способах оценки качества рекомендаций.

Если, например, у нас есть данные об описании продуктов, профиль пользователей и история его покупок, то мы можем улучшить рекомендательную систему путем объединения методов коллаборативной фильтрации и алгоритмов фильтрации по содержанию. Таким образом, в случае появления нового пользователя в системе, о котором нет истории покупок, мы сможем использовать рекомендации на основе алгоритмов фильтрации по содержанию, а в случае большого объема статистических данных строить более точный прогноз, используя методы коллаборативной фильтрации.

Несмотря на то, что гибридные системы помогают бороться с недостатками описанных ранее методов, они все же оставляют достаточно вопросов, на которые нужно ответить при проектировании такой системы:

1) Какие подходы могут быть объединены, и какие условия должны выполняться, чтобы это могло быть сделано?

2) Должны ли разные техники использоваться в разных ситуациях, либо результат каждой должен браться с определенным весом?

3) Как результат разных методов должны быть взвешены, чтобы на выходе получить один результат?

Список литературы

[1] Jannach D., Zanker M., Felfering A., Friedrich G., Recommender Systems: An Introduction.. — M.: Cambridge University Press, 2011.

[2] Ricci F., Rokach L., Shapira B., Kantor P. Recommender Systems: Handbook.. — M.: Springer, 2010.

[3] Bollacker K.D., Lawrence S., Giles C.L., CiteSeer: An autonomous web agent for automatic retrieval and identification of interesting publications // 2nd international conference on Autonomous agents — 1998. — C. 116-123.

[4] Goldberg D., Nichols D., Oki B.M., Terry D. Using collaborative filtering to wave an information tapestry // Communication of the ACM — 1992. — 35(12):61-70.

[5] Linden G., Smith B., York J., Amazon.com recommendations: item-to-item collaborative filtering // Internet Computing — IEEE 7 2003 — C. 76-80.

[6] Miller B., An open Architecture for collaborative filtering // GroupLens — 1995

[7] Gomez-Uribe C., Hunt N., The Netflix Recommender System: Algorithms, Business Value, and Innovation // ACM Transactions on Management Information Systems — 2015

[8] Melville P., Mooney R.J., Nagarajan R. Content-boosted collaborative filtering for improved recommendations // in Proceedings of the National Conference on Artificial Intelligence — 2002 — C. 187–192.

[9] Shi Y., Larson M., Hanjalic A., Collaborative Filtering Beyond the User-Item Matrix: A Survey of the State of the Art and Future Challenges // ACM Comput. Surv. — 2014 — C. 3:1–3:45.

[10] Gunawardana A., Shani G., A survey of accuracy evaluation metrics of recommendation tasks // The Journal of Machine Learning Research — 2009 — C. 2935–2962.

[11] Herlocker, J.L., Konstan J.A., Terveen L.G., Riedl J.T., Evaluating collaborative filtering recommender systems // ACM Transactions on Information Systems (TOIS) — 2004 — С. 5–53.

[12] Ge, M., Delgado-Battenfeld C., Jannach D., Beyond accuracy: evaluating recommender systems by coverage and serendipity // in Proceedings of the fourth ACM conference on Recommender systems — 2010 — С. 257–260.

[13] McNee, S., Kapoor N., Konstan J.A., Don't look stupid: avoiding pitfalls when recommending research papers // in Proceedings of the 2006 20th anniversary conference on Computer supported cooperative work — 2006 — С. 171–180.

[14] McNee, S., Albert I., Cosley D., On the Recommending of Citations for Research Papers // in Proceedings of the ACM Conference on Computer Supported Cooperative Work — 2002 — С. 116–125.

[15] Geyer-Schulz A., Hahshler M., Comparing two recommender algorithms with the help of recommendations by peers // in Proceedings of the WEBKDD 2002 - Mining Web Data for Discovering Usage Patterns and Profiles — 2003 — С. 137–158.

[16] Geyer-Schulz A., Hahshler M., Evaluation of recommender algorithms for an internet information broker based on simple association rules and on the repeatbuying theory // in Proceedings of the 4th WebKDD Workshop: Web Mining for Usage Patterns & User Profiles — 2002 — С. 100–114.

[17] Geyer-Schulz A., Hahshler M., Jahn M., A customer purchase incidence model applied to recommender services // in Proceedings of the Third International Workshop on Mining Web Log Data Across All Customers Touch Points — 2002 — С. 25–47.

[18] Geyer-Schulz A., Hahshler M., Jahn M., Educational and scientific recommender systems: Designing the information channels of the virtual university // International Journal of Engineering Education — 2001 — С. 153–163.

[19] Huang Z., Chung W., Ong T.H., Chen H., A graph-based recommender system for digital library // in Proceedings of the 2nd ACM/IEEE-CS joint conference on Digital libraries — 2002 — С. 65–73.

[20] Balabanovic M., An Adaptive Web Page Recommendation Service // in Proceedings of the First international Conference on Autonomous Agents — 1997

[21] Menczer F., ARACHNID:Adaptive retrieval agents choosing heuristic neighborhoods for information discovery // In Machine Learning: Proceedings of the fourteenth International Conference — 1997 — C. 227–235.

[22] Liang Y., Li Q., Qian T., Finding relevant papers based on citation relations // in Proceedings of the 12th international conference on Web-age information management Conference — 2011 — C. 403–414.

[23] Pennock D.M., Horvitz E., Lawrence S., Giles C.L., Collaborative filtering by personality diagnosis: A hybrid memory-and model-based approach // in Proceedings of the Sixteenth conference on Uncertainty in artificial intelligence — 2000 — C. 473–480.

Recommender systems: overview of main statements and results

Kutyanin A.R.

Main recommender algorithms are considered in this paper. A meaningful formulation of the problem of finding N best recommendations is given, pros and cons are described for all mentioned approaches. Also reader can find papers and results overview starting from appearance of the first recommendation systems up to the present.

Keywords: Recommender systems, recommendation, collaborative filtering, item-based, user-based, content-based filtering, knowledge-based filtering, hybrid recommender system.

Верификация программ методом инвариантов

Миронов А.М.

Излагается метод инвариантов для доказательства правильности компьютерных программ. Основные концепции, связанные с этим методом, иллюстрированы примерами верификации последовательных и параллельных программ.

Ключевые слова: верификация программ, метод Флойда, инварианты.

1. Введение

Проблемы верификации (т.е. доказательства правильности) программ занимают центральное положение в теории и практике разработки программного обеспечения. Под правильностью программ понимается их соответствие различным условиям корректности, безопасности, устойчивости в случае непредусмотренного поведения окружения, эффективности использования ресурсов времени и памяти, оптимальности реализованных в программе алгоритмов, и т.п.

Как правило, для обоснования правильности программы её тестируют, т.е анализируют её поведение на некоторых входных данных. Однако тестирование обладает очевидным недостатком: если его возможно провести не для всех допустимых входных данных, а только лишь для их небольшой части (что имеет место почти всегда), то оно не может служить гарантированным обоснованием того, что тестируемая программа обладает проверяемыми свойствами. Как отметил Дейкстра ([1], стр. 41), тестирование может лишь помочь выявить некоторые ошибки, но отнюдь не доказать их отсутствие. Ошибки в программах могут быть весьма тонкими, и чем тоньше ошибка, тем сложнее обнаружить её тестированием. Но во многих программах наличие даже незначительных ошибок категорически недопустимо. Например, наличие даже небольших ошибок в таких программах, как

- программы управления атомными электростанциями,
- программы, управляющие работой медицинских устройств,
- программы в бортовых системах управления самолетов и космических аппаратов,
- программы в системах управления секретными базами данных, системах электронной коммерции, и т.п.

может привести к существенному ущербу для экономики и жизни людей.

Гарантированное обоснование правильности программ может быть получено только при помощи альтернативного подхода, принципиально отличного от тестирования. Данный подход называется верификацией. В самом общем виде верификация программы может пониматься как построение математического доказательства утверждения о том, что верифицируемая программа соответствует своему предназначению. Предназначение программы может быть выражено, например, путем описания функции, которую должна вычислять эта программа, или правил взаимодействия этой программы с другими программами, т.е. реакции, которую эта программа должна обеспечивать в ответ на получение сигналов или сообщений от других программ.

Формальное описание предназначения программы (или некоторых свойств, которыми она должна обладать) в виде математического утверждения называется **спецификацией** этой программы. Спецификация может представлять собой формальное описание самых разнообразных свойств программы, например:

- её входные и выходные данные находятся в заданном соотношении,
- программа всегда завершает свою работу,
- во время работы программы не происходит сбоев и ненормальных ситуаций (например, деления на 0, извлечения квадратного корня из отрицательного числа, выхода индекса за границы массива, неавторизованных утечек информации, и т.п.),
- программа решает свою задачу за установленное время,
- программа использует не более установленного объема памяти.

Для верификации программы P необходимо определить

- математический смысл всех конструкций, используемых в P, называемый **формальной семантикой** (или просто **семантикой**) этих конструкций, и
- спецификацию $Spec$ этой программы, выражающую то свойство программы P, которое необходимо верифицировать,

после чего можно ставить вопрос о верификации P относительно $Spec$, т.е. о построении математического доказательства утверждения о том, что P удовлетворяет $Spec$.

В настоящем тексте излагается один из наиболее широко распространённых методов верификации программ, известный под названием **метод инвариантов**. Одними из первых работ, в которых изложен этот метод, являются статьи [3] и [4]. Метод инвариантов основан на понятиях математической логики, с которыми можно познакомиться по книге [5]. Различные изложения метода инвариантов содержатся в [6]–[11].

2. Программы, представленные в виде блок-схем

Одним из языков описания программ является язык блок-схем. На данном языке программа представляется в графовой форме. Отметим, что графовая форма представления программ в последнее время завоевывает все большую популярность, по причине того, что такая форма представления программ облегчает их понимание и упрощает их анализ.

2.1. Вспомогательные понятия

Мы будем предполагать, что заданы следующие множества.

- Множество $\mathcal{T}$, элементы которого называются **типами**. Каждому типу $t \in \mathcal{T}$ сопоставлено множество D_t **значений** типа t.
- Множество $\mathcal{X}$, элементы которого называются **переменными**. Каждой переменной $x \in \mathcal{X}$ сопоставлен тип $t(x) \in \mathcal{T}$. Каждая переменная $x \in \mathcal{X}$ может принимать **значения** в множестве $D_{t(x)}$, т.е. в различные моменты времени переменная x может быть связана с различными элементами множества $D_{t(x)}$.
- Множество $\mathcal{C}$, элементы которого называются **константами**. Каждой константе $c \in \mathcal{C}$ сопоставлены тип $t(c) \in \mathcal{T}$ и значение из $D_{t(c)}$, обозначаемое тем же символом c, и называемое **интерпретацией** константы c.

- Множество $\mathcal{F}$, элементы которого называются **функциональными символами (ФС)**. Каждому ФС $f \in \mathcal{F}$ сопоставлены
  - **функциональный тип** $t(f)$, который представляет собой запись вида $(t_1, \ldots, t_n) \to t$, где $t_1, \ldots, t_n, t \in \mathcal{T}$, и
  - функция вида $D_{t_1} \times \ldots \times D_{t_n} \to D_t$, где $(t_1, \ldots, t_n) \to t \ = t(f)$, данная функция обозначается тем же символом f и называется **интерпретацией** ФС f.

  Ниже мы будем использовать ФС $+, -, \cdot, div, mod$, где
  - ФС $+, -, \cdot$ имеют функциональный тип $(\texttt{int}, \texttt{int}) \to \texttt{int}$, и $\texttt{int}$ – тип, значениями которого являются целые числа,
  - ФС div, mod имеют функциональный тип $(\texttt{int}, \texttt{int}_0) \to \texttt{int}$, и $\texttt{int}_0$ – тип, значениями которого являются целые числа, отличные от 0,

  и функции $+$, $-$, и $\cdot$ представляют собой соответствующие арифметические операции, div и mod вычисляют частное и остаток соответственно от деления первого аргумента на второй.

Выражения строятся из переменных, констант и ФС. Множество всех выражений обозначается символом $\mathcal{E}$. Каждое выражение e имеет тип $t(e) \in \mathcal{T}$, определяемый структурой выражения e. Правила построения выражений имеют следующий вид:

- каждая переменная и константа является выражением того типа, который сопоставлен этой переменной или константе, и
- если $e_1, \ldots, e_n \in \mathcal{E}$, $f \in \mathcal{F}$, и $t(f)$ имеет вид $(t(e_1), \ldots, t(e_n)) \to t$, то знакосочетание $f(e_1, \ldots, e_n)$ является выражением типа t.

Выражения $+(e_1, e_2)$, $-(e_1, e_2)$, $\cdot(e_1, e_2)$, $div(e_1, e_2)$, $mod(e_1, e_2)$ будут записываться в более привычном виде $e_1 + e_2$, $e_1 - e_2$, $e_1 e_2$, e_1/e_2 и $e_1 \% e_2$ соответственно.

Среди типов, входящих в $\mathcal{T}$, имеется тип $\texttt{bool}$, множество значений которого имеет вид $\{0, 1\}$. Выражения типа $\texttt{bool}$ называются **формулами**. Множество всех формул обозначается символом $\mathcal{B}$. При построении формул могут использоваться обычные булевские ФС ($\wedge, \vee, \to$ и т.д.), которым соответствуют функции коньюнкции, дизъюнкции, и т.д. Символ 1 обозначает тождественно истинную формулу, а символ 0 – тождественно ложную формулу. Формулы вида $\wedge(e_1, e_2)$, $\vee(e_1, e_2)$, и т.п. мы будем

записывать в более привычном виде $e_1 \wedge e_2$, $e_1 \vee e_2$, и т.д. В некоторых случаях формулы вида $e_1 \wedge \ldots \wedge e_n$ будут записываться в виде $\left[\!\left[\begin{array}{l} e_1 \\ \ldots \\ e_n \end{array} \right]\!\right]$.

$\forall\, e \in \mathcal{E}$ запись X_e обозначает множество переменных, входящих в e.

$\forall\, X \subseteq \mathcal{X}$ записи $\mathcal{E}(X)$ и $\mathcal{B}(X)$ обозначают множества $\{e \in \mathcal{E} \mid X_e \subseteq X\}$ и $\mathcal{E}(X) \cap \mathcal{B}$ соответственно.

Пусть $X \subseteq \mathcal{X}$. **Означиванием** переменных из множества X называется произвольное соответствие ξ, которое сопоставляет каждой переменной $x \in X$ некоторое значение $x^\xi \in D_{t(x)}$. Множество всех означиваний переменных из X обозначается записью $X^\bullet$.

Если $X \subseteq \mathcal{X}$, $e \in \mathcal{E}(X)$, $\xi \in X^\bullet$, то e^ξ обозначает значение из $D_{t(e)}$, называемое **значением** e **на** ξ, и определяемое следующим образом:

- если $e \in \mathcal{X}$, то e^ξ предполагается заданным,
- если $e \in \mathcal{C}$, то e^ξ является интерпретацией константы e, и
- если $e = f(e_1, \ldots, e_n)$, то $e^\xi = f(e_1^\xi, \ldots, e_n^\xi)$.

Выражения $e_1, e_2 \in \mathcal{E}$ считаются равными, если

$$\forall\, \xi \in (X_{e_1} \cup X_{e_2})^\bullet \;\; e_1^\xi = e_2^\xi.$$

Пусть $X \subseteq \mathcal{X}$, $\xi \in X^\bullet$, $x \in X$ $e \in \mathcal{E}(X)$, $t(x) = t(e)$. Запись $\xi(x := e)$ обозначает означивание из $X^\bullet$, определяемое следующим образом:

$$x^{\xi(x:=e)} = e^\xi, \;\; \forall\, x \in X \setminus \{x\} \;\; x^{\xi(x:=e)} = x^\xi. \tag{1}$$

Если $x \in \mathcal{X}$ $e, e' \in \mathcal{E}$, и $t(x) = t(e)$, то запись $(x := e)e'$ обозначает выражение, получаемое из e' заменой всех вхождений x в e' на e.

2.2. Понятие блок-схемы

Блок-схема (БС) представляет собой конечный ориентированный граф, каждая вершина которого имеет один из следующих типов:

- **начальная** вершина, она обозначается символом $\bigodot$, в каждой БС имеется только одна начальная вершина, из неё выходит одно ребро, и в неё не входит ни одного ребра,
- **присваивание**, вершина такого типа обозначается записью вида $[x := e]$, где $x \in \mathcal{X}$, $e \in \mathcal{E}$, $t(x) = t(e)$, из каждого присваивания выходит только одно ребро,

- **условный переход**, вершина такого типа обозначается записью вида (b) где $b \in \mathcal{B}$, из каждого условного перехода выходит два ребра, одно из которых имеет метку 1, а другое - метку 0,

- **пустой оператор**, такая вершина обозначается символом $\bigcirc$, из неё выходит только одно ребро,

- **терминальная** вершина, она обозначается символом $\bigotimes$, из каждой терминальной вершины не выходит ни одного ребра.

Пусть P – БС. Будем обозначать записями V_P и X_P совокупность всех вершин P и переменных, входящих в P, соответственно.

Выполнение БС P – это обход вершин P, начиная с начальной вершины (т.е. последовательность переходов по рёбрам от одной вершины P к другой), с выполнением действий, сопоставленных проходимым вершинам. С каждым шагом $i \geq 0$ выполнения P связаны вершина $v_i \in V_P$ и означивание $\xi_i \in X_P^\bullet$ (называемые **текущей вершиной** и **текущим означиванием** на шаге i). Если v_i – нетерминальная вершина, то определен $i+1$-й шаг выполнения P, в P есть ребро из v_i в v_{i+1}, и

- если v_i – начальная вершина или пустой оператор, то $\xi_{i+1} = \xi_i$,

- если $v_i = [x := e]$, то $\xi_{i+1} = \xi_i(x := e)$,
  (можно интерпретировать действие, соответствующее этой вершине, как обновление значения переменной x: после исполнения этого действия значение x становится равным значению выражения e на текущих значениях переменных БС P)

- если $v_i = (b)$, то $\xi_{i+1} = \xi_i$, и ребро из v_i в v_{i+1} имеет метку b^{ξ_i},

а если v_i – терминальная вершина, то выполнение БС на шаге i завершается.

2.3. Задача верификации блок-схем

Пусть задана БС P, и её спецификация представляет собой пару формул $(Pre, Post)$ с переменными из X_P, имеющих следующий смысл:

- формула Pre называется **предусловием**, и выражает условие, которому должны удовлетворять значения переменных БС P в момент начала её выполнения, и

- формула $Post$ называется **постусловием**, и выражает условие, которому должны удовлетворять значения переменных БС P после завершения её выполнения.

Пусть P – БС, и $\xi \in X_P^\bullet$. Мы будем говорить, что P **выполняется с начальным означиванием** ξ, если в момент начала выполнения P значение каждой переменной $x \in X_P$ было равно x^ξ.

Запись $\xi \xrightarrow{P} \otimes$ обозначает утверждение: если P выполняется с начальным означиванием ξ, то выполнение P когда-либо завершится.

Если верно $\xi \xrightarrow{P} \otimes$, то запись ξP обозначает означивание из $X_P^\bullet$, определяемое следующим образом: пусть P выполняется с начальным означиванием ξ, тогда $\forall\, x \in X_P$ значение $x^{\xi P}$ равно тому значению, которое будет иметь переменная x после завершения выполнения P.

Задача **верификации** БС P относительно спецификации $(Pre, Post)$ заключается в доказательстве следующего утверждения:

$$\forall\, \xi \in X_P^\bullet \quad \text{если} \;\; Pre^\xi = 1, \;\; \text{то} \;\; \xi \xrightarrow{P} \otimes \;\; \text{и} \;\; Post^{\xi P} = 1. \qquad (2)$$

Данное утверждение обозначается записью $Pre \xrightarrow{P} Post$.

3. Метод инвариантов для верификации блок-схем

В этом параграфе излагается метод доказательства утверждений вида $Pre \xrightarrow{P} Post$, называемый **методом инвариантов**. Для его формулировки введём понятия базового множества и базового пути.

3.1. Базовые множества и базовые пути

Пусть задана БС P. **Путь** в P – это последовательность $\pi = \alpha_1 \dots \alpha_k$ рёбер P, такая, что $\forall\, i = 1, \dots, k-1$ конец ребра α_i совпадает с началом ребра α_{i+1}. Путь $\pi = \alpha_1 \dots \alpha_k$ называется **циклом** в P, если $\alpha_1 = \alpha_k$.

Множество M точек на некоторых ребрах БС P называется **базовым** для P, если каждый цикл в P содержит ребро, на котором имеется точка из M. Если M – базовое множество для P, то путь π в P называется **базовым** относительно M, если он непуст, на первом и последнем ребрах этого пути имеются точки из M, и на других рёбрах этого пути точек из M нет. Множество всех базовых относительно M путей в P обозначается записью $\Pi_{P,M}$.

Докажем, что множество $\Pi_{P,M}$ конечно. Если $\Pi_{P,M}$ бесконечно, то оно содержит пути сколь угодно большой длины. Тогда в $\Pi_{P,M}$ есть путь π вида $\alpha_1 \dots \alpha_i \dots \alpha_j \dots \alpha_k$, где $\alpha_i = \alpha_j$, $1 < i < j < k$. Последовательность $\alpha_i \dots \alpha_j$ является циклом, и, согласно определению базового множества, одно из рёбер этого цикла содержит точку из M, что противоречит предположению о том, что путь π – базовый относительно M.

$\forall\, \pi \in \Pi_{P,M}$ будем обозначать записями $start(\pi)$ и $end(\pi)$ точки из M, которые лежат на первом и последнем ребре π, соответственно.

Базовое множество M для БС P называется **полным**, если

- на ребре, выходящем из начальной вершины P, есть точка из M (такая точка называется **начальной**), и
- на каждом ребре, входящем в какую-либо терминальную вершину P, есть точка из M (такие точки называются **терминальными**).

Пусть M – полное базовое множество для БС P. Каждому выполнению $Exec$ БС P соответствует последовательность $\pi = \alpha_1\alpha_2 \dots$ рёбер P, в которой каждое ребро α_i является тем ребром, по которому происходит перемещение на шаге i этого выполнения от текущей вершины v_i к вершине v_{i+1}. Обозначим записью $i_1 i_2 \dots$ последовательность номеров тех рёбер из π, на которых есть базовые точки. Согласно определению понятий полного базового множества и выполнения БС, $i_1 = 1$, и для каждой пары (i_j, i_{j+1}) соседних индексов в $i_1 i_2 \dots$ последовательность $\pi_j = \alpha_{i_j} \dots \alpha_{i_{j+1}}$ является базовым относительно M путём. Если путь π конечен, то его последнее ребро входит в терминальную вершину, и, следовательно, содержит точку из M. Таким образом, можно представить π в виде последовательности $\pi_1\pi_2 \dots$, где π_1, π_2, $\dots$ – базовые относительно M пути. Каждый член π_j этой последовательности мы будем называть **компонентой** выполнения $Exec$.

$\forall\, \pi \in \Pi_{P,M}, \forall\, \xi, \xi' \in X_P^\bullet$ запись $\xi \xrightarrow{\pi} \xi'$ означает, что π – компонента некоторого выполнения БС P, и ξ, ξ' – текущие означивания в моменты прохода через точки $start(\pi)$ и $end(\pi)$ соответственно при движении по пути π во время этого выполнения.

3.2. Описание метода инвариантов для верификации блок-схем

Пусть заданы БС P и её спецификация, выражаемая предусловием Pre и постусловием $Post$. Доказательство утверждения $Pre \xrightarrow{P} Post$ **методом инвариантов** имеет следующий вид.

1) Выбирается полное базовое множество точек M для P, и с каждой точкой $m \in M$ связывается формула $\varphi_m \in \mathcal{B}$, называемая **инвариантом** в точке m, причем выполнены следующие условия:

   - если m – начальная точка, то $\varphi_m = Pre$,
   - если m – терминальная точка, то $\varphi_m = Post$,
   - для любого пути $\pi \in \Pi_{P,M}$, и любых означиваний $\xi, \xi' \in X_P^\bullet$, таких, что $\xi \xrightarrow{\pi} \xi'$, верна импликация

   $$\varphi^{\xi}_{start(\pi)} = 1 \quad \Rightarrow \quad \varphi^{\xi'}_{end(\pi)} = 1. \tag{3}$$

   Ниже, при анализе импликаций вида (3), $\forall\, x \in X_P$ будем обозначать значения x^ξ и $x^{\xi'}$ записями x и x' соответственно.

2) Свойство завершаемости P ($\forall\, \xi \in X_P^\bullet \;\; Pre^\xi = 1 \Rightarrow \xi \xrightarrow{P} \otimes$) обосновывается путем указания

   - базового множества N для P,
   - частично упорядоченного множества L, являющегося **фундированным**, т.е. такого, что не существует бесконечной строго убывающей последовательности $l_0, l_1, \ldots$ элементов L, и
   - множества выражений $\{u_n \in \mathcal{E}(X_P) \mid n \in N\}$,

   таких, что

   - при каждом выполнении P, $\forall\, n \in N$ при каждом проходе через n текущее означивание ξ удовлетворяет условию $u_n^\xi \in L$, и
   - для любого пути $\pi \in \Pi_{P,N}$, и любых означиваний $\xi, \xi' \in X_P^\bullet$, таких, что $\xi \xrightarrow{\pi} \xi'$, верно неравенство $u^{\xi}_{start(\pi)} > u^{\xi'}_{end(\pi)}$.

Изложенный в этом пункте метод инвариантов был открыт Р. Флойдом [2] и впервые был изложен в [3].

3.3. Обоснование метода инвариантов

Докажем, что если выполнены условия в пунктах 1 и 2 параграфа 3.2, то $Pre \xrightarrow{P} Post$, т.е. $\forall\, \xi \in X_P^\bullet$ из $Pre^\xi = 1$ следует $\xi \xrightarrow{P} \otimes$ и $Post^{\xi P} = 1$.

Пусть $Exec$ – произвольное выполнение БС P с начальным означиванием ξ, удовлетворящим условию $Pre^\xi = 1$. Этому выполнению соответствует последовательность $\pi = \alpha_1\alpha_2\ldots$ рёбер P, в которой каждое ребро α_i является тем ребром, по которому происходит перемещение на шаге i этого выполнения от текущей вершины v_i к вершине v_{i+1}.

Если бы выполнение $Exec$ было бесконечным, то π тоже была бы бесконечной, и некоторое ребро встречалось бы в ней бесконечно много раз. Пусть $i_1 i_2 \ldots$ – бесконечная последовательность номеров рёбер из π, таких, что $\alpha_{i_1} = \alpha_{i_2} = \ldots$. Подпоследовательности $\pi_{i_j} = \alpha_{i_j} \ldots \alpha_{i_{j+1}}$ $(j \geq 1)$ последовательности π являются циклами, и т.к. N – базовое множество, то $\forall\, j \geq 1 \quad \pi_{i_j}$ содержит ребро, на котором есть точка из N. Таким образом, точки из N присутствуют на бесконечном числе членов последовательности π. Обозначим номера таких членов записями $j_1, j_2, \ldots$, а точки из N на них – записями $n_1, n_2, \ldots$. Пути $\alpha_{j_k} \ldots \alpha_{j_{k+1}}$ $(k \geq 1)$ являются базовыми относительно N, поэтому, согласно пункту 2 параграфа 3.2, текущие означивания $\xi_{j_1}, \xi_{j_2}, \ldots$ удовлетворяют неравенствам

$$u_{n_1}^{\xi_{j_1}} > u_{n_2}^{\xi_{j_2}} > \ldots \tag{4}$$

т.е. в L есть бесконечная строго убывающая последовательность (4), что противоречит предположению о фундированности L.

Таким образом, выполнение $Exec$ является конечным. В соответствии со сказанным в конце пункта 3.1, можно представить π в виде конечной последовательности $\pi_1 \ldots \pi_k$ путей из $\Pi_{P,M}$.

Согласно определениям выполнения и полного базового множества, а также условиям в пункте 1 параграфа 3.2

- $m_0 = start(\pi_1)$ – начальная точка, поэтому $\varphi_{m_0} = Pre$,
- $m_k = end(\pi_k)$ – терминальная точка, поэтому $\varphi_{m_k} = Post$ и
- $\forall\, i = 1, \ldots, k-1 \quad m_i = end(\pi_{i-1}) = start(\pi_i) \in M$.

Кроме того, $\forall\, i = 1, \ldots, k$ текущие означивания ξ_{i-1} и ξ_i вычисления $Exec$ до и после прохождения компоненты π_i последовательности $\pi_1 \ldots \pi_k$ соответственно удовлетворяют условию $\xi_{i-1} \xrightarrow{\pi_i} \xi_i$, поэтому, согласно (3),

$$\forall\, i = 1, \ldots, k \quad \varphi_{m_{i-1}}^{\xi_{i-1}} = 1 \;\Rightarrow\; \varphi_{m_i}^{\xi_i} = 1.$$

Суммируя все вышесказанное, получаем цепочку импликаций:

$$(Pre^{\xi} = 1) \Rightarrow (\varphi^{\xi}_{m_0} = 1) \Rightarrow (\varphi^{\xi_1}_{m_1} = 1) \Rightarrow \ldots$$
$$\ldots \Rightarrow (\varphi^{\xi_k}_{m_k} = 1) \Rightarrow (Post^{\xi P} = 1). \blacksquare$$

3.4. Примеры фундированных множеств

В качестве фундированных множеств, требуемых для обоснования завершаемости, можно брать, например, следующие множества:

- множество $\mathbf{N}$ натуральных чисел $(0, 1, \ldots)$,
- множество L^k кортежей длины k элементов произвольного фундированного множества L, с лексикографическим порядком, который определяется следующим образом: для любой пары кортежей $a = (a_1, \ldots, a_k)$, $b = (b_1, \ldots, b_k)$ из L^k

  $a \le b \Leftrightarrow a = b$ или $\exists\, i \in \{1, \ldots, k\} : a_i < b_i, \forall\, j \in \{1, \ldots, i-1\}\ a_j = b_j$.

  Обоснуем фундированность этого множества. Пусть существует бесконечная строго убывающая последовательность

  $$(a^1_1, \ldots, a^1_k) > (a^2_1, \ldots, a^2_k) > \ldots \tag{5}$$

  элементов множества L^k. Из (5) и из определения лексикографического порядка следует, что $a^1_1 \ge a^2_1 \ge \ldots$. Поскольку L фундировано, то в этой последовательности неравенств не может быть бесконечного количества строгих неравенств, т.е.

  $$\exists\, i_1 :\ a^{i_1}_1 = a^{i_1+1}_1 = \ldots. \tag{6}$$

  Из последнего соотношения и из (5) следует, что

  $$(a^{i_1}_2, \ldots, a^{i_1}_k) > (a^{i_1+1}_2, \ldots, a^{i_1+1}_k) > \ldots \tag{7}$$

  Применяя изложенные выше рассуждения к (7), получаем, что

  $$\exists\, i_2 \ge i_1 :\ a^{i_2}_2 = a^{i_2+1}_2 = \ldots, \tag{8}$$

  откуда на основании (7) следует, что

  $$(a^{i_2}_3, \ldots, a^{i_2}_k) > (a^{i_2+1}_3, \ldots, a^{i_2+1}_k) > \ldots.$$

  Продолжая так и дальше, в конце концов получим, что

  $$\exists\, i_k \ge i_{k-1} :\ a^{i_k}_k = a^{i_k+1}_k = \ldots. \tag{9}$$

  Из (6), (8), …, (9) следует, что кортежи в (5), начиная с кортежа с номером i_k, совпадают, что противоречит предположению. $\blacksquare$

4. Процессные представления блок-схем

Для автоматизации верификации БС P методом инвариантов целесообразно сначала преобразовать эту БС в определяемый ниже граф G_P, который называется **процессным представлением** БС P. Для определения процессного представления БС необходимо ввести излагаемые в следующих пунктах вспомогательные понятия.

4.1. Действия и последовательности действий

Будем понимать под **действием** запись $x := e$ или $[\![b]\!]$, где $x \in \mathcal{X}$, $e \in \mathcal{E}$, $t(x) = t(e)$, $b \in \mathcal{B}$. Действие вида $x := e$ называется **присваиванием** переменной x значения выражения e, а действие вида $[\![b]\!]$ называется **проверкой условия**, выражаемого формулой b.

Действие вида $[\![b]\!]$, где формула b сама имеет вид $[\![b']\!]$ (т.е. является конъюнкцией), будет обозначаться без дублирующих квадратных скобок (т.е. записью $[\![b']\!]$).

Множество всех действий обозначается символом $\mathcal{A}$, множество всех конечных последовательностей действий обозначается записью $\mathcal{A}^*$. Если $a \in \mathcal{A}$ и $A, A' \in \mathcal{A}^*$, то записи aA, Aa и AA' обозначают конкатенации действия a и последовательности A, или A и A', соответственно.

$\forall\, a \in \mathcal{A}$ и $\forall\, A \in \mathcal{A}^*$ записи X_a и X_A обозначают множества переменных, содержащихся в a и A соответственно.

$\forall\, a \in \mathcal{A}$, $\forall\, X : X_a \subseteq X \subseteq \mathcal{X}$, $\forall \xi \in X^\bullet$ ξa обозначает объект, который

- равен означиванию $\xi(x := e)$ (см. (1)), если $a = (x := e)$,
- равен означиванию ξ, если $a = [\![b]\!]$ и $b^\xi = 1$
- не определен, если $a = [\![b]\!]$ и $b^\xi = 0$.

$\forall A \in \mathcal{A}^*$, $\forall X : X_A \subseteq X \subseteq \mathcal{X}$, $\forall \xi \in X^\bullet$ ξA обозначает объект, который

- совпадает с $(\ldots((\xi a_1)a_2)\ldots)a_k$, если $A = a_1 \ldots a_k$, и все объекты ξa_1, $(\xi a_1)a_2$, …, $(\ldots((\xi a_1)a_2)\ldots)a_k$ определены,
- не определен, в противном случае.

Последовательность из $\mathcal{A}^*$ называется **редуцируемой**, если она имеет вид $Aa[\![b]\!]A'$, где $A, A' \in \mathcal{A}^*$, $a \in \mathcal{A}$. **Редукцией** последовательности $Aa[\![b]\!]A'$ называется последовательность

- $A[\![b' \wedge b]\!]A'$, если $a = [\![b']\!]$,

- $A[\![(x := e)b]\!](x := e)A'$, если $a = (x := e)$
  (определение выражений вида $(x := e)e'$ см. в конце пункта 2.1).

Нетрудно доказать, что $\forall\, A \in \mathcal{A}^*$

- если A' – редукция A, то $\forall X : X_A \subseteq X \subseteq \mathcal{X}$, $\forall \xi \in X^\bullet$ $\xi A = \xi A'$ (т.е. ξA и $\xi A'$ либо оба не определены, либо определены и совпадают),

- $\exists\, A_1, \ldots, A_k\ (k \geq 1) : A_1 = A$, $\forall\, i = 1, \ldots, k-1$ A_{i+1} – редукция A_i, и A_k нередуцируема.

Последовательность A_k из предыдущего абзаца называется **нормальной формой** последовательности A, и обозначается записью $NF(A)$. Множество всех нередуцируемых последовательностей из $\mathcal{A}^*$ обозначается записью $\mathcal{A}_n^*$. Из определения понятия редукции следует, что каждая последовательность из $\mathcal{A}_n^*$ содержит не более одной проверки условия.

$\forall\, A \in \mathcal{A}_n^*$ запись $[\![A]\!]$ обозначает действие $[\![b]\!]$, если оно является первым в A, и действие $[\![1]\!]$, если A не содержит проверки условия.

$\forall A \in \mathcal{A}_n^*$, $\forall e \in \mathcal{E}$ запись Ae обозначает выражение

$$(x_1 := e_1) \ldots (x_k := e_k)e,$$

где $(x_1 := e_1) \ldots (x_k := e_k)$ – список всех присваиваний, входящих в A.

