Компьютерные инструменты в образовании, 2024
№ 3: 5–13
УДК: 519.175.1
http://cte.eltech.ru
doi:10.32603/2071-2340-2024-3-5-13

алгоритмическая математика и математическое моделирование

# АЛГОРИТМ НАХОЖДЕНИЯ НАИБОЛЬШЕГО ОБЩЕГО ПОДГРАФА ДВУХ ОРИЕНТИРОВАННЫХ ГРАФОВ*

Чжоу Ц.[1], аспирант, ✉ st103098@student.spbu.ru, orcid.org/0009-0007-2350-987X

[1]Санкт-Петербургский государственный университет,
Университетская наб., д. 7–9, Санкт-Петербург, 199034, Россия

**Аннотация**

Проверка графов на изоморфизм имеет важное значение в теории графов и её прикладных областях, особенно по отношению к изоморфизму ориентированных графов, нахождение биекции которого представляет собой NP-трудную задачу.

Задача нахождения наибольшего общего подграфа двух ориентированных графов является сужением задачи изоморфизма предикатных формул, когда в элементарной конъюнкции содержится только один двухместный предикат. Ранее алгоритм был разработан для нахождения наибольшей общей подформулы двух элементарных конъюнкций. В данной работе представлены псевдокоды алгоритмов нахождения наибольшего общего подграфа для ориентированных графов и их пример использования.

**Ключевые слова:** *изоморфизм графов, максимальный общий ориентированный подграф, биекция графов.*

**Цитирование:** Чжоу Ц. Алгоритм нахождения наибольшего общего подграфа двух ориентированных графов // Компьютерные инструменты в образовании. 2024. № 3. С. 5–13. doi:10.32603/2071-2340-2024-3-5-13

## 1. ВВЕДЕНИЕ

Представление данных в виде графа широко распространено для моделирования исходных данных при решении многих дискретных задач [1–5]. Одной из важных задач является задача нахождения наибольшего общего подграфа двух графов. Эта задача является NP-трудной [6]. В настоящее время разрабатывается много различных алгоритмов для её решения.

Некоторые задачи с графами оказались тесно связаны с задачами, исходными данными для которых являются предикатные формулы [7].

В частности, в работе [8] доказана полиномиальная эквивалентность задач изоморфизма предикатных формул и изоморфизма графов.

В статье [9] рассматривается задача выделения наибольших общих подформул двух элементарных конъюнкций предикатных формул. В ней для решения этой задачи пред-

* Исследование проведено при финансовой поддержке программы Китайского совета по стипендиям, грант ID: 202108620001.

ставлен алгоритм **MCF1**. Вычислительная сложность этого алгоритма составляет $O(n^n)$, где $n$ — максимальное количество аргументов в формулах.

Идея предлагаемого ниже алгоритма нахождения наибольшего общего подграфа возникла после разработки алгоритма **MCF1**.

## 2. НЕОБХОДИМЫЕ ОПРЕДЕЛЕНИЯ

**Определение 1.** *Два ориентированных графа* $G_1 = (V_1, A_1)$ *и* $G_2 = (V_2, A_2)$ *изоморфны и обозначаются*

$$G_1 \cong G_2,$$

*если существует биекция* $f : V_1 \to V_2$ *такая, что любые две вершины* $u$ *и* $v$ *графа* $G_1$ *смежны тогда и только тогда, когда вершины* $f(u)$ *и* $f(v)$ *смежны в графе* $G_2$*, то есть* $\exists f\Big(f : V_1 \to V_2 \;\&\; f$ *— биекция* $\;\&\; \forall u\, v\,\big(u \in V_1 \;\&\; v \in V_1 \to ((u, v) \in A_1 \leftrightarrow ((f(u), f(v)) \in A_2))\big)\Big)$.

Биекцию $f$ будем записывать в виде множества пар $m_{G_1,\,G_2} = \{a_{i_1} : f(a_{i_1}),\ \ldots,\ a_{i_n} : f(a_{i_n})\}$ (где $n = |V_1|$), задающих изоморфизм. То есть если $b_{j_1} = f(a_{i_1}),\ \ldots,\ b_{j_n} = f(a_{i_n})$, то $m_{G_1,\,G_2} = \{a_{i_1} : b_{j_1},\ \ldots,\ a_{i_n} : b_{j_n}\}$.

