УДК 004.421.2:519.178

ОПТИМИЗАЦИЯ АЛГОРИТМОВ ИДЕНТИФИКАЦИИ ГРАФОВОГО ИЗОМОРФИЗМА

С.В. Москаленко, Ю.А. Гатчин

Предложен ряд оптимизирующих моментов для современных алгоритмов идентификации графового изоморфизма, призванных увеличить скорость работы данных алгоритмов с сохранением точности вычисления результатов.

Ключевые слова: теория графов, распознавание графических образов, определение межграфового изоморфизма.

Введение

В последнее время растет объем обрабатываемой инженерной документации, в частности, приборостроительных и машиностроительных чертежей. Эта тенденция возникла в связи с активизацией использования электронной версии инженерной документации в САПР [1]. Довольно частой практикой для инженеров и проектировщиков является обращение к уже существующей документации, однако данный процесс всегда оказывается трудоемким, так как требует просмотра всей базы чертежей. Для упрощения поиска часто используется текстовая составляющая документа, когда происходит просмотр по ключевым словам. Такой подход весьма эффективен, но существуют трудности, связанные с получением этой текстовой информации (необходимо вмешательство человека), кроме того, набор ключевых слов не всегда точно может описать содержание чертежа. Поиск требуемого изображения в БД есть не что иное, как проблема сопоставления графических объектов, которая может быть решена путем вычисления степени соответствия входного объекта объекту в базе данных (БД) и сведена к процессу сопоставления графов. Для этого инженерная документация сначала приводится к представлению атрибутивных графов, где вершины графов отображают примитивы, полученные из исходных документов (линии, кривые) [2], а топологические связи

между этими примитивами описаны ребрами графов. Далее вершины графов задаются метками, которые вычисляются по некоторой функции. Метки, как правило, являются целыми числами и используются для сравнения входных графов с заданными в БД графами. Таким образом, сравнивая ребра и метки графов, можно обнаружить чертежи, находящиеся в БД, целиком или частично схожие с текущим чертежом.

В статье рассматривается подкласс процесса сопоставления графов – проверка графов на изоморфизм (далее графовый изоморфизм или ГИ). ГИ можно определить следующим образом: два графа $G=(V_1, E_1)$ и $H=(V_2, E_2)$ изоморфны, если между парами множеств их вершин, ребер и дуг существуют взаимно однозначные соответствия, сохраняющие смежность и ориентацию для дуг.

Понятие ГИ находит применение в различных направлениях – при определении поврежденных клеток на медицинских снимках, поиске изоморфных химических соединений [3], сравнении пар кинематических цепей [4], и т.д. Для решения проблемы поиска ГИ, однако, не существует достаточно эффективных алгоритмов, обладающих полиномиальным временем исполнения, даже если алгоритм ограничен по области применения (например, операции проходят над строго однородными графами). В работе [5] представлен эффективный алгоритм, обладающий сложностью вычисления $O(n^{1/3} \log n)$, но оперирующий лишь со строго однородными графами. В работе [6] приведен еще один алгоритм, являющийся линейным для большинства случайных графов. Однако последний имеет ряд недостатков, которые связаны с невозможностью разрешения неопределенных ситуаций, возникающих при обнаружении множества идентичных друг другу вершин. В рамках данной статьи предложены 3 оптимизационных момента, призванные ускорить поиск ГИ в современных алгоритмах сопоставления графов.

Реализация алгоритма

Сформулируем обобщенный алгоритм работы многих современных систем идентификации ГИ [6, 7]. На старте алгоритм имеет некоторый входной помеченный граф (вершинам которого приписаны метки), а также база данных (БД) эталонных помеченных графов. Требуется найти граф в БД, изоморфный входному графу (см. определение 1, с. 101). Вначале алгоритм строит связи между всеми вершинами входного графа со всеми вершинами эталонного графа в БД, которые имеют одинаковые метки. Ситуацию, когда одна вершина в одном графе идентична с несколькими вершинами в другом, называют неопределенной. На следующих шагах после обнаружения удовлетворяющего эталона алгоритм пытается сократить число ложных связей. Для снижения числа случаев ошибочного определения изоморфизма отдельные алгоритмы, например [6], реализуют вовлечение информации о соседних вершинах в функцию вычисления меток.

Итак, обобщенный алгоритм идентификации ГИ выглядит следующим образом.