$\forall A \in \mathcal{A}_n^*, \forall\, \varphi, \psi \in \mathcal{B}$ запись $\varphi \stackrel{A}{\to} \psi$ обозначает формулу

$$\varphi \wedge [\![A]\!] \to A\psi,$$

которая выражает утверждение: $\forall\, \xi \in X^\bullet$, где $X_\varphi \cup X_A \cup X_\psi \subseteq X \subseteq \mathcal{X}$, если $\varphi^\xi = 1$ и ξA определено, то $\psi^{\xi A} = 1$.

4.2. Понятие процессного графа

Процессным графом (ПГ) называется конечный граф G со следующими свойствами:

- граф G имеет выделенные вершины G^0 и $G^\otimes$, называемые **начальной** и **терминальной** вершинами соответственно, при изображении ПГ в виде рисунка данные вершины обозначаются символами $\odot$ и $\otimes$ соответственно,

- каждому ребру α графа G сопоставлена метка $A_\alpha \in \mathcal{A}_n^*$,

- $G^{\otimes}$ – единственная вершина G, из которой не выходят рёбра.

Пусть G – ПГ. Будем обозначать записями V_G и X_G совокупность всех вершин G и переменных, входящих в G, соответственно. Будем предполагать, что в каждом ПГ G среди его переменных присутствует переменная at_G, множеством значений которой является V_G, и метка каждого ребра α содержит присваивание вида $at_G := v$, где v – конец ребра α. В записи меток рёбер ПГ данное присваивание будет опускаться.

Выполнение ПГ G – это обход вершин G, начиная с G^0, с выполнением действий, сопоставленных проходимым рёбрам. С каждым шагом $i \geq 0$ выполнения G связаны вершина v_i этого графа и означивание $\xi_i \in X_G^{\bullet}$ (называемые **текущей вершиной** и **текущим означиванием** на шаге i). Если $v_i = G^{\otimes}$, то выполнение G на шаге i завершается, иначе выполнение на шаге i заключается в

- выборе произвольного ребра α, которое выходит из v_i и удовлетворяет условию $[\![A_\alpha]\!]^{\xi_i} = 1$,
- замене текущего означивания ξ_i на означивание $\xi_{i+1} = \xi_i A_\alpha$, и
- переходе в конец выбранного ребра, который будет текущей вершиной v_{i+1} на $i+1$–м шаге выполнения.

4.3. Построение процессного представления блок-схем

Пусть задана БС P. Алгоритм построения ПГ G_P, называемого **процессным представлением** БС P, состоит из следующих шагов.

1) Удаляется начальная вершина P и выходящее из неё ребро, конец этого ребра – начальная вершина G_P^0 графа G_P.

2) Удаляются терминальные вершины P, кроме одной, которая является терминальной вершиной $G_P^{\otimes}$ графа G_P, рёбра с концами в удаляемых вершинах перенаправляются в $G_P^{\otimes}$.

3) Каждая оставшаяся вершина v БС P является вершиной ПГ G_P, и

   - если v имеет вид $[x := e]$, и P содержит ребро $v \to v'$, то G_P содержит ребро $v \overset{x:=e}{\to} v'$
   - если v имеет вид (b), и P содержит рёбра $v \overset{1}{\to} v'$ и $v \overset{0}{\to} v''$, то G_P содержит рёбра $v \overset{[\![b]\!]}{\to} v'$ и $v \overset{[\![\neg b]\!]}{\to} v''$.

Получившийся ПГ G_P может быть редуцирован путем применения операции **удаления вершины**: если v – вершина ПГ G_P, отличная от G_P^0 и $G_P^\otimes$, в G_P нет ребер вида $v \xrightarrow{A} v$, и списки ребер, входящих в v и выходящих из v, имеют вид

$$v_1 \xrightarrow{A_1} v, \ldots, v_k \xrightarrow{A_k} v \quad \text{и} \quad v \xrightarrow{A'_1} v'_1, \ldots, v \xrightarrow{A'_{k'}} v'_{k'} \tag{10}$$

соответственно, то операция удаления v из G_P заключается в удалении этой вершины и замене всех ребер из (10) на ребра вида $v_i \xrightarrow{A_{ij}} v'_j$, где $i \in \{1, \ldots, k\}$, $j \in \{1, \ldots, k'\}$, и $A_{ij} = NF(A_i A'_j)$.

Результат применения этой операции обозначается той же записью G_P и рассматривается как ПГ, эквивалентный в некотором смысле исходному ПГ. Данная операция может быть применена несколько раз.

4.4. Верификация блок-схем с использованием процессного представления

Пусть задан ПГ G. Для каждого пути $v_0 \xrightarrow{A_1} \ldots \xrightarrow{A_k} v_k$ в G записи $start(\pi)$ и $end(\pi)$ обозначают начало v_0 и конец v_k пути π соответственно, и запись A_π обозначает последовательность $NF(A_1 \ldots A_k)$. Путь π называется **циклом**, если $start(\pi) = end(\pi)$.

Множество вершин M ПГ G называется **базовым**, если для каждого цикла в G хотя бы одна из его вершин лежит в M. Если, кроме того, M содержит G^0 и $G^\otimes$, то M называется **полным** базовым множеством. Путь π в G называется **базовым** относительно M, если он непуст, $start(\pi)$ и $end(\pi)$ лежат в M, а остальные вершины π не лежат в M. Нетрудно доказать, что множество $\Pi_{G,M}$ всех базовых относительно M путей в G конечно.

Теорема.

Пусть заданы БС P и её спецификация, выражаемая предусловием Pre и постусловием $Post$. Тогда утверждение $Pre \xrightarrow{P} Post$ верно, если

- существует полное базовое множество M вершин G_P, причем с каждой вершиной $m \in M$ связана формула $\varphi_m \in \mathcal{B}$, и
  - $\varphi_{G_P^0} = Pre$, $\varphi_{G_P^\otimes} = Post$,
  - $\forall\, \pi \in \Pi_{G_P,M} \;\; (\varphi_{start(\pi)} \xrightarrow{A_\pi} \varphi_{end(\pi)}) = 1$,

- существуют базовое множество N вершин G_P и фундированное множество L, такие, что с каждой вершиной $n \in N$ связано выражение $u_n \in \mathcal{E}(X_P)$, причем выполнены условия:

  – если при каком-либо выполнении G_P текущей вершиной в какой-либо момент является вершина $n \in N$, то текущее означивание ξ в этот момент удовлетворяет условию $u_n^\xi \in L$,

  – $\forall\, \pi \in \Pi_{G_P,N} \quad \Big(\llbracket A_\pi \rrbracket \to \big(u_{start(\pi)} > A_\pi u_{end(\pi)}\big)\Big) = 1.$

Для доказательства теоремы заметим, что если существуют множества M, N и L, упоминаемые в формулировке этой теоремы, то по ним нетрудно построить аналогичные множества M_P, N_P, L_P, необходимые для доказательства $Pre \overset{P}{\to} Post$ на основе метода инвариантов. Действительно, если вершина m графа G_P принадлежит множеству M, то этой вершине соответствует некоторая вершина в исходной БС P. Мы зачисляем в M_P точки на всех рёбрах P, входящих в эту вершину, и сопоставляем каждой такой точке формулу φ_m. Аналогично определяется N_P. В качестве L_P можно взять L. Проверка всех условий, упоминаемых в формулировке метода инвариантов, является несложной. ■

5. Заключение

В настоящем тексте изложено описание метода инвариантов для верификации различных классов программ, и приведено несколько иллюстраций применения этого метода.

Наиболее актуальная проблема, связанная с применением метода инвариантов для верификации программ, заключается в автоматизации построения инвариантов (или в автоматизации построения доказательства их отсутствия в том случае, когда верифицируемая программа не удовлетворяет своей спецификации). В общем случае эта проблема алгоритмически неразрешима, поэтому данную проблему целесообразно рассматривать в следующей постановке: разработать средства интерактивного построения требуемых инвариантов, которые бы давали программисту рекомендации по поводу того, какой вид могли бы иметь инварианты, т.е. предлагали ему некоторые шаблоны инвариантов, которые он бы уже самостоятельно конкретизировал до формул, обладающих необходимыми свойствами. Некоторые продвижения в решении данной проблемы можно найти в работах [12]–[24].

Другим направлением исследований в этой области является распространение данного метода на другие классы программ, в том числе на

- программы с потенциально неограниченным количеством взаимодействующих процессов (например, MPI–программы),
- программы, при выполнении которых могут порождаться новые процессы (например, аналогичные тем, которые порождаются функцией `fork` в ОС UNIX),
- сети Петри,
- программы с использованием нейронных сетей,
- функциональные и логические программы, и т.п.

Список литературы

[1] Лекции лауреатов премии Тьюринга. - М.: Мир, 1993.

[2] Страничка Р. Флойда в Википедии.
`https://ru.wikipedia.org/wiki/Флойд,_Роберт`

[3] Floyd R. W. Assigning meanings to programs. In J. T. Schwartz, editor, Proc. Symp. Appl. Math., volume 19 of Mathematical Aspects of Computer Science, pages 19-32, Providence, R.I., 1967.

[4] Hoare C. A. R. An axiomatic basis for computer programming. Communications of the ACM, 12(10):576580, October 1969.

[5] Мендельсон Э. Введение в математическую логику. - М.: Наука, 1984.

[6] Андерсон Р. Доказательство правильности программ. - М.: Мир, 1982.

[7] Абрамов С. А. Элементы анализа программ. Частичные функции на множестве состояний. - М.: Наука, 1986.

[8] Непомнящий В. А., Рякин О. М. Прикладные методы верификации программ. - М.: Радио и связь, 1988.

[9] Francez N. Verication of programs. -Addison-Wesley Publishers Ltd., 1992.

[10] Loeckx J., and Sieber K. The Foundations of Program Verication. Wiley. -Teubner, Stuttgart, 1984.

[11] Manna Z. Mathematical theory of computation. -McGraw-Hill, 1974.

[12] Bjorner N., Browne A., and Manna Z. Automatic generation of invariants and intermediate assertions. Theoretical Computer Science, Volume 173, Issue 1, 1997, P. 49–87.

[13] S. Bensalem, Y. Lakhnech, H. Saidi. Powerful techniques for the automatic generation of invariants. Proc. 8th Internat. Conf. on Computer Aided Verification, Lecture Notes in Computer Science, Vol. 1102, Springer, Berlin (1996), pp. 323-335.

[14] R. Chadha, D.A. Plaisted. On the mechanical derivation of loop invariants. J. Symbol. Comput., 15 (5) (1993), pp. 705-744

[15] P. Cousot, R. Cousot. Abstract interpretation: a unified lattice model for static analysis of programs by construction or approximation of fix points. 4th ACM Symp. Principles of Programming Languages, ACM Press, New York (1977), pp. 238-252

[16] P. Cousot, N. Halbwachs. Automatic discovery of linear restraints among the variables of a program. 5th ACM Symp. Principles of Programming Languages (1978), pp. 84-97

[17] D. Dill, H. Wong-Toi. Verification of real-time systems by successive over and under approximation. Proc. 7th Internat. Conf. on Computer Aided Verification. Lecture Notes in Computer Science, Springer, Berlin (1995), pp. 409-422.

[18] S.M. German, B. Wegbreit. A synthesizer of inductive assertions. IEEE Trans. Software Eng., 1 (1975), pp. 68-75.

[19] N. Halbwachs, P. Raymond, Y.-E. Proy. Verification of linear hybrid systems by means of convex approximations. 1st Internat. Static Analysis Symp., Lecture Notes in Computer Science, Vol. 864, Springer, Berlin (1994), pp. 223-237.

[20] S. Bensalem, M. Bozga, J.-C. Fernandez, L. Ghirvu, and Y. Lakhnech. A transformational approach for generating non-linear invariants. In SAS, 2000, pp. 58–74.

[21] S. Sankaranarayanan, H. B. Sipma, and Z. Manna. Non-linear loop invariant generation using grobner bases. 2004.

[22] A. Tiwari, H. Rueb, H. Saidi, and N. Shankar. A technique for invariant generation. In TACAS 2001 – Tools and Algorithms for the Construction and Analysis of Systems, ser. LNCS, vol. 2031. Genova, Italy: Springer-Verlag, Apr. 2001, pp. 113–127.

[23] M. D. Ernst, J. H. Perkins, P. J. Guo, S. McCamant, C. Pacheco, M. S. Tschantz, and C. Xiao. The Daikon system for dynamic detection of likely invariants. Science of Computer Programming, vol. 69, no. 1–3, pp. 35–45, Dec. 2007.

[24] A. Gupta and A. Rybalchenko. Invgen: An efficient invariant generator. In Computer Aided Verification. Springer, 2009, pp. 634–640.

Verification of programs by the method of invariants
Mironov A.M.

We present a method of invariants for proving the correctness of computer programs. The basic concepts associated with this method are illustrated by examples of verification of sequential and parallel programs.

Keywords: program verification, Floyd's method, invariants

Об одной модели влияния в социальных сетях

Рыжов А.П., Ильин И.Ю.

В работе приводится модель влияния в социальных сетях и изучаются некоторые ее свойства. Основное внимание уделяется изучению матрицы влияния. Влияние рассматривается как величина, зависящая от времени, а также состояния и мнений других агентов. Для рассмотрения подобного влияния интерпретируется противоположность мнений агентов. Приведены примеры социальных сетей, основанных на данной модели. Рассмотрены несколько простых случаев сходимости матрицы влияния. Показаны отличия в структуре модели социальной сети с предельной матрицей влияния от некоторых существующих моделей.

Ключевые слова: анализ социальных сетей, модели влияния.

1. Введение.

Социальные сети являются одним из ярких феноменов современных информационно-коммуникационных технологий и общепризнанным трендом их развития. Приведем лишь несколько примеров [1]:

• Каждую секунду 8 человек на планете становятся частью какой-либо из существующих социальных сетей.

• Facebook по численности 3-я страна в мире, после Китая и Индии, с населением около миллиарда человек.

• Шансы того, что среднестатистический человек младше тридцати лет состоит в какой-либо социальной сети — более 50%.

Они все сильнее влияют на такие важные аспекты повседневной жизни, как семья (например, более 10% браков, заключенных в США, произошли благодаря социальным сетям; социальные сети служат причиной каждого третьего развода в мире [1]), экономика (у 24 из 25 крупнейших мировых газет упали тиражи из-за того, что новости начали приходить к людям через социальные сети; для поиска работников 4 из 5 компаний используют социальные сети [1]), и многие другие. Естественно, такой инструмент взаимодействия с огромной аудиторией не мог остаться не

замеченным для политиков, бизнесменов и других заинтересованных в этом людей и организаций. Социальные сети используются как для решения вполне понятых «традиционных» задач (например, маркетинга), так и достаточно необычных [2]. Среди большого набора таких задач нам будут интересны задачи влияния.

2. Краткая история развития социальных сетей.

Отношения между людьми, возникающие при этом структуры отношений, их анализ и управление ими, связанные с этим вопросы власти волновали людей с первых шагов организации общества. С развитием общества эти вопросы стали изучаться на систематической основе. В качестве примера можно привести всем известную Школу Пифагора: «Начиналось всё со ступеней обучения нравственной чистоте. Он объяснял законы взаимоотношений человека — с человеком, с природой, с Богом» [4]. Однако, собственно термин «социальная сеть» впервые введен Джоном Барнсом (John A. Barnes) только в 1954 г. в работе [3].

Интересующие нас компьютерные социальные сети возникли в результате достижений информационно-коммуникационных технологий, среди которых можно отметить следующие:

• 1971 г.: электронная почта (социальная сеть с использованием компьютерной техники), Рэй Томлинсон (Ray Samuel Tomlinson)

• 1988 г.: «IRC» (Internet Relay Chat) - общение в реальном времени, финский студент Ярко Ойкариненом (Jarkko Oikarinen)

• 1991 г.: Интернет, Тимоти Джон Бернерс-Ли (Sir Timothy John «Tim» Berners-Lee)

Интересующийся деталями читатель без труда сможет найти в Интернете всю необходимую информацию по упомянутым авторам. Перечисленное составило инфраструктурную основу современных социальных сетей. Собственно сети в современном их понимании возникли не так давно, и знаковыми событиями здесь были создание Рэнди Конрадом (Randy Conrad) сети Classmates.com в 1995 г. и создание Марком Эллиот Цукербергом (Mark Zuckerberg) сети Facebook в 2004 г.

Заметим, что мир реальный и мир виртуальный существуют не отдельно друг от друга. В качестве примера можно привести сеть Groupon, созданную в 2008 г. (https://www.groupon.com) – соединение общения и бизнеса, on-line и off-line миров [5, 6]. Заметим также, что многие исследователи и лидеры высокотехнологической индустрии (например, технический директор Google Рэй Курцвейл (Ray Kurzweil) - см., например, [7], а также множество ссылок в Интернет) видят будущее социальных

сетей как сетей смешанных, где общаются, развиваются, совместно решают задачи люди и роботы. Более подробно про это можно прочитать в [8].

В заключение этого краткого обзора отметим, что в анализе социальных сетей можно выделить два подхода, которые можно условно назвать взгляд снаружи и взгляд изнутри. Первый подход является более распространенным в настоящее время. Отчасти это связано с тем, что с математической точки зрения социальная сеть представляет собой достаточно хорошо изученный объект – граф, представляющий собой набор узлов или вершин и связей или ребер между ними. Такой взгляд позволяет визуализировать, сделать наглядным, состояние сети или ее части и понимать некоторые процессы в ней происходящие. Наиболее впечатляющая коллекция таких визуализаций представлена в блоге известного ученого и предпринимателя Стивена Вольфрама (Stephen Wolfram), разработчика системы компьютерной алгебры Mathematica и системы извлечения знаний WolframAlpha [9]. Некоторые из них приведены на рис. 1 и 2.

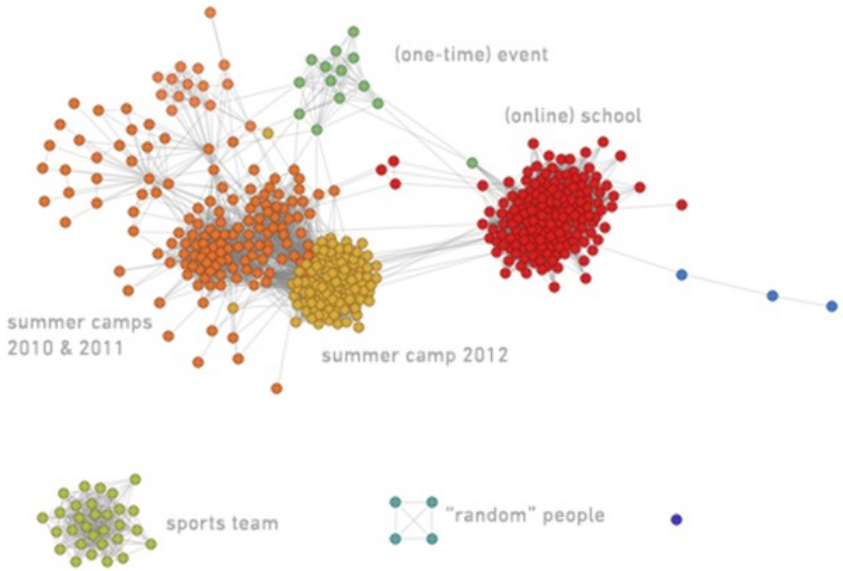


Рис. 1. Социальной граф 14 летней дочки С. Вольфрама.

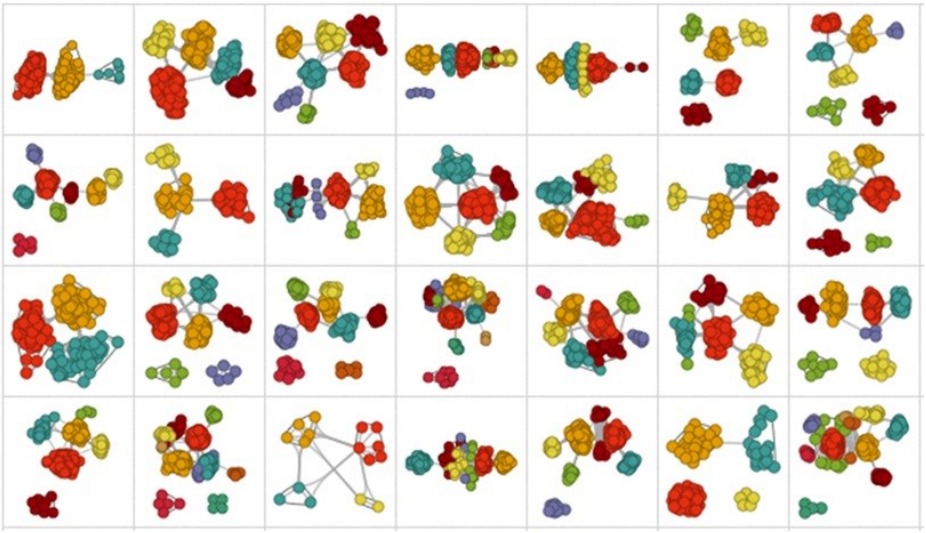


Рис. 2. Социальные графы друзей С. Вольфрама.

В рамках этого подхода вводятся различные меры важности, ценности и других качеств вершин и ребер графа для конкретных задач или классов задач, изучаются их свойства. Мы не будем повторять здесь все эти определения; интересующегося читателя отошлем к книге [10], где они представлены наиболее полно. В этой же книге содержатся очень интересное переосмысление классических задач управления, игр и других для сетевых объектов. Среди обширной англоязычной литературы кроме упомянутой можно выделить книги [11] и [12]. Второй подход, или взгляд изнутри, изучает свойства социальной сети как модели реального мира. Эти свойства и возникающие в связи с этим вопросы конструирования оптимальных сетей изучаются в работах [8, 13]. Ниже мы будем придерживаться первого подхода.

3. Влияние в социальных сетях.

В повседневности мы постоянно сталкиваемся с ситуациями, в которых нам приходится делать выбор. К примеру, утром вам может представится возможность встретиться с друзьями, для которой придется отменить ваши планы, а может быть вы захотите провести это время дома и посмотреть кино, или же вы решите заняться какими-либо важными вопросами и делами. Что бы вы не выбрали, на вас будут оказывать влияние различные факторы – друзья, обстановка, ваши личные предпочтения. Это простейший пример влияния, с которым мы сталкиваемся

в нашей жизни [15]. Тем не менее, он хорошо иллюстрирует важность этого явления для принятия человеком каких-либо решений.

Дадим формальное определение. В психологии влияние — процесс и результат изменения индивидом поведения другого человека, его установок, намерений, представлений, оценок и прочего в ходе взаимодействия с ним. Различаются влияние направленное и ненаправленное. Механизм первого — убеждение и внушение. При этом субъект ставит задачу добиться определенного результата от объекта влияния. Ненаправленное влияние подобной специальной задачи не имеет, хотя эффект воздействия возникает, часто проявляясь в действии механизмов заражения и подражания [16].

В рамках Марковской модели информационного влияния считается, что влияние формирует мнение по какому-либо вопросу [10]. Однако, как показывают исследования социальных психологов, мнение также в некоторой степени формирует и влияние. Для того чтобы пояснить это, сошлемся на социальный эксперимент Шехтера, где группы людей, занимающие одну позицию, переставали общаться с диссидентом, занимающим противоположную, а границы групп перестраивались таким образом, чтобы исключить диссидента и изолировать [15]. При этом способна возникать и другая ситуация: «влиятельное меньшинство» может изменить мнение большой группы и не станет диссидентами [28].

Именно поэтому замечание о воздействии мнения на влияние и представляется важным. Для того чтобы учесть это, мы немного изменим марковскую модель информационного влияния и зададим влияние агентов как зависящее от их мнений. Причем связь влияния и мнения мы будем задавать так, чтобы новая модель смогла явно отражать рассмотренные выше социальные явления: изоляцию диссидентов и влиятельное меньшинство. В какой-то мере введенная зависимость сблизит её с марковскими моделями с репутацией и управления доверием (более подробно про отличия с ними мы поговорим немного позже, когда будем определять модель математически). Мы сможем ставить схожие по смыслу задачи: «максимизировать репутацию агента» (для модели с репутацией), «максимизировать влияние агента» (для модели влияния), где под агентом мы будем понимать члена социальной сети.

4. Модель. Мнение, влияние, противоположность мнений. Структура социальной сети.

Для того чтобы построить нашу модель, обратимся к [10] и по аналогии с Марковской моделью информационного влияния дадим неко-

торые определения и понятия. Так, в Марковской модели социальная сеть представляет собой ориентированный граф $G = (V, E)$, где множество вершин V – множество действующих агентов (групп, людей, сообществ), E – множество ребер, связей между действующими агентами. Сами агенты, входящие в социальную сеть, будем описывать множеством $N = 1, 2, 3, \ldots, n$.

Итак, в Марковской модели агент – вершина ориентированного графа, агенты взаимодействуют друг с другом, высказывая мнение по определенному вопросу, и если i-й агент влияет на агента, то i-я и j-я вершины в графе связаны ориентированным ребром, причем ребру приписано некоторое положительное число, обозначающее доверие j-го агента i-му агенту, или же влияние i-го агента на j-го агента. Данное число может меняться со временем в зависимости от предыдущего состояния социальной сети и мнения агентов. Проинтерпретируем мнение, доверие и влияние в рамках данной модели:

Мнение i-го агента – функция $r_i(t) \to [s, s]$, $s \in \mathbb{R}^+$, $t \in \mathbb{N}$ - т.е. ограниченная вещественная функция, зависящая от дискретного момента времени t.

В определении мнения никаких отличий от Марковской модели нет, формироваться оно будет идентично тому, как это происходит в Марковской модели, но для того чтобы выписать его в явном виде нам необходимы следующие определения:

Влияние i-го агента на j-го агента – функция $a_{ij}(t, r_i(t-1), r_j(t-1)) \to [0, m]$, $m \in \mathbb{R}^+$, $t \in \mathbb{N}$ - ограниченная неотрицательная функция, также зависящая от дискретного момента времени t и предыдущих мнений. В дальнейшем конструкцию $a_{ij}(t, r_i(t-1), r_j(t-1))$ будем обозначать $a_{ij}(t)$ для сокращения записи.

Доверие – функция $b_{ij}(t, r_i(t-1), r_j(t-1)) \to [0, m]$, $m \in \mathbb{R}^+$, $t \in \mathbb{N}$ «двойственная» к функции влияния, т.е. $a_{ij}(t) = b_{ji}(t)$. В целях упрощения в дальнейшем будем использовать только функции влияния.

В дальнейшем, для того чтобы было удобно показывать зависимость влияния от мнения нам будет необходимо следующее определение противоположных мнений: будем говорить, что мнения i-го агента и j-го *противоположны*, если $sgn(r_i(t) \cdot r_j(t)) = -1$, и схожи, если $sgn(r_i(t) \cdot r_j(t)) = 1$.

После того как мы дали необходимые нам понятия, мы можем заметить, что функции мнения агентов образуют вектор $\vec{r}(t)$, на i-й строчке которого располагается функция мнения i-го агента, который называется вектором мнения или мнением социальной сети, а функции влияния

образуют матрицу влияния $\mathcal{A}(t)$, на i-й строчке в j-м столбце которой располагается функция влияния i-го агента на j-го. И теперь мы можем показать как будет формироваться мнение в момент времени $t + 1$ в матричном виде: $\vec{r}(t+1) = \mathcal{A}(t) \cdot \vec{r}(t)$.

Заметим, что функции влияния, в отличие от функций мнения, не будут указаны в явном виде. С какой-то стороны это может показаться упущением, так как не позволяет явно судить об их сходимости, однако в то же время мы сможем задавать более сложные функции влияния.

Перед тем как описать структуру социальной сети, мы должны дать определения итоговой матрицы влияния и итогового вектора мнения, иначе нам трудно будет давать ответы на следующие вопросы: «В каком состоянии будет находиться социальная сеть через некоторый большой период времени? Какие мнения в итоге установятся в социальной сети? Как будут влиять агенты друг на друга по прошествии некоторого большого периода времени? Перестанут ли они «дружить» друг с другом или нет?».

Пусть существует предел $\lim\limits_{t\to\infty} \mathcal{A}(t) = \mathbf{A}$. Тогда матрица $\mathbf{A}$ называется матрицей результирующего влияния или итогового влияния. Также, если существует предел $\lim\limits_{t\to\infty} \vec{r}(t) = \mathbf{r}$, то вектор $\mathbf{r}$ называется вектором результирующего мнения, или итоговым мнением. Из того, как формируется вектор $\vec{r}(t)$ следует, что $\mathbf{r} = \mathbf{rA}$, а сам вектор $\mathbf{r}$ будет являться собственным вектором матрицы $\mathbf{A}$ с $\lambda = 1$.

До того, как мы приступим к изучению итогового влияния и мнения, необходимо задать структуру социальной сети, чтобы результаты были легко интерпретируемы, и мы смогли сравнить их с результатами, полученными в других моделях социальных сетей. Поэтому в соответствии с [10] мы введем несколько понятий:

Сообществом S называется множество агентов, которые не подвергаются влиянию агентов вне него: $S \subseteq N|$ и для $\forall i \in S, \forall j \in N \setminus S$ выполнено $a_{ij}(t) = 0$, $\forall t \in \mathbb{N}$.

Группой называется множество агентов, которые влияют друг на друга прямым или косвенным образом.

Спутником группы называется агент, подвергающийся влиянию этой группы, но не оказывающий на нее влияния.

Смысл этих определений подробно описан в книге [10], для Марковской модели информационного влияния. Так как мы задаем их абсолютно идентично, обратившись к вышеуказанной книге, читатель может по-

смотреть на основные примеры сообществ, групп и спутников, здесь же мы оставим данные определения без примеров.

Представляется важным заметить, что в каком-то сообществе могут содержаться другие сообщества. Для нашего удобства и во избежание путаницы, мы можем выделить максимальные сообщества, то есть такие, в которые нельзя включить любого агента не из него. А затем мы можем описывать взаимодействия агентов только в отдельно взятом максимальном сообществе, и не изучать другие максимальные сообщества (ведь агенты из других сообществ на него никак не влияют!), то есть матрицу $\mathcal{A}(t)$ можно "разбить"на подматрицы, характеризующие отдельные максимальные сообщества.

5. Вопрос существования результирующей матрицы и результирующего вектора. Свойства матрицы результирующего влияния.

Ранее мы вывели необходимое условие на собственное значение результирующих матриц, нам будут нужны такие, что у них будет собственный вектор $\mathbf{r} = \mathbf{Ar}$. Теперь посмотрим, какие свойства достаточны для матрицы $\mathcal{A}(t)$ для того чтобы существовал её предел.

Перед тем как сформулировать первое утверждение уточним, что в нём под колебанием элементов матрицы понимается колебание функций на множестве, см. [24].

Утверждение 1. Пусть колебание элементов матрицы влияния стремится к нулю. Тогда существует матрица итогового влияния.
Доказательство:
В данном случае функции $a_{ij}(t)$ будут удовлетворять критерию Коши и сходиться к некоторому пределу, следовательно, и матрица $\mathcal{A}(t)$ сходится к некоторой матрице *Утверждение доказано.*

Приведём пример двумерной дважды стохастической матрицы $\mathcal{A}(t)$ с $a_{11}(t) = a_{22}(t)$, которая удовлетворяет **утверждению 1** и имеет предел **A**. Зададим элементы $\mathcal{A}(t)$ следующим образом:

В начальный момент времени $a_{11}(0) = C$, $a_{12} = 1 - C$, $\frac{1}{2} \leq C < 1$. Во все остальные моменты времени $a_{11}(t) = 1 - a_{12}(t)$, $a_{12}(t) = a_{12}(t - 1) \cdot (1 + 2^{-2^t})$, если мнения агента 1 и 2 схожие, и $a_{12}(t) = \frac{a_{12}(t-1)}{2}$, если противоположные.

Покажем корректность и сходимость. Рассмотрим вектор $\vec{r}(0)$ такой, что мнения агентов 1 и 2 всегда схожие. Покажем, что $\lim\limits_{t\to\infty} a_{12}(t) < 1$.

Докажем по индукции, что $a_{12}(t) < 1 - 2^{-t-1}$. Для $t = 1$ это будет верно. Проверим для $t > 1$. Предположим, что верно для $t-1$, и получим для t: $a_{12}(t-1) \cdot (1 + 2^{-2^t}) < (1 - 2^{-t}) \cdot (1 + 2^{-2^{-t}}) < 1 - 2^{-t-1}$. А значит $\lim\limits_{t\to\infty} a_{12}(t) < \sum\limits_{i=1}^{\infty} 2^{-t} = 1$. Колебание стремится к нулю, и данный пример корректен, соответствует **утверждению 1** и сойдётся либо к единичной матрице (в случае, если мнения долго будут противоположны), либо куда-то ещё, если оба агента выработают схожие мнения.

Перед тем как к перейти к следующему примеру и сформулировать **утверждение 2**, дадим следующее определение.

Будем говорить, что *влияние монотонно по знаку*, если для любых $t > k$ при условии $sgn(r_i(t-1) \cdot r_j(t-1)) = 1$ выполнено $a_{ij}(t) \geq a_{ij}(t-1)$, и при $sgn(r_i(t-1) \cdot r_j(t-1)) = -1$ $a_{ij}(t) < a_{ij}(t-1)$, $i \neq j$. Если $sgn(r_i(t) \cdot r_j(t-1)) = sgn(r_i(t-1) \cdot r_j(t)) = 1$, то будем говорить, что мнения агентов остаются схожими (противоположными) в момент времени t.

Утверждение 2. Пусть влияние монотонно по знаку. Тогда если существует такое k, что для любых $t > k$ мнения любых агентов i и j будут оставаться схожими (или противоположными), то существует результирующая матрица.

Доказательство: Не ограничивая общности, пусть для некоторого k верно, что для любых $t > k$ и любых $\vec{r}(0)$ $sgn(r_i(t) \cdot r_j(t-1)) = sgn(r_i(t-1) \cdot r_j(t)) = 1$, $a_{ij}(t) \leq m$, и $a_{ij}(t) \uparrow\uparrow$ при $t > k$. Тогда мы можем применить теорему Вейерштрасса о пределе монотонной ограниченной последовательности. Значит $\exists \lim\limits_{t\to\infty} a_{ij}(t) = \mathbf{a_{ij}}$. *Утверждение доказано.*

Приведем пример матрицы влияния, которая подходит под формулировку **утверждения 2**. Так же, как и в предыдущем примере, возьмем lдвумерную дважды стохастическую матрицу с дополнительным условием $a_{11}(t) = a_{22}(t)$ и зададим элементы матрицы $\mathcal{A}(t)$ следующим образом: в начальный момент времени $a_{11}(0) = 1-C$, $a_{12}(0) = C$, $0 < C < \frac{1}{2}$, а для всех остальных моментов времени t функции влияния будут выглядеть следующим образом: $a_{12}(t) = C + C \cdot sgn(r_1(t-1) \cdot r_2(t-1)) \cdot \frac{2 \cdot arctg(t)}{\pi}$, $a_{11}(t) = 1 - a_{12}(t)$. Докажем, что для данной матрицы выполнены условия **утверждения 2**.