**Определение 2.** *Граф* $R = (V_R, A_R)$ *называется наибольшим общим (с точностью до имён вершин) подграфом двух графов* $G_1$ *и* $G_2$*, если он изоморфен некоторым подграфам этих графов, но после добавления в него хоть одного ребра становится не изоморфным ни одному подграфу либо графа* $G_1$*, либо графа* $G_2$*.*

Имеется два отображения графа $R$ на графы $G_1$ и $G_2$: $m_{R,\,G_1} = \{x_1 : a_{i_1}, \ldots, x_n : a_{i_n}\}$ и $m_{R,\,G_2} = \{x_1 : b_{j_1}, \ldots, x_n : b_{j_n}\}$.

Пусть $R$ и $G$ — два ориентированных графа, причём именами вершин графа $R$ являются переменные, а именами вершин графа $G$ — только константы.

**Определение 3.** *Подстановки* $\{\overline{x} : \overline{b}\}$*, где* $\overline{x}$ *— список имён некоторых вершин из* $R$*,* $\overline{b}$ *— список имён некоторых различных вершин из* $G$*, называются частичным отображением* $R$ *на* $G$*, если результат применения этой подстановки к графу* $R$ *содержит подграф, совпадающий с некоторым подграфом* $G$*.*

Частичное отображение будем обозначать посредством $m_p$. Ниже все отображения будут частичными.

**Определение 4** [9]. *Две подстановки называются противоречивыми, если одной и той же переменной* $x$ *соответствуют две разные константы* $a_1$ *и* $a_2$*, то есть* $\{x : a_1, x : a_2\}$*, или разным переменным* $x_1$ *и* $x_2$*, соответствует одна и та же константа* $a$*, то есть* $\{x_1 : a, x_2 : a\}$.

Предлагаемый ниже алгоритм нахождения наибольшего общего подграфа двух ориентированных графов будем называть **MCOG** (Maximal Common Oriented subGraph). В **MCOG** применяется метод отслеживания с возвратом (**backtracking**), в котором вызывается алгоритм **CONS-T**. Псевдокоды **backtracking** и **CONS-T** описаны ниже как алгоритм 2 и алгоритм 3.

Метод **backtracking** заключается в рекурсивном поиске с попыткой расширить текущие дуги MCOG на каждом шаге, пока не будут найдены MCOG, удовлетворяющие заданному условию[1], или не будет установлено, что MCOG нет. В процессе поиска MCOG, если текущие дуги MCOG противоречат уже найденным, происходит возврат к предыдущему шагу, чтобы сделать другой выбор и продолжить поиск.

[1] Условие написано в тексте алгоритма 2 на строке 2.

Алгоритм **CONS-T** предназначен для формирования непротиворечивого дерева возможных отображений. **CONS-T** проверяет наличие противоречий между отображениями, где первое отображение было подтверждено в качестве родительского узла дерева, а второе остается в состоянии неопределенности в качестве дочерних узлов. Если во втором отображении есть подстановка, которая противоречит подстановкам первого отображения, то она удаляется; в противном случае подстановки во втором отображении группируются таким образом, чтобы подстановки в группе не противоречили друг другу, и все подстановки в каждой группе и в первом отображении добавляются в дерево в качестве дочернего узла.

## 3. ПСЕВДОКОД АЛГОРИТМА MCOG

Для дальнейшего псевдокода алгоритма **MCOG**, которому посвящена эта статья, использованы следующие обозначения:

- $\overline{x}$, $\overline{y}$ — списки переменных,
- $\overline{a}$, $\overline{b}$ — списки констант,
- $G_1, G_2$ — ориентированные графы, содержащие только списки констант $\overline{a}$ и $\overline{b}$ соответственно в качестве списков имён вершин, причём количество вершин в $G_1$ меньше количества вершин в $G_2$,
- $R$ — граф, полученный из $G_1$ путём замены имён вершин на переменные $\overline{x}$,
- $G$ — граф $G_2$,
  (в процессе работы алгоритма текущие дуги (и количество вершин) графов $R$ и $G$ будут меняться),
- $R\_ISOM$ — множество, предназначенное для хранения найденных (но не обязательно максимальных) графов $R$, изоморфных подграфам графов $G_1, G_2$;
  $R\_ISOM = \{(R,\ m_{R,\,G_1},\ m_{R,\,G_2}) : R$ — изоморфен подграфам $G_1$ и $G_2$; $m_{R,\,G_1}, m_{R,\,G_2}$ — отображения графа $R$ на подграф графа $G_1$ и $G_2$ соответственно; отображения $m_{R,\,G_1}$ и $m_{R,\,G_2}$ задают этот изоморфизм$\}$;
  отображение $m_{R,\,G_1}$ выписывать не будем, поскольку все подстановки в этом изоморфизме $m_{R,\,G_1}$ имеют вид $x_i : a_i$ для всех переменных $x_i$, вошедших в $R$,
- $R\_ISOM\_c$ — граф, являющийся элементом множества $R\_ISOM$, рассматриваемый в текущий (current) момент как лист в одной ветви непротиворечивого дерева, и соответствующее отображение $m_{R,\,G}$,
- $arcs_{u,v}(x_i, x_j)_R$, $arcs_{u,v}(b_i, b_j)_G$ — списки дуг из $R$ и $G$, в которых обе вершины $x_i$ и $x_j$ (соответственно $b_i$ и $b_j$) находятся в позициях $u$ и $v$ соответственно ($u$ и $v$ принимают значения 1 и 2),
- $used$ — метка для использованных в отображении $m$ ($m := \{\overline{x} : \overline{b}\}$) вершин графа $G$,
- $const$ — списки дуг в $R$, не содержащих вершины, имена которых являются переменными.