- Шаг 1 (предобработка). Для входного графа и для всех графов в БД высчитываются степени у вершин и собирается другая информация, необходимая для последующих шагов, например, тип линии для примитивов инженерных чертежей. Далее полученные данные используются в качестве аргументов функции вычисления меток. На данном шаге также может собираться информация о соседних вершинах (количество степеней и пр.) для включения в функцию вычисления меток.
- Шаг 2 (построение связей). Связи строятся между двумя сравниваемыми графами для всех пар вершин, которые имеют одинаковые метки. Пример таких связей приведен на рис. 1, где различные метки показаны черным, белым или серым цветом, вершина *u* графа G связана с вершиной *v* графа H. Две вершины соединены потому, что метка от *u* совпадает с меткой от *v*, т.е. метка и степень вершины *u* равны соот-

ветствующим значениям вершины *v*, а каждый потомок (смежная вершина) от узла *u*, а именно *i, j, k*, имеет соответствующего потомка от узла *v* с такой же меткой и степенью. На рис. 1 потомки *i, j, k* графа *G* совпадают с *x, y, z* на графе *H*. Однако ясно видно, что графы *G* и *H* не изоморфны.

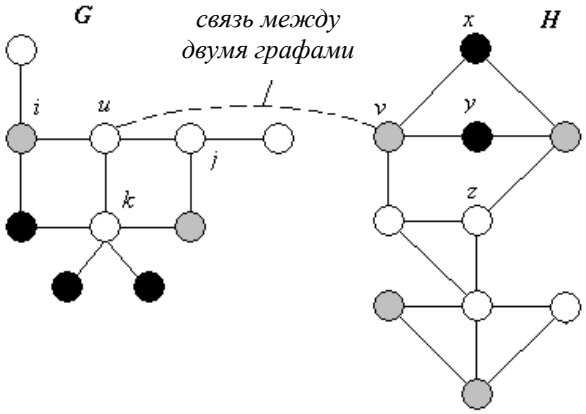

Рис. 1. Построение связи между двумя графами

– Шаг 3 (редукция энтропии). Данный шаг является самой затратной частью алгоритма по количеству вычислений. Происходит попытка разрешить неопределенные ситуации, если таковые возникли на предыдущем шаге, и сделать вывод о наличии изоморфизма между входным графом и эталонным графом.

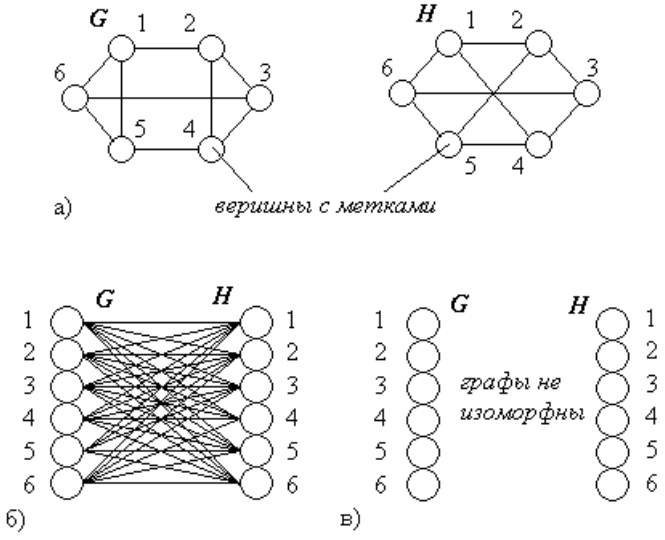

Рис. 2. Иллюстрация работы алгоритма поиска ГИ: а – шаг предобработки, даны два графа G и H; б – шаг построения связей; в – шаг редукции энтропии

На рис. 2 изображены все шаги представленного алгоритма. На этапе предобработки (рис. 2, а) для обоих графов вычисляются метки для каждой вершины. Здесь каждая вершина содержит метку с информацией о текущем узле, а также о его соседях. Вдобавок следует отметить, что все метки графов G и H одинаковы как внутри графов, так и между G и H. Это равенство обусловлено тем, что метки самих вершин изначально равны (все кружочки отмечены белым цветом), оба графа являются однородными (одинаковое число ребер, смежных с каждым узлом). Данное равенство дает высокую степень неопределенности, когда каждая вершина графа G может быть соединена с каждой вершиной H (рис. 2, б). Неопределенность разрешается на шаге редукции энтропии (рис. 2, в), когда стартует детальный анализ графов.

Пути оптимизации алгоритма

В предыдущем разделе был показан пример (рис. 2), когда при сопоставлении двух графов возникали избыточные неопределенные ситуации. Разрешение таких ситуаций – весьма трудоемкий процесс, требующий привлечения процедуры детального анализа графов, во время работы которой происходит последовательное продвижение по уровням графа (каждый уровень характеризуется набором всех вершин, смежных с любой текущей вершиной). Существует еще ряд проблем, связанных с трудозатратами, которые линейно возрастают по мере добавления в БД эталонных графов. В данном разделе приведены два критерия и один момент, которые могут быть вовлечены в любой алгоритм поиска ГИ в качестве оптимизирующих мер.