Пусть это не так, и мнения не остаются схожими на бесконечном множестве $S \subseteq \mathbb{N}$. Тогда оба мнения в любой момент времени t отличны от нуля, при этом $a_{11}(t) = a_{22}(t) > C$, $a_{12} < 1 - C$, $\forall t \in \mathbb{N}$, кроме того, для $s \in S$ верно, что $sgn(r_1(t) \cdot r_2(s)) = -1$. Не ограничивая общности, можем считать, что $sgn(r_1(t)) = 1$. Из определения модели $r_1(s+1) =$

$a_{11}(s+1) \cdot r_1(s) + a_{12}(s+1) \cdot r_2(s)$; $r_2(s+1) = a_{21}(s+1) \cdot r_1(s) + a_{22}(s+1) \cdot r_2(s)$. Возьмем такое s, что $r_1(s+1) > 0$, $r_2(s+1) < 0$. Мы можем это сделать так как $a_{12}(s)$ при противоположных знаках будет мало. В моменты $s + 1$ и s мнения остаются схожими. Значит, влияние монотонно по знаку $r_1(s+2) \geq r_1(s+1)$, $r_2(s+2) \leq r_2(s+1)$, и это будет верно для всех моментов $s + t$. Следовательно, сходимость достигается для любых начальных векторов $\vec{r}(0)$, и матрица удовлетворяет **утверждению 2**.

Теперь мы можем применить это утверждение и посмотреть, чему равен предел матрицы $\mathcal{A}(t)$ в бесконечности. Если мнения долго будут противоположными, то предел их влияния друг на друга будет равен нулю, а если же нет, то он будет равен $2C$, то есть в одном случае агенты перестанут контактировать друг с другом, а в другом их влияние друг на друга увеличится в два раза, и они выработают схожие мнения.

Если рассматривать матрицы $\mathcal{A}(t)$, подходящие под данные утверждения, то можно найти серьёзные идеологические отличия от моделей с доверием и репутацией, которые рассмотрены в [10].

В модели с доверием вводится функция управления, кроме того, рассматриваемое доверие агента связано только с управлением и мнением агента, сама матрица влияния не изменяется. В модели с репутацией формируются лишь общие репутации агентов, то есть репутация одного агента едина для всех остальных. Существенным математическим отличием являются условия сходимости. Сходимость данной модели исследовалась в довольно тривиальных случаях, тогда как Марковская модель информационного влияния сходится всегда, а модели с репутацией и управления доверием сходятся в зависимости от вида функций репутации и матрицы управлений. В целом, изучение условий сходимости данной матрицы представляется авторами хорошим направлением для дальнейшего исследования.

6. О виде матрицы результирующего влияния.

Ранее мы привели два частных случая существования результирующей матрицы и получили её вид для этих случаев. Эти примеры были довольно простыми, включающими лишь одну группу и двух агентов. Теперь, в предположении, что выполнено **утверждение 2** мы попробуем ответить на общий вопрос «как должна выглядеть результирующая матрица?», когда в начале у нас есть неопределенное количество сообществ, групп, спутников, агентов. В предыдущем разделе было отмечено, что матрицу $\mathcal{A}(t)$ можно изучать с помощью разделения её на подматрицы изначальных максимальных сообществ, то есть, если два сообщества

никак между собой не связаны, то мы будем рассматривать их матрицы по отдельности. Воспользуемся этим и посмотрим, что происходит с социальной сетью, у которой влияния монотонны по знаку, а мнения после некоторого момента времени остаются схожими. Оказывается, что верны следующие два утверждения:

Утверждение 3.

Пусть выполнены условия **утверждения 2** и для любых агентов i и j с противоположными мнениями $a_{ij}(t) \downarrow\downarrow 0$ для $\forall t > k$; $k, t \in \mathbb{N}$. Тогда результирующие мнения агентов максимальных сообществ итоговой матрицы влияния будут схожими.

Доказательство:

Предположим, что это не так. Пусть в результирующем сообществе **S** есть два множества **Q** и **R**, мнения агентов в каждом из них схожи, но при этом мнения агентов из множества **Q** противоположны мнениям агентов из **R**. Рассмотрим произвольного агента i из множества **Q** и произвольного агента j из множества **R**. Существует момент времени k такой, что их мнения будут оставаться противоположными для любых $t > k$. По условию $a_{ij}(t) \downarrow\downarrow 0$, значит предел $a_{ij}(t)$ будет равен нулю. Так как агенты i и j произвольны, то любой агент из **Q** (**R**) не влияет на агента из **R** (**Q**). Что приводит нас к противоречию с тем, что **S** – сообщество. *Утверждение доказано.*

Для стохастических матриц из этого можно вывести интересный факт о виде собственного вектора.

Утверждение 4.

Пусть выполнены условия **утверждения 3** и матрица $\mathcal{A}(t)$ – стохастическая. Тогда существует результирующий вектор $\mathbf{r}$ следующего вида: $\mathbf{r} = (c_1, ..., c_1, ..., c_k, ..., c_k)$, где k – количество итоговых сообществ, и каждый элемент c_i встречается столько раз, сколько содержится элементов в итоговом сообществе.

Доказательство: Для стохастической матрицы любого результирующего сообщества вектор $c \cdot (1, 1, ..., 1)$ будет являться собственным. (см. [25].) Результирующий вектор из единиц будет существовать всегда, т.к. матрице сообщества соответствует неразложимая Марковская цепь, у которой корень имеет кратность равную единице [27]. В силу того, что матрица A будет состоять из квадратных матриц для различных максимальных сообществ, результирующий вектор будет состоять из суммы собственных векторов указанного выше вида для матриц различных сообществ. *Утверждение доказано.*

Итак, для стохастических матриц с указанными свойствами мы получаем хорошо интерпретируемый результат – у каждого сообщества агенты будут иметь схожее мнение и их числовой показатель мнения также будет одинаков. В Марковской модели информационного влияния наблюдается немного другое: во всех группах будет принято одинаковое мнение с одинаковым числовым показателем. То есть всюду, где агенты прямо или косвенно влияют друг на друга. Здесь мы получаем несколько иную возможность развития событий: любое множество агентов, которое не будет подвергаться серьёзному влиянию извне, придёт к единому мнению, и не будет общаться с теми, кто займёт противоположную позицию, даже если ранее они контактировали. Таким образом, мы решили изначально поставленную задачу, состоящую в том, чтобы показать связь мнения и влияния с учетом согласования с затронутыми ранее исследованиями социальных психологов [28] о взаимодействии агентов в группах.

7. Заключение.

В данной работе была представлена новая модель социальной сети, основанная на изменении влияния агентов друг на друга в соответствии с занимаемыми ими позициями. В работе не ставилась цель показать все условия сходимости модели и не проводился её полный анализ, однако были показаны отличия в структуре получаемой социальной сети, сделана попытка учесть противоположность взглядов различных агентов. Представляется интересным дальнейший её анализ и исследование её связи с другими явлениями, наблюдаемыми в социальных сетях.

Перспективным для последующего исследования выглядит постановка задачи информационного управления, введение «СМИ» и управления их сообщениями для того чтобы получить наибольшее количество агентов с нужным мнением, а также постановка задачи управления влиянием. Как отмечалось ранее, схожие задачи можно увидеть в [10] у моделей с репутацией и управления доверием.

Еще одним направлением развития данной модели является применение её на реальных социальных сетях. Подбор оптимальных функций влияния, получение практических результатов, прогнозов, а затем соотнесение полученного с реальными данными, исследованиями психологов и с другими моделями.

Список литературы

[1] Сорок фактов о социальных сетях. http://promaketing.by/40-faktov-o-socialnyx-setyax.html

[2] Расследование Das Magazin: как Big Data и пара ученых обеспечили победу Трампу и Brexit. The Insider, 06.12.2016 - http://theins.ru/politika/

[3] John Arundel Barnes. Class and Committees in a Norwegian Island Paris. http://pierremerckle.fr/wp-content/uploads/2012/03/Barnes.pdf

[4] Диалоги с Пифагором - http://ru.pythagoras.name/dialogues-with-pythagoras.html

[5] Grupon - https://ru.wikipedia.org/wiki/Groupon

[6] Фрэнк Сеннетт. Groupon. Бизнес-модель, которая изменила то, как мы покупаем. Перевод с английского. Альпина Паблишер, Москва, 2013, 300 с.

[7] Васильев А. Прогноз развития технологий до 2099 года. Компьютерра, 2015. - http://www.computerra.ru/122163/predictions-of-raymond-kurzweil/

[8] Рыжов А.П. Некоторые задачи оптимизации и персонификации социальных сетей. Saarbrucken, LAP, 2015, 96 с. (ISBN: 978-3-659-68661-0).

[9] Data Science of the Facebook World. - http://blog.stephenwolfram.com/2013/04/data-science-of-the-facebook-world/

[10] Губанов Д.А., Новиков Д.А., Чхартишвили А.Г. Социальные сети: модели информационного влияния, управления и противоборства. М., Физматлит, 2010, 228 с.

[11] John H. Miller and Scott E. Page. Complex Adaptive Systems: an introduction to computational models of social life. Princeton University Press, 2007, 264 с.

[12] Witold Pedrycz and Shyi-Ming Chen (Eds.). Social Networks: A

Framework of Computational Intelligence. Springer International Publishing Switzerland 2014, .440 p.

[13] Alexander Ryjov. Personalization of Social Networks: Adaptive Semantic Layer Approach. In: Social Networks: A Framework of Computational Intelligence. Witold Pedrycz and Shyi-Ming Chen (Eds.). Springer International Publishing Switzerland 2014, pp. 21-40.

[14] Д. А. Губанов, Д. А. Новиков, А. Г. Чхартишвили. Модели влияния в социальных сетях, Управление большими системами, 2009, выпуск 27, с. 205–281. -
http://www.mathnet.ru/links/ee5b0fe42833edce87554d4fdc7cf91d/ubs367.pd

[15] Зимбардо Ф., Ляппе М. Социальное влияние — СПб.: Питер, 2001 – 448 с.

[16] Словарь психолога-практика / Сост. С. Ю. Головин. — 2-е изд., перераб. И доп. — Мн: Харвест, М.: АСТ, 2001. — 976 с.

[17] Градосельская Г. В. Сетевые измерения в социологии: Учебное пособие / Под ред. Г. С. Батыгина. М.: Издательский дом «Новый учебник», 2004. — 248 с.

[18] D. Kempe, J. Kleinberg, E. Tardos. Maximizing the Spread of Influence through a Social Network. ACM SIGKDD, 2003.
https:/www.cs.cornell.edu/home/kleinber/kdd03-inf.pdf

[19] Богачев В. И., Смолянов О. Г. Действительный и функциональный анализ: Университетский курс. РХД, 2009. 724 стр.

[20] Винберг Э.Б., Курс Алгебры 2-е / изд., испр. и доп. М.: Изд-во Факториал Пресс, 2001. — 544 с.

[21] Колмогоров, Фомин, А.Н. Колмогоров, С.В. Фомин, Элементы теории функций и функционального анализа / 7-е изд. — М.: Физматлит, 2004. — 572 стр.

[22] Булинский А.В., Ширяев А.Н. Теория случайных процессов, М.: Физматлит, 2005. — 408 с.

[23] Бахвалов Н . С . Численные методы / Н. С. Бахвалов, Н. П. Жидков, Г. М. Кобельков. — 6-е изд. — М. : БИНОМ. Лаборатория знаний, 2008. — 636 с. : ил.

[24] Зорич В.А. Математический анализ. В 2-х ч. М.: ФАЗИС; Наука; Ч.I. - 1997, 568с.; Ч.II. - 1984, 640с

[25] Гантмахер Ф.Р. Теория матриц. 2-е изд., доп. — М.: Наука, 1966. — 576 с.

[26] Андреева Г.М. Социальная психология. Учебник для высших учебных заведений / Г. М. Андреева. — 5-е изд., испр. и доп. — М.: Аспект Пресс, 2007. — 363 с.

[27] Маркус, М.; Минк, Х. Обзор по теории матриц и матричных неравенств, М.: Наука. Главная редакция Физико-математической литературы, 1972. – 232 с.

[28] Майерс Д. Социальная психология. – СПб.: Питер, 2002.

About one model of influence in social networks

Ryjov A., Ilin I.

In this paper we provide a model of influence in social networks and study some of its properties. Primary purpose of the research was the study of influence matrix. The influence is considered as a function dependent on time, status and opinion of agents, for further analysis we give the interpretation of opposed opinions of agents. We also give two simple examples of social network based on this model and several cases of convergence of influence matrix. We show some differences between structure of this model with a limit matrix of influence and several other influence models of social networks.

Keywords: social network analysis, influence models.

Часть 2.
Специальные вопросы теории интеллектуальных систем

К вопросу о восстановлении трехмерного тела по его плоским проекциям

Д.В.Алексеев

В работе рассматривается задачи восстановления трехмерного изображения по его плоским проекций с точностью до аффинной и метрической эквивалентности. Найдено необходимое и достаточное условие разрешимости этих задач.

Ключевые слова: аффинные преобразования, изометрические преобразования, распознавание изображений, восстановление трехмерных изображений, стереозрение.

1. Введение

В работе рассматривается задача восстановления по плоским проекциям. В работе [4] описан процесс восстановления тела по плоским проекциям с точностью до аффинной эквивалентности. В [3] описан оптимизированный алгоритм решения этой задачи. В указанных работах не приводится критерия возможности восстановления тела, т.е. условия, позволяющие по двум наборам точек определить, являются ли они проекциями одного и того же трехмерного тела. В данной работе такой критерий приводится в главе 3.

Также рассматривается более сложная задача восстановления трехмерного тела с точностью до метрической эквивалентности проекции. Задача состоит в том, что надо построить трехмерное тело и задать две плоскости и два направления проектирования так, чтобы проекции были равны данным (как геометрические фигуры). Эта задача решается в главе 4.

2. Определения и формулировки задач

Определение 1. *Будем называть* ***изображением*** *(двумерным) произвольный занумерованный набор (непустое конечное множество) то-*

чек на плоскости. Будем называть ***телом*** *произвольный занумерованный набор (непустое конечное множество) точек в трехмерном пространстве (с указанием порядка).*

Определение 2. *Пусть дано 2-мерное изображение, состоящее из n точек $\mathcal{A} = (A_1, \ldots, A_n)$. Кроме того, пусть задан вектор $\bar{p}$ и прямая l, не параллельная вектору $\bar{p}$. Будем называть такие прямые (не параллельные $\bar{p}$)* ***допустимыми****. Рассмотрим прямые a_i, проходящие через соответствующие точки A_i параллельно вектору $\bar{p}$ ($i = 1, \ldots, n$). Проекцией изображения $\mathcal{A}$ по направлению $\bar{p}$ на прямую l называется изображение $\mathcal{A}' = A'_1, \ldots, A'_n$, в котором A'_i есть точка пересечения прямых a_i и l ($i = 1, \ldots, n$). Пусть на прямой l введена система координат, тогда вектор координат точек $X(A'_1), \ldots, X(A'_n)$ будем называть* ***отпечатком*** *множества $\mathcal{A}$ и обозначать $F_{l,p}(\mathcal{A})$.*

Определение 3. *Пусть дано 3-мерное изображение, состоящее из n точек $\mathcal{A} = (A_1, \ldots, A_n)$. Кроме того, пусть задан вектор $\bar{p}$ и плоскость Π, не параллельная вектору $\bar{p}$. Рассмотрим прямые a_i, проходящие через соответствующие точки A_i, $i = 1, \ldots, n$, параллельно вектору $\bar{p}$ (см. рис. 2) .* ***Проекцией*** *изображения $\mathcal{A}$ по направлению $\bar{p}$ на плоскость Π называется изображение $\mathcal{A}' = A'_1, \ldots, A'_n$, в котором A'_i есть точка пересечения прямой a_i с плоскостью Π ($i = 1, \ldots, n$).*

Задача 1. *Пусть даны два плоских изображения. Построить трехмерное тело и указать плоскости и направления проектирования, такие, что проекции указанного тела на эти плоскости аффинно эквивалентны данным плоским изображениям.*

Задача 2. *Пусть даны два плоских изображения. Построить трехмерное тело и указать плоскости и направления проектирования, такие, что проекции указанного тела на эти плоскости метрически эквивалентны данным плоским изображениям. Т.е. существуют изометрические преобразования, переводящие проекции в данные изображения.*

Дальше будут указаны необходимы и достаточные условия, для каждого из этих построений, также будут описаны алгоритмы этих построений.

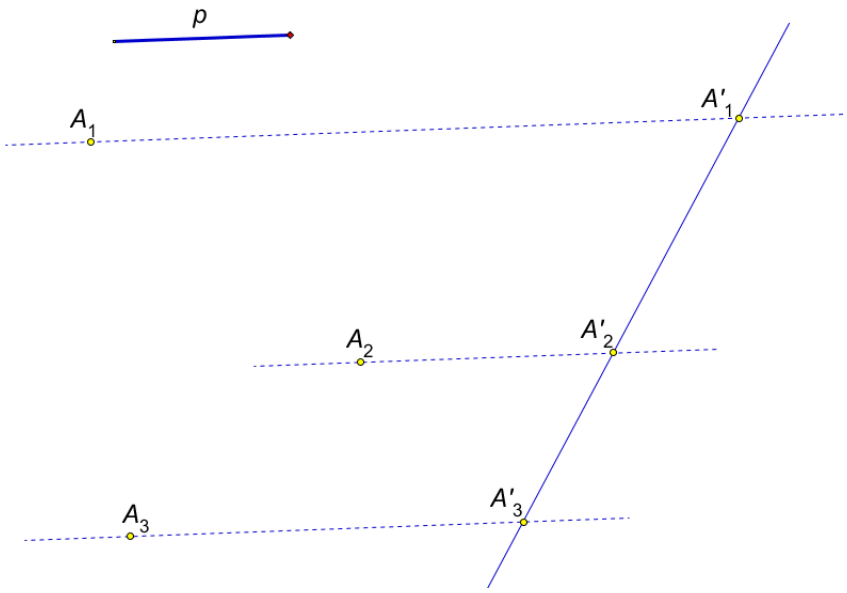


Рис. 1. Проекция на прямую

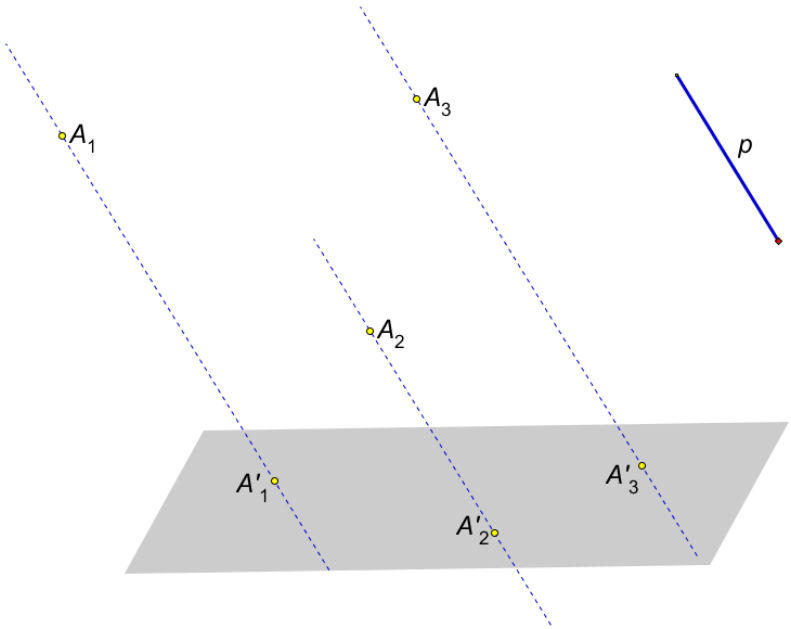


Рис. 2. Проекция на плоскость

3. Восстановление трехмерного тела с точностью до аффинной эквивалентности.

Несложно показать, что для двух произвольных плоских изображений задача 1 не всегда имеет решение.

Пример 4. *Рассмотрим точки с координатами* $A'_1 = A''_1 = (0,0)$, $A'_2 = A''_2 = (1,0)$, $A'_3 = A''_3 = (0,1)$, $A'_4 = (1,1)$, $A''_4 = (2,1)$, $A'_5 = (3,2)$, $A''_5(2,4)$. *Допустим задача 1 имеет решение для данных плоских изображений. Тогда точки* A_1, A_2, A_3, A_4 *не могут быть расположены в одной плоскости, т.к. иначе их проекции были бы аффинно эквивалентны. Следовательно, векторы* $e_1 = A_1A_2$, $e_2 = A_1A_3$ *и* $e_3 = \overrightarrow{A_1A_4}$ *образуют базис в трехмерном пространстве. Разложим вектор* $\overrightarrow{A_1A_5}$ *по этому базису:* $A_1A_5 = x_1e_1 + x_2e_2 + x_3e_3$. *Поскольку проекция на плоскость сохраняет линейную комбинацию векторов, то это же соотношение выполнено для проекций, откуда получаем систему линейных уравнений*

$$\begin{pmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 0 & 2 \\ 0 & 1 & 1 \end{pmatrix} \cdot \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 3 \\ 2 \\ 2 \\ 4 \end{pmatrix}.$$

Эта система, очевидно, не имеет решений, поскольку ранг расширенной матрицы равен 4.

Вообще говоря, в случае n точек получается $3(n-4)$ переменных и $4(n-4)$ уравнений, т.е. решение существует далеко не для любых наборов проекций. Ниже приводится необходимое и достаточное условие существования решения.

Определение 5. ***Условие коллинеарности двух проекций***. *Пусть заданы изображения* $\mathcal{A}' = (A'_1, \ldots, A'_n)$ *и* $\mathcal{A}'' = (A''_1, \ldots, A''_n)$. *Пусть в изображениях* $\mathcal{A}'$ *и* $\mathcal{A}''$ *точки* $A'_{i_1}, A'_{i_2}, A'_{i_3}$ *не лежат на одной прямой и* A''_{i_1}, A''_{i_2}, A''_{i_3} *не лежат на одной прямой. Рассмотрим аффинное отображение* $\mathcal{T}$, *которое переводит* $\mathcal{T} : A'_{i_1} \mapsto A''_{i_1}$, $\mathcal{T} : A'_{i_2} \mapsto A''_{i_2}$, $\mathcal{T} : A'_{i_2} \mapsto A''_{i_2}$. *Будем говорить, что изображения* $\mathcal{A}'$ *и* $\mathcal{A}''$ ***коллинеарны*** *относительно точек* i_1, i_2, i_3, *если все векторы* $\mathcal{T}(A'_i)A''_i$ *попарно взаимно коллинеарны (нулевой вектор коллинеарен любому). Будем это обозначать как* $\mathcal{A}' \|_{i_1,i_2,i_3} \mathcal{A}''$.

Теорема 6. *Даны изображения* $\mathcal{A}' = (A'_1, \ldots, A'_n)$ *и* $\mathcal{A}'' = (A''_1, \ldots, A''_n)$. *Пусть в изображениях* $\mathcal{A}'$ *и* $\mathcal{A}''$ *выбраны по 4 соответствующие*

точки (не ограничивая общности можно считать, что их индексы 1,2,3,4), обладающие следующими свойствами: a) Первые три A'_1, A'_2, A'_3 не лежат на одной прямой и A''_1, A''_2, A''_3 не лежат на одной прямой. b) Изображения $A'_1A'_2A'_3A'_4$ и $A''_1A''_2A''_3A''_4$ не являются аффинно-эквивалентными. Тогда необходимым и достаточном условием существования решения Задачи 1 (восстановления с точностью до аффинной эквивалентности) является условие коллинеарности изображений $\mathcal{A}'$ и $\mathcal{A}''$ относительно тройки A_1, A_2, A_3.

Замечание: Если изображения $\mathcal{A}'$ и $\mathcal{A}''$ аффинно эквивалентны, то задача 1 допускает тривиальное решение — любое из изображений можно взять в качестве исходного и спроектировать по произвольному направлению — результат будет аффинно-эквивалентен.

Докажем теорему 6.

Выберем какие-либо четыре точки изображения, не лежащие в одной плоскости. В дальнейшем будем называть эти точки опорными. Не ограничивая общности, можно считать, что это точки A_1, A_2, A_3, A_4. Введем трехмерный базис $\overrightarrow{e_1} = A_1A_2$, $\overrightarrow{e_2} = A_1A_3$ и $\overrightarrow{e_3} = A_1A_4$, начало координат поместим в A_1 (см. рис. 3), будем называть этот базис **опорным**. Тогда любую точку A_i можно задать координатами в опорном базисе (x_i, y_i, z_i), где $\overrightarrow{A_1A_i} = x_i \cdot \overrightarrow{e_1} + y_i \cdot \overrightarrow{e_2} + z_i \cdot \overrightarrow{e_3}$. Заметим, что при проектировании на плоскость линейные комбинации векторов сохраняются, т.е. $P(\overrightarrow{A_1A_i}) = x_i \cdot P(\overrightarrow{e_1}) + y_i \cdot P(\overrightarrow{e_2}) + z_i \cdot P(\overrightarrow{e_3})$. То же верно и для аффинных преобразований, следовательно, $\overrightarrow{A'_1A'_i} = x'_i \cdot \overrightarrow{e_1} + y'_i \cdot \overrightarrow{e_2} + z'_i \cdot \overrightarrow{e_3}$. Таким образом, если взять произвольную проекцию тела $\mathcal{A} = (A_1, ..., A_n)$ а потом произвести невырожденные аффинные преобразования, то указанные соотношения будут выполняться.

Пусть есть две проекции $(A'_1, ..., A'_n)$ и $(A''_1, ..., A''_n)$. Построим аффинные преобразования $\mathcal{T}'$ и $\mathcal{T}''$, переводящие A'_1 и A''_2 в точку $(0, 0)$, A'_2 и A''_2 в точку $(1, 0)$, A'_3 и A''_3 в точку $(0, 1)$, соответственно.

Допустим, что при таких преобразованиях точка A'_4 переходит в точку с координатами $\mathcal{T}'(A'_4) = B'_4(x'_4, y'_4)$, а точка A''_4 — в точку с координатами $\mathcal{T}''(A''_4) = B''_4(x''_4, y''_4)$. Допустим, что исходная точка A_i имела аффинные координаты (x_i, y_i, z_i). Тогда, спроектировав и применив аффинные отображения $\mathcal{T}'$ и $\mathcal{T}''$, получим, что координаты (двумерные) будут равны $\mathcal{T}'(A'_i) = B'_i(x_i + x'_4z_i, y_i + y'_4z_i)$ и $\mathcal{T}''(A''_i) = B''_i(x_i + x''_4z_i, y_i + y''_4z_i)$. Несложно заметить, что вектор $\overrightarrow{B'_iB''_i}$ имеет координаты $\overrightarrow{B'_iB''_i} = z_i \cdot (x''_4 - -x'_4, y''_4 - y'_4)$. Таким образом, получаем, что все эти векторы коллинеарны

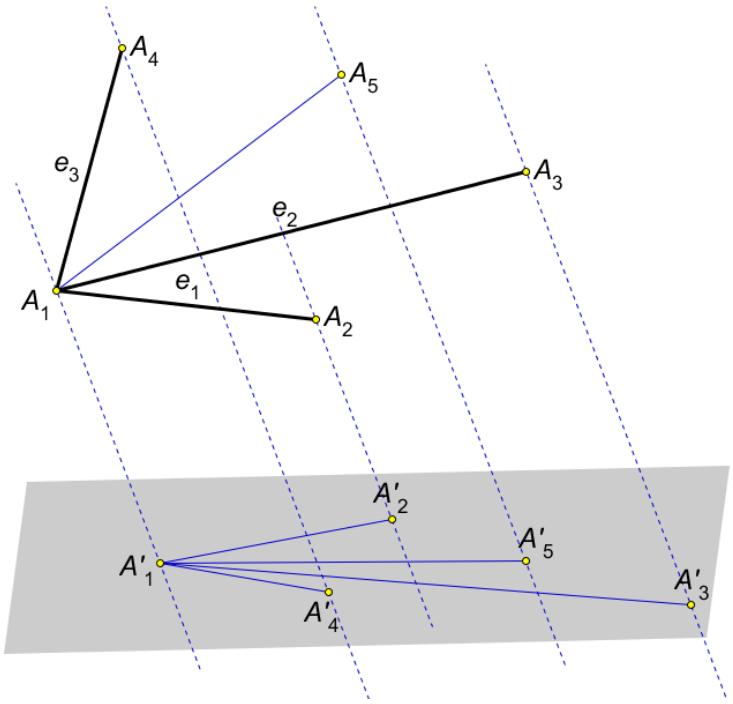


Рис. 3. К доказательству теоремы 6.

вектору $\overrightarrow{B'_4B''_4}$, а их длины пропорциональны величине z_i. Это доказывает необходимость условия коллинеарности изображений.

Покажем, что это же условие коллинеарности является и достаточным для существования решения. Пусть $\mathcal{A}'\|_{1,2,3}\mathcal{A}''$, т.е. изображения $\mathcal{A}'$ и $\mathcal{A}''$ коллинеарны относительно A_1, A_2, A_3, $\mathcal{T}$ — аффинное отображение, переводящее $A'_i \mapsto A''_i$, $i = 1, 2, 3$. Обозначим $C_i = \mathcal{T}(A'_i)$, $i = 1, \ldots, n$. Выберем евклидову систему координат и построим точки $\tilde{A}_i$, $i = 1, \ldots, n$ с координатами $x'_i, y'_i, \tilde{z}_i$, где $\tilde{z}_i$ выбрано из условия

$$\overrightarrow{C_iA''_i} = \tilde{z}_i \cdot \overrightarrow{C_4A''_4}. \tag{1}$$

Рассмотрим тело, состоящее из точек $\tilde{A}_i$, $i = 1, \ldots, n$. Выберем плоскость $z = 0$ и направление $p_1(0, 0, 1)$. Очевидно, проекции точек в указанном направлении $\tilde{A}_i$ будут иметь координаты (x'_i, y'_i), $i = 1, \ldots, n$.

Рассмотрим теперь вектор $\overrightarrow{p_2}(x''_4 - x'_4, y''_4 - y'_4, -1)$ в качестве направления проектирования (на ту же плоскость). Точки $\tilde{A}_1, \tilde{A}_2, \tilde{A}_3$ лежат в указанной плоскости, поэтому их проекции совпадут с исходными точками. Точка $\tilde{A}_4(0, 0, 1)$ будет спроектирована в точку $P(\tilde{A}_4) = A''_4(x''_4, y''_4)$, поскольку вектор $\overrightarrow{p_2}\|\overrightarrow{A_4A''_4}$.

Покажем, что остальные точки $\tilde{A}_i$, тоже будут проектироваться в точки $A''_i(x''_i, y''_i)$, $i = 5, \ldots, n$. Отложим вектор $\tilde{z}_i \cdot \overrightarrow{p_2}$ от точки $\tilde{A}_i(x_i, y_i, z_i)$. Тогда конец вектора попадет в точку с координатами $(x'_i, y'_i) + z_i \cdot (x''_4 - x'_4, y''_4 - y'_4, 0)$, которая и будет проекцией $\tilde{A}_i$ на плоскость $z = 0$ по направлению $\overrightarrow{p_2}$. Поскольку z_i выбрано из условия 1, то указанная проекция совпадет с A''_4.

Следствие 7. *Поскольку существование трехмерного тела не зависит от выбора опорных точек, то условие коллинеарности также не зависит от выбора четырех исходных точек (не лежащих в одной плоскости). Поэтому в дальнейшем будем говорить просто про условие коллинеарности изображений и обозначать* $\mathcal{A}'\|\mathcal{A}''$.

4. Восстановление трехмерного тела с точностью до метрической эквивалентности проекций

4.1. Некоторые свойства проекции двумерного изображения на прямую

Пусть заданы точки A_1, ..., A_n, прямая L и $\overrightarrow{p}$ — допустимое направление проекции. Будем обозначать a_i прямые, проходящие через A_i параллель-

но $\overrightarrow{p}$. Будем обоозначать A'_i — проекции точек A_i $(i = 1, ..., n)$. Пусть на прямой L введена система координат — начало координат O и единичный вектор $\vec{e_1}$. Рассмотрим $F_{L,p}(\mathcal{A}) = (X(A'_1), ..., X(A'_n))$ — отпечаток изображения $\mathcal{A}$.

Лемма 8. *Пусть $\varphi \neq \frac{\pi}{2}$ — угол между нормалью к прямой l и вектором $\overrightarrow{p}$. Тогда расстояние между прямыми a_i и a_j равно $|\left(X(A'_i) - X(A'_j)\right) \cdot \cos\varphi|$.*

Доказательство. Очевидно следует из определения косинуса. (см. рис. 4)

Рассмотрим $\rho(a_i, a_j)$ — ориентированное расстояние между прямыми a_i и a_j, которое положительно, если прямая a_j лежит в положительной полуплоскости по отношению к a_i (т.е. в той полуплоскости, куда направлен вектор нормали $\vec{n}_i$), и отрицательно в противном случае.

Лемма 9. *Пусть $\varphi \neq \frac{\pi}{2}$ — угол между нормалью к прямой L и вектором $\overrightarrow{p}$. Пусть нормали к прямым a_i выбраны так, что образуют острый угол с $\vec{e_1}$ — положительным направлением прямой L. Тогда ориентированное расстояние равно*

$$\rho(a_i, a_j) = (X(A'_j) - X(A'_i)) \cdot \cos\tilde{\varphi}.$$

Доказательство. Очевидно следует из леммы 8 и сохранения порядка точек при проектировании.

Лемма 10. *Пусть $T_{\vec{v}}$ — параллельный перенос на вектор $\vec{v}$, Пусть $L_1 = T_{\vec{v}}(L)$ — образ прямой L. Тогда между координатами проекций на прямые L и L_1 выполнено соотношение*

$$X_{L_1}(A_i) = X_L(A_i) + C,$$

где $C = C(\vec{v}, \vec{p}, \vec{e})$. Таким образом, при параллельном переносе прямой L координаты отпечатка $F_{L,p}$ изменяются на одну и ту же величину.