Псевдокод алгоритма **MCOG** представлен ниже (см. алгоритм 1).

Для псевдокода алгоритма **backtracking**, потребуются следующие обозначения:

- $arcs\_maxR$ — список всех дуг, выделенных в графе $R$, одновременно включающих максимальное количество вершин с именами, рассматриваемыми как константы, помеченных как «*used*»,
- $arcs\_maxG$ — упорядоченный список дуг, извлеченных из графа $G$. Вершины этих дуг совпадают с вершинами, имеющими константные имена из $arcs\_maxR$. Исключением являются позиции, в которых дуги $arcs\_maxR$ содержат вершины с пере-

**Algorithm 1:** алгоритм **MCOG**

**Input:** два ориентированных графа $G_1 = (V_1, A_1)$ и $G_2 = (V_2, A_2)$.
**Output:** $R\_ISOM$.

```
1  R_ISOM ← ∅, R_ISOM_c ← ∅, составляем arcs_{u,v}(x_{i_1}, x_{j_1})_R и arcs_{u,v}(b_{i_2}, b_{j_2})_G для каждой
     (u, v);
2  if arcs_{u,v}(x_{i_1}, x_{j_1})_R ≠ ∅ then
3      отбор пар вершин в R и G, имеющих наибольшее количество одновременных
         вхождений в одну дугу;
4      уборка дуг с выбранными вершинами из arcs_{u,v}(x_i, x_j)_R и arcs_{u,v}(b_i, b_j)_G;
5      помещение в очереди QR и QG дуг, включающих отобранные пары вершин;
6      for первая дуга A_R ∈ QR to последная do
7          for первая дуга A_G ∈ QG to последная do
8              нахождение частичного отображения m_{A_R,A_G}, m_p ← m_{A_R,A_G};
9              корневой узел дерева нахождения отображений ← m_p;
10             применение m_p к R, установка used;
11             извлечение const, R ← R \ const, G ← G \ const;
12             запись const, замененных переменными отображения m_p, в R_ISOM_c;
13             вызов backtracking (R, G, used, m_p, R_ISOM_c, R_ISOM);
14         end
15     end
16 end
```

менными именами. В этих позициях имена вершин найденной дуги не включают константы, помеченные как «*used*».

Ниже представлен псевдокод алгоритма **backtracking** (см. алгоритм 2).

## 4. ПСЕВДОКОД АЛГОРИТМА CONS-T ФОРМИРОВАНИЯ НЕПРОТИВОРЕЧИВОГО ДЕРЕВА ВОЗМОЖНЫХ ОТОБРАЖЕНИЙ

В псевдокоде описанного ниже алгоритма **CONS-T** использованы следующие обозначения:

- $R, G$ — ориентированные графы, содержащие $\overline{x}$ и $\overline{b}$ соответственно в качестве имён вершин,
- $UP1$, $UP2$ — два частичных отображения графа $R$ на граф $G$.
  $UP1 := m_p$.
  $UP1 - \{\overline{x} : \overline{a}\}$ (отображение, находящееся в рассматриваемом узле).
  $UP2 := m_{arcs\_maxR,\ arcs\_maxG}$.
  $UP2 - \{\overline{y} : \overline{b}\}$ (возможное отображение для переменных, ещё не имеющих значения),
- $isChildNode()$ — булева функция, имеющая в качестве аргумента подстановку и имеющая значение истина, если эта подстановка является подстановкой дочернего узла непротиворечивого дерева.