Ниже даны несколько ключевых определений.

Определение 1. Графовый изоморфизм. Даны 2 графа $G=(V_1, E_1)$ и $H=(V_2, E_2)$, где $|V_1|=|V_2|$.

Граф G изоморфен H ($G=H$), при $V_1 \rightarrow V_2$ $|\exists (u, v) E_1$ && $\exists (f(u), f(v)) \in E_2$ && $l(u) \in l(f(u))| \forall u \in V_1$, где l – функция вычисления метки для вершины.

Определение 2. Соответствие (идентичность) вершин. Пусть $u \in V_1$ – вершина графа G, а $v \in V_2$ – вершина графа H. Считается, что узел u идентичен узлу v, если метка от u равна метке от v.

Определение 3. Степенью неопределенности узла u, принадлежащего графу G, называется число узлов графа H, с которыми u идентичен. Обозначение: $deg(u)=m$, где m – число вершин $\in H$, которым соответствует узел u.

Критерий 1. При поиске в БД эталонного графа, изоморфного входному, для каждого графа в БД запускается алгоритм идентификации ГИ, описанный выше. Ясно, что это весьма трудоемкий подход, требующий большого количества вычислений. Для оптимизации поиска ГИ можно ввести условие: во время первого пробега (последовательного сопоставления входного графа с каждым эталонным) алгоритм идентификации ГИ должен запускаться лишь для эталонных графов с числом вершин, равным или меньшим числа вершин входного графа. Данное условие применимо лишь для графов, которые отражают графическую информацию, заложенную в изображениях (инженерная документация, чертежи). Для таких изображений характерно наличие шумов, сказывающихся появлением на графах дополнительных вершин и ребер. Действительно, при разрыве линии на чертеже или при появлении случайного штриха, вызванного недостатками процесса сканирования, появляются шумы, которые, однако, не повлияют на вывод о наличии ГИ.

На вход процедуры сравнения графов подаются G – входной граф, $\{H\}$ – множество эталонных графов в БД, $G=(V_1, E_1)$ и $H=(V_2, E_2)$.

```
1.Для каждого H
2.      если ( |V1| ≥ |V2| )
3.              выполняется алгоритм поиска ГИ
4.                      если (ГИ обнаружен)
5.                              тогда выход их программы с результатом «ГИ найден»
6.Для каждого H
7.      если ( |V1| < |V2| )
8.              выполняется алгоритм поиска ГИ
9.                      если (ГИ обнаружен)
10.                             тогда выход их программы с результатом «ГИ найден»
11.     Выход из программы с результатом «ГИ не найден»
```

Критерий 2. При поиске в БД эталонного графа, изоморфного входному, можно добавить еще одно условие запуска алгоритма идентификации ГИ: запуск должен про-

исходить лишь тогда, когда набор меток входного графа равен набору меток эталонного. Здесь под равенством понимается, что для каждого элемента набора эталонного графа существует идентичный элемент набора входного графа (набор – совокупность элементов, уникальных внутри данной совокупности).

$G=(V_1, E_1)$ – входной граф, $H=(V_2, E_2)$ – эталонный граф, $\{H\} \in$ БД.

$\exists$ набор меток F_1, заданных функцией $l(u) \rightarrow F_1$, $\forall u \in V_1$

& $\exists$ набор меток F_2, $l(v) \rightarrow F_2$, $\forall v \in V_2$

$\rightarrow \{ l(u)\} \geq \{ l(v)\}$

Количество элементов набора входного графа может быть больше или равно количеству элементов эталонного. Это допущение основано на природе графических изображений, получение которых может сопровождаться шумами (см. также критерий 1), обусловливающими увеличение числа вершин и ребер соответствующего графа.

```
1. Для каждого H
2.       если ({ l(u)} ≥ { l(v)})
3.             выполняется алгоритм поиска ГИ
4.                   если (ГИ обнаружен)
5.                         тогда выход их программы с результатом «ГИ найден»
6. Для каждого H
7.       если ({ l(u)} < { l(v)})
8.             выполняется алгоритм поиска ГИ
9.                   если (ГИ обнаружен)
10.                        тогда выход их программы с результатом «ГИ найден»
11.          Выход из программы с результатом «ГИ не найден»
```

Оптимизационный момент. Еще один момент может быть применен непосредственно в самом алгоритме идентификации ГИ. При этом на шаге предобработки графов, когда вычисляются степени для вершин, все узлы со степенью 2 должны быть удалены, а соответствующие две вершины, смежные с только что удаленным узлом, объединены в одну. Во время объединения двух вершин происходит формирование новой метки, связанной с новым узлом, которая хранит атрибуты удаленной вершины.