Доказательство. Пусть точка $O_1 = T_{\vec{v}}(O)$, а O'_1 — ее проекция (см. рис. 5). Тогда $X_{L_1}(A''_i) = X_L(A'_i) - X_L(O'_1)$. □

Лемма 11. *Пусть $T_{\vec{v}}$ — параллельный перенос на вектор $\vec{v}$, точки $B_i = T_{\vec{v}}(A_i)$ — образы точек исходного изображения. Тогда между координатами проекций выполнено соотношение*

$$X(B'_i) = X(A'_i) + C,$$

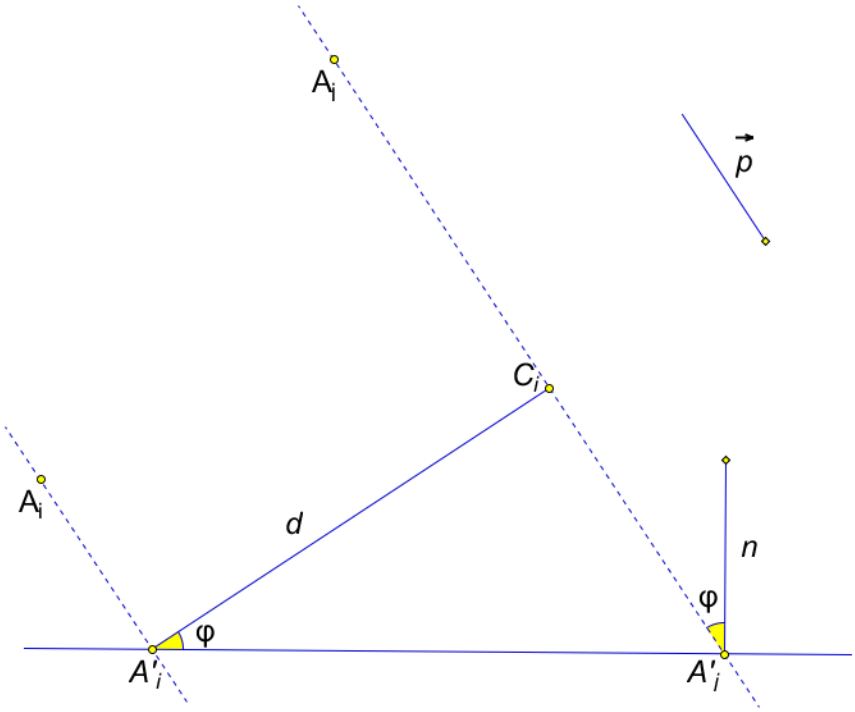


Рис. 4. К лемме 8

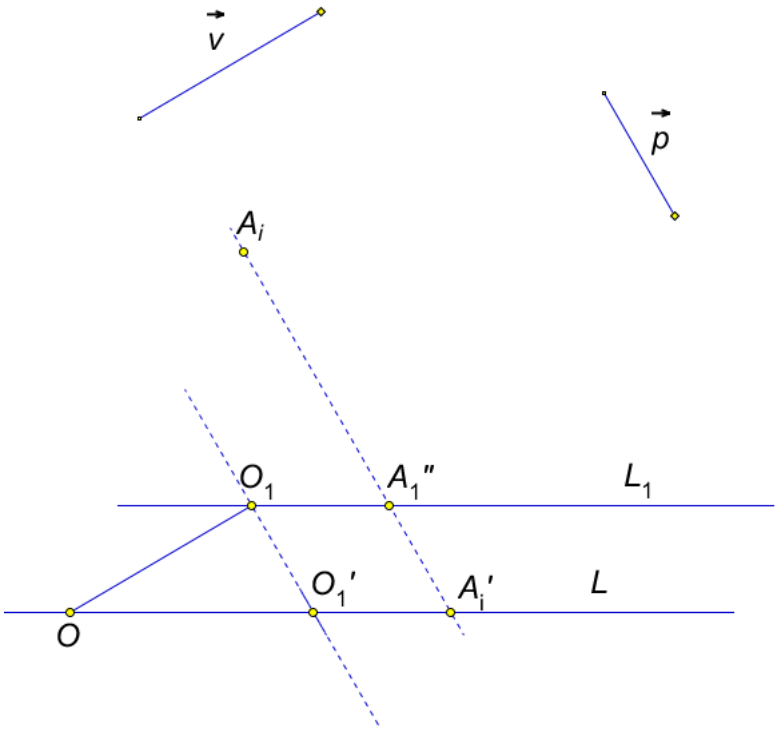


Рис. 5. К лемме 10

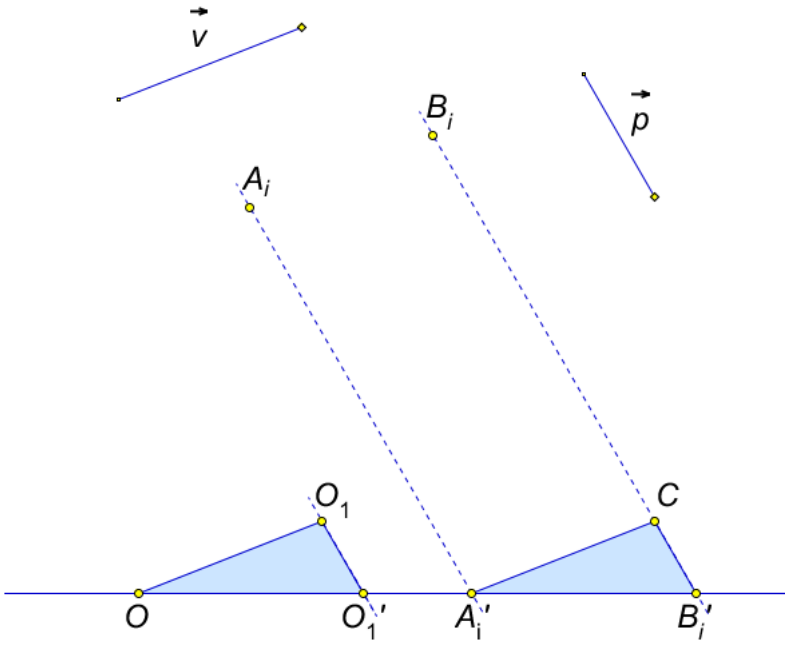


Рис. 6. К лемме 11

где $C = C(\vec{v}, \vec{p}, \vec{e})$. *Таким образом, при параллельном переносе изображения координаты отпечатка изображения* $X(A'_1), \ldots, X(A'_n)$ *изменяются на одну и ту же величину.*

Доказательство. Пусть точка $O_1 = T_{\vec{v}}(O)$, а O'_1 — ее проекция (см. рис. 6). Тогда из равенства треугольников $\triangle OO_1O'_1$ и $\triangle A'_iC_iB'_i$ следует, что $X(B'_i) = X(A'_i) + X(O'_1)$. □

Выделим некоторую точку изображения. Не ограничивая общности рассуждений, можно считать, что выделена точка A_1. Параллельным переносом получим из прямой L прямую L_1, проходящую через точку A_1. Пусть точки $A_1, ..., A_n$ проектируются в точки $A'_1, \ldots, A'_n$, расположенные на прямой L_1. На прямой зададим систему координат: выберем $A'_1 = A_1$ в качестве начала координат и направление $A'_1A'_2$ в качестве

положительного (если проекции A'_1 и A'_2 совпали, то выберем вместо A_2 какую-то другую точку). Верна следующая

Лемма 12. *Пусть задана прямая L_1, проходящая через точку A_1 , $\varphi \neq$ $\neq \frac{\pi}{2}$ — угол между нормалью к прямой L_1 и вектором $\overrightarrow{p}$. Пусть задан угол $\theta \in [0; 2\pi)$, такой, что $\varphi+\theta \neq \frac{\pi}{2}+\pi k$, $k \in Z$. Рассмотрим поворот $R^{\theta}_{A_1}$ относительно A_1, обозначим $B_i = R^{\theta}_{A_1}(A_i)$, $i = 1, \ldots, n$ — образы точек при этом повороте. Рассмотрим проекцию изображения $\mathcal{B} =$ $= (B_1, \ldots, B_n)$ на ту же прямую L_1 (не повернутую) по направлению $\overrightarrow{p'} = R^{\theta}_{A_1}(\overrightarrow{p})$. Тогда верны следующие соотношения ($i = 1, \ldots, n$):*

$$X(B'_i) \cdot \cos(\varphi + \theta) = X(A'_i) \cdot \cos(\varphi).$$

Доказательство. Применим лемму 8 к исходному изображению $\rho(a_1, a_i) = X(A'_i) \cdot \cos\varphi$ и к повернутому изображению: $\rho(b_1, b_i) =$ $= X(B'_i) \cdot \cos(\varphi + \theta)$. Заметим, что расстояние между прямыми не меняется при повороте, откуда и вытекает утверждение леммы.

Таким образом, из леммы 12 вытекает, что при повороте исходного изображения координаты его отпечатка остаются пропорциональными друг другу.

Определение 13. *Будем считать эквивалентными отпечатки $X(A'_1)$, $\ldots$, $X(A'_n)$ и $X(B'_1), \ldots, X(B'_n)$ и обозначать $X(A'_1), ...X(A'_n)) \equiv$ $\equiv X(B'_1), ..., X(B'_n)$, если существует существует $\lambda \neq 0$, такое, что $X(A'_i) = \lambda X(B'_i)$. Класс эквивалентности будем называть проективными координатами пучка прямых $(a_1, \ldots, a_n)$. Видно, что при повороте изображения как целого (или, соответственно, при повороте прямой L) эти координаты не меняются.*

Зафиксируем теперь изображение и прямую l и будем менять только направление проектирования $\overrightarrow{p}$. Следующая лемма описывает, как при этом меняются проективные координаты пучка прямых.

.

Лемма 14. *Введем полярную систему координат следующим образом: полюс разместим в точке A_1, луч (A_1A_2[выберем в качестве полярной оси. Тогда каждая точка изображения A_i будет обладать полярными координатами (r_i, φ_i) ($i = 2, \ldots, n$). Пусть направление проектирования образует угол ψ с полярной осью, а проектирование производится на прямую A_1A_2. Тогда координаты проекции точки A_i будут равны $X_\psi(A'_i) = r_i \cos\varphi_i - r_i \sin\varphi_i \operatorname{ctg}\psi$.*

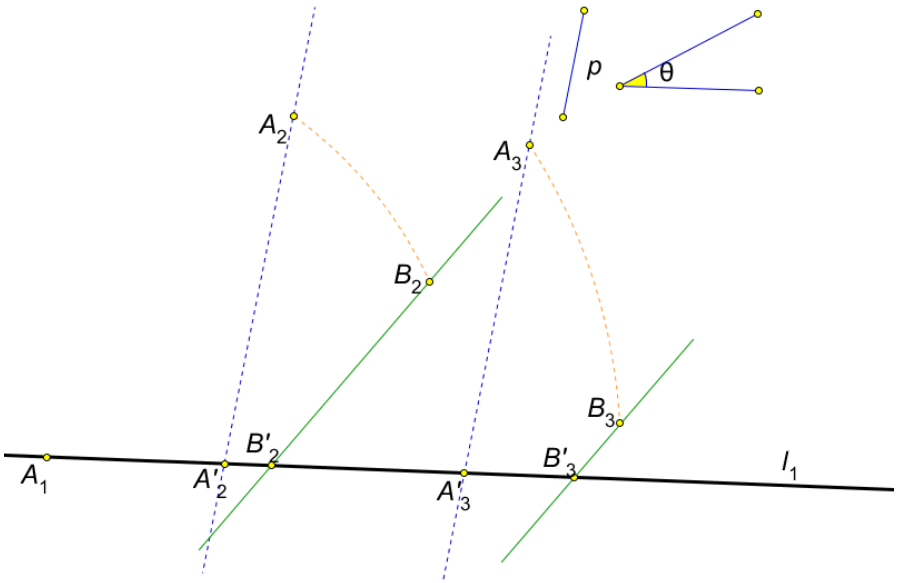


Рис. 7. Поворот изображения

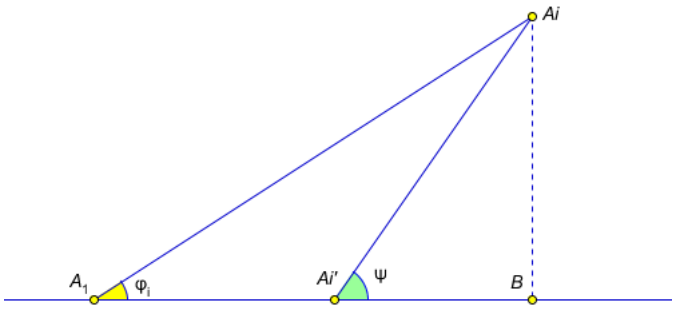


Рис. 8. К лемме 14.

Доказательство. Очевидно, $A_1'B = r_i \cos\varphi_i$, откуда получаем равенство $A_i'B = A_iB \cdot ctg\psi = r_i \sin\varphi_i \operatorname{ctg}\psi$ (см. рис. 8).

Следствие 15. *Заметим, что* $t = \operatorname{ctg}\psi$ *пробегает все значения из* $\mathbb{R}$. *Таким образом, в лемме 14 в параметрическом виде задана прямая в* $n-1$ *мерном пространстве. Если же искусственно добавить бесконечно удаленную точку, соответствующую* $\operatorname{ctg}\frac{\pi}{2} = \infty$, *то получим прямую в соответствующем проективном пространстве.*

4.2. Совмещение двух изображений при проекции на прямую

Теорема 16. *Пусть заданы два изображения, состоящие из 4 точек* $\mathcal{A} = (A_1, A_2, A_3, A_4)$ *и* $\mathcal{B} = (B_1, B_2, B_3, B_4)$. *Можно разместить эти изображения, прямую* L *и выбрать направления проекций таким образом, что* $F_{p_1,L}(\mathcal{A}) = F_{p_2,L}(\mathcal{B})$.

Доказательство. Рассмотрим изображения $\mathcal{A}$ и $\mathcal{B}$ и поставим в соответствие прямые на проективной плоскости (как в Следствии 15). На проективной плоскости прямые пересекаются в одной точке Ξ с координатами (ξ_1, ξ_2, ξ_3) (если прямые совпадают, то возьмем произвольную точку, принадлежащую им).

Этой точке соответствуют значения углов ψ_a и ψ_b, при которых

$$F_{\psi_a,l}(A_1, A_2, A_3) \sim (\xi_1, \xi_2, \xi_3)$$

$$F_{\psi_b,l}(B_1, B_2, B_3) \sim (\xi_1, \xi_2, \xi_3).$$

Рассмотрим соответствующие направления проектирования p_a и p_b. Расположим изображения $\mathcal{A}$ и $\mathcal{B}$ на плоскости таким образом, чтобы точки A_1 и B_1 совпали, а прямые L_1 и L_2 выберем перпендикулярно направлениям p_1 и p_2 и проходящими через $A_1 = B_1$. Тогда проекцией точек $A_1 = B_1$ будет начало координат а отпечатки $(X(A'_1), X(A'_2), X(A'_3), X(A'_4)) \sim X(B'_1), X(B'_2), X(B'_3), X(B'_4)$, — пропорциональны, т.к. $X(A'_1) = X(B'_1) = 0$, а $(X(A'_2), X(A'_3), X(A'_4)) \sim$ $\sim (X(B'_2), X(B'_3), X(B'_4))$, как соответствующие одной и той же точке $\Xi(\xi_1, \xi_2, \xi_3)$ на проективной плоскости. Пусть $\lambda \neq 0$ — коэффициент пропорциональности, т.е.

$$(X(A'_2), X(A'_3), X(A'_4)) = \lambda(X(B'_2), X(B'_3), X(B'_4)).$$

Можно считать, что $\lambda < 1$ не ограничивая общности рассуждений (в противном случае поменяем изображения $\mathcal{A}$ и $\mathcal{B}$ местами). Повернем первое изображение вместе с направлением p_1 так, чтобы угол между L_1 и L_2 стал равен $\gamma = \arcsin\lambda$ (см. рис 9). Очевидно, при этом проекции точек A_i совпадут с проекциями соответствующих точек B_i $(i = 2, 3, 4)$.

4.3. Восстановление трехмерного изображения

Докажем теорему 6. Рассмотрим две произвольные плоскости Π_1 и Π_2, пересекающиеся по прямой L. Расположим изображение $\mathcal{A}$ в плоскости Π_1, а $\mathcal{B}$ в плоскости Π_2, выберем направления проектирования p_1 и p_2 таким образом, чтобы проекции точек изображений $\mathcal{A}$ и $\mathcal{B}$ на прямую L по соответствующим направлениям совпали. Согласно теореме 16 это можно сделать. Выберем произвольное $i = 1, \ldots, n$ и проведем плоскость (см. рис. 10) через точку $C_i = F_{L,p_1}(A_i) = F_{L,p_2}(B_i)$ параллельно направлениям p_1 и p_2. Такая плоскость существует и единственна, т.к. $p_1 \not\parallel p_2$. Построим в этой плоскости прямые α_i и β_i следующим образом:

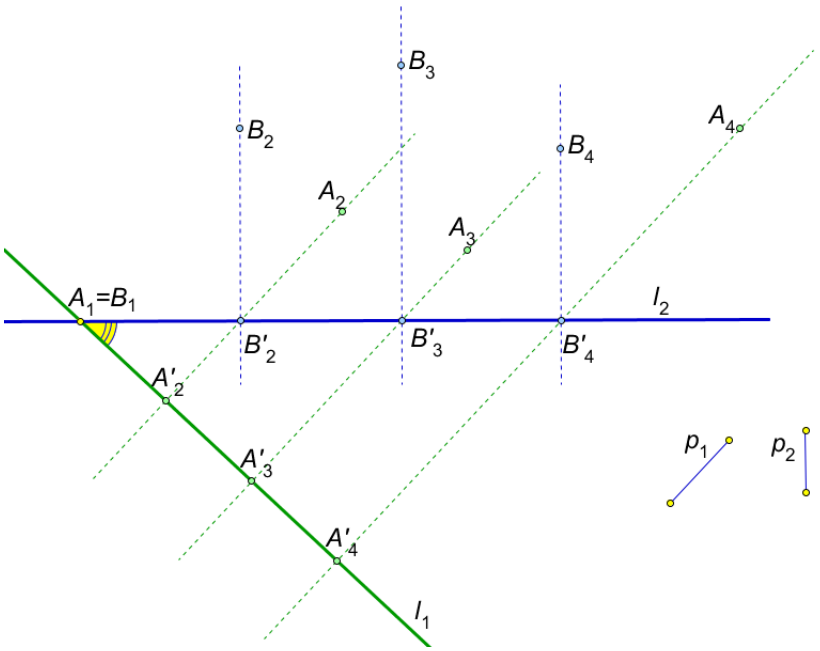


Рис. 9. К лемме 16.

α_i проходит через точку A_i параллельно p_2, β_i — через B_i параллельно p_1. Прямые пересекаются в некоторой точке X_i (параллельны они быть не могут). Тогда, очевидно, точка A_i является проекцией X_i на плоскость Π_1 по направлению p_2 и B_i — проекция X_i на плоскость Π_2 по направлению p_1. Таким образом, построен тетраэдр $X_1X_2X_3X_4$, проекции которого на две плоскости в точности совпадают с изображениями $\mathcal{A}$ и $\mathcal{B}$ (см. рис. 11). Теорема доказана.

Рассмотрим теперь тетраэдр, построенный в главе 3. Существует единственное аффинное преобразование F трехмерного пространства, переводящее этот тетраэдр в $X_1X_2X_3X_4$. Тогда выберем $X_i = F(A_i)$, $i = 5, ..., n$. Итак, образами точек X_1, X_2, X_3, X_4 при указанных проекциях будут точки A'_1, A'_2, A'_3, A'_4 и A''_1, A''_2, A''_3, A''_4. Поскольку линейные комбинации векторов сохраняются, то и для образов оставшихся точек X_i $(i = 5, \ldots, n)$ будут выполнены условия $\overrightarrow{X'_1X'_i} = x_i \cdot \overrightarrow{X'_1X'_2} + y_i \cdot \overrightarrow{X'_1X'_3} + + z_i \cdot \overrightarrow{X'_1X'_4}$ и $\overrightarrow{X''_1X''_i} = x_i \cdot \overrightarrow{X''_1X''_2} + y_i \cdot \overrightarrow{X''_1X''_3} + z_i \cdot \overrightarrow{X''_1X''_4}$ Следовательно, X'_i будут совпадать с совпадать с точками A'_i, а X''_i — с A''_i.

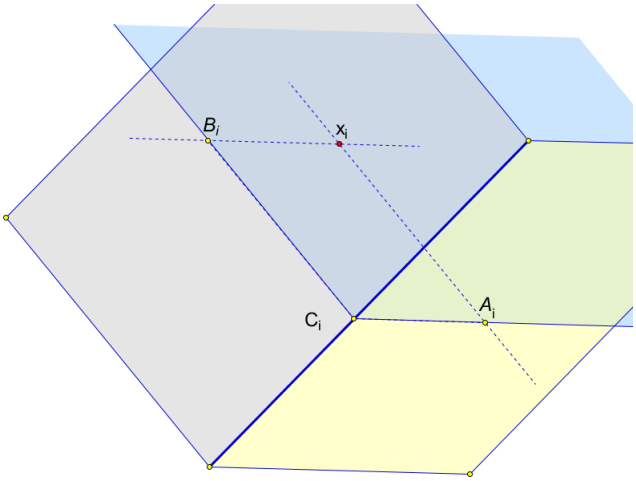


Рис. 10. Восстановление точки по проекциям

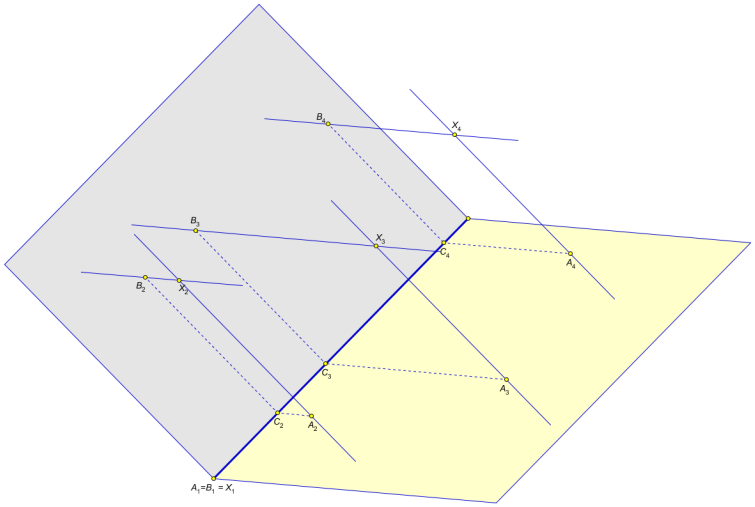


Рис. 11. Восстановление тетраэдра

5. Выводы

Доказанные теоремы дают необходимое и достаточное условие восстановления исходного изображения по плоским проекциям. С алгоритмической точки зрения это позволяет проверять возможность восстановления за $\underline{\underline{O}}(n)$ (где n — число точек). Во многих практических задачах машинного зрения взаимное соответствие точек заранее неизвестно (или известно неточно). Таким образов, вообще говоря, задача требует перебора $n!$ вариантов соответствия, что делает задачу неразрешимой на практике при $n > 20$.

Но для проверки коллинеарности изображений достаточно выбрать по 4 точки на одном и другом изображении. Это дает алгоритмическую сложность $\underline{\underline{O}}(n^9)$, что все еще не позволяет использовать такой алгоритм на практике. Если же заранее известно, что некоторые 4 точки не лежат в одной плоскости, то перебор точек на другом изображении дает C_n^4 вариантов, т.е. проверку условия за $\underline{\underline{O}}(n^5)$, что позволяет решать задачи 1-2 при достаточно больших значениях n ($n \sim 100$).

Также в некоторых задачах распознавания соответствие точек частично известно. Например, известны некоторые дескрипторы точек двух изображений, например, SIFT ([10]), SURF ([11]) или HOG ([12]). Тогда соответствие точек возможно только при условии близости их дескрипторов, что может существенно сократить перебор вариантов.

Список литературы

[1] Александров П.С., Курс аналитической геометрии и линейной алгебры: Учебник. 2-е изд., стер., СПб.: изд-во «Лань», 2009, 512 стр.

[2] Алексеев Д.В., Использование метода В.Н. Козлова в образовательном процессе на кафедре МаТИС // Интеллектуальные системы, т. 17, № 1-4, с. 16-20

[3] Алексеев Д.В., К вопросу о восстановлении тела по плоским проекциям // Интеллектуальные системы. Теория и приложения, т. 18, № 3, с. 47-60, 2014.

[4] Козлов В.Н., Элементы математической теории зрительного восприятия, М.: Изд-во ЦПИ при мех.-мат. ф-те МГУ, 2001, 128 стр.

[5] Козлов В.Н., Доказательность и эвристика при распознавании визуальных образов // Интеллектуальные системы, т. 14, № 1-4, 2010 г., с. 35-52

[6] Kozlov V.N., Image Coding and Recognition and Some Problems of Stereovision // Pattern Recognition and Image Analysis, Vol.7, No. 4, 1997, pp. 448 - 466.

[7] Kozlov V., Mathematical model of reconstructing a three-dimensional image from plane projections // Pattern Recognition and Image Analysis. — 2011. — Vol. 2. — P. 279-282.

[8] Kozlov V., Conclusiveness and heuristics in visual recognition // Pattern Recognition and Image Analysis. — 2014. — Vol. 24, no. 4. — P. 1-7.

[9] Кудрявцев В.Б., Гасанов Э.Э., Подколзин А.С. Введение в теорию интеллектуальных систем. М.: Изд-во ф-та ВМиК МГУ, 2006, 208 с.

[10] Lowe D. G., Object recognition from local scale-invariant features // Proceedings of the International Conference on Computer Vision 2., 1999, pp. 1150-1157.

[11] Bay H., Ess A., Tuytelaars T., Van Gool L. // SURF: Speeded Up Robust Features, Computer Vision and Image Understanding (CVIU), Vol. 110, No. 3, pp. 346-359, 2008

[12] Dalal N., Triggs B., Histograms of oriented gradients for human detection // IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 1, pp.886-893, 2005.

On a 3-dimensional body reconstruction using its planar projections

Alexeev D.V.

This article is devoted to the problem of 3D image reconstruction from its planar projections up to affine and metric equivalence.

The necessary and sufficient conditions for the existence of solution are found.

Keywords: affine transformation, congruent transformation, isometry, image recognition, stereo reconstruction, stereovision.

Персептронный алгоритм и линейное программирование

Носов М.В.

В работе персептронный алгоритм применён для решения задачи линейного программирования с непустой внутренней областью.

Ключевые слова: персептронный алгоритм, линейное программирование.

Пусть задача линейного программирования сформулирована для целых коэффициентов в следующем виде:

$$\begin{cases} a_{11}x_1 + \ldots + a_{1n}x_n + a_{10} \geq 0, \\ \ldots \\ a_{m1}x_1 + \ldots + a_{mn}x_n + a_{m0} \geq 0, \end{cases}$$

$$b_1x_1 + \ldots + b_nx_n \to max.$$

С использованием персептронного алгоритма исследуем случай совместности системы:

$$\begin{cases} a_{11}x_1 + \ldots + a_{1n}x_n + a_{10} > 0, \\ \ldots \\ a_{n1}x_1 + \ldots + a_{mn}x_n + a_{m0} > 0. \end{cases}$$

Введём положительную переменную и получим систему:

$$\begin{cases} a_{11}x_1 + \ldots + a_{1n}x_n + a_{10}x_{n+1} > 0, \\ \ldots \\ a_{n1}x_1 + \ldots + a_{mn}x_n + a_{m0}x_{n+1} > 0, \\ x_{n+1} > 0. \end{cases}$$

Совместность последней системы эквивалентна существованию целых переменных, удовлетворяющей следующей системе:

$$\begin{cases} a_{11}x_1 + \ldots + a_{1n}x_n + a_{10}x_{n+1} \geq 1, \\ \ldots \\ a_{n1}x_1 + \ldots + a_{mn}x_n + a_{m0}x_{n+1} \geq 1, \\ x_{n+1} \geq 1. \end{cases}$$

Из теории линейного программирования следует, что существует квадратный невырожденный минор, для которого совместна система равенств, остальные переменные равны 0, и такой набор даст решение системы неравенств. Определитель минора - целое число, по правилу Крамера и неравенству Адамара следует, что существует решение $(x_1^0, \ldots, x_n^0, x_{n+1}^0)$, удовлетворяющее условию:

$$|x_i^0| = \left|\frac{\triangle_i}{\triangle}\right| \leq |\triangle_i| \leq \left(\max_{pq}|a_{pq}| \cdot \sqrt{min(n,m)+1}\right)^{min(n,m)+1},$$
$$i = 1, \ldots, n, n+1.$$

Значит для точек $A_1 = (a_{11}, \ldots, a_{1n}, a_{10}), \ldots, A_m = (a_{m1}, \ldots, a_{mn}, a_{m0}), A_{m+1} = (0, \ldots, 0, 1)$ существует плоскость, задаваемая уравнением $x_1^0 x_1 + x_n^0 x_n + x_{n+1}^0 x_{n+1} = 0$, от которой точки $A_1, \ldots, A_m, A_{m+1}$ удалены на расстояние δ, которое удовлетворяет следующему условию:

$$\delta \geq \min_i \frac{|a_{i1}x_1^0 + \ldots + a_{in}x_n^0 + a_{i0}x_{n+1}^0|}{\sqrt{(x_1^0)^2 + \ldots + (x_{n+1}^0)^2}} \geq$$
$$\geq \min_i \frac{|a_{i1}\triangle_1 + \ldots + a_{in}\triangle_n + a_{i0}\triangle_{n+1}|}{\sqrt{\triangle_1^2 + \ldots + \triangle_n^2 + \triangle_{n+1}^2}} \geq$$
$$\geq \left(\max_{pq}|a_{pq}| \cdot \sqrt{min(n,m)+1}\right)^{-min(n,m)-1} \cdot (n+1)^{-\frac{1}{2}}.$$

Таким образом, по теореме Новикова вопрос о существовании внутренней точки области решается циклическим поступлением на вход персептронного алгоритма точек $A_1, \ldots, A_m, A_{m+1}$ с начальным нулевым вектором и числом исправлений не более

$$\max_{pq}|a_{pq}|^2 \cdot (n+1)^2 \left(\max_{pq}|a_{pq}|^2 \cdot (min(n,m)+1)\right)^{min(n,m)+1}.$$

При положительном решении вопроса о внутренней точке $(x_1^0, \ldots, x_n^0, x_{n+1}^0)$ системы

$$\begin{cases} a_{11}x_1 + \ldots + a_{1n}x_n + a_{10}x_{n+1} > 0, \\ \ldots \\ a_{n1}x_1 + \ldots + a_{mn}x_n + a_{m0}x_{n+1} > 0, \\ x_{n+1} > 0, \end{cases}$$

определяем величину

$$f^0 = b_1\frac{x_1^0}{x_{n+1}^0} + \ldots + b_n\frac{x_n^0}{x_{n+1}^0}$$

и формируем новую систему неравенств

$$\begin{cases} a_{11}x_1 + \ldots + a_{1n}x_n + a_{10} > 0, \\ \ldots \\ a_{n1}x_1 + \ldots + a_{mn}x_n + a_{m0} > 0, \\ b_1x_1 + \ldots + b_nx_n - f^0 > 0. \end{cases}$$

Вводим новую переменную x_{n+1}, получаем систему

$$\begin{cases} a_{11}x_1 + \ldots + a_{1n}x_n + a_{10}x_{n+1} > 0, \\ \ldots \\ a_{n1}x_1 + \ldots + a_{mn}x_n + a_{m0}x_{n+1} > 0, \\ x_{n+1} > 0, \\ b_1x_1 + \ldots + b_nx_n - f^0x_{n+1} > 0. \end{cases}$$

Проводим персептронную процедуру для точек $A_1, \ldots, A_m, A_{m+1}, A^0_{m+2}$, где $A^0_{m+2} = (b_1, \ldots, b_n, -f^0)$ с начальным вектором $(x^0_1, \ldots, x^0_n, x^0_{n+1})$ и находим вектор $(x^1_1, \ldots, x^1_n, x^1_{n+1})$ и т.д. Теперь после нахождения очередного вектора $(x^j_1, \ldots, x^j_n, x^j_{n+1})$, находим величину

$$f^j = b_1\frac{x^j_1}{x^j_{n+1}} + \ldots + b_n\frac{x^j_n}{x^j_{n+1}}$$

и строим новую систему

$$\begin{cases} a_{11}x_1 + \ldots + a_{1n}x_n + a_{10}x_{n+1} > 0, \\ \ldots \\ a_{n1}x_1 + \ldots + a_{mn}x_n + a_{m0}x_{n+1} > 0, \\ x_{n+1} > 0, \\ b_1x_1 + \ldots + b_nx_n - f^jx_{n+1} > 0. \end{cases}$$

Проводим персептронную процедуру для точек $A_1, \ldots, A_m, A_{m+1}, A^j_{m+2}$, где $A^j_{m+2} = (b_1, \ldots, b_n, -f^j)$ с начальным вектором $(x^j_1, \ldots, x^j_n, x^j_{n+1})$.

Очевидно, что последовательность $\{f^j\}$ возрастающая. Покажем, что если она сходится, то сходится к f^* - максимуму целевой функции исходной задачи. Допустим, что это не так и возрастающая последовательность сходится к $\hat{f}$, причём $\hat{f} < f^*$. Следовательно, в области, задаваемой системой

$$\begin{cases} a_{11}x_1 + \ldots + a_{1n}x_n + a_{10} > 0, \\ \ldots \\ a_{n1}x_1 + \ldots + a_{mn}x_n + a_{m0} > 0, \\ b_1x_1 + \ldots + b_nx_n - \hat{f} > 0, \end{cases}$$

найдётся внутренняя точка $(y_1, \ldots, y_n)$, что $b_1y_1 + \ldots + b_ny_n > \hat{f}$. Возьмём бесконечную последовательность векторов $\{A_1, \ldots, A_{m+1}, A^0_{m+2}, A^2_{m+2}, \ldots\}$, как множество оно ограничено. Возьмём вектор $Y = (y_1, \ldots, y_n, 1)$, для каждого из первых $m+1$ векторов, указанного множества скалярное произведение на вектор Y очевидно строго больше 0, а для остальных векторов имеем

$$(Y, A^j_{m+2}) = b_1y_1 + \ldots + b_ny_n - f^j =$$
$$(b_1y_1 + \ldots + b_ny_n - \hat{f}) + (\hat{f} - f^j) > b_1y_1 + \ldots + b_ny_n - \hat{f} > 0.$$

Таким образом, последовательность векторов удовлетворяет теореме Новикова, следовательно если их подавать на вход персептронного алгоритма, то количество исправлений должно быть ограничено, а это противоречит построению последовательности.Следовательно предположение неверно и возрастающая последовательность $\{f^j\}$ если сходится, то к максимальному значению.

Пример(из Галеев Э.М. "Оптимизация:Теория, примеры,задачи")

$$\begin{cases} -9x_1 - 3x_2 - x_3 + 4 \geq 0, \\ -x_1 - 4x_2 - 5x_3 + 2 \geq 0, \\ x_1 \geq 0, x_2 \geq 0, x_3 \geq 0, \end{cases}$$

$$15x_1 + 27x_2 + 20x_3 \to \max.$$

Точное решение задачи:

$$x_1 = \frac{10}{33} = 0,303030...,$$
$$x_2 = \frac{14}{33} = 0,424242...,$$
$$x_3 = 0,$$
$$\max(15x_1 + 27x_2 + 20x_3) = 16.$$

При решением представленным алгоритмом первая внутренняя точка была найдена через 14 циклов по всем 5 неравенствам. Затем было проведено 15 циклов по параметру j, т.е. 15 раз определялась внутренняя

точка области. Статистика представлена в таблице

Номер цикла	Число исправлений	Текущий максимум
1	161	13.2034
2	670	14.2322
3	948	14.9
4	3480	15.3198
5	7419	15.6032
6	23129	15.7468
7	55051	15.8515
8	170715	15.90558
9	373437	15.9403
10	963769	15.9633
11	2650496	15.9768
12	6316983	15.9856
13	16736385	15.9909
14	41069869	15.9944
15	112748541	15.9965

Решение задачи

$$\begin{aligned} x_1 &= 0.302649, \\ x_2 &= 0.424316, \\ x_3 &= 0.000014817, \\ \max(15x_1 + 27x_2 + 20x_3) &= 15.99656334 \end{aligned}$$

Perceptron algorithm and linear programming
Nosov M.V.

In the work perceptron algorithm is applied to solve linear programming problem with a nonempty interior region.

Keywords: perceptron algorithm, linear programming.

Об одной модели цифровых привычек

А.П. Рыжов, П.А. Новиков

В работе предложена формализация способов использования человеком информационно-коммуникационных технологий на основе анализа логов смартфона. Полученная модель взаимодействия человека с другими людьми и информационными ресурсами (цифровым миром) может быть использована для персонализации такого цифрового окружения и, соответственно, оптимизации такого взаимодействия. Предложен сценарий использования модели для персонализации новостной ленты.

Ключевые слова: персонализация, цифровой след, нечеткая кластеризация.

1. Введение.

Широкое использование информационно-коммуникационных технологий в повседневной жизни открывает новые возможности для государств (e-government, smart city), экономики (цифровая экономика, цифровая трансформация), оптимизации различных функций бизнеса - маркетинга (цифровой маркетинг), продаж (cross-selling, up-selling) и многих других аспектов функционирования бизнеса и жизни человека. Интересующемуся читателю можно порекомендовать ознакомиться с [1], где эти возможности представлены достаточно полно и описаны достаточно убедительно и глубоко.

Основой значительной доли таких возможностей является обычный мобильный телефон. Еще в 2014 году количество активных SIM-карт, используемых в телефонах, смартфонах и планшетах, впервые в истории превысило число живущих на Земле людей [2]. То же самое произошло в 2016 году с мобильными телефонами [3]. Интересная статистика распределения телефонов по странам представлена в [4]: от 11,6 на 100 человек на Кубе до 240,2 на 100 человек в Гонконге.