Алгоритм **CONS-T** описывается следующим псевдокодом (см. алгоритм 3).

**Algorithm 2:** Алгоритм **backtracking**

**Input:** $R, G, used, m_p, R\_ISOM\_c, R\_ISOM$.
**Output:** $R\_ISOM$.

$arcs\_maxR \leftarrow \emptyset$;
**if** *В дугах графа R нет вершин с переменным именем OR в дугах R есть, но в дугах G нет вершин, совпадающих с вершинами из* $arcs\_maxR$ *в тех позициях, в которых дуги R имеют вершины, обладающие константным именем,* **then**
  **if** $R\_ISOM\_c \notin R\_ISOM$ **then**
    $R\_ISOM \leftarrow R\_ISOM \cup \{R\_ISOM\_c\}$
  **end**
  **return**
**end**
нахождение $arcs\_maxR$;
**if** $arcs\_maxR \neq \emptyset$ **then**
  создание $arcs\_maxG$;
  **if** $arcs\_maxG \neq \emptyset$ **then**
    нахождение $m_{arcs\_maxR,\ arcs\_maxG}$;
    проверка непротиворечивости $m_p$ и $m_{arcs\_maxR,\ arcs\_maxG}$ с помощью алгоритма **CONS-T** $(R, G, m_p, m_{arcs\_maxR,\ arcs\_maxG})$;
    **if** $m_p$ *и* $m_{lits\_maxR,\ lits\_maxG}$ *непротиворечивы* **then**
      $m_{pb} \leftarrow (m_p \cup m_{arcs\_maxR,\ arcs\_maxG})$;
      **for** *первое частичное отображение* $m_{pbe} \in m_{pb}$ **to** *последнее* **do**
        установка $used_b$, которая будет действительна только внутри одной ветки обхода дерева;
        применение $m_{pbe}$ к $R$;
        получение $const$, $R_b \leftarrow R \setminus const$, $G_b \leftarrow G \setminus const$;
        запись $const$, замененные переменными $m_{pbe}$, в $R\_ISOM\_c$;
        вызов **backtracking** $(R_b, G_b, used_b, m_{pbe}, R\_ISOM\_c, R\_ISOM)$;
        отмена замен в дугах $R$;
      **end**
    **end**
  **end**
**end**

## 5. ПРИМЕНЕНИЕ АЛГОРИТМА MCOG К НАХОЖДЕНИЮ НАИБОЛЬШЕГО ОБЩЕГО ПОДГРАФА ДВУХ ОРИЕНТИРОВАННЫХ ГРАФОВ

Пусть имеются два ориентированных графа $G_1$ и $G_2$ (см. рис. 1).

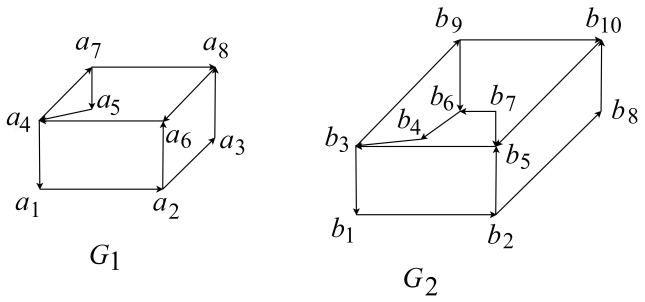


**Рис. 1.** Два ориентированных графа $G_1$ и $G_2$

**Algorithm 3:** алгоритм **CONS-T**

**Input:** $R$ и $G$, $UP1$ и $UP2$.
**Output:** непротиворечивое дерево возможных отображений $R$ на $G$.

родительский узел дерева ← $UP1$;
**for** *первая подстановка* $up2 \in UP2$ **to** *последная* **do**
  **for** *первая подстановка* $up1 \in UP1$ **to** *последная* **do**
    **if** $up2$ *и* $up1$ *противоречивы* **then**
      удаление $up2$ из $UP2$;
    **end**
  **end**
**end**
**for** *первая подстановка* $up2 \in UP2$ **to** *последная* **do**
  $isChildNode(up2) \leftarrow FALSE$;
**end**
**for** *первая подстановка* $up2 \in UP2$ **to** *последная* **do**
  **if** $isChildNode(up2) == FALSE$ **then**
    поиск всех подстановок $sub$ из $UP2 \setminus up2$, не противоречащих $up2$;
    добавления $UP1, up2, sub$ в качестве дочернего узла к родительскому узлу дерева;
    **for** *первая подстановка* $se \in sub$ **to** *последная* **do**
      $isChildNode(se) \leftarrow TRUE$;
    **end**
  **end**
**end**

Так как граф $G_1$ включает меньшее количество дуг, чем $G_2$, берем $G_1$ в качестве графа $R$.