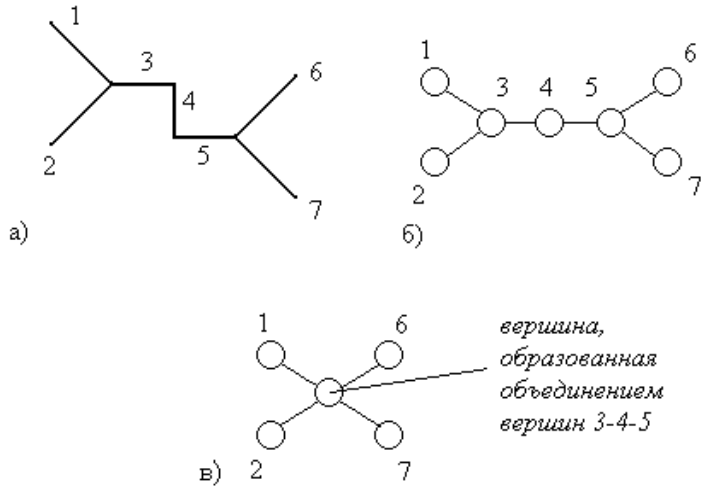

Рис. 3. Пример работы оптимизационного момента: а – входной чертеж с пронумерованными примитивами; б – графовое отображение чертежа; в – объединение вершин графа

На рис. 3 представлен пример работы данного момента. Дан фрагмент инженерного чертежа (рис. 3, а), разбитого на примитивы, каждому из которых приписан свой порядковый номер. Далее происходит переход к графовому представлению чертежа (рис.

3, б). Проанализировав последнее представление, можно сделать вывод о наличии избыточных узлов (узел 4 со степенью, равной 2), которые можно удалить, объединив две смежные вершины 3 и 5 в одну (рис. 3, в).

Использование этой оптимизационной меры сокращает количество узлов в графах, что повышает скорость работы алгоритма, при этом степень информативности графов не снижается. Вдобавок метки приобретают более содержательный характер, что снижает степень неопределенности (см. определение 3).

Заключение

В статье представлены оптимизационные меры (два критерия и один момент), которые могут быть включены в любой современный алгоритм поиска изоморфизма среди графов. Данные меры не снижают точность работы данных алгоритмов и в то же время являются весьма надежными.

Первые два критерия вовлекаются в процесс сопоставления входного графа с эталонными графами в БД. Здесь на ранних стадиях сопоставления происходит сравнение вершин и меток, что обеспечивает пропуск не удовлетворяющих критериям графов без их последующего детального анализа и, тем самым, рост производительности системы. Оптимизационный момент должен быть применен непосредственно в самом алгоритме идентификации ГИ, снижая степень неопределенности графов и повышая информативность меток, что дает более быструю и точную работу процедуры сопоставления графов. В дальнейших работах планируется реализация приведенных оптимизационных мер на языке программирования и проведение нагрузочного тестирования для исследования изменения производительности алгоритмов поиска ГИ.

Литература

1. Tombre K. Analysis of engineering drawings: state of the art and challenges // Graphics recognition. Algorithms and systems. V. 1389. – Lecture Notes. – Computer Science, Springer-Verlag, 1998. – P. 257–264.
2. Москаленко С.В., Гатчин Ю.А. Волновой алгоритм векторизации линейных растровых изображений // Научно-технический вестник СПбГУ ИТМО. – 2008. – № 51.
3. Xu J. A Generic Match Algorithm for structural homomorphism, isomorphism, and maximal common substructure match and its applications // J Chem Infor Comput Sci. – 1996. – № 36(1). – P. 25–34.
4. Rao A.C., Varada Raju D. Application of the Hamming number technique to detect isomorphism among kinematic chains and inversions // Mech. Mach. Theory. – 1991. – № 26 (1). – P. 55–75.
5. Spielman D.A. Faster isomorphism testing of strongly regular graphs: Technical report. – University of California Berkeley, Computer Science Division, Berkeley, California, 1996.
6. Abdulrahim M., Misra M. A Graph Isomorphism Algorithm for Object // Recognition, Pattern Analysis and Applic. – 1998. – V. 1. – № 3. – P. 189–201.
7. von der Malsburg C, Bienenstock E.A. Neural network for retrieval of superimposed connection patterns // Europhysics Let. – 1987. – № 3 (11). – P. 1243–1249.

Москаленко Станислав Владимирович – Санкт-Петербургский государственный университет информационных технологий, механики и оптики, аспирант, stan.moskalenko@gmail.com

Гатчин Юрий Арменакович – Санкт-Петербургский государственный университет информационных технологий, механики и оптики, доктор технических наук, профессор, зав. кафедрой, gatchin@mail.ifmo.ru