Приведенные цифры говорят о том, что мобильный телефон является действительно персональным устройством для большинства людей, и стиль его использования во многом характеризует самого человека, которому он принадлежит. Это давно подметили крупные компании -

ритейл, банки, страховые, энергетические и многие другие компании, которые научились использовать это в сегментации клиентов, маркетинге, продажах и во многих других бизнес-процессах. Читатель может без труда найти много таких примеров в интернете, мы не будем их анализировать. Заметим лишь, что в этой парадигме человек - источник данных для всех подобных алгоритмов - является объектом, его классифицируют, сегментируют, а потом еще и навязывают различную рекламу на основе проведённых классификаций и сегментаций. Авторы видят в этом несправедливость. Настоящая работа - первый шаг сделать человека субъектом в этом процессе, когда он на основе своей активности формирует «персональный API» и указывает различным бизнесам что, когда, и в каком формате ему предлагать. Эта концепция глубокой персонализации (deep personalization) нами только осмысливается, она частично описана в [5 - 7]. Мы видим ее большой потенциал как для людей, так и для компаний.

Из вышесказанного вытекает две особенности решаемой задачи. Во-первых, большинство сервисов, решающих задачи персонализации и построения рекомендаций принципиально ограничены обработкой индивидуальных событий (“пользователь купил товар”, “пользователь высоко оценил просмотренный фильм” и т.п.), которые отражают лишь статические характеристики интересов отдельных пользователей (исключение составляет информация об эволюции этих интересов, которая рассматривается, наепример, в [20]), либо обобщённые характеристики больших групп пользователей. Данные же об использовании телефона открывают возможность рассматривать привычки пользователя в привязке ко времени.

Во-вторых, поскольку мы предполагаем, что запрос на персонализацию поступает от пользователя, необходим инструмент, позволяющий описать данные, передаваемые сервисам, в понятных человеку терминах. В таком контексте естественным выбором, позволяющим совместить информационную нагруженность с понятностью, является язык нечёткой логики.

Итак, изучая логи использования мобильного телефона, мы можем что-то сказать о человеке и его привычках работы с информацией. Описываемая ниже модель, несмотря на массу допущений, дает представление о потенциале подобного анализа.

2. Модель.

2.1. Предварительные замечания.

Ниже мы будем пытаться извлечь полезную для решения определённой задачи информацию из логов смартфона определённого человека. Это имеет отношение к анализу цифровых следов человека. В настоящее время нет четкого определения термина «цифровой след», это больше журналистский прием (см., например, [16]). Предлагаемые определения (например, [17] - «Цифровой след (или цифровой отпечаток; англ. digital footprint) — совокупность информации о посещениях и вкладе пользователя во время пребывания в цифровом пространстве. Может включать в себя информацию, полученную из Интернета, мобильного Интернета, веб-пространства и телевидения») вызывают много вопросов, поэтому мы не будем пытаться их анализировать и критиковать, а будем использовать более нейтральный термин - цифровые привычки пользователя.

Под цифровыми привычками будем понимать набор лингвистических переменных [18], определяемых на логах смартфона, достаточный для решения определённой задачи. Например, *Активность* в терминах «активный», «средней активности», «не активный»; *Предпочтения типа контента* в терминах «голос», «картинки», «видео»; *Предпочтения размера контента* в терминах «большой», «средний», «маленький»; *Предпочтения времени* в терминах «утро», «день», «вечер» и т.п. Набор лингвистических переменных определяется задачей (то есть нас не будет волновать абстрактная полнота набора лингвистических переменных, но будет важно понимать, как их строить).

Формально, будем считать, что наш объект описывается конечным набором признаков $A = \{A_1, ..., A_n\}$. Каждому признаку A_q ставится в соответствие U_q множество его «физических» значений и множество $\{a_{1q}, ..., a_{n_q q}\}$ лингвистических значений ($1 \leq q \leq n$) (то есть признак - это лингвистическая переменная). Каждому такому лингвистическому значению a_{wq} ставится в соответствие функция принадлежности $\mu_{a_{wq}}(u_q)$ в универсальном множестве U_q ($1 \leq w \leq n_q$). Множества U_q определяются набором доступных данных. Такой набор данных (dataset) был предоставлен компанией ООО “Технологии обработки данных” и включает в себя запись истории запуска приложений смартфона (логов) 800 пользователей за 4 недели, в формате <идентификатор, транзакция>, где транзакция имеет формат <время начала транзакции, приложение, время окончания транзакции>, где приложение - имя одного из приложений, установленных на смартфоне пользователя, с которым происходила работа в указанное время. Из такого dataset мы можем извлекать различные данные (множества U_q), например, среднее количество звонков, среднее количество использования телефона в сутки / за неделю и над

ними (множествами U_q) строить лингвистические значения, описывающие интересующие нас признаки A_q $(1 \leq q \leq n)$.

2.2. Обозначения.

Обозначим через X_i $(1 \leq i \leq N)$ каналы коммуникации, которые мы умеем измерять. Это могут быть звонки, смс, чаты, мессенджеры, социальные сети и пр. Единицы измерения - время (для звонков), количество символов (для смс и мессенджеров), время нахождения в приложении и т.п. должны быть определены.

Обозначим через Δt некоторый промежуток времени, «естественный» для задачи. Ниже будем считать этот промежуток равным суткам. Разобьём сутки на элементарные единицы - минуты и обозначим через Δt_j $(1 \leq j \leq 1440)$ промежуток времени, соответствующий j минуте.

Обозначим через $v_{i,j}^k$ количество единиц, потраченных k-ым пользователем по каналу i на j-ой минуте $(1 \leq k \leq K)$. Для времени $v_{i,j}^k$ - число из отрезка $[0, 1]$, соответствующее активности в рассматриваемый промежуток, для других единиц измерения (например, символов), это может быть усредненная величина за время набора текста.

Замечание 1. Нами не учитывается направленность коммуникации, только интенсивность. Это упрощение модели позволяет тем не менее решать достаточно важные задачи.

2.3. Разбиение универсального множества.

Допустим, нам будет интересно понимание готовности конкретного пользователя потреблять информацию в определенное время суток. Сколько таких осмысленных интервалов суток существует? Общего ответа нет, поэтому правильно выбрать такое количество «естественных интервалов», которое обеспечит лучшее качество покрытия ими универсального множества $[0, 1440]$. Качество определяется различными показателями - ниже мы будем использовать, аналогично [8], дисбаланс классов и степень нечеткости.

Для этого соберем все активности всех пользователей за определенное время, то есть вычислим для каждого Δt_j величину $\bar{v}_j^k = \sum_{i=1}^{N} \bar{v}_{i,j}^k$, где $\bar{v}_{i,j}^k = 1$, если k-ый пользователь использовал i-ый канал коммуникации во временя Δt_j и $\bar{v}_{i,j}^k = 0$, если у k-го пользователя не было никакой активности в промежуток времени Δt_j. Для разбиения времени суток выберем «типичного» пользователя (например, центр кластера после кластеризации пользователей)и кластеризацию объединения мно-

жеств $\{i|\bar{v}_i > 0\}$ стандартным алгоритмом c-means для разного количества кластеров и посчитаем качество кластеризации.
Замечание 2. Мы неявно считаем, что все каналы коммуникаций имеют одинаковую ценность. Это еще одно упрощение модели.
Замечание 3. 1 секунда оказалась не самым удачным разбиением суток - человек если и делает что-то регулярное, то не с точностью до секунды. Проведенные нами эксперименты показали наилучшие результаты для интервала времени 15 мин., что вполне разумно. Поэтому ниже мы будем использовать именно такую агрегацию времени, то есть $(1 \leq j \leq 95)$ Результаты собраны в таблице 1.

Таблица 1. Характеристики качества кластеризации времени использования

Количество кластеров	Степень нечеткости	Дисбаланс классов
2	0.17368421	0.230382367079
3	0.29473684	0.199661103355
4	0.32631579	0.389432788809
5	0.33684211	0.488659684614

Как видно из таблицы 1, оптимальным с точки зрения дисбаланса классов является разбиение на 3 класса, которые можно интерпретировать как «утро», «рабочее время» и «вечер» (центры кластеров 22.84787803, 54.46499606, 78.82015172). Будем далее использовать именно такое разбиение суток (рис. 1).

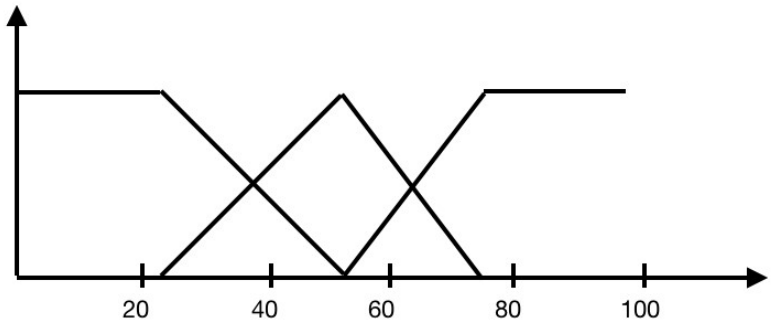


Рис. 1. Кластеризация времени использования для трех кластеров.

Аналогично мы можем определять, например, активность пользователя:

1. Вычисляем среднее значение потраченных единиц за сутки для каждого пользователя.

2. Проводим кластеризацию пользователей.

3. Интерпретируем полученные кластеры.

Если нас интересует описание пользователя в терминах «Активный/ не активный», «Любит читать новости утром/ днем/ вечером», «Любит видео/ текст» и пр., то, используя аналогичные рассуждения и при наличии данных, мы можем построить такие описания. Эти описания (лингвистические переменные, построенные на универсальных множествах, полученных из dataset) будем называть первичными. На основе полученных первичных описаний, мы можем проводить классификацию пользователей с помощью нечеткого классификатора [9] и получать вторичные описания. Заметим, что мы можем делать это с минимальной неопределенностью для большого количества случаев [9].

Такие описания и являются описанием цифровых привычек пользователя. Архитектура системы построения цифровых привычек представлена на рис. 2

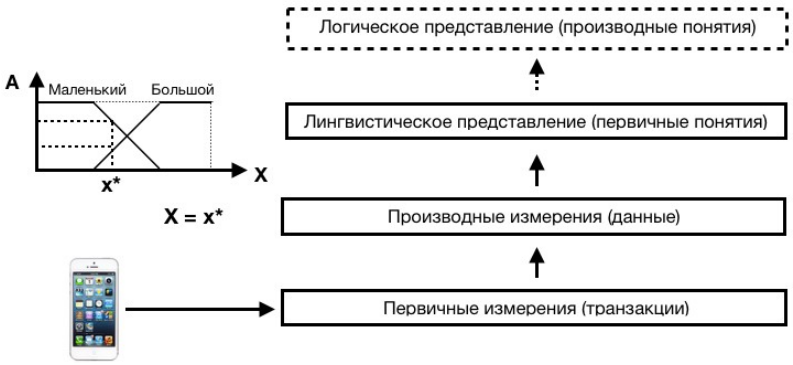


Рис. 2. Архитектура системы построения цифровых привычек

Описания цифровых привычек, или профили, могут быть использованы для персонализации различных сервисов (например, рекомендательных).

3. Сценарий использования.

В качестве возможного примера использования такой формализации информационных привычек пользователя, рассмотрим персонализацию новостной ленты. Эта задача активно обсуждается [10 – 15], и, по мне-

нию многих аналитиков, является трендом развития интернет СМИ и новостных агрегаторов.

Пусть у нас есть поток новостей, и мы хотим упорядочить их так, чтобы пользователь испытывал минимальных дискомфорт при просматривании новостей. Каждая новость представляет собой файл, про который известно:

- Наличие и размер текста
- Наличие и размер картинок
- Наличие и размер видеоклипов
- Тематика (по классификатору агрегатора) и другие параметры (зависит от агрегатора или СМИ)

Локальной задачей является оптимизация времени подачи новости. Давно подмечено, что время существенно влияет на эффективность коммуникации (даже существует термин prime-time). Если мы умеем вычислять какую информацию предпочитает пользователь утром, а какую - вечером, то, соединяя это знание с параметрами новости, мы можем определить насколько она будет соответствовать его привычкам. Например, если пользователь утром предпочитает короткие текстовые файлы (мы можем это видеть из используемых им утром приложений и времени их использования), а вечером - большие видеофайлы (информация так же доступна), то мы можем вычислить насколько конкретная новость соответствует этим критериям (вычисляя степень принадлежности к понятиям «короткий текст» и «большой видео»). Добавляя степень принадлежности текущего времени к понятиям «утро» и «вечер», мы можем вычислить рейтинг новости в данный момент времени для данного пользователя (как t-норму упомянутых выше степеней принадлежности [19]) и на основе этого - позицию новости в списке новостей для данного пользователя. В результате мы будем иметь «персональную газету», которая будет отличаться для пользователей с разными цифровыми привычками, и, более того, в разное время для одного и того же пользователя персональная лента будет разной. Архитектура такого персонализатора новостей представлена на рис. 3.

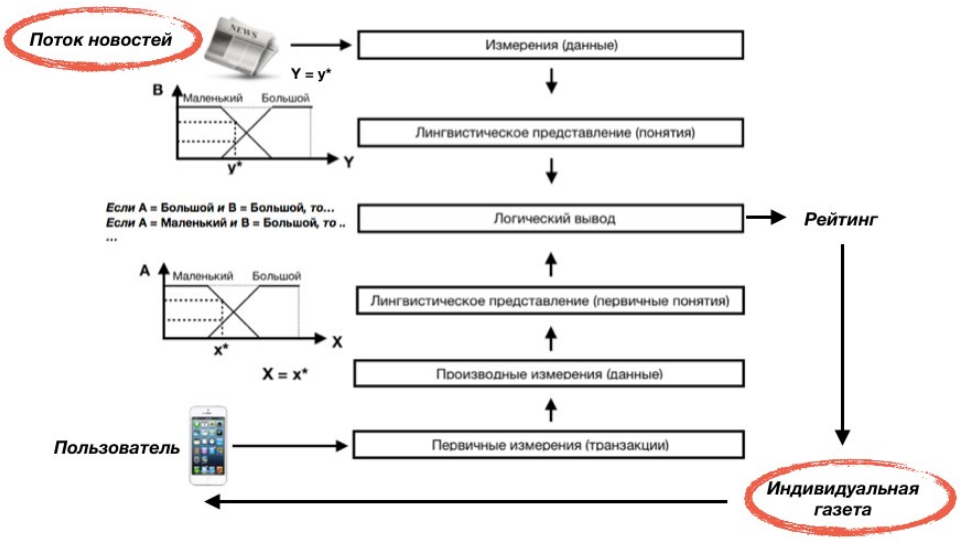


Рис. 3. Архитектура персонализатора новостной ленты.

Подобные дополнения могут сделать более персонализированными (а, следовательно, и более эффективными) любые коммуникации с клиентами, включая рекомендации, рекламные кампании и т.п.

Список Литературы

[1] James Manyika, Michael Chui, Jacques Bughin, Richard Dobbs, Peter Bisson, Alex Marrs. Disruptive technologies: Advances that will transform life, business, and the global economy. McKinsey Global Institute (MGI), May 2013, 176 p. - *http* : *//www.mckinsey.com/insights/business*$_t$*echnology/disruptive*$_t$*echnologies*

[2] Eric Mack. There are now more gadgets on Earth than people - https://www.cnet.com/news/there-are-now-more-gadgets-on-earth-than-people/

[3] Мобильных телефонов – больше, чем людей на планете - http://apps4all.ru/post/10-09-14-mobilnyh-telefonov-bolshe-chem-lyudej-na-planete

[4] Количество мобильных телефонов по странам мира. 30-04-2017. -

http://total-rating.ru/1970-kolichestvo-mobilnyh-telefonov-po-stranam-mira.html

[5] Рыжов А.П. Некоторые задачи оптимизации и персонификации социальных сетей. Saarbrucken, LAP, 2015, 88 с. (ISBN: 978-3-659-68661-0)

[6] Alexander Ryjov. Personalization of Social Networks: Adaptive Semantic Layer Approach. In: Social Networks: A Framework of Computational Intelligence. Witold Pedrycz and Shyi-Ming Chen (Eds.). Springer International Publishing Switzerland 2014, pp. 21-40.

[7] Alexander Ryjov. Towards an optimal task-driven information granulation. In: Information Granularity, Big Data, and Computational Intelligence. Witold Pedrycz and Shyi-Ming Chen (Eds.). Springer International Publishing Switzerland 2015, pp. 191-208.

[8] Рыжов А. П., Журавлев А. Д., Вахов А. Н., Кривцов В. В. Об одном подходе к персонализации обучения в рамках компьютерных обучающих систем. Интеллектуальные Системы. Теория и приложения. Т. 20, Вып. 3, 2016, с. 180-185.

[9] Рыжов А.П. О качестве классификации объектов на основе нечетких правил. Интеллектуальные системы, Т.9, вып. 1-4, Москва, МНЦ КИТ, 2005, с. 253 – 264.

[10] Персонализация в новостях - агрегатор Top Story - http://www.mforum.ru/phones/tests/113536.htm

[11] Новостной агрегатор Gong использует новый алгоритм персонализации -https://hightech.fm/2016/11/25/gong-news

[12] Новостной агрегатор SmartNews собрал $38 млн. - https://hightech.fm/2016/07/08/smartnews

[13] Мифы и факты об алгоритмах формирования новостной ленты Facebook - https://vc.ru/6615-facebook-news-feed

[14] «ВКонтакте» изменит алгоритм формирования ленты новостей - http://www.rbc.ru/rbcfreenews/56f502f39a79478da23507bd

[15] Instagram изменит порядок показа публикаций - http://www.rbc.ru/technology_and_media/16/03/2016/56e914e09a794700ce80e798

[16] Стечкин И. Кому принадлежит наш «цифровой след»? - http://jrnlst.ru/komu-prinadlezhit-nash-cifrovoy-sled

[17] Цифровой след. Материал из Википедии — свободной энциклопедии. https://ru.wikipedia.org/wiki/Цифровой_след

[18] Заде Л.А. Понятие лингвистической переменной и его применение к принятию приблизительных решений. М., Мир, 1976. - 165 с.

[19] Рыжов А.П. Элементы теории нечетких множеств и измерения нечеткости. Москва, Диалог-МГУ, 1998, 116 с.

[20] Li L, Zheng L, Yang F, Li T. Modeling and broadening temporal user interest in personalized news recommendation. Expert Systems with Applications. 2014;41:3168-77.

One model of digital habits

A.P. Ryjov P.A. Novikov

The paper proposes the formalization of ways of using the information technologies on the basis of the analysis of smartphone's logs. The resulting model of human interaction with other people and information resources (digital world) can be used to personalize such a digital environment and, accordingly, to optimize such interaction. A scenario for using the model to personalize the news feed is described.

Keywords: personalization, digital footprint, fuzzy clustering.

Часть 3.
Математические модели

О функциональной сложности двумерной задачи о доминировании

Гасанов Э.Э.

В работе исследуются двумерная задача о доминировании, в которой база данных представляет собой множество точек на плоскости, и надо для произвольной точки плоскости, интерпретируемой как запрос на поиск, найти все точки из базы данных, которые по обоим координатам превосходят запрос. Под функциональной сложностью понимается функция зависимости времени поиска от объема памяти, которую можно выделить под структуры данных. В работе показано, как, используя метод Бентли-Маурера, можно получить алгоритмы с разными соотношениями времени поиска и объема памяти.

Ключевые слова: информационно-графовая модель данных, двумерная задача о доминировании, функциональная сложность, метод сеток.

1. Введение

Для оценки алгоритмов поиска в базах данных обычно используются три характеристики: количество памяти, требуемой под структуры данных, время поиска в худшем случае, и время поиска в среднем. При этом алгоритмы, которые могут обеспечить быстрое время поиска, как правило требуют много памяти для хранения своих структур данных. Поэтому всегда желательно иметь серию различных алгоритмов решения одной и той же задачи поиска с разными соотношениями времени поиска и объема памяти. Тогда разработчик базы данных, зная свои ограничения по памяти, сможет подобрать самый быстрый алгоритм, удовлетворяющий этим требованиям.

Известно, что если задача поиска такая, что на элементах базы данных можно задать линейный порядок, ассоциированный с отношением поиска, то такие задачи можно как правило решать быстро без существенных затрат по памяти, но когда отношение поиска, определяющее,

когда элементы базы данных удовлетворяют запросам на поиск, является отношением частичного порядка, тогда для получения хороших характеристик по времени поиска необходимо затратить существенные объемы памяти.

В работе [1] Бентли и Маурером для решения задачи интервального поиска был предложен многоэтапный метод прямого доступа — метод сеток. Он позволяет снижать порядок требуемой памяти, но при этом возрастает время поиска. Этот метод может быть использован и для других задач информационного поиска.

Двумерная задача о доминировании одна из самых простых задач с отношением поиска, являющимся отношением частичного порядка, и на ней идею метода Бетли-Маурера можно продемонстрировать наиболее просто.

В двумерной задаче о доминировании элементы базы данных и запросы на поиск являются точками евклидовой плоскости. Задача состоит в том, чтобы для произвольного запроса на поиск перечислить все элементы базы данных, которые по обоим координатам не меньше, чем запрос.

Скажем, что алгоритм поиска является $(f(k), g(k), h(k))$ алгоритмом, где k — число элементов в базе данных, если при $k \to \infty$ объем памяти, требуемый алгоритму, время поиска в худшем случае и среднее время поиска асимптотически не превосходят соответственно величин $f(k)$, $g(k)$ и $h(k)$.

В работе используется информационно-графовая модель данных, которая позволяет строго формально определить как задачу поиска, так и алгоритмы ее решения и ее сложностные характеристики. Среди последних работ с использованием информационно-графовой модели можно можно отметить [2, 3, 4, 5, 6, 7, 8, 9, 10, 11].

В работе с помощью метода Бентли-Маурера получена серия алгоритмов решения двумерной задачи о доминировании, которые являются $(q \cdot k^{1+1/q}/2, 4q + \log_2 k, 4q - 1)$ алгоритмами, где q — некоторый целочисленный параметр.

Автор выражает благодарность В.Б.Кудрявцеву и А.С.Подколзину за поддержку в работе.

2. Основные понятия и формулировка результата

В соответствии с [12, 13] опишем информационно-графовую модель данных.

Пусть X — множество запросов с заданным на нем вероятностным пространством $\langle X, \sigma, \mathbf{P}\rangle$, где σ — алгебра подмножеств множества X, $\mathbf{P}$ — вероятностная мера на σ; Y — множество записей (объектов поиска); ρ — бинарное отношение на $X \times Y$, называемое отношением поиска. Пятерку $S = \langle X, Y, \rho, \sigma, \mathbf{P}\rangle$ будем называть *типом*. Тройку $I = \langle X, V, \rho\rangle$, где V — некоторое конечное подмножество множества Y, в дальнейшем называемое *библиотекой*, будем называть задачей информационного поиска (ЗИП) типа S, и будем считать, что ЗИП $I = \langle X, V, \rho\rangle$ содержательно состоит в перечислении для произвольно взятого запроса $x \in X$ всех тех и только тех записей $y \in V$ таких, что $x\rho y$.

Пусть f — одноместный предикат, определенный на X , то есть $f : X \to \{0, 1\}$. Множество $N_f = \{x \in X : f(x) = 1\}$ назовем *характеристическим множеством* предиката f.

Множество $O(y, \rho) = \{x \in X : x\rho y\}$ назовем *тенью* записи $y \in Y$.

Функцию $\chi_{y,\rho} : X \to \{0, 1\}$ такую, что $N_{\chi_{y,\rho}} = O(y, \rho)$ назовем *характеристической функцией* записи y.

Пусть F — множество символов одноместных предикатов, определенных на множестве X, G — множество символов одноместных переключателей, определенных на множестве X. Под переключателем будем понимать функцию, областью значений которой является начальный отрезок натурального ряда. Пару $\mathcal{F} = \langle F, G\rangle$ назовем базовым множеством.

Понятие *информационного графа* (ИГ) над базовым множеством $\mathcal{F} = \langle F, G\rangle$ определяется следующим образом. Берется конечная многополюсная ориентированная сеть. В ней выбирается некоторый полюс, который называется корнем. Остальные полюса называются листьями и им приписываются записи из Y, причем разным листьям могут быть приписаны одинаковые записи. Некоторые вершины сети (в том числе это могут быть и полюса) называются переключательными и им приписываются переключатели из G. Ребра, исходящие из каждой из переключательных вершин, нумеруются подряд, начиная с 1, и называются переключательными ребрами. Ребра, не являющиеся переключательными, называются предикатными и им приписываются предикаты из множества F. Таким образом нагруженную многополюсную ориентированную сеть называем ИГ над базовым множеством $\mathcal{F} = \langle F, G\rangle$.

Функционирование ИГ определяется следующим образом. Скажем, что предикатное ребро проводит запрос $x \in X$, если предикат, приписанный этому ребру, принимает значение 1 на запросе x, Переключательное ребро, которому приписан номер n, проводит запрос $x \in X$, если переключатель, приписанный началу этого ребра, принимает значение n на

запросе x. Ориентированная цепочка ребер проводит запрос $x \in X$, если каждое ребро цепочки проводит запрос x. Запрос $x \in X$ проходит в вершину β ИГ, если существует ориентированная цепочка, ведущая из корня в вершину β, которая проводит запрос x. Запись y, приписанная листу α, попадает в ответ ИГ на запрос $x \in X$, если запрос x проходит в лист α. Ответом ИГ U на запрос x назовем множество записей, попавших в ответ ИГ на запрос x, и обозначим его $\mathcal{J}_U(x)$. Эту функцию $\mathcal{J}_U(x)$ будем считать результатом функционирования ИГ U.

Пусть нам дана ЗИП $I = \langle X, V, \rho \rangle$. Множество $\mathcal{J}_I(x) = \{y \in V : x \rho y\}$ назовем *ответом* ЗИП I на запрос x. Скажем, что ИГ U *решает* ЗИП $I = \langle X, V, \rho \rangle$, если для любого запроса x из X выполнено $\mathcal{J}_U(x) = \mathcal{J}_I(x)$.

Пусть β — некоторая вершина ИГ. Предикат, определенный на множестве запросов, который принимает значение 1 на запросе x, если запрос проходит в вершину β, и 0 — в противном случае, назовем функцией фильтра вершины β и обозначим $\varphi_\beta(x)$.

Объемом $Q(U)$ ИГ U назовем число ребер в графе U.

Сложностью ИГ U на запросе x назовем величину

$$T(U, x) = \sum_{\beta \in \mathcal{R} \setminus \mathcal{P}} \psi_\beta \varphi_\beta(x) + \sum_{\beta \in \mathcal{P}} \varphi_\beta(x),$$

где $\mathcal{R}$ — множество вершинИГ U, $\mathcal{P}$ — множество переключательных вершин ИГ U, ψ_β — количество ребер, исходящих из вершины β. $T(U, x)$ равно числу переключателей и предикатов, вычисленных алгоритмом поиска, соответствующим ИГ U на запросе x, т.е. $T(U, x)$ характеризует время поиска на запросе x.

Сложностью ИГ U в худшем случае назовем величину

$$\widehat{T}(U) = \max_{x \in X} T(U, x)$$

$\widehat{T}(U)$ характеризует время поиска в худшем случае.

Скажем, что базовое множество $\mathcal{F}$ *измеримое*, если каждая функция из $\mathcal{F}$ — измеримая (относительно алгебры σ).

В [13, Лемма 17] доказана справедливость следующей леммы.

Лемма 1. *Если базовое множество $\mathcal{F}$ измеримое, то для любого ИГ U над базовым множеством $\mathcal{F}$ функция $T(U, x)$, как функция от x, является случайной величиной.*

Далее всюду будем предполагать, что базовое множество измеримое, и тем самым $T(U, x)$ будет случайной величиной.

Сложностью ИГ U назовем математическое ожидание величины $T(U,x)$, то есть число

$$T(U) = \mathbf{M}_x T(U,x).$$

Через $\mathcal{U}(I,\mathcal{F})$ обозначим множество всех ИГ над базовым множеством $\mathcal{F}$, решающих ЗИП I.

Согласно [13, Теорема 17] для любой ЗИП I, любого ИГ $U \in \mathcal{U}(I,\mathcal{F})$ и любого запроса $x \in X$ выполнена мощностная нижняя оценка

$$T(U,x) \geq |\mathcal{J}_I(x)|.$$

Поэтому введем следующие сложностные характеристики ИГ:

- $T'(U,x) = T(U,x) - |\mathcal{J}_I(x)|$, которая характеризует время поиска на запросе x без времени перечисления ответа;
- $\widehat{T}'(U) = \max_{x\in X} T'(U,x)$, которая характеризует время поиска в худшем случае без учета времени перечисления ответа:
- $T'(U) = \mathbf{M}_x T'(U,x)$, которая характеризует среднее время поиска без учета времени перечисления ответа.

На множестве $[0,1]^2 \times [0,1]^2$ введем отношение "$\leq$"следующим образом:

$$(x^1,x^2) \leq (y^1,y^2) \Longleftrightarrow x^1 \leq y^1 \& x^2 \leq y^2.$$

Пусть $Y = (0,1] \times [0,1]$ — множество записей; $V = \{y_1,\ldots,y_k : y_i = (y_i^1, y_i^2),\ i = 1,\ldots,k\} \subseteq Y$ — библиотека; $X = (0,1] \times [0,1]$ — множество запросов; отношение поиска ρ задается на $X \times Y$ следующим соотношением:

$$x \rho y \Longleftrightarrow x \leq y,$$

где $x = (x^1,x^2)$, $y = (y^1,y^2)$, т.е. $x\rho y \Longleftrightarrow x \in (0,y^1] \times [0,y^2]$. Тогда ЗИП $I = \langle X, V, \rho\rangle$ называется двумерной задачей о доминировании.

Здесь у квадратов X и Y выброшена левая сторона только в целях удобства дальнейшего изложения.

Определим переключатели и предикаты

$$f_a(x^1,x^2) = \begin{cases} 1, & \text{если } x^2 \leq a \\ 0, & \text{в противном случае} \end{cases},\ a \in [0,1], \tag{1}$$

$$g_a(x^1,x^2) = \begin{cases} 1, & \text{если } x^1 \leq a \\ 2, & \text{в противном случае} \end{cases},\ a \in [0,1], \tag{2}$$

$$g_{\cdot,m}(x^1, x^2) =](x^1) \cdot m[,\ m \in \mathbb{N}, \tag{3}$$

где $\mathbb{N}$ — множество натуральных чисел, $]a[$ — наименьшее целое, не меньшее, чем a.

Пусть

$$F = \{f_a(x^1, x^2) : a \in [0, 1]\}, \tag{4}$$

$$G_1 = \{g_a(x^1, x^2) : a \in [0, 1]\}, \tag{5}$$

$$G_2 = \{g_{\cdot,m}(x^1, x^2) : m \in \mathbb{N}\}, \tag{6}$$

$$\mathcal{F} = \langle F, G_1 \cup G_2 \rangle - \tag{7}$$

базовое множество

Будем писать $A(n) \precsim B(n)$ при $n \to \infty$, если существует такая положительная константа c, что, начиная с некоторого номера n_0, $A(n) \leq c \cdot B(n)$. Если $A \precsim B$ и $B \precsim A$, то будем говорить, что A и B *равны по порядку* при $n \to \infty$ и обозначать $A = O(B)$.

Справедлива следующая теорема.

Теорема 1. *Пусть $I_k = \langle X, V_k, \rho \rangle$ — произвольная последовательность двумерных задач о доминировании, где $|V_k| = k$, $\mathcal{F}$ — базовое множество, определяемое соотношениями (1)–(7). Пусть на X задано вероятностное пространство $\langle X, \sigma, \mathbf{P} \rangle$, и функция $\mathbf{P}$ — равномерная вероятностная мера. Тогда для любого натурального $q \leq \log_2 k$ существует ИГ $U^q_{V_k} \in \mathcal{U}(I_k, \mathcal{F})$, такой что при $k \to \infty$*

$$Q(U^q_{V_k}) = \frac{q}{2} \cdot k^{1+\frac{1}{q}} + O(k),$$

$$\widehat{T}'(U^q_{V_k}) \leq 4q - 1 + \log_2 k,$$

$$T'(U^q_{V_k}) \leq 4q - 1.$$

3. Вспомогательные структуры

Предположим, что библиотека $V = \{y_1, \ldots, y_k : y_i = (y_i^1, y_i^2),\ i = 1, \ldots, k\}$ и запрос $x = (x^1, x^2)$ такие, что мы точно знаем, что $x^1 \leq y_i^1$ для любого $i \in \{1, \ldots, k\}$. Тогда нам для данного запроса x фактически надо решать одномерную задачу о доминировании, т.е. осуществлять проверку только по второй координате.

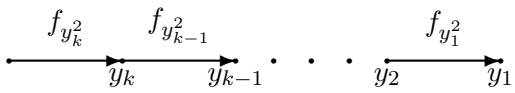


Рис. 1. Граф C_V

Без ограничения общности можно считать, что записи в библиотеке V упорядочены в порядке возрастания вторых координат, т.е. $y_1^2 < y_2^2 < \ldots < y_k^2$.

Согласно [12, Теорема 35] оптимальным алгоритмом для этой задачи будет следующий: просматриваем числа y_i^2, начиная с самого большого и далее в порядке убывания и сравниваем с x^2. Пока $x^2 \leq y_i^2$ запись y_i включаем в ответ, если $x^2 > y_i^2$ или просмотрены все точки библиотеки, то поиск прекращается. Тем самым, время работы этого алгоритма отличается от времени перечисления ответа не более чем на 1.

ИГ, соответствующий этому алгоритму, изображен на рисунке 1. Этот ИГ мы будем обозначать через C_V. Очевидно, что

$$\widehat{T}'(C_V) \leq 1. \tag{8}$$

Далее нам понадобится вспомогательный алгоритм, который будет решать первую задачу о близости, описанную в [12, раздел 2.3]. Эта задача состоит в следующем. Дана библиотека $V = \{y_1, \ldots, y_k : y_i = (y_i^1, y_i^2),\ i = 1, \ldots, k\}$, где записи упорядочены по возрастанию абсцисс, т.е. $y_1^1 < y_2^1 < \ldots < y_k^1$. Для запроса $x = (x^1, x^2)$ В библиотеке V надо найти наименьшую по абсциссе запись, абсцисса которой не меньше, чем x^1.

Дадим неформальное описание алгоритма решения первой задачи о близости.

Разбиваем интервал $(0, 1]$ на k подинтервалов $(\frac{i-1}{k}, \frac{i}{k}]$, $i = 1, 2, \ldots, k$. Разбиваем библиотеку V на части V_i, где в V_i включаются те записи из V, абсциссы которых принадлежат интервалу $(\frac{i-1}{k}, \frac{i}{k}]$.

Пусть на вход алгоритма поступает запрос $x = (x^1, x^2)$. Вычисляем значение переключателя $g_{\cdot,k}(x)$. Пусть $g_{\cdot,k}(x) = i$. Это означает, что $x^1 \in (\frac{i-1}{k}, \frac{i}{k}]$. Если V_i — непустое множество, то бинарным поиском находим в V_i запись, абсцисса которой наименьшая, но не меньшая, чем x^1. Если x^1 больше любой абсциссы записей из V_i или если V_i — пустое множество, то искомая запись есть первая запись в первом непустом множестве V_j таком, что $j > i$.

Согласно [12, Теорема 19] описанное решение имеет константное в среднем время поиска и линейные затраты памяти.

Дадим формальное ИГ D_V, который будет решать первую задачу о близости для множества V и соответствовать описанному выше алгоритму.

Возьмем вершину β_0 и объявим ее корнем графа. Выпустим из β_0 k ребер, припишем им числа от 1 до k, объявим β_0 точкой переключения и припишем ей переключатель $g_{\cdot,k}$.

Пусть $B_i = \{y_j^1, y_j^2) \in V : y_j^1 \in (\frac{i-1}{k}, \frac{i}{k}]$, $l_i = |B_i|$, $i = 1, \ldots, k$,

Конец ребра с номером i обозначим β_i.

Для всех i таких, что $B_i \neq \emptyset$, выполним следующую процедуру. Выпустим из вершины β_i бинарное сбалансированное дерево D^{B_i} с $l_i + 1$ концевыми вершинами и высоты $]\log_2(l_i + 1)[$.

Здесь и далее *бинарное дерево* — это ориентированное дерево, в каждую вершину которого, кроме корня, входит одно ребро, и из каждой вершины которого, кроме концевых вершин, исходит два ребра; *корень* — это вершина, в которую не входит ни одно ребро; *концевые вершины* — это вершины из которых не исходит ни одно ребро; *высота вершины* — это длина пути от корня к данной вершине; *сбалансированное дерево* — это дерево, высота концевых вершин которого отличается не более чем на 1; *высота дерева* — это максимальная высота концевых вершин дерева.

Сопоставим концевым вершинам этого дерева D^{B_i}, кроме последней (самой правой), слева направо в порядке возрастания элементы из B_i.

Здесь и далее будем представлять дерево в виде некоторой его укладки на плоскости и понимать направления "вправо "влево" как направления на этой плоскости относительно данной укладки.

Для произвольной внутренней вершины β дерева D^{B_i} обозначим через V_β множество чисел, соответствующих концевым вершинам, достижимым из β. Здесь и далее вершина α *достижима* из вершины β, если существует ориентированный путь ведущий из β в α.

Для произвольной внутренней вершины β дерева D^{B_i} обозначим

$$b_\beta = \max_{b \in V_{\beta'}} b,$$

где β' — конец левого ребра, исходящего из β.

Объявим все внутренние вершины дерева D^{B_i} вершинами переключения и для каждой внутренней вершины β левому ребру, из нее выходящему, припишем 1, а правому — 2, а самой вершине припишем переключатель $g_{b_\beta}(x)$.

Пусть $i \in \{1, \ldots, k\}$. Обозначим $j(i)$ такой номер, что $j(i) > i$, $l_{j(i)} > 0$ и не существует j' такого, что $l_{j'} > 0$, $j' > i$ и $j' < j(i)$, то есть $j(i)$ — индекс ближайшего сверху непустого множества B_j. Если такого множества нет, то $j(i) = 0$.

Теперь для каждого дерева D^{B_i} самую правую концевую вершину дерева отождествим с самым левым листом дерева $D^{B_{j(i)}}$, если $j(i) \neq 0$.

Для каждого такого i, что $l_i = 0$, вершину β_i отождествим с самым левым листом дерева $D^{B_{j(i)}}$, если $j(i) \neq 0$.

Полученный граф и будет ИГ D_V.

Обозначим через α_i концевую вершину, которой приписано число y_i^1, $i = 1, \ldots, k$, а через α_{k+1} самую правую концевую вершину в самом правом дереве D^{B_i}. Легко проверить, что функция проводимости между корнем и вершиной α_i $(i = 1, 2, \ldots, k+1)$ имеет вид:

$$h_i(x^1, x^2) = \begin{cases} 1, & \text{если } y_{i-1}^1 < x^1 \leq y_i^1, \\ 0, & \text{в противном случае,} \end{cases}$$

где $y_0^1 = 0$, $y_{k+1}^1 = 1$.

Подсчитаем объем графа U_m^1.

Поскольку каждое из D^{B_i} есть дерево, в котором полустепень исхода любой вершины равна 2, то в D^{B_i} будет ровно $2 \cdot (l_i + 1) - 2 = 2\, l_i$ ребер. Отсюда следует, что

$$Q(D_V) = k + 2 \sum_{i=1}^{k} l_i = 3\, k. \tag{9}$$

Согласно [12, Теорема 19]

$$\widehat{T}(D_V) \leq 1 +] \log_2(k+1) [, \tag{10}$$

$$T(D_V) \leq 2. \tag{11}$$

4. Доказательство теоремы 1

Доказывать теорему 1 будем индукцией по параметру q.

Базис индукции: $q = 1$.

Пусть $V_k = \{y_1, \ldots, y_k : y_i = (y_i^1, y_i^2),\ i = 1, \ldots, k\}$, причем записи упорядочены по возрастанию абсцисс, т.е. $y_1^1 < y_2^1 < \ldots < y_k^1$.

Обозначим $Z_i^1 = \{y_j : j = i, i+1, \ldots, y_k\}$, $i = 1, 2, \ldots, k$.

Построим ИГ $U_{V_k}^1$.

Возьмем описанный ранее граф D_{V_k}. Корень этого графа объявим корнем графа $U^1_{V_k}$.

Как и ранее, обозначим через α_i концевую вершину ИГ D_{V_k}, которой приписано число y^1_i, $i = 1, \ldots, k$. Для каждого $i = 1, \ldots, k$ из вершины α_i выпустим цепь $C_{Z^1_i}$, которую мы описывали в предыдущем разделе и пример которой приведен на рисунке 1.

Полученный ИГ и будет графом $U^1_{V_k}$.

Убедимся, что этот граф решает задачу $I_k = \langle X, V_k, \rho \rangle$.

В самом деле, если запрос $x = (x^1, x_2)$ попал в вершину α_i, то это означает, что $y^1_{i-1} < x^1 \le y^1_i$. Что в свою очередь означает, что записи из $V_k \setminus Z^1_i$ не удовлетворяют запросу x, а все записи из Z^1_i подходят по абсциссе. А именно в этих условиях цепь $C_{Z^1_i}$ позволяет решить задачу I_k.

Подсчитаем сложностные характеристики ИГ $U^1_{V_k}$, учитывая соотношения (8) - (11) и, что запрос x может выйти только к одной из вершин α_i:

$$Q(U^1_{V_k}) = Q(D_{V_k}) + \sum_{i=1}^{k} Q(C_{Z^1_i}) = 3\,k + \sum_{i=1}^{k} i = k^2/2 + O(k),$$

$$\widehat{T}'(U^1_{V_k}) \le \widehat{T}(D_{V_k}) + 1 \le 1 +]\log_2(k+1)[+ 1 \le 3 + \log_2 k,$$

$$T'(U^1_{V_k}) \le T(D_{V_k}) + 1 \le 3.$$

Тем самым базис индукции доказан.

Индуктивный переход. Предположим мы построили ИГ $U^{q-1}_{V_k}$, решающий ЗИП I_k и имеющий требуемые в условиях теоремы характеристики.

Чтобы в тяжелых технических выкладках не потонула идея метода Бентли-Маурера, мы в дальнейших рассуждениях допустим некоторую вольность и будем рассматривать, вообще говоря, натуральные числа как вещественные.

Пусть

$$p(q,i) = i \cdot k^{1-\frac{1}{q}},\ i = 0, 1, \ldots, k^{\frac{1}{q}},$$

$$R^q = \{y_{p(q,i)},\ i = 1, \ldots, k^{\frac{1}{q}}\},$$

$$W^q_i = \{y_j : p(q, i-1) + 1 \le j \le p(q,i),\ i = 1, \ldots, k^{\frac{1}{q}}\},$$

$$Z^q_i = \cup_{j=i+1}^{k^{\frac{1}{q}}} W^q_j,\ i = 1, \ldots, k^{\frac{1}{q}} - 1.$$

Отметим, что $|R^q| = k^{\frac{1}{q}}$, $|W^q_i| = k^{1-\frac{1}{q}}$, $|Z^q_i| = (k^{\frac{1}{q}} - 1 - i) \cdot k^{1-\frac{1}{q}}$.

Построим ИГ $U^q_{V_k}$.

Возьмем описанный ранее граф D_{R^q}. Корень этого графа объявим корнем графа $U^q_{V_k}$.

Обозначим через α_i концевую вершину ИГ D_{R^q}, которой приписано число $y^1_{p(q,i)}$, $i = 1, \ldots, k^{\frac{1}{q}}$. Для каждого $i = 1, \ldots, k^{\frac{1}{q}}$ из вершины α_i выпустим предикатное ребро, которому припишем предикат тождественная 1, а из конца этого ребра выпустим ИГ $U^{q-1}_{W^q_i}$. Кроме того для каждого $i = 1, \ldots, k^{\frac{1}{q}} - 1$ из вершины α_i выпустим цепь $C_{Z^q_i}$, которую мы описывали в предыдущем разделе и пример которой приведен на рисунке 1.

Полученный ИГ и будет графом $U^q_{V_k}$.

Убедимся, что этот граф решает задачу I_k.

В самом деле, если запрос $x = (x^1, x_2)$ попал в вершину α_i, то это означает, что $y^1_{p(q,i-1)} < x^1 \leq y^1_{p(q,i)}$. Что в свою очередь означает, что записи из $\cup_{j=1}^{i-1} W^q_j$ не удовлетворяют запросу x, а все записи из Z^1_i подходят по абсциссе, поэтому из α_i выпускается цепь $C_{Z^q_i}$. А для записей из W^q_i нам надо решать полноценную задачу о доминировании, что и делается с помощью ИГ $U^{q-1}_{W^q_i}$.

Подсчитаем сложностные характеристики ИГ $U^1_{V_k}$, учитывая соотношения (8) - (11) и предположение индукции:

$$Q(U^q_{V_k}) = Q(D_{R^q}) + \sum_{i=1}^{k^{\frac{1}{q}}} Q(U^{q-1}_{W^q_i}) + \sum_{i=1}^{k^{\frac{1}{q}}-1} Q(C_{Z^1_i}) =$$

$$= 3\,k^{\frac{1}{q}} + k^{\frac{1}{q}} \cdot \Big(\frac{q-1}{2} k^{\frac{q-1}{q} \cdot \frac{q}{q-1}} + O(k^{\frac{q-1}{q}})\Big) + k^{1-\frac{1}{q}} \cdot \sum_{i=1}^{k^{\frac{1}{q}}-1} i \leq$$

$$\leq \frac{q-1}{2} k^{1+\frac{1}{q}} + O(k) + \frac{1}{2} k^{1-\frac{1}{q}} \cdot k^{\frac{2}{q}} = \frac{q}{2} k^{1+\frac{1}{q}} + O(k),$$

$$\widehat{T}'(U^q_{V_k}) \leq \widehat{T}(D_{R^q}) + 1 + \widehat{T}'(U^{q-1}_{V_k}) \leq$$

$$\leq 4 + \frac{1}{q} \log_2 k + 4(q-1) - 1 + \frac{q-1}{q} \log_2 k = 4q - 1 + \log_2 k,$$

$$T'(U^q_{V_k}) \leq T(D_{D_{R^q}}) + 2 + T'(U^{q-1}_{V_k}) \leq 4 + 4(q-1) - 1 = 4q - 1.$$

Тем самым, теорема 1 доказана.

Список литературы

[1] Bentley J.L., Maurer H.A. Efficient worst-case data structures for range searching. *Acta Informatica* (1980), **13** 155–168.

[2] Перпер Е.М. Порядок сложности задачи поиска в множестве слов вхождений подслова // Интеллектуальные системы. — 2014. — Т. 19, вып. 1. — С. 99–116.

[3] Плетнев А.А. Информационно-графовая модель динамических баз данных и ее применение// Интеллектуальные системы. — 2014. Т. 18, Вып. 1. — С. 111-140.

[4] Е. М. Перпер. Нижние оценки временной и объемной сложности задачи поиска подслова // Дискретная математика, 2014, том 26:2, 58–70.

[5] Гасанов Э.Э., Ефремов Д.В. Фоновый алгоритм решения двумерной задачи о доминировании // Интеллектуальные системы. — 2014. — Т. 18, вып. 3. — С. 133–158.

[6] Шуткин Ю.С. Моделирование схемных управляющих систем // Интеллектуальные системы. — 2014. — Т. 18, вып. 3. — С. 253–261.

[7] Плетнев А.А. Динамическая база данных, допускающая параллельную обработку произвольных потоков запросов // Интеллектуальные системы. — 2015. Т. 19, Вып. 1. — С. 117–145.

[8] Плетнев А.А. Логарифмическая по сложности параллельная обработка автоматами произвольных потоков запросов в динамической базе данных // Интеллектуальные системы. — 2015. Т. 19, Вып. 1. — С. 171–213.

[9] Плетнев А.А. Нижняя оценка на область видимости автомата, обрабатывающего произвольный поток запросов к динамической базе данных // Интеллектуальные системы. — 2015. Т. 19, Вып. 4. — С. 117–151.

[10] Плетнев А.А. Минимально возможный по степени ветвления информационный граф с радиусом видимости один, обрабатывающий произвольный поток запросов к динамической базе данных // Интеллектуальные системы. — 2016. Т. 20, Вып. 1. — С. 223–254.

[11] Гасанов Э.Э., Плетнев А.А. Моделирование динамических баз данных // Интеллектуальные системы. — 2016. Т. 20, Вып. 3. — С. 146–150.

[12] Гасанов Э. Э, Кудрявцев В. Б. Интеллектуальные системы. Теория хранения и поиска информации. — М.: Юрайт, 2016.

[13] Кудрявцев В. Б, Гасанов Э. Э, Подколзин А. С. Основы теории интеллектуальных систем. — М.: МАКС Пресс, 2016.

On the functional complexity of the two-dimensional domination problem

Gasanov E.E.

In this paper we study the two-dimensional domination problem in which the database is a set of points on the plane, and it is necessary for an arbitrary point of the plane, interpreted as a search query, find all the points from the database, which exceed the query by both coordinates. Functional complexity is the function of the dependence of search time on the memory size. The paper shows how, using the Bentley-Maurer method, we can obtain algorithms with different search time and memory size ratio.

Keywords: information-graph data model, two-dimensional domination problem, functional complexity, grid method.

О максимальном накрытии начала натурального ряда с ограничениями

Дергач Пётр Сергеевич

В статье рассматривается следующая задача: необходимо определить, какое максимальное по длине начало натурального ряда можно накрыть арифметическими прогрессиями, не накрыв при этом весь ряд. При этом может вводиться ряд ограничений на начало и разность(шаг) этих прогрессий, а также на их общее количество. В зависимости от того, какие из ограничений имеют место, возникает класс различных задач, часть из которых успешно решается в данной статье. Самыми интересными случаями оказываются ограничения типа "начало+шаг", "количество".

Ключевые слова: натуральный ряд, арифметическая прогрессия, максимальное накрытие.

Введение

В статье приводятся точные оценки на решение задачи с ограничениями типа "начало", "шаг", "непересечение", "количество". Тем самым окончательно закрывается вопрос о решении поставленной проблемы с единственным типом ограничения. Для задачи с парой ограничений возникающие при этом выкладки становится на порядок нетривиальнее. Тем не менее, для ограничений типа "начало+шаг" удается найти и доказать соответствующую точную оценку.О решении похожих проблем можно прочитать в статьях [1-8]. О других интересных аспектах исследований автора и других ученых в смежных областях к тематике данной работы можно прочитать в [9-20].

Основные определения

Множество натуральных чисел обозначаем через $\mathbb{N}$.

Множество целых чисел обозначаем через $\mathbb{Z}$.

Множество целых неотрицательных чисел обозначаем через $\mathbb{N}_0$.

Множество действительных чисел обозначаем через $\mathbb{R}$.

Для произвольного $n \in \mathbb{N}$ множество натуральных чисел от 1 до n обозначаем через E_n.

Через $\mathbf{P}$ обозначаем множество простых чисел.

Для произвольного $n \in \mathbb{N}$ Через $\mathbf{P}_n$ обозначаем множество $\mathbf{P} \cap E_n$.

Пусть $a, b \in \mathbb{N}$. Тогда *арифметической прогрессией с началом a и шагом b* называется множество

$$\{a + ib \mid i \in \mathbb{N}_0\}.$$

Для краткости обозначаем эту прогрессию через (a, b).

Множество всех арифметических прогрессий обозначаем через $\mathbb{P}$.

Для произвольного $n \in \mathbb{N}$ множество всех арифметических прогрессий, начало которых не превосходит n, обозначаем через $\mathbb{P}_1(n)$.

Множество всех арифметических прогрессий, шаг которых не превосходит n, обозначаем через $\mathbb{P}_2(n)$.

Через f_1 обозначаем максимальное $m \in \mathbb{N}$ такое, что множество E_m можно накрыть объединением любого конечного количества попарно непересекающихся элементов из $\mathbb{P}$, не накрыв при этом весь натуральный ряд $\mathbb{N}$.

Через $f_2(n)$ обозначаем максимальное $m \in \mathbb{N}$ такое, что множество E_m можно накрыть объединением любого конечного количества элементов из $\mathbb{P}_1(n)$, не накрыв при этом весь натуральный ряд $\mathbb{N}$.

Через $f_3(n)$ обозначаем максимальное $m \in \mathbb{N}$ такое, что множество E_m можно накрыть объединением любого конечного количества элементов из $\mathbb{P}_2(n)$, не накрыв при этом весь натуральный ряд $\mathbb{N}$.

Через $f_4(n)$ обозначаем максимальное $m \in \mathbb{N}$ такое, что множество E_m можно накрыть объединением не более чем n элементов из $\mathbb{P}$, не накрыв при этом весь натуральный ряд $\mathbb{N}$. При этом никаких ограничений на попарное пересечение арифметических прогрессий, их начало и шаг не накладывается.

Через $f_{2,3}(n)$ обозначаем максимальное $m \in \mathbb{N}$ такое, что множество E_m можно накрыть объединением любого конечного количества элементов из $\mathbb{P}_1(n) \cap \mathbb{P}_2(n)$, не накрыв при этом весь натуральный ряд

$\mathbb{N}$. При этом никаких ограничений на попарное пересечение арифметических прогрессий или их количество не накладывается.

Любую арифметическую прогрессию (a, b) можно представить в виде объединения

$$(a, b) = (a, bp) \cup (a + b, bp) \cup \ldots \cup (a + b(p - 1), bp).$$

Множество $(a+b(i-1), bp)$ называем *i-ым слоем* прогрессии (a, b) *с шагом* bp.

Называем подмножество S натурального ряда *плотно-упакованным с шагом p* и *глубиной k*, если его можно получить отбрасыванием из множества $\mathbb{N}$ нескольких(хотя бы одного) слоев, взятых в каком-нибудь фиксированном слое прогрессии $(1, 1)$ с шагом p^{k-1}.

Для произвольного $n \in \mathbb{N}$ обозначаем

$$t(n) := \max\{a_1 \cdot a_2 \cdot \ldots \cdot a_k \mid k, a_1, \ldots, a_k \in \mathbb{N}, a_1 + \ldots + a_k = n\}.$$

Для любого $x \in \mathbb{R}$ через $[x]$ обозначаем наибольшее целое число $a \in \mathbb{Z}$, для которого $a \leq x$.

Теорема 1. *Пусть $n, m \in \mathbb{N}$, $m > 1$. Тогда*

$$f_1 = \infty,\ f_2(1) = 1,\ f_2(m) = \infty,\ f_3(n) = HOK(1, 2, \ldots, n) - 1.$$

Теорема 2. *Пусть $n \in \mathbb{N}$. Тогда* $f_{2,3}(n) = \prod\limits_{p \in \mathbf{P}_n} p^{[\log_p n]}$.

Теорема 3. *Пусть $n \in \mathbb{N}$. Тогда $f_4(n) = 2^n - 1$.*

Доказательство вспомогательных утверждений

Лемма 1. Критерий пересечения. *Для любых* $a, c \in \mathbb{N}_0$ *и* $b, d \in \mathbb{N}$ *верно*

$$(a, b) \cap (c, d) \neq \emptyset \Longleftrightarrow a \equiv c \ (mod\ НОД(b, d)).$$

Доказательство леммы см. в [1].

Лемма 2. О примарности. *Пусть* $k, n \in \mathbb{N}$ *и для* $1 \leq i \leq k$ *выполнено* $a_i, b_i \in \mathbb{N}$, $b_i > 1$. *Пусть, кроме того,*

$$E_n \subset \bigcup_{i=1}^{k} (a_i, b_i) \neq \mathbb{N}$$

и $b_1 = p_1^{a_1} p_2^{a_2} \dots p_s^{a_s}$ *— разложение числа* b_1 *на простые множители. Тогда существует такое* $t \in E_s$, *для которого*

$$E_n \subset \bigcup_{i=2}^{k} (a_i, b_i) \cup (a_1, p_t^{a_t}) \neq \mathbb{N}.$$

Доказательство.

Предположим, что утверждение неверно и для любого $t \in E_s$ не выполнено условие

$$E_n \subset \bigcup_{i=2}^{k} (a_i, b_i) \cup (a_1, p_t^{a_t}) \neq \mathbb{N}. \tag{1}$$

Очевидно, что для любого $t \in E_s$ выполнено

$$E_n \subset \bigcup_{i=1}^{k} (a_i, b_i) \subset \bigcup_{i=2}^{k} (a_i, b_i) \cup (a_1, p_t^{a_t}). \tag{2}$$

Из невыполнения условия (1) отсюда тогда следует, что для каждого $t \in E_s$ верно

$$\bigcup_{i=1}^{k} (a_i, b_i) \subset \bigcup_{i=2}^{k} (a_i, b_i) \cup (a_1, p_t^{a_t}) = \mathbb{N}. \tag{3}$$

Обозначим через S множество $\mathbb{N} \setminus \bigcup\limits_{i=1}^{k} (a_i, b_i)$. По условию оно непусто. Однако, из (3) получаем, что при всех $t \in E_s$ выполнено

$$S \subset (a_1, p_t^{a_t}). \tag{4}$$

Значит

$$S \subset \bigcap_{i=1}^{s} (a_1, p_t^{a_t}). \tag{5}$$

Из взаимной простоты чисел $p_t^{a_t}$, $t \in E_s$ получаем из (5), что

$$\mathbb{N} \setminus \bigcup_{i=1}^{k} (a_i, b_i) = S \subset (a_1, \prod_{i=1}^{s} p_t^{a_t}) = (a_1, b_1).$$

Полученное противоречие очевидно завершает доказательство леммы.

■

Лемма 3. О плотной упаковке. *Пусть p — простое число, дано $L = (a_i, b_i)_{i=1}^{n}$ — семейство попарно непересекающихся прогрессий с шагами, равными степеням числа p и пусть это семейство не накрывает все $\mathbb{N}$. Тогда его можно накрыть плотно-упакованным множеством с шагом p и глубиной $]\frac{n}{p-1}[$, использовав при этом не более n прогрессий.*

Доказательство.

Суть доказательства очень проста. Обозначим через k число $]\frac{n}{p-1}[$. Рассмотрим слои прогрессии $(1, 1)$ с шагом p. Возможны два случая: или хотя бы один из этих слоев не пересекается с элементами из L, или же L хотя бы частично накрывает каждый слой прогрессии $(1, 1)$ с шагом p. Ясно, что каждая прогрессия из L будет находиться целиком внутри какого-то одного слоя. В первом случае нужно заменить все элементы из L, находящиеся внутри своего слоя на сам этот слой. Во втором случае этого сделать нельзя. Однако, можно заменить таким образом все слои, кроме одного. А уже к этому слою применить такое же рассуждение, которое выше было использовано для прогрессии $(1, 1)$. Всего на каждом шаге итерации этого процесса будет взято $p - 1$ слоев. И лишь на последнем шаге может потребоваться взять только некоторую часть всех слоев. Очевидно, что при такой замене количество использованных прогрессий не возрастает и, значит, их будет не больше n. И на последнем шаге итерации мы будем брать слои с шагом p^k. Значит образованное нами множество является плотно упакованным с шагом p и глубиной $k =]\frac{n}{p-1}[$.

■

Лемма 4. Про t(n). *Для любого $k \in \mathbb{N}$ верны следующие равенства:*

$$t(3k) = 3^k,\ t(3k+1) = 4 \cdot 3^{k-1},\ t(3k+2) = 2 \cdot 3^k.$$

Доказательство.

Пусть для фиксированного $n \in \mathbb{N}, n \geq 3$ числа $a_1, \ldots, a_k \in \mathbb{N}$ доставляют максимальное значение для $a_1 \cdot \ldots \cdot a_k$ с ограничением

$$a_1 + \ldots + a_k = n.$$

Тогда все $a_i > 1$, так как в противном случае пару $1, a_i$ можно было бы, не поменяв сумму и увеличив произведение, заменить на a_i+1. Аналогично, все $a_i < 5$, так как при $a_i \geq 5$ было бы верно, что

$$2(a_i - 2) = 2a_i - 4 = a_i + (a_i - 4) > a_i$$

и замена числа a_i на пару чисел 2 и a_i не изменила бы сумму, но увеличила произведение. Также можно считать, что $a_i \neq 4$, так как 4 можно, не изменив суммы и произведения, заменить на 2 и 2. Таким образом, все a_i равны или двум, или трем. Больше двух двоек среди a_i быть не может, так как иначе мы могли бы заменить 3 двойки на 2 тройки и, не изменив сумму, увеличили бы произведение. После сказанного утверждение леммы становится очевидным.

■

Следствие из леммы 4. Для любого $n \in \mathbb{N}$ верно, что $t(n) \leq 3^{\frac{n}{3}}$.

Доказательство.

Для $n = 1, 2$ утверждение проверяется непосредственно. При $n > 2$ утверждение тривиально вытекает из верных неравенств

$$4 \cdot 3^{k-1} \leq 3^{k+\frac{1}{3}},\ 2 \cdot 3^k \leq 3^{k+\frac{2}{3}}.$$

■

Лемма 5. Индуктивный переход. *Пусть для всех $m \in E_n$ верно, что $f_4(m) = 2^m - 1$. И пусть дано $L = (a_i, b_i)_{i=1}^{n+1}$ — семейство из $n+1$ прогрессий, содержащее плотно-упакованное множество с шагом $p \in \mathbf{P}$ и глубиной больше 1. Тогда L не может накрывать $E_{2^{n+1}}$, не накрыв при этом весь ряд $\mathbb{N}$.*

Доказательство.

Будем доказывать утверждение от противного. По определению плотно упакованного множества в L есть $p-1$ различных слоев с шагом p. Удалим из L эти слои и обозначим получившееся семейство через L'. В оставшемся слое (x,p) будет, как минимум, $\left[\frac{2^{n+1}}{p}\right]$ чисел из $E_{2^{n+1}}$. Так как L не накрывало $\mathbb{N}$, то и L' не накрывает (x,p). Заменим каждый элемент (a,b) из L' на $(a,b)\cap(x,p)$. Полученное семейство обозначим через L''. Можно считать, что в нем столько же элементов, сколько и в L', иначе какой-то элемент $(a,b)\in L'$ не пересекался бы с (x,p) и его можно было бы изначально выкинуть из L. Значит в L'' всего $n+1-(p-1)$ элементов. Так как $n+1-(p-1)\in E_n$, то по условию

$$f(n+1-(p-1)) = 2^{n+1-(p-1)} - 1. \tag{6}$$

Однако, все элементы из L'' лежат в (x,p), но L'' не накрывает его целиком. Поэтому

$$f(n+1-(p-1)) \geq \left[\frac{2^{n+1}}{p}\right]. \tag{7}$$

Из (6) и (7) получаем, что

$$2^{n+1-(p-1)} - 1 \geq \left[\frac{2^{n+1}}{p}\right]. \tag{8}$$

При $p=2$ это условие, очевидно, нарушается. Если же $p>2$, то из (8) получаем $\frac{2^{n+1}}{2^{p-1}} \geq \frac{2^{n+1}}{p}$, а значит $p \geq 2^{p-1}$, что, конечно же, неверно.

■

Доказательство основных утверждений

Теорема 1. *Пусть* $n, m \in \mathbb{N}$, $m > 1$. *Тогда*

$$f_1 = \infty,\ f_2(1) = 1,\ f_2(m) = \infty,\ f_3(n) = HOK(1, 2, \ldots, n) - 1.$$

Доказательство.

Докажем, что $f_1 = \infty$. Пусть $k \in \mathbb{N}$. Рассмотрим следующее множество:

$$P_1 = (1, 2) \sqcup (2, 4) \sqcup (4, 8) \sqcup \ldots \sqcup (2^{k-1}, 2^k). \tag{9}$$

Ясно, что в (9) все прогрессии попарно не пересекаются и в совокупности накрывают все числа, не делящиеся нацело на 2^k. Значит $f_1 \geq 2^k - 1$, т.е., из произвольности k, $f_1 = \infty$.

Докажем от противного, что $f_2(1) = 1$. Пусть $f_2(1) > 1$. Тогда существует такое конечное семейство арифметических прогрессий с началом в 1, объединение которых содержит 1 и 2. Но это, в свою очередь, означает, что в какой-то из прогрессий этого семейства есть и 1, и 2. Но тогда эта прогрессия будет накрывать весь ряд. Полученное противоречие доказывает, что $f_2(1) = 1$.

Докажем, что $f_2(m) = \infty$. Для этого достаточно доказать, что

$$f_2(2) = \infty.$$

Рассмотрим произвольное $l \in \mathbb{N}$. Так как простых чисел бесконечно много, то существует простое число p, которое больше l. Посмотрим на следующее множество:

$$P_2 = (1, 2) \cup (1, 3) \cup (1, 4) \cup \ldots \cup (1, l) \cup (2, p).$$

Ясно, что P_2 накрывает все натуральные числа, меньшие или равные $l + 1$. В то же время это множество точно не содержит числа $1 + p$. Поэтому $f_2(m) = \infty$.

Докажем, что $f_3(n) = \text{НОК}(1, 2, \ldots, n) - 1$. При $n = 1$ ясно, что все разрешенные прогрессии будут иметь шаг 1 и поэтому не смогут накрыть в совокупности 1, не накрыв весь натуральный ряд. Значит $f_3(1) = 0 = \text{НОК}(1) - 1$. Формула выполнена. Допустим теперь, что $n > 1$. Рассмотрим следующее множество:

$$P_3 = \Big((1, 2)\Big) \cup \Big((1, 3) \cup (2, 3)\Big) \cup \ldots \cup \Big((1, n) \cup (2, n) \cup \ldots (n - 1, n)\Big). \tag{10}$$

Ясно, что семейство прогрессий из (10) не накрывает в совокупности те и только те числа, которые одновременно делятся на 2, на 3, на 4 и так далее до n. Значит P_3 заведомо содержит в себе все натуральные числа, строго меньшие чем $\mathrm{НОК}(1, 2, \ldots, n)$. Поэтому

$$f_3(n) \geq \mathrm{НОК}(1, 2, \ldots, n) - 1. \tag{11}$$

Докажем теперь от противного, что

$$f_3(n) \leq \mathrm{НОК}(1, 2, \ldots, n) - 1. \tag{12}$$

Пусть это не так. Тогда существует такое конечное семейство X арифметических прогрессий, объединение которых содержит все натуральные числа от 1 до $\mathrm{НОК}(1, 2, \ldots, n)$ и при этом не накрывает весь натуральный ряд. Пусть $s \in \mathbb{N}$ — произвольное натуральное число между 1 и $\mathrm{НОК}(1, 2, \ldots, n)$. Его накрывает одна из прогрессий семейства X. Без ограничения общности, эта прогрессия имеет вид (x, y). Здесь $1 \leq y \leq n$. Но тогда число $s + \mathrm{НОК}(1, 2, \ldots, n)$ тоже принадлежит (x, y), ведь $\mathrm{НОК}(1, 2, \ldots, n)$ делится без остатка на y. Но число s можно выбрать произвольно среди натуральных чисел от 1 до $\mathrm{НОК}(1, 2, \ldots, n)$. Значит семейство X накрывает весь $\mathbb{N}$, а это невозможно. Итак, неравенство (12) доказано. Объединяя неравенства (11), (12), получаем, что $f_3(n) = \mathrm{НОК}(1, 2, \ldots, n) - 1$.

■

Теорема 2. *Пусть* $n \in \mathbb{N}$. *Тогда* $f_{2,3} = \prod\limits_{p \in \mathbf{P}_n} p^{[\log_p n]} - 1$.

Доказательство.
Обозначим через $t(n)$ число $\prod\limits_{p \in \mathbf{P}_n} p^{[\log_p n]}$. Докажем, что

$$f_{2,3} \geq t(n) - 1. \tag{13}$$

Для произвольного $p \in \mathbf{P}_n$ обозначим через $a(p)$ число $p^{[\log_p n]}$ и для произвольного $i \in E_{a(p)-1}$ вводим обозначение

$$R(p, i) = (i, a(p)). \tag{14}$$

Очевидно, что в (14) все $R(p, i) \in \mathbb{P}_1(n) \cap \mathbb{P}_2(n)$ и выполнено

$$\bigcup_{i=1}^{a(p)-1} R(p, i) = \mathbb{N} \setminus (a(p), a(p)). \tag{15}$$

Для доказательства (13) осталось заметить, что из (15) можно получить

$$\bigcup_{p\in\mathbf{P}_n}\left(\bigcup_{i=1}^{a(p)-1}R(p,i)\right)=\bigcup_{p\in\mathbf{P}_n}\mathbb{N}\setminus(a(p),a(p))=\mathbb{N}\setminus\left(\bigcap_{p\in\mathbf{P}_n}(a(p),a(p))\right)=$$

$$=\mathbb{N}\setminus\left(\prod_{p\in\mathbf{P}_n}a(p),\prod_{p\in\mathbf{P}_n}a(p)\right)=\mathbb{N}\setminus(t(n),t(n))\,.$$

Докажем теперь, что

$$f_{2,3}\le t(n)-1. \tag{16}$$

Пусть это неверно и нашлись такие $k\in\mathbb{N}$ $a_i,b_i\in\mathbb{N}$, $i\in E_k$, для которых

$$E_{t(n)}\subset\bigcup_{i=1}^{k}(a_i,b_i)\ne\mathbb{N}, \tag{17}$$

$$(a_i,b_i)\in\mathbb{P}_1(n)\cap\mathbb{P}_2(n),\ i\in E_k. \tag{18}$$

Можно сразу считать, что все b_i в (16) больше 1, так как иначе прогрессия (a_i,b_i) не пересекалась бы с $E_{t(n)}$ и ее можно было бы выкинуть. По лемме 2 о примарности можем считать, что все b_i, $i\in E_k$ — примарные числа, так как проводимая в лемме 2 замена прогрессии (a_1,b_1) на прогрессию $(a_1,p_t^{a_t})$ не выводит нас за класс $\mathbb{P}_1(n)\cap\mathbb{P}_2(n)$. Из (18) следует, что $a_i,b_i\le n$ при $i\in E_k$. Для каждого $p\in\mathbf{P}_n$ обозначаем

$$v(p):=\{i\in E_k\mid b_i\text{ делится на }p\}.$$

Так как все $b_i\le n$, то их примарные основания тоже не превосходят n, а значит лежат в $\mathbf{P}_n$. Поэтому

$$\bigcup_{p\in\mathbf{P}_n}v(p)=E_k. \tag{19}$$

Теперь проделаем следующее преобразование. Для всех $p\in\mathbf{P}_n$ и для всех $i\in v(p)$ прогрессию (a_i,b_i) можно представить в виде конечного объединения непересекающихся прогрессий с максимально возможным для класса $v(p)$ шагом $a(p)$. Это верно, так как все такие b_i — степени числа p и эти степени не превосходят $a(p)$. При этом, возможно, мы выйдем за границы класса $\mathbb{P}_1(n)$. Заметим, что хотя бы одна из прогрессий в списке

$$(1,a(p)),(2,a(p)),\dots,(a(p),a(p))$$

не войдет в наше представление. Другими словами такой прогрессии (обозначим ее за $(b(p), a(p))$) не будет в множестве

$$\bigcup_{i \in v(p)} (a_i, b_i). \tag{20}$$

Рассмотрим теперь множество

$$\bigcap_{p \in \mathbf{P}_n} (b(p), a(p)). \tag{21}$$

Оно непусто, так как при разных $p_1, p_2 \in \mathbf{P}_n$ числа $a(p)$ будут взаимно простыми. Более того, из китайской теоремы об остатках известно, что (21) образует арифметическую прогрессию с шагом $\prod\limits_{p \in \mathbf{P}_n} a(p)$ и началом не выше этого же числа. Осталось заметить, что

$$\prod_{p \in \mathbf{P}_n} a(p) = \prod_{p \in \mathbf{P}_n} p^{[\log_p n]} = t(n).$$

Получили противоречие с условием (17).

■

Теорема 3. *Пусть* $n \in \mathbb{N}$. *Тогда* $f_4(n) = 2^n - 1$.

Доказательство.

Докажем сначала, что $f_4(n) \geq 2^n - 1$. Для этого достаточно рассмотреть накрытие

$$P = (1, 2) \sqcup (2, 4) \sqcup (4, 8) \sqcup \ldots \sqcup (2^{n-1}, 2^n).$$

Ясно, что в правой части этого накрытия ровно n прогрессий и они в совокупности накрывают все числа, строго меньшие чем 2^n. Поэтому $f_4(n) \geq 2^n - 1$.

Докажем теперь, что $f_4(n) \leq 2^n - 1$. Будем доказывать утверждение индукцией по n. При $n = 1$ утверждение очевидно. Предположим, что при $n \in E_k$ утверждение доказано. Докажем, что и при $n = k + 1$ утверждение тоже верно. Пусть это не так и $f_4(k + 1) \geq 2^{k+1}$. Рассмотрим тогда какое-нибудь семейство

$$L = (a_i, b_i)_{i=1}^{k+1},$$

накрывающее $E_{2^{k+1}}$, но не накрывающее весь ряд $\mathbb{N}$. Обозначим через $U(L)$ объединение его элементов. Можно сразу считать, что все $b_i > 1$, так как иначе прогрессии $(a_i, 1)$ не пересекались бы с $E_{2^{k+1}}$ и их можно было бы выкинуть. По лемме 2 считаем, что все b_i примарны, то есть являются степенями простых чисел. Тогда семейство L можно представить в конечного объединения семейств $(L_i)_{i=1}^s$ таких, что шаги всех прогрессий из L_i являются степенями некоторого $p_i \in \mathbf{P}$.

Обозначим количество элементов в L_i через a_i. Обозначим через $U(L_i)$ объединение элементов из L_i. Из леммы 1 очевидно следует, что при всех $i = 1, \ldots, s$ элементы из L_i попарно не пересекаются. По лемме 3 можно считать, что каждое из множеств $U(L_i)$ является плотноупакованным множеством с шагом p_i. Причем из леммы 5 следует, что глубина каждого из таких множеств равна 1, то есть все шаги b_i прогрессий из L — простые числа вида $(p_i)_{i=1}^s$. Тогда $U(L)$ является периодическим множеством с периодом $t := p_1 \cdot p_2 \cdot \ldots \cdot p_s$. Значит найдется такое $x \in E_t$, что все элементы из L не пересекаются с прогрессией (x, t). Оценим сверху количество элементов в множестве

$$U(L) \cap \{x + i \mid i \in E_{t-1}\},$$

то есть количество элементов из $U(L)$ на его периоде. По китайской теореме об остатках таких элементов будет не больше чем $a_1 \cdot a_2 \cdot \ldots \cdot a_s$. Но мы знаем, что $a_1 + a_2 + \ldots + a_s = k + 1$. По следствию из леммы 4 получаем

$$a_1 \cdot a_2 \cdot \ldots \cdot a_s \leq 3^{\frac{k+1}{3}}.$$

Осталось заметить, что $3^{\frac{k+1}{3}} < 2^{k+1}$. Значит на периоде множество $U(L)$ имеет менее чем 2^{k+1} элементов, что противоречит тому факту, что семейство L накрывает $E_{2^{k+1}}$.

■

Список литературы

[1] П. С. Дергач. *О каноническом регулярном представлении S-тонких языков.* Интеллектуальные системы, 2014. Т.18, вып. 1, М., Сс. 211-242.системы, 2014. Т.18, вып. 1, М., Сс. 211-242.

[2] П. С. Дергач. *О проблеме вложения допустимых классов.* Интеллектуальные системы, 2015. Т.19, вып. 2, М., Сс. 143-174.

[3] П. С. Дергач, Э. С. Айрапетов. *О прогрессивном разбиении некоторых подмножеств натурального ряда.* Интеллектуальные системы, 2015. Т.19, вып. 3, М., Сс. 79-86.

[4] П. С. Дергач. *О двух размерностях спектров тонких языков.* Интеллектуальные системы, 2015. Т.19, вып. 3, М., Сс. 155-174.

[5] П. С. Дергач, Э. С. Айрапетов. *О прогрессивном разбиении последовательности натуральных чисел, имеющей пропуск длины 2.* Интеллектуальные системы, 2016. Т.20, вып. 2, М., Сс. 67-86.

[6] П. С. Дергач, Е. Д. Данилевская. *О покрытиях и разбиениях натуральных чисел, имеющих два последовательных пропуска длины 1.* Интеллектуальные системы, 2017. Т.21, вып. 1, М., Сс.192-237.

[7] П. С. Дергач. *О структуре вложения прогрессивных множеств сложности два.* Интеллектуальные системы, 2017. Т.21, вып. 2, М., Сс.117-162.

[8] П. С. Дергач, Ж. И. Раджабов *О длине минимальной алфавитной склейки для класса линейных регулярных языков.* Интеллектуальные системы, 2017. Т.21, вып. 3, М., Сс.120-130.

[9] Д. Е. Александров. *Эффективные методы реализации проверки содержания сетевых пакетов регулярными выражениями.* Интеллектуальные системы, 2014. Т.18, вып. 1, М., Сс. 37-60.

[10] Д. Н. Бабин. *Частотные регулярные языки.* Интеллектуальные системы, 2014. Т.18, вып. 1, М., Сс. 205-210.

[11] Д. Е. Александров. *Об оценках автоматной сложности распознавания классов регулярных языков.* Интеллектуальные системы, 2014. Т.18, вып. 4, М., Сс. 161-190.

[12] В. М. Дементьев. *О звездной высоте регулярного языка и циклической сложности минимального автомата.* Интеллектуальные системы, 2014. Т.18, вып. 4, М., Сс. 215-222.

[13] И. Е. Иванов. *О сохранении периодических последовательностей автоматами с магазинной памятью с однобуквенным магазином.* Интеллектуальные системы, 2015. Т.19, вып. 1, М., Сс. 145-160.

[14] А. А. Петюшко. *О контекстно-свободных биграмных языках.* Интеллектуальные системы, 2015. Т.19, вып. 2, М., Сс. 187-208.

[15] И. Е. Иванов. *Нижняя оценка на максимальную длину периода выходной последовательности автономного автомата с магазинной памятью.* Интеллектуальные системы, 2015. Т.19, вып. 3, М., Сс. 175-194.

[16] В. А. Орлов. *О конечных автоматах с максимальной степенью различимости состояний.* Интеллектуальные системы, 2016. Т.20, вып. 1, М., Сс. 213-222.

[17] П. С. Дергач. *О проблеме проверки однозначности алфавитного декодирования в классе регулярных языков с полиномиальной функцией роста.* Интеллектуальные системы, 2016. Т.20, вып. 2, М., Сс. 147-202.

[18] А. М. Миронов. *Основные понятия теории вероятностных автоматов.* Интеллектуальные системы, 2016. Т.20, вып. 2, М., Сс. 283-330.

[19] А. А. Петюшко, Д. Н. Бабин. *Классификация Хомского для матриц биграммных языков.* Интеллектуальные системы, 2016. Т.20, вып. 2, М., Сс. 331-336.

[20] С. Б. Родин. *О связи линейно реализуемых автоматов и автоматов с максимальной вариативностью относительно кодирования состояний.* Интеллектуальные системы, 2016. Т.20, вып. 2, М., Сс. 337-348.

Сведения об авторах

Дергач Петр Сергеевич

Младший научный сотрудник МГУ имени М. В. Ломоносова в городе Москве;

адрес: Россия, г. Москва, 125565, Ленинградское ш., 88-19;

тел. моб.: +79037189288;

e-mail: dergachpes@mail.ru.

About the maximum coverage of positive integers with some kind of restrictions

Dergach Peter Sergeevich

In this thesis it is necessary to find the maximum length of the beginning of natural set, that can be covered by the union of arithmetic progressions without covering this way all natural set. There are also some kind of restrictions on beginning and step of these progressions, and on their total number. Depending on what type of restrictions take place, we have a class of various tasks. Some of them were solved in this paper. The most interesting cases are types of restrictions like "beginning+step"and "quantity".

Keywords: natural set, arithmetic progression, maximum coverage.

Проблема стабилизации в булевых сетях

М. Ж. Жолбарысов, Ю. С. Шуткин
(МГУ имени М.В. Ломоносова)

В данной работе рассматривается проблема стабилизации булевых сетей, а именно, вопрос наличия точечных аттракторов в асинхронной булевой сети. Найден критерий стабилизации в зависимости от выбора компонент булевой сети: граф, булевы функции, начальное состояние, порядок обновления.

Ключевые слова: булевы сети, стабилизация. Проблема стабилизации в булевых сетях

Введение

Исследование биологических сетей имеет важное место в биоинформатике, системной биологии и алгебраической биологии. Одной из многочисленных математических моделей биологических сетей являются булевы сети. Впервые булевы сети рассмотрены Стюартом Кауффманом в 1969 году [1], как случайная модель регулярной генной сети, которая в свою очередь является частным случаем клеточного автомата. Булева сеть имеет очень простую модель в виде ориентированного графа: каждая вершина может иметь состояние либо 0, либо 1, и состояния этих вершин меняются в некой последовательности и дискретно, под действием некоторых булевых функций. Количество булевых сетей имеющих ровно n вершин — $\left(\frac{2^{2^n}n!}{(n-K)!}\right)^n$, где K — максимальное количество аргументов булевых функции. Поведение и стабильность булевой сети характеризуется аттрактором, понятие которого занимает центральное место в теории булевых сетей. Количество возможных аттракторов и их длины изучались С. Кауффманом [2], Д.Ж. Айроном [3]. Проблема нахождения аттрактора в булевой сети исследовалась Т. Файзуровым [5], Е. Дубровой [4], Т. Акутсой [12] и др. В данной работе рассматриваются проблема стабилизации булевых сетей, а именно, вопрос наличия точечных аттракторов в булевой сети.

Основные понятия

Булева сеть N — ориентированный псевдограф G, который определяется конечным набором вершин $V = \{v_1, v_2, ..., v_n\}$, множеством ребер E, начальным состоянием вершин $\mathbf{v}(0)$ и набором булевых функций $F = \{f_1, f_2, ..., f_n\}$, приписанных этим вершинам соответственно. При этом количество аргументов функции, приписанной каждой вершине, в точности равно количеству ребер, входящих в эту вершину, и установлено соответствие между входящими ребрами и аргументами функции путем нумерации входящих ребер от 1 до k, где k — степень входа вершины. В случае, если порядок аргументов функции не важен (т. е. функция симметричная) или соответствие между входящими ребрами и аргументами следует из контекста, нумерация ребер будет опускаться, чтобы не загромождать рисунки.

Обозначим через v_i^t *состояние*(значение) *вершины* v_i в момент времени t, которое может быть равно либо 0, либо 1. Например, булевая сеть на рис. 3, состоит из $V = \{v_1\}, F = \{\bar{x}\}$ и $v_1^0 = 0$.

Состоянием или *траекторий* T булевой сети в момент времени t будем называть вектор $\mathbf{v}(t) = (v_1^t, v_2^t, ..., v_n^t)$. Начальное состояние булевой сети N задается вектором $\mathbf{v}(0)$. Заметим, что существует только 2^n возможных состоянии булевой сети.

Существует два вида булевых сетей: *синхронная* и *асинхронная*.

Дадим сначала определение *асинхронной булевой сети*. В каждый момент времени, выбирается одна вершина, и она может быть обновлена, а все остальные не меняют свое значение. За промежуток времени равный n, каждая вершина должна обновиться ровно один раз. Состояние v_i в момент времени $t+1$ может быть найдено, как

$$v_i^{t+1} = \begin{cases} f_i(v_{i_1}^t, v_{i_2}^t, ..., v_{i_k}^t), & \text{если выбрана для обновления именно } v_i; \\ v_i^t, & \text{иначе;} \end{cases}$$

где $v_{i_1}, ..., v_{i_k}$ — набор вершин, из которых ведут ориентированные ребра в v_i, записанных в той последовательности, в какой от них зависит функция f_i, а k — степень входа вершины v_i.

Синхронная булева сеть отличается от асинхронной тем, что в каждый момент времени обновляются все вершины. Формально это выглядит так: $v_i^{t+1} = f_i(v_{i_1}^t, v_{i_2}^t, ..., v_{i_k}^t)$, $\forall i = 1, 2, ..., n$, а $v_{i_1}, ..., v_{i_k}$ — набор вершин, из которых ведут ориентированные ребра в v_i, записанных в той последовательности, в какой от них зависит функция f_i, также k — степень входа вершины v_i. Булеву сеть V будем называть стабильной,

если ее траектория $T(t)$ не зависит от времени $t \geq M_0$ при некотором M_0. Ориентированное *ребро* (v_i, v_j) булевой сети N называется *несущественным*, если f_j не зависит от значения вершины v_i.

Порядком D булевой сети V назовем последовательность, состоящую из n вершин, расставленных в ней в той очередности, в какой обновлялась булева сеть в первые n моментов времени.

Пусть $\omega \in \{0,1\}^n$ — состояние булевой сети и $R(\omega)$ — всевозможные состояния, которые могут быть достигнуты, начиная с ω. Тогда множество состояний S называется *аттрактором* булевой сети, если $R(\omega) = S$ для любого $\omega \in S$. S называется *точечным аттрактором*, если $|S| = 1$, и называется *циклическим аттрактором* в случае $|S| > 1$. Таким образом, булева сеть N определяется множеством вершин V, множеством ребер E, функций F, соответствием между ребрами входящими в вершины и аргументами приписанных функций A, порядком обновления D, начальным значением вершин $\mathbf{v}(0)$ (иначе говоря, $N = (V, E, F, A, D, \mathbf{v}(0))$. Заметим, что длина аттрактора может быть любой в пределах от 1 до 2^n. Приведем пример асинхронной булевой сети, имеющей аттрактор длины 2^n: $(V, E, F, A, D, \mathbf{v}(0)) = (\{v^1, v^2, ..., v^n\}, \{(v^1, v^1), ..., (v^n, v^n)\}, (\bar{x}, ..., \bar{x})$, любая нумерация входящих ребер, любой порядок, $\mathbf{v}(0) = \{0, 0, ..., 0\})$.

Обзор литературы

С. Кауффман [1] определил, что среднее количество аттракторов в синхронной булевой сети с n вершинами примерно равно $\sqrt{n}$. Однако исследования 1997-2003 годов показали ошибочность вышеуказанного. Б. Самуэлльсон и К. Троин [10] показали суперполимиальный рост числа аттракторов в зависимости от числа вершин синхронных булевых сетей ($> n^\gamma$ для $\forall\gamma$). Ф. Греиль и Б. Дроссель [6] уже в асинхронных булевых сетях показали полимиальный рост среднего числа аттракторов, а также, что средняя длина аттрактора имеет экспоненциальный рост ($exp(N^\alpha)$ для некоторого фиксированного α). В свою очередь Т. Михайлев и Б. Дроссель в [8] показали суперполимиальный рост длины аттрактора в синхронной булевой сети. Средняя длина же цикличных аттракторов в сихронных булевых сетях пропорциональна 2^n [2].

Большинство задач, касающихся булевых сетей, принадлежат классу NP. (см. Таблица 1). Существует несколько задач связанных с булевыми сетями. Одной из них является построение синхронной булевой сети с заданной структурой аттракторов. Для формулировки ее нам понадобится следующие определения. Подмножество $W \subseteq V$ называется предсказа-

Таблица 1. Оценки для синхронной булевой сети

	дерево	граф без циклов (ориентированный)	граф
Построение сети	P	P	P
Поиск аттрактора	P	P	NP
Поиск сети	NP	NP	NP
Поиск сети (ограниченные K)	P	P	P

(K означает максимальное количество существенных аргументов булевой функции)

тельным множеством для вершины v_i, если ограничение $f_i|_{W_i}$ функции f_i равно f_i. Набор $W = (W_1, W_2, ..., W_n)$ называется предсказательным набором для булевой сети N. Состояния в итерационном графе (см. последнюю часть) содержатся во множестве уровней: уровень l_j содержит все состояния, откуда можно за ровно j переходов (времени) достичь аттрактора, которые находятся в уровне l_0. Итерационный граф порождает однокоренное дерево, если в нем содержится ровно один точечный аттрактор (цикличные аттракторы отсутствуют), такой, что аттрактор достижим в сети через это дерево. Если же он содержит k точечных аттракторов (но не содержит цикличные аттракторы), то здесь уже k однокоренных деревьев, которые будем называть k-лесом (от англ. forest).

Пусть дан набор из n вершин V, семейство из n подмножеств $W_1, W_2, ..., W_n$ множества V с мощностями не меньше m и не больше M, $0 < m \leq M$, и набор A содержащий k состояний и два положительных числа $l \leq L$, которые соответствуют булевой сети N с множеством вершин V, с предсказательным набором $W = (W_1, W_2, ..., W_n)$, цикличными аттракторами $A_1, A_2, ..., A_r$, где $A = \bigcup_{j=1}^{r} A_j$, которые содержатся между l и L уровнями. Ранадип Пал, Иван Иванов и др. [9] построили алгоритм, позволяющий находить в указанных выше условиях булевы сети в пространстве k-лесов ($k = \overline{1, n}$) с вероятностью $\binom{n}{m}^n N^{2^m}/(N+1)^{N-1}$.

Центральной проблемой в булевых сетях является нахождение точечных аттракторов. Задача поиска точечного аттрактора булевой сети лежит в классе NP. На данный момент существует множество алгоритмов нахождения точечных аттракторов синхронной булевой сети. Отметим, что $O(2^n)$ времени достаточно, ввиду того, что булева сеть имее всего 2^n состояний. Однако неизвестно существование алгоритма со сложностью

$O((2-\epsilon)^n)(\epsilon > 0$, n- количество вершин булевой сети) для общего случая. Тем не менее, существуют алгоритмы для ограниченного класса булевых функций (см. Таблицу 2). А именно, в зависимости от аргументов булевой функций, класса $AND\backslash OR$ $((v_{i_1}^{a_1} \wedge v_{i_2}^{a_2} \wedge ... \wedge v_{i_{k_i}}^{a_{k_i}})^b$, $(v_{i_1}^{a_1} \vee v_{i_2}^{a_2} \vee ... \vee v_{i_{k_i}}^{a_{k_i}})^b$, где $v^a = v + a \ mod\ 2)$), и так называемые nested canalyzing функции (ее определение дано ниже).

Булеву функцию $f(x_1, ..., x_k)$ будем называть nested canalyzing относительно переменных $(x_1, ..., x_k)$, если можно представить ее как $f = l_1 \vee l_2 \vee ... \vee l_{k_1-1} \vee (l_{k_1} \wedge ... \wedge l_{k_2-1} \wedge (l_{k_2} \vee ... \vee l_{k_3-1} \vee (...)))$, где l_i — либо v_j, либо $\overline{v_j}$, и $1 \leq k_1 < k_2 < ...$

Таблица 2. Оценки для синхронной булевой сети

Вид функции	Сложность алгоритма
K=1	$O(1.09^n)$[13]
K=3	$O(1.21^n)$[13]
K=5	$O(1.37^n)$[13]
K=7	$O(1.45^n)$[13]
nested canalyzing	$O(1.799^n)$[11]
$AND\backslash OR$	$O(1.587^n)$[7]

(K означает максимальное количество существенных аргументов булевой функции)

Стабилизация булевых сетей

В данном разделе рассматривается проблема поиска точечного аттрактора асинхронной булевой сети.

Будем решать следующую задачу. Пусть у нас имеется 4 степени свободы: булев псевдограф, булева функция, начальные значения, порядок. Тогда делая выбор в этой вышеуказанной четверке необходимо добиться, чтобы булева сеть была стабильной. Считаем. что если мы не выбираем какую-то из степеней свободы, она может быть фиксирована произвольным образом (в том числе и наихудшим образом). Назовем эти условия У1.

Утверждение 1. Для любой булевой функции f, кроме константы, можно построить булеву сеть, в которой всем вершинам приписана

функция f, для которой существует порядок и начальные значения, при котором сеть не стабилизируется.

Доказательство.

Рассмотрим несколько случаев. Случай 1: $f(\tilde{0}) = 0, f(\tilde{1}) = 1$, где $\tilde{0} = (0, 0, ..., 0)$ и $\tilde{1} = (1, 1, ..., 1)$. Заметим, что тогда существует такой набор $\tilde{\alpha}$, что без ограничения общности выполнено $f(1, \tilde{\alpha}) = 1, f(0, \tilde{\alpha}) = 0$. Рассмотрим сеть на рис. 2.

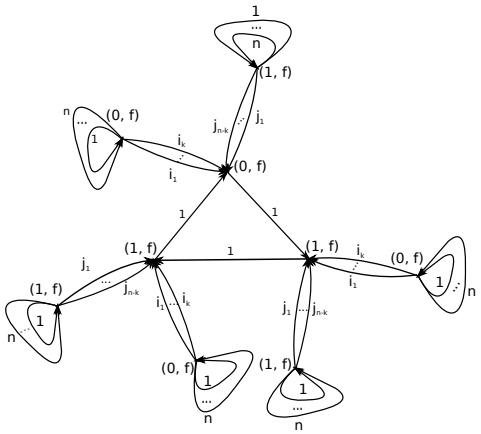


рис. 2

рис. 3

Случай 2: $f(\tilde{0}) = 1, f(\tilde{1}) = 0$. В этом случае рассмотрим сеть на рис. 3.

Случай 3: $f(\tilde{0}) = f(\tilde{1}) = 0$. Заметим, что тогда существует такой набор $\tilde{\alpha}$, что без ограничения общности выполнено $f(1, \tilde{\alpha}) = 1, f(0, \tilde{\alpha}) = 0$. Тогда рассмотрим сеть, изображенную на рис. 4.

Случай 4: $f(\tilde{0}) = f(\tilde{1}) = 1$. Этот случай аналогичен случаю 3. Сеть изображена на рис. 5.

Осталось рассмотреть порядок обновления полученных булевых графов. В случаях 1,3,4 порядок характеризуется числами в “основном треугольнике” (точнее набором $(1, 0, 1)$). Тогда действуем следующим образом: $(1, 0, 1) \to (1, 0, 0) \to (1, 1, 0) \to (0, 1, 0) \to (0, 1, 1) \to$

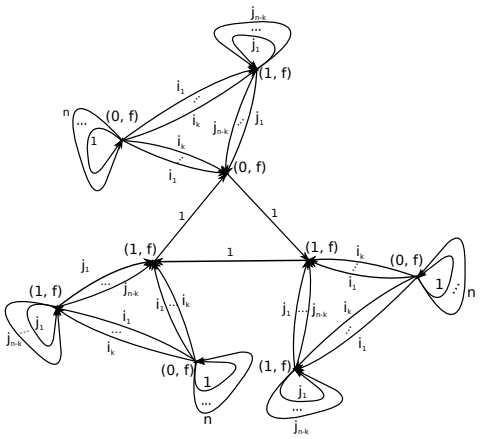


рис. 4

$(0,0,1) \to (1,0,1)$, иначе говоря против часовой стрелки. Таким образом, аттрактор в случаях 1,3,4 состоит из 6 наборов, а именно $S = \{(0,1,1,0,1,0,0,1,1),\ (0,1,1,0,1,0,0,1,0),\ (0,1,1,0,1,1,0,1,0),\ (0,1,0,0,1,1,0,1,0), (0,1,0,0,1,1,0,1,1), (0,1,0,0,1,0,0,1,1)\}$. А во втором случае аттрактор состоит из двух наборов: $S = \{(0),(1)\}$.

Утверждение доказано.

Утверждение 2. Существуют булев псевдограф и булева функция, такие что для $\forall$ начальных значений и порядка можно построить булевую функцию, при которой граф не стабилизируется.

Доказательство.

Рассмотрим булевую функцию $f = \bar{x}$. Тогда можно рассмотреть следующий граф N на рис. 6. Пусть начальные значения графа N имеют вид $\mathbf{v}(0) = (\alpha, \beta, \gamma)$. Всевозможные порядки D графа N имеет вид $\{(1,2,3),(1,3,2),(2,1,3),(2,3,1),(3,1,2),(3,2,1)\}$. Тогда траектория $\mathbf{v}(3)$ может быть равна $\{(\bar{\gamma},\gamma,\bar{\gamma}),(\bar{\gamma},\gamma,\bar{\beta}),(\bar{\gamma},\bar{\alpha},\alpha),(\bar{\alpha},\bar{\alpha},\alpha),(\beta,\bar{\beta},\bar{\beta}),(\beta,\bar{\alpha},\bar{\beta})\}$. Отсюда следует, что $\mathbf{v}(3) \neq \mathbf{v}(0)$, при любых α, β, γ. Доказательство завершено.

Теорема 1. Пусть фиксирован класс булевых функций $M = P_2 \backslash \{0,1\}$. Тогда, если мы находимся в условиях У1, можно гарантировать, что сеть будет стабильной в том и только в том случае, если позволено выбирать булевы функции из M, а также одну из следующих степеней свободы булевой сети: булев граф, начальные значения или порядок.

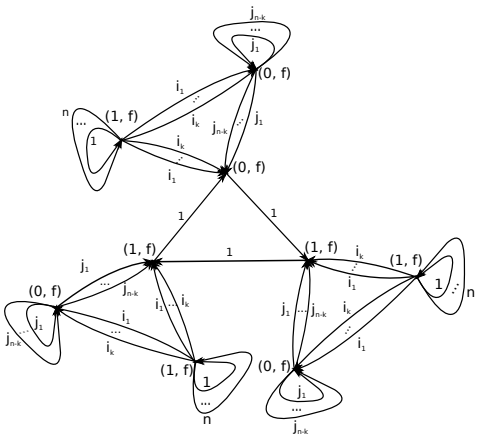


рис. 5

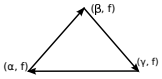


рис. 6

Доказательство. Рассмотрим следующие случаи, в которых можно гарантировать стабильность (кроме последнего):

Случай 1: пусть выбрали булев граф и булеву функцию, тогда выбрав следующий граф на рис. 7 получаем стабильную булеву сеть.

рис. 7

Случай 2: пусть выбрали булеву функцию и начальные значения, тогда нужно выбрать булеву функцию $f(\tilde{0}) = 0$ и ко всем вершинах приписать 0.

Случай 3: пусть выбрали булеву функцию и порядок. Обозначим тогда получаемую булеву сеть через $N = (V, E, F, A, D, \mathbf{v}(0))$. Осуществим разбиение множества вершин V графа на непересекающиеся подмножества следующим образом: выберем произвольную вершину, без ограничения общности будем считать, что это v_1, и рассмотрим максимальную (по количеству вершин) простую ориентированную цепь $C_1 = (v_1, v_{i_1}, ..., v_{i_j})$. Пусть $V_1 = \{v_1, v_{i_1}, ..., v_{i_j}\}$. Назовем его порождаемым множеством для

v_1 и в этом случае также будем говорить, что вершина v_1 порождает множество V_1. Выберем вершину, из которой существует ориентированное ребро в v_1, и будем считать, что это v_2. Тогда вершине v_1 (v_{i_1}) припишем селекторную функцию, которая существенно зависит от значения вершины v_2 (v_1). Остальным вершинам v_{i_k} из C_1 (если они есть) припишем селекторную функцию, которая существенно зависит от значения вершин $v_{i_{k-1}}$. Если $V \diagdown V_1$ непуста, то выберем другую вершину из этого множества и проделаем для нее то же самое, что и для v_1, с единственной оговоркой — порождаемое для нее множество не должно пересекаться с V_1. Отметим, что если невозможно построить простую цепь для некоторой выбранной вершины, то порождаемое ей множество будет состоять только из нее. Будем продолжать этот процесс до тех пор, пока не разобьем V (порождаемые множества не должны пересекаться). Пусть $V = \bigsqcup V_p$, где $V_i = (v_1, v_{i_1}, ..., v_{i_j})$. Рассмотрим порядок $D = (C_1, C_2, ..., C_p)$, т. е. сначала последовательно обновляются вершины цепи C_1, потом C_2 и т. д. Очевидно, что построенная булева сеть будет стабильной.

Случай 4: пусть выбрали только булевы функции. Допустим, тогда что булева сеть имеет граф из рис. 6. Тогда можно функции выбирать только из следующего множества $(\bar{x}, x)$. В силу утверждения 2 не ограничиваемся выбором только функции вида $\bar{x}$. Построим простым перебором все возможные булевы сети $N = (V, E, F, A, D, \mathbf{v}(0))$ (V, E порождена графом из рис. 11). Пусть $V = (v_1, v_2, v_3)$. Положим, что начальные значения имеют вид $\mathbf{v}(0) = (0, 0, 1)$. Также в силу симметрии можно считать, что $F = (f_1, f_2, f_3) \in \{(\bar{x}, \bar{x}, x), (\bar{x}, x, x), (x, x, x)\}$. Таким образом у нас может быть три вида булевой сети:

1) Пусть $F = \{(\bar{x}, \bar{x}, x)\}$. Тогда можно взять $D = (3, 2, 1)$. Отсюда аттрактор сети не точечный: $\{(0, 0, 1), (1, 1, 0)\}$.

2) Пусть $F = \{(\bar{x}, x, x)\}$. Тогда можно взять $D = (3, 2, 1)$. Отсюда аттрактор сети также не точечный: $\{(0, 0, 1), (1, 0, 0), (1, 1, 0), (0, 1, 1)\}$.

3) Пусть $F = \{(x, x, x)\}$. Тогда можно взять $D = (1, 3, 2)$. Отсюда аттрактор сети не точечный: $\{(0, 0, 1), (1, 1, 0)\}$.

Случай 5: пусть выбрали булев граф, но не выбрали функцию. Пусть в выбранном графе сети N есть петли. Допустим петлю содержит вершина v_1. Тогда к этой вершине припишем функцию $f_1 = \bar{x}$, т. е. $v_1^{t+1} = \bar{v}_i^t$. Очевидно, что булева сеть не будет стабильным. Значит нужно выбирать граф, который не содержит петель. Так как можно выбирать только неконстантные булевые функции, то в выбранном графе должны быть циклы. Пусть цикл состоит из k вершин $u_1, ...u_k$. Тогда если приписать к

вершинам цикла функции $u_1^{t+1} = f_1(..., u_k^t, ...) = \bar{u}_k^t, u_2^{t+1} = f_2(..., u_1^t, ...) = u_1^t, ..., u_k^{t+1} = f_k(..., u_{k-1}^t, ...)$, т. е. $f_1 = \bar{x}, f_2 = x, ..., f_k = x$, получается нестабильная булева сеть, вне зависимости от порядка и начальных значений (рис. 8).

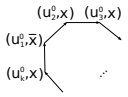


рис. 8

Предположим противное, пусть булева сеть стабильная. Допустим, сеть стабилизируется через t момент времени. Пусть $u_k^t = \alpha$, тогда $u_1^t = \bar{\alpha}, u_2^t = \alpha$ и т. д. Однако, из селекторных функций невозможно получить отрицание. Предположение неверное.

Рассмотрим теперь последний случай 6: если не выбрали ни булеву функцию ни булев граф, тогда из утверждения 2 следует теорема 1. Доказательство завершено.

Теорема 2. Пусть фиксирован класс булевых функций $M = P_2$. Тогда если мы находимся в условиях У1 можно гарантировать, что сеть будет стабильной в том и только в том случае, если позволено выбирать булевы функции из M или булев граф.

Доказательство. Рассмотрим следующие случаи, в которых можно гарантировать стабильность (кроме последнего):

Случай 1: пусть выбрали булев граф, тогда выбирам как на рис. 9. Откуда при любых булевых функциях, начальных значениях, порядке, очевидно, что данная сеть стабильная.

рис. 9

Случай 2: пусть выбрали булеву функцию, стабильность следует из того, что достаточно выбрать в качестве функций константы.

Случай 3: пусть не выбрали ни булеву функцию, ни булев граф. Тогда этот случай непосредственно следуют из утверждения 2. Доказательство завершено.

Будем решать теперь другую задачу. Пусть у нас имеется также 4 степени свободы: булев граф, булева функция, начальные значения, порядок. Тогда, делая выбор в этой вышеуказанной четверке, необходимо добиться, чтобы булева сеть была стабильной. Также будем считать, что

на сей раз в отличие от предыдущей задачи, порядок и начальные значения могут генерироваться случайным образом. Положим также, что если мы не выбрали какую-то из степеней свободы, и она не выбрана случайным образом, она может быть фиксирована произвольным образом (в том числе наихудшим для нас образом). Назовем эти условия У2.

Если разрешено выбирать константные функции, тогда наличие случайности начального значения и порядка не изменяет условие теоремы 2. Поэтому будем теперь считать, что булевы сети содержат только неконстантные булевые функции.

Теорема 3. Пусть фиксирован класс булевых функций $M = P_2\backslash\{0,1\}$. Тогда если мы находимся в условиях У2, можно гарантировать, что сеть будет стабильной, если позволено выбирать булевы функции из M, а также одну из следующих степеней свободы булевой сети: булев граф, начальные значения и порядок, в остальных случаях нельзя гарантировать, что сеть будет стабильной с какой-либо вероятностью большей 0.

Доказательство. Случаи из теоремы 1 будут верны и для этой теоремы. Поэтому достаточно рассмотреть следующие случаи:

Случай 1: допустим, что выбираем функцию, а порядок случаен. Покажем, что тогда вероятность стабильности булевой сети почти наверно равно 0. Обозначим вероятность стабильности через p_1.

Пусть булева сеть имеет граф как на рис. 11

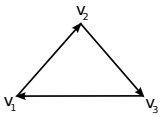


рис. 11

Тогда $f_i \in \{x, \bar{x}\}, i = 1, 2, 3$. Если $f_1 = f_2 = f_3 = \bar{x}$, то в силу утверждения 2, булева сеть будет не стабильной. Если $f_1 = \bar{x}, f_2 = x, f_3 = x$, то булева сеть также будет не стабильной. Действительно, иначе верно следующее для некоторого t: $v_1^t = f_1(v_3^t) = \bar{v_3^t}, v_2^t = f_2(v_1^t) = v_1^t, v_3^t = f_3(v_2^t) = v_2^t$, что невозможно. Значит множество функций $F = \{x, x, x\}$, либо $F = \{\bar{x}, \bar{x}, x\}$.

Пусть $F = \{\bar{x}, \bar{x}, x\}$ и начальные значения графа N имеют вид $\mathbf{v}(0) = (\alpha, \beta, \gamma)$. Всевозможные порядки D графа N имеют вид $\{(1,2,3),(1,3,2),(2,1,3),(2,3,1),(3,1,2),(3,2,1)\}$. Тогда траектория $\mathbf{v}(3)$ может быть равна $\{(\bar{\gamma},\gamma,\gamma),(\bar{\gamma},\gamma,\beta),(\bar{\gamma},\bar{\alpha},\bar{\alpha}),(\alpha,\bar{\alpha},\bar{\alpha}),(\bar{\beta},\beta,\beta),(\bar{\beta},\bar{\alpha},\beta)\}$.

Положим $\mathbf{v}(0) = (\alpha, \beta, \gamma) = (1, 0, 1)$, тогда если $D = (1, 3, 2)$ или $D = (3, 2, 1)$, то булева сеть N нестабильна. Значит, булева сеть N с вероятностью $\frac{4}{6} = \frac{2}{3}$ стабильна. Рассмотрим теперь булеву сеть N_0, состоящий из n булевых сетей N. Тогда в булевой сети N_0 вероятность стабильности $p_1 = (\frac{2}{3})^n \to 0$ при $n \to \infty$.

Аналогично, можно показать для $F = \{x, x, x\}$, что $p_1 = (\frac{2}{3})^n \to 0$ при $n \to \infty$.

Случай 2: допустим, что также выбираем функцию, а начальные значения случайны. Покажем, что тогда так же вероятность стабильности булевой сети почти наверно равна 0.

Будем действовать по аналогии с предыдущим случаем. Положим $F = \{\bar{x}, \bar{x}, x\}$ (случай $F = \{x, x, x\}$ аналогичен). Выберем порядок D графа N равным $(1, 3, 2)$. Тогда булева сеть будет стабильной, если $\gamma = \beta$, а значит вероятность стабильности сети будет равна $(\frac{4}{8})^n \to 0$ при $n \to \infty$.

Случай 3: выбираем функцию, а порядок и начальные значения случайны. Покажем, что тогда так же вероятность стабильности булевой сети почти наверно равна 0. Обозначим вероятность стабильности сети через p_3.

Будем действовать также как и в двух предыдущих случаях 1, 2. Пусть $F = \{\bar{x}, \bar{x}, x\}$ (случай $F = \{x, x, x\}$ аналогичен). Для каждого возможного порядка D (а их всего 6) существует несколько начальных значений, для которых сеть стабилизируется. Общее их число $2 + 4 + 4 + 2 + 2 + 4 = 18$ штук. Так как количество всевозможных порядков и начальных значений $6 \times 8 = 48$, то $p_3 = (\frac{18}{48})^n \to 0$ при $n \to \infty$. Доказательство завершено.

Теперь потребуем, чтобы булев граф был только связным. Это условие в дополнении с У2 назовем условия У3. Однако это условие, как видно из следующей теоремы 4, никак не повляет на утверждение теоремы 3.

Теорема 4. Пусть фиксирован класс булевых функций $M = P_2 \backslash \{0, 1\}$. Тогда если мы находимся в условиях У3 можно гарантировать, что сеть будет стабильной, если позволено выбирать булевы функции из M, а также одну из следующих степеней свободы булевой сети: булев граф, начальные значения и порядок, в остальных случаях нельзя гарантировать, что сеть будет стабильной с какой-либо вероятностью большей 0.

Доказательство. Доказательство полностью повторяет доказательство теоремы 3, единственное различие в случаях 1,2,3 теоремы 3. По-

этому нужно подобрать связный граф, вместо множества несвязанных n графов из рис. 11. "Свяжем" в буквальном смысле эти n графов из следующих фрагментов "общего" графа рис. 12:

а) б)

рис. 12

Граф из рис. 12.а полностью "автономный", т. е. можно построить сеть из n частей графа из рис. 12.а и рис. 12.б (очевидно, что нельзя соединять граф из рис 12.а. с самим собой, только с графом из рис. 12.б и наоборот), при этом сохраняется оценка (из доказательства теоремы 4) для p_1, p_2, p_3 из общей сети. Заметьте, что граф уже состоит не из n частей рис. 11, а $2n$. Доказательство завершено.

Литература

[1] Kauffman S. A. Metabolic stability and epigenesis in randomly constructed genetic nets. *Journal of Theoretical Biology*, (22):437–467, 1969.
[2] Kauffman S. A. *The origins of order*. Oxford University press, 1993.
[3] Irons D.J. Improving the efficiency of attractor cycle identification in boolean networks. *Physica D*, (217):7–21, 2006.
[4] E. Dubrova and M. Teslenko. A sat-based algorithm for finding attractors in synchronous boolean networks. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 2010.
[5] T. Fayruzov, De Cock M., Cornelis C., and D. Vermeir. Modeling protein interaction networks with answer set programming, proceedings of. *2009 IEEE International Conference on Bioinformatics and Biomedicine*, pages 99–104, 2009.
[6] Florian Greil and Barbara Drossel. Dynamics of critical kauffman networks under asynchronous stochastic update. *Physical Review Letters*, 95(048701), 2005.
[7] A. Melkman, T. Tamura, and T. Akutsu. Determining a singleton attractor of an AND/OR boolean network in O(1.587^n) time. *Information Processing Letters*, (110):565–569, 2010.

[8] Tamara Mihaljev and Barbara Drossel. Scaling in a general class of critical random boolean networks. *Physical Review Letters*, 74(046101), 2006.
[9] Pal R., Ivanov I., A. Datta, Bittner M. L., and E. R. Dougherty. Generating boolean networks with a prescribed attractor structure. *Bioinformatics*, 21(21):4021–4025, 2005.
[10] Bjorn Samuelsson and Carl Troein. Superpolynomial growth in the number of attractors in kauffman networks. *Physical Review Letters*, 90(098701), 2003.
[11] Akutsu T., Melkman A.A., Tamura T., and Yamamoto M. Determining a singleton attractor of a boolean network with nested canalyzing functions. *Journal of computational biology*, 18(10):1275–90, 2011.
[12] Akutsu T., M. Hayashida, and Ching W.K. Control of boolean networks: Hardness results and algorithms for tree structured networks. *Journal of Theoretical Biology*, (244):670–679, 2007.
[13] Shu-Qin Zhang, Morihiro Hayashida, Tatsuya Akutsu, Wai-Ki Ching, and Michael K Ng. Algorithms for finding small attractors in boolean networks. *EURASIP Journal on Bioinformatics and Systems Biology*, 2007(20180), 2007.

Boolean networks stabilization problem

M. Zh. Zholbaryssov, Yu. S. Shutkin (Moscow State University)

Problem of Boolean networks stabilization is considered. A criterion of stabilization is found depending on fixed components of Boolean network: graph, Boolean transition functions, initial state, order of update.

Keywords: Boolean networks, stabilization.

Линейные автоматы над полем рациональных чисел

Ронжин Д.В.

Исследуется класс линейных автоматов над полем рациональных чисел. В указанном классе доказано отсутствие конечной K-полной системы, отсутствие конечных Σ полных с добавкой определенного вида систем, выделен бесконечный K-базис и бесконечный Σ-базис, а так же бесконечная Σ-полная система не содержащая Σ-базиса.

Ключевые слова: линейные автоматы, поле рациональных чисел, операции композиции, операции суперпозиции, K-замыкание, Σ-замыкание.

1. Введение

Настоящая работа посвящена классу линейных автоматов над полем рациональных чисел. Ранее, другими авторами [1-4] рассматривался класс линейных автоматов над конечными полями. В этих работах для рассматриваемого класса был использован аппарат формальных степенных рядов. Показано, что линейный автомат осуществляет преобразование таких рядов, и в терминах рядов были описаны операции композиции. Классическими проблемами теории автоматов являются проблемы выразимости и полноты по операциям композиции и суперпозиции (K и Σ соответственно)[5]. В настоящей работе для класса линейных автоматов над рациональными числами показано отсутствие конечной K-полной системы, найден бесконечный базис и бесконечная K-полная система, из которой нельзя выделить базис, получен бесконечный Σ-базис.

2. Используемые понятия и обозначения

Часть обозначений, используемых в этой работе введены в [1-4].

Будем обозначать множество рациональных чисел через $\mathbb{Q}$, а множество целых чисел через $\mathbb{Z}$.
Инициальным автоматом с n входами и m состояниями над полем рациональных чисел будем называть следующую упорядоченную шестерку:

$$V = (\mathbb{Q}^n, \mathbb{Q}^m, \mathbb{Q}, \vec{\varphi}, \psi, \vec{q_0}), \text{ где}$$

1) $\mathbb{Q}^n$ - входной алфавит $\mathbb{Q}^m$ - алфавит состояний и $\mathbb{Q}$ - выходной алфавит автомата

2) $\vec{\varphi} = (\varphi_1, ..., \varphi_m)$, где $\varphi_i \colon \mathbb{Q}^n \times \mathbb{Q}^m \to \mathbb{Q}$, $i \in [1, m]$ - функции переходов

3) $\psi \colon \mathbb{Q}^n \times \mathbb{Q}^m \to \mathbb{Q}$ - функция выхода

4) $\vec{q_0} = (q_1^0, q_2^0, ..., q_m^0), q_i^0 \in \mathbb{Q}, i \in [1, m]$ - вектор начальных состояний.

Назовем *основной системой* $\mathbf{B}$ следующее бесконечное множество линейных автоматов над $\mathbb{Q}$:

$$\mathbf{B} = \{\xi_0, \xi_1, V_+, V_{\cdot c} | c \in \mathbb{Q}\}, \text{ где}$$

1) ξ_0 - задержка с нулевым начальным состоянием над $\mathbb{Q}$.

2) ξ_1 - задержка с единичным начальным состоянием над $\mathbb{Q}$.

3) V_+ - сумматор с двумя входами над $\mathbb{Q}$.

4) $V_{\cdot c}$ - автомат-умножитель на константу c с одним входом.

Для произвольного множества линейных автоматов V будем обозначать замыкание этого множества по операциям суперпозиции и композиции через $K[V]$ и $\Sigma[V]$ соответственно.
Множество *линейных автоматов* над полем рациональных чисел - $L(\mathbb{Q})$ определяется как:

$$L(\mathbb{Q}) = K[\mathbf{B}]$$

Множество всевозможных формальных степенных рядов с коэффициентами из $\mathbb{Q}$ с формальной переменной ξ будем обозначать следующим образом:

$$\mathbb{Q}_\xi^\infty = \{\alpha(\xi) = a_0 + a_1 \cdot \xi + a_2 \cdot \xi^2 + ... + a_k \cdot \xi^k + ... | a_i \in \mathbb{Q}, i \in \mathbb{N} \cup \{0\}\}$$

Для удобства записи, где возможно, будем обозначать ряд $\alpha(\xi)$ через α.

Операции сложения, вычитания и умножения формальных степенных рядов определяются покомпонентно.

Формальный степенной ряд $\alpha = \sum_{i=1}^{\infty} a_i \cdot \xi^i \in \mathbb{Q}_\xi^\infty$ называется обратимым при условии $a_0 \neq 0$, (т.е. $\alpha(0) \neq 0$), и обратным к нему является ряд $\beta \in \mathbb{Q}_\xi^\infty$, такой что $\alpha \cdot \beta = 1$.

Будем называть множеством дробно-рациональных функций от переменной ξ с коэффициентами из поля рациональных чисел, со свободным членом в знаменателе отличным от нуля следующее множество:

$$\mathbf{R}(\mathbb{Q}) = \{\tfrac{P(\xi)}{Q(\xi)} | \text{ где } P(\xi), Q(\xi) \text{ - многочлены от } \xi \text{ над } \mathbb{Q}, \text{причем } (P(\xi), Q(\xi)) = 1 \text{ и } Q(0) \neq 0\}.$$

Скажем что $R = \frac{P(\xi)}{Q(\xi)} \in \mathbf{R}(\mathbb{Q})$ кратно ξ, если $P(\xi)$ делится на ξ без остатка как многочлен.

Несложно видеть, что элементы $\mathbf{R}(\mathbb{Q}) \subset \mathbb{Q}_\xi^\infty$, поскольку многочлены являются подмножеством формальных степенных рядов и знаменатели в указанных дробях - обратимы.

Пусть $V \in L(\mathbb{Q})$ - автомат с n входами из множества линейных автоматов над рациональными числами. Будем говорить, что на i-й вход автомата V подается степенной ряд $\alpha_i \in \mathbb{Q}_\xi^\infty$, $\alpha_i = \sum_{j=0}^{\infty} a_j \cdot \xi^j$, если $x_i(t) = a_t$, и ряд α_i будем называть *i-м входным рядом.*

Выходным рядом автомата V будем называть формальный степенной ряд $\beta \in Q_\xi^\infty$, $\beta = \sum_{j=0}^{\infty} b_j \cdot \xi^j$, такой что $\psi(t) = b_t$. В таком случае V реализует отображение вида $V\colon (\mathbb{Q}_\xi^\infty)^n \to \mathbb{Q}_\xi^\infty$.

Обозначим через $x_i(t) \in \mathbb{Q}$ значение, которое подается на i-й вход автомата V в момент времени t. Будем говорить, что автомат $V \in L(\mathbb{Q})$ *зависит от i-го входа непосредственным образом*, если выходная функция $y(t)$ автомата V существенным образом зависит от значения $x_i(t)$ в момент времени t для любого t. Нетрудно видеть, что отсутствие непосредственной зависимости i-го входа позволяет применять к нему операцию обратной связи.

Автомат $V \in L(\mathbb{Q})$ будем называть *константным автоматом* в случае, если его выходной ряд не зависит от входных значений. Последовательность выходных символов константного автомата будем называть *константным выходом.*

3. Вспомогательные утверждения

Нам понадобятся следующие леммы, которые мы приводим без доказательств.

Лемма 1. *Пусть $V \in L(\mathbb{Q})$ - линейный автомат с n входами, который реализует преобразование $V\colon (\mathbb{Q}_\xi^\infty)^n \to \mathbb{Q}_\xi^\infty$, и для некоторых $R_i \in \boldsymbol{R}(\mathbb{Q})$, при произвольных входных рядах $\alpha_i \in \mathbb{Q}_\xi^\infty, i \in [1,n]$ выходной ряд β имеет вид:*

$$\beta = R_0 + R_1 \cdot \alpha_1 + R_2 \cdot \alpha_2 + ... + R_n \cdot \alpha_n.$$

Тогда V не зависит от j-го входа непосредственным образом тогда и только тогда, когда R_j кратно ξ, $j \in [1,n]$.

Лемма 2. *Для всякого линейного автомата $V \in L(\mathbb{Q})$ с n входами существуют $R_i \in \boldsymbol{R}(\mathbb{Q}), i \in [0,n]$, такие что для любых входных рядов $\alpha_i \in \mathbb{Q}_\xi^\infty, i \in [1,n]$ выходной ряд $\beta \in \mathbb{Q}_\xi^\infty$ имеет вид:*

$$\beta = R_0 + R_1 \cdot \alpha_1 + R_2 \cdot \alpha_2 + ... + R_n \cdot \alpha_n,$$

Лемма 3. *Любой формальный степенной ряд вида*

$$\beta = a_0 + \textstyle\sum_{k=1}^{\infty} 0 \cdot \xi^t = a_0; a_0 \in \mathbb{Q}$$

реализуется некоторым константным автоматом без входа.

Лемма 4. *Пусть $f\colon (\mathbb{Q}_\xi^\infty)^n \to \mathbb{Q}_\xi^\infty$ и существуют такие P_i - многочлены от формальной переменной ξ над $\mathbb{Q}$, $0 \le i \le n$, что при любых $\alpha_i \in \mathbb{Q}_\xi^\infty, i \in [1,n]$ выполняется:*

$$f(\alpha_1, ..., \alpha_n) = \beta = P_0 + P_1 \cdot \alpha_1 + P_2 \cdot \alpha_2 + ... + P_n \cdot \alpha_n.$$

Тогда существует $V \in L(\mathbb{Q})$ - линейный автомат с n входами, который реализует f, т.е. выходной ряд V совпадает с β при любых входных рядах α_i, $i \in [1,n]$.

Лемма 5. *Пусть $f\colon (\mathbb{Q}_\xi^\infty)^n \to \mathbb{Q}_\xi^\infty$ и существуют такие $R_i = \frac{P_i}{Q_i} \in \boldsymbol{R}(\mathbb{Q}), i \in [0,n]$, что при любых $\alpha_i \in \mathbb{Q}_\xi^\infty, i \in [1,n]$ выполняется:*

$$f(\alpha_1, ..., \alpha_n) = \beta = R_0 + R_1 \cdot \alpha_1 + R_2 \cdot \alpha_2 + ... + R_n \cdot \alpha_n.$$

Тогда существует $V \in L(\mathbb{Q})$ - линейный автомат с n входами, который реализует отображение f.

Пример

Рассмотрим формальный степенной ряд, который представляет последовательность чисел Фибоначчи:

$$\alpha_F = 1 + 1 \cdot \xi + 2 \cdot \xi^2 + 3 \cdot \xi^3 + 5 \cdot \xi^4 + 8 \cdot \xi^5 ... + a_k \cdot \xi^k + ..., \text{ где}$$
$$a_k = a_{k-1} + a_{k-2}, k \geq 2$$

Нетрудно вычислить, что обратным к нему является ряд:

$$\frac{1}{\alpha_F} = 1 - \xi - \xi^2$$

Следовательно:

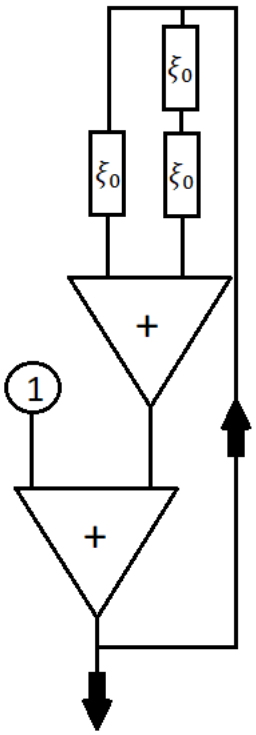
$$\alpha_F = \frac{1}{1-\xi-\xi^2}.$$

Построим константный линейный автомат, реализующий данный ряд.

Умножим это выражение на знаменатель, перенесем все коэффициенты кратные ξ вправо и заменим α_F в правой части уравнения на ряд α, запомнив что необходимо будет добавить обратную связь в конце построения на указанный новый вход. Получим уравнение:

$$\alpha_F = 1 + (\xi + \xi^2)\alpha.$$

Поскольку $(\xi + \xi^2)$ - многочлен степени 2, мы можем реализовать умножение на него согласно лемме 4. Подставим на один вход сумматора этот автомат, а на другой - автомат, реализующий константу 1 в первый такт, и 0 во все остальные такты, а так же применим операцию обратной связи. Получим автомат, изображенный на схеме, реализующий последовательность чисел Фибоначчи.



4. Основные результаты

Теорема 1. *В классе $L(\mathbb{Q})$ не существует конечной K-полной системы.*

Доказательство: Рассмотрим некоторую конечную систему линейных автоматов $\mathbf{V} = \{V_1, V_2, ..., V_m\}$. Рассмотрим $\alpha_{1,j}, ..., \alpha_{n_j,j} \in \mathbb{Q}_\xi^\infty$, при подаче которых на входы автомата V_j с n_j входами на выходе получается ряд:

$$\beta_j = R_{0,j} + R_{1,j} \cdot \alpha_{1,j} + R_{2,j} \cdot \alpha_{2,j} + ... + R_{k_j,j} \cdot \alpha_{n_j,j},$$

$$R_{i,j} = \frac{P_{i,j}}{Q_{i,j}} \in \mathbf{R}(\mathbb{Q}), 1 \leq j \leq m, 0 \leq i \leq n_j$$

Множество всех указанных $R_{i,j}$ обозначим через $\mathbf{R}$. Пусть $R_j \in \mathbf{R}$, тогда:

$$R_j = \sum_{k=0}^{\infty} a_{j,k} \cdot \xi^k, a_{j,k} \in \mathbb{Q}$$

Согласно указанным обозначениям, определим следующее множество коэффициентов:

$$\mathbf{V}_0 = \{a_{j,0} = \frac{p_{j,i}}{q_{j,i}} | j \in [1,n], i \in [1,n_j], (p_{j,0}, q_{j,0}) = 1\}$$

Рассмотрим $V \in K[\mathbf{V}]$ - автомат с k входами. Пусть $\mathbf{0} = \sum_{k=0}^{\infty} 0 \cdot \xi^k$. Подадим на входы V ряды $\alpha_j = \mathbf{0}, j \in [1,k]$. Тогда на выходе V получается ряд:

$$\beta = R'_0 = \sum_{k=0}^{\infty} b_k \cdot \xi^k,\ R'_0 \in \mathbf{R}(\mathbb{Q})$$

Заметим, что R'_0 получена из некоторых $R_{i,j}$ путем конечного числа применений следующих операций:

1) Сложение двух имеющихся на некотором шаге дробно-рациональных R_1 и R_2.

2) Умножение двух имеющихся на некотором шаге дробно-рациональных R_1 и R_2.

3) Деление образованной на некотором шаге дробно-рациональной R_1 на $1 - R_2$, где R_2 делится на ξ. *(Заметим, что эта операция не влияет на свободный член ряда R_1, поскольку, по определению деления, происходит умножение свободного члена на единицу).*

Очевидно, что b_0 будет получаться из элементов множества $\mathbf{V}_0$ применением операций сложения, вычитания, умножения и деления.

В силу конечности системы $\mathbf{V}_0$ мы можем выбрать число A следующим образом: $A = max_{j \in [1,n]}(q_{j,0})$ либо, если все $q_{j,0} = 1$, положить A = 2.

Однако в таком случае мы никогда не получим $b_0 = \frac{1}{A'}$, где $A' > A$ и A' - простое. Следовательно $\mathbf{V}$ - не полна. Теорема доказана.

Теорема 2. *В классе $L(\mathbb{Q})$ не существует конечной системы автоматов $\boldsymbol{V}$, такой что $\Sigma[\boldsymbol{V} \cup \boldsymbol{B}] = L(\mathbb{Q})$.*

Доказательство: Для начала покажем, что если $R_1, R_2 \in \mathbf{R}(\mathbb{Q})$, причем $R_1 \neq R_2$ как функции, тогда отображения $R_1 \cdot \alpha$ и $R_2 \cdot \alpha$ будут выдавать различные ряды при некоторых α.

В самом деле, предположим противное, тогда $(R_1 - R_2) \cdot \alpha = \mathbf{0}$, где

$$\mathbf{0} = 0 + 0 \cdot \xi + ... + 0 \cdot \xi^n + ... \in \mathbb{Q}^{\infty}_{\xi}.$$

Однако поскольку $(R_1 - R_2) \neq 0$, существует индекс i такой, что

$$(R_1 - R_2) = a_0 + a_1 \cdot \xi + a_2 \cdot \xi^2 + ... + a_i \cdot \xi^i + ..., \text{ причем } a_i \neq 0.$$

Рассмотрим $(R_1 - R_2)(\mathbf{1})$, где:

$$\mathbf{1} = 1 + \xi + \xi^2 + ...\xi^n + ... \in \mathbb{Q}_\xi^\infty$$

Отсюда сразу следует, что $(R_1 - R_2)(\mathbf{1}) \neq \mathbf{0}$, а это противоречит нашему предположению.

Рассмотрим некоторую конечную систему линейных автоматов из $L(\mathbb{Q})$:

$$\mathbf{V} = \{V_1, V_2, ..., V_m\}$$

Согласно лемме 2, V_j с n_j входами на входных рядах $\alpha_{k,j} \in \mathbb{Q}_\xi^\infty$ реализует выходной ряд:

$$\beta_j = R_{0,j} + R_{1,j} \cdot \alpha_{1,j} + R_{2,j} \cdot \alpha_{2,j} + ... + R_{n_j,j} \cdot \alpha_{n_j,j},$$

$$R_{i,j} = \frac{P_{i,j}}{Q_{i,j}} \in \mathbf{R}(\mathbb{Q}),\ 1 \leq j \leq m.$$

Определим множество из неприводимых многочленов $\mathbf{V}_Q$:

$$\mathbf{V}_Q = \{P(\xi) | P(\xi) - \text{ неприводимый многочлен: } \exists i, j : P(\xi) \text{ делит } Q_{i,j}\}$$

Пусть $V \in \Sigma[\mathbf{V} \cup \mathbf{B}]$ - автомат с k входами. При подаче на вход V рядов $\alpha_1, ..., \alpha_k \in \mathbb{Q}_\xi^\infty$ выходной ряд имеет вид:

$$\beta = R'_0 + R'_1 \cdot \alpha_1 + ... + R'_k \cdot \alpha_k$$

Рассмотрим выходной ряд при $\alpha_i = \mathbf{0}, i \in [1, k]$. Тогда $\beta = R'_0$, где $R'_0 \in \mathbf{R}(\mathbb{Q})$, полученная в результате построения V из некоторых $R_{i,j}$ путем применения конечного числа операций:

1) Сложение двух имеющихся на некотором шаге R_1 и R_2.

2) Умножение двух имеющихся на некотором шаге R_1 и R_2.

3) Умножение произвольной R_1 на $c \in \mathbb{Q}$.

4) Умножение произвольной R_1 на формальную переменную ξ.

5) Прибавление произвольной константы к R_1.

Нетрудно видеть, что используя указанный набор операций мы будем получать $R'_0 \neq \frac{P'}{Q'} \in \mathbf{R}(\mathbb{Q})$, где $Q' \notin \mathbf{V}_Q$. Следовательно $\mathbf{V}$ не полна. Теорема доказана.

При построении класса линейных автоматов мы использовали полную, но избыточную *основную систему* $\mathbf{B}$, которая не является базисной.

Следующие далее утверждения касаются вопроса существования базисов в классе линейных автоматов над полем рациональных чисел.

Теорема 3. *В классе $L(\mathbb{Q})$ можно выделить бесконечный K - базис.*

Доказательство: Рассмотрим счетное множество линейных автоматов $\mathbf{B_1}$:

$$\mathbf{B_1} = \{\xi_1, V_+, V_{\cdot(-1)}, V_{\cdot\frac{1}{p}} \mid p \text{ - простые числа}\}$$

Покажем что $K[\mathbf{B_1}] = L(\mathbb{Q})$, а именно, сведем все к заведомо полной системе $\mathbf{B}$:

1) $V_{\cdot 0}$ получим из V_+ и $V_{\cdot(-1)}$, подстановкой $V_{\cdot(-1)}$ на один из входов V_+ и отождествлением входов полученного автомата.

2) $V_{\cdot n}$, где $n \neq 0$ - целое число несложно получить из $V_{\cdot(-1)}$ и V_+.

3) $V_{\cdot\frac{a}{b}}$, где $b = p_1^{s_1} \cdot p_2^{s_2} \cdot ... \cdot p_k^{s_k}$, p_i -простые числа, s_i - натуральные, получим очевидным образом из $V_{\cdot(-1)}$, V_+,$V_{\cdot\frac{1}{p_i}}$, $i \in [1, k]$.

Таким образом мы целиком восстановим $\mathbf{B}$, а значит построим все автоматы в классе $L(\mathbb{Q})$.

Покажем что $\mathbf{B_1}$ - базис:

1) $K[\mathbf{B_1} \setminus \xi_1]$ содержит только автоматы сохраняющие ноль в первый момент времени.

2) $K[\mathbf{B_1} \setminus V_+]$ не содержит автоматов, зависящих более чем от одного входа.

3) $K[\mathbf{B_1} \setminus V_{\cdot\frac{1}{p}}]$ где p-простое, не содержит автоматов, выдающих в первый такт константу $\frac{1}{p}$.

4) $K[\mathbf{B_1} \setminus V_{\cdot(-1)}]$ не содержит автоматов, выдающих в первый такт отрицательное число.

Следовательно $\mathbf{B_1}$ - K - базис. Теорема доказана.

Теорема 4. *В классе $L(\mathbb{Q})$ существует бесконечная Σ -полная система, не содержащая Σ-базиса.*

Доказательство: Рассмотрим множество линейных автоматов $\mathbf{B_2}$:

$\mathbf{B_2} = \{\xi_1, V_+, V_{\cdot(-1)}, V_{\cdot\frac{1}{p_1 \cdot p_2 \cdot \ldots \cdot p_k}} \mid p_i$ - подряд идущие простые числа$\}$

Очевидно, что $\mathbf{B_1} \subset \Sigma[\mathbf{B_2}]$, т.к. всякий $V_{\cdot\frac{1}{p_i}}$ может быть получен из $V_{\cdot\frac{1}{p_1 \cdot p_2 \cdot \ldots \cdot p_k}}$, где $k \geq i$ и V_+.

В указанной системе невозможно выделить базис, поскольку любое конечное подмножество $\mathbf{B_2}$ не является полным, так как не позволит нам получить все умножители на простые числа, а из любого бесконечного подмножества всегда можно удалить элементы без потери свойства системы быть полной в $L(\mathbb{Q})$ по операция композиции. Теорема доказана.

Теорема 5. *В классе $L(\mathbb{Q})$ существует бесконечный Σ-базис.*

Доказательство: Определим автоматы-умножители на многочлены и рациональные функции:

1) $V_{\cdot P(\xi)}$ - линейный автомат с одним входом, выходной ряд которого равен:

$$\beta = P(\xi) \cdot \alpha,$$

где $P(\xi)$ - многочлен от ξ над $\mathbb{Z}$, α - входной ряд.

2) $V_{\cdot\frac{P(\xi)}{Q(\xi)}}$ - линейный автомат с одним входом, выходной ряд которого равен:

$$\beta = \frac{P(\xi)}{Q(\xi)} \cdot \alpha,$$

где $\frac{P(\xi)}{Q(\xi)} \in \mathbf{R}(\mathbb{Q})$, α - входной ряд.

Назовем системой $\mathbf{B_3}$ следующую систему:

$$\mathbf{B_3} = \mathbf{B_1} \cup \mathbf{B_4},$$

где $\mathbf{B_4} = \{V_{\cdot\frac{1}{Q(\xi)}} \mid \text{Q - неразложимый, } deg(Q(\xi)) > 0, Q(0) = 1\}$

Условие $Q(0) = 1$ введено для того, что бы избавиться от многочленов, равных с точностью до ассоциированного (умножения на +1 и -1) и добиться единичного содержания многочленов в знаменателе.
Покажем, что система $\mathbf{B_3}$ полна.
В $\Sigma[\mathbf{B_1}]$ содержатся все $V_{\cdot c}$ и $V_{\cdot P(\xi)}$. Для полноты системы необходимо реализовать произвольный $V_{\cdot\frac{P(\xi)}{Q(\xi)}}$, $\frac{P(\xi)}{Q(\xi)} \in \mathbf{R}(\mathbb{Q})$, т.к. тогда при подаче на входы $\alpha_1, ..., \alpha_n \in \mathbb{Q}_\xi^\infty$ реализуется любое отображение вида:

$$\beta = R_0 + R_1 \cdot \alpha_1 + R_2 \cdot \alpha_2 + ... + R_n \cdot \alpha_n$$

Очевидно, что любая $R_i = \frac{P_i}{Q_i} \in \mathbf{R}(\mathbb{Q})$, может быть представлена как $R_i = \frac{P_i'}{Q_i'} \in \mathbf{R}(\mathbb{Q})$, где $Q_i'(0) = 1$. Кольцо многочленов над целыми числами - факториально[6], поэтому любая дробно рациональная функция $R_i' = \frac{P_i'}{Q_i'}$ представима суммой дробно-рациональных, с неразложимыми в знаменателе, а значит $R_i' \in \Sigma[\mathbf{B_3}]$, поскольку всякий $V_{\cdot P_i'(\xi)} \in \Sigma[\mathbf{B_3}]$.

Поскольку мы реализовали произвольные $V_{\cdot\frac{P(\xi)}{Q(\xi)}}$, несложно получить константные отображения, реализующие ряд R_0, т.к. используя $V_+, V_{\cdot(-1)}$ и ξ_1 можно получить константный автомат с выходным рядом вида:

$$\beta = 1 + \textstyle\sum_{k=1}^{\infty} 0 \cdot \xi^k$$

Полнота системы $\mathbf{B_3}$ доказана.
Покажем, что система $\mathbf{B_3}$ является базисом.

1) $\Sigma[\mathbf{B_3} \setminus V_+]$ не содержит автоматов, зависящих более чем от одного входа.

2) $\Sigma[\mathbf{B_3} \setminus V_{\cdot(-1)}]$ не содержит автоматов, выдающих отрицательные числа в первый момент времени.

3) $\Sigma[\mathbf{B_3} \setminus V_{\cdot\frac{1}{p}}]$ не содержит автоматов, выдающих константу $\frac{1}{p}$ в первый момент времени.

4) $\Sigma[\mathbf{B_3} \setminus \xi_1]$ не содержит автоматов, реализующих формальные степенные ряды вида $R \cdot x$, где R - неправильная рациональная функция от ξ (степень числителя больше степени знаменателя).

5) $\Sigma[\mathbf{B_3} \setminus V_{\cdot\frac{1}{Q(\xi)}}]$ не содержит константного автомата, реализующего выходной ряд $\frac{1}{Q(\xi)}$.

Таким образом, $\mathbf{B_3}$ - базис. Теорема доказана.

Автор выражает искреннюю признательность А.А.Часовских за постановку задачи, а так же ценные советы и замечания.

Список литературы

[1] Гилл А. «Линейные последовательные машины.» — М.: Наука, 1974.

[2] Кудрявцев В. Б. Алешин С. В. Подколзин А. С. «Элементы теории автоматов» — М.: Издательство Московского Университета – 1978

[3] Часовских А. А. «О полноте в классе линейных автоматов» // Математические вопросы кибернетики –Выпуск 03 – С. 140–166 – 1991.

[4] Часовских А. А. «О полноте в классе линейных 2-адических автоматов» // Интеллектуальные системы – Том 20 – Выпуск 4 – С. 209–227 – 2016.

[5] Кудрявцев В. Б. Алешин С. В. Подколзин А. С. «Введение в теорию автоматов» — М.: Наука – Главная редакция физико-математической литературы – 1985

[6] Кострикин А. И. «Введение в алгебру. Часть III. Основные структуры: Учебник для вузов. -3-е изд.»— М.: Физматлит – 2004

Linear automata over rational numbers field

Ronzhin D.V.

We consider a class of linear automata over the field of rational numbers. In this class we prove there are no finite K-full systems and no finite Σ-full systems with infinte additive of special form. We construct an infinite K-basis and an infinite Σ-basis, and also an infinite Σ-full sistem which contains no Σ-basis.

Keywords: linear automata, rational numbers field, composition operations, superposition operations, K-closure, Σ-closure.

К сведению авторов публикаций в журнале «Интеллектуальные системы. Теория и приложения»

В соответствии с требованиями ВАК РФ к изданиям, входящим в перечень ведущих рецензируемых научных журналов и изданий, в которых могут быть опубликованы основные научные результаты диссертаций на соискание ученой степени доктора и кандидата наук, статьи в журнал «Интеллектуальные системы. Теория и приложения» предоставляются авторами в следующей форме:

1. Статьи, набранные в пакете LaTeX, предоставляются к загрузке через WEB-форму http://intsysjournal.org/generator_form .
2. К статье прилагаются файлы, содержащие название статьи на русском и английском языках, аннотацию на русском и английском языках (не более 50 слов), список ключевых слов на русском и английском языках (не более 20 слов), информация об авторах: Ф.И.О. полностью, место работы, должность, ученая степень и/или звание (если имеется), контактные телефоны (с кодом города и страны), e-mail, почтовый адрес с индексом города (домашний или служебный).
3. Список литературы оформляется в едином формате, установленном системой Российского индекса научного цитирования.
4. За публикацию статей в журнале «Интеллектуальные системы. Теория и приложения» с авторов (в том числе аспирантов высших учебных заведений) статей, рекомендованных к публикации, плата не взимается. Оттиски статей авторам не предоставляются. Журнал распространяется по подписке, экземпляры журнала рассылаются подписчикам наложенным платежом. Условия подписки публикуются в каталоге НТИ «Роспечать», индекс журнала 64559.
5. Доступ к электронной версии последнего вышедшего номера осуществляется через НЭБ «Российский индекс научного цитирования». Номера, вышедшие ранее, размещаются на сайте http://intsysjournal.org, и доступ к ним бесплатный. Там же будут размещены аннотации всех публикуемых статей.

Подписано в печать: 15.12.2017
Дата выхода: 25.12.2017
Тираж: 200 зкз.
Цена свободная
Свидетельство о регистрации СМИ: ПИ № ФС77–58444 от 25 июня 2014 г., выдано Федеральной службой по надзору в сфере связи, информационных технологий и массовых коммуникаций(Роскомнадзор).