В результате работы программы были получены два наибольших общих подграфа $R_1$ и $R_2$ графов $G_1$ и $G_2$ (см. рис. 2).

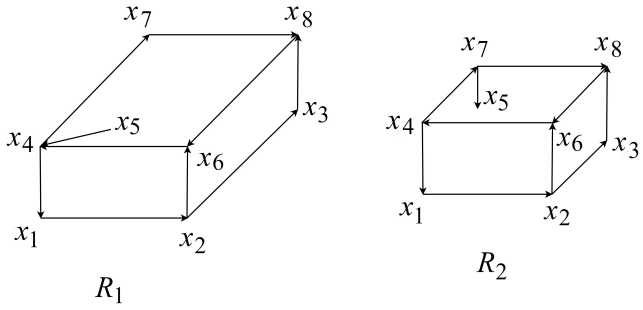


**Рис. 2.** Наибольшие общие подграфы графов $G_1$ и $G_2$

Подграф $R_1$ содержит 10 дуг:

$$(x_1, x_2),\ (x_2, x_3),\ (x_2, x_6),\ (x_3, x_8),\ (x_4, x_1),\ (x_4, x_7),\ (x_5, x_4),\ (x_6, x_4),\ (x_7, x_8),\ (x_8, x_6),$$

отображение $m_{R_1,\,G_2}$ графов $R_1$ и $G_2$ представляет собой

$$\{x_1 : b_1,\ x_2 : b_2,\ x_3 : b_8,\ x_4 : b_3,\ x_5 : b_4,\ x_6 : b_5,\ x_7 : b_9,\ x_8 : b_{10}\}.$$

Подграф $R_2$ тоже включает 10 дуг:

$$(x_1, x_2),\ (x_2, x_3),\ (x_2, x_6),\ (x_3, x_8),\ (x_4, x_1),\ (x_4, x_7),\ (x_6, x_4),\ (x_7, x_5),\ (x_7, x_8),\ (x_8, x_6),$$

отображение $m_{R_2,\,G_2}$ графов $R_2$ и $G_2$ является

$$\{x_1 : b_1,\ x_2 : b_2,\ x_3 : b_8,\ x_4 : b_3,\ x_5 : b_6,\ x_6 : b_5,\ x_7 : b_9,\ x_8 : b_{10}\}.$$

## 6. ЗАКЛЮЧЕНИЕ

В статье разработан алгоритм нахождения наибольшего общего подграфа двух ориентированных графов. Реализация была осуществлена на языке программирования Python [10].

Изоморфизм графа является одной из ключевых задач в теории графов. Опираясь на графовые нейронные сети [11] и тест Вайсфейлера-Лемана, авторы [12] разработали сеть изоморфизма графов. Данная архитектура применяется в различных областях, особенно в биоинформатике [13] и медицине [14]. Кроме того, в [15] предложен алгоритм в качестве решения задачи изоморфизма гиперграфов.

## Список литературы


1. *Chen X., Jia S., Xiang Y.* A review: Knowledge reasoning over knowledge graph // Expert Systems with Applications. 2020. Т. 141. С. 1–21. doi:10.1016/j.eswa.2019.112948
2. *Bouritsas G. et al.* Improving graph neural network expressivity via subgraph isomorphism counting // IEEE Transactions on Pattern Analysis and Machine Intelligence. 2023. Т. 45, № 1. С. 657–668. doi: 10.1109/TPAMI.2022.3154319
3. *Zeng X. et al.* GNNGL-PPI: multi-category prediction of protein-protein interactions using graph neural networks based on global graphs and local subgraphs // BMC Genomics. 2024. Т. 25, № 1. С. 1–13. doi: 10.1186/s12864-024-10299-x
4. *Wu S. et al.* Graph neural networks in recommender systems: a survey // ACM Computing Surveys. 2022. Т. 55, № 5. С. 1–37. doi: 10.1145/3535101
5. *Cheng D. et al.* Graph neural network for fraud detection via spatial-temporal attention // IEEE Transactions on Knowledge and Data Engineering. 2020. Т. 34, № 8. С. 3800–3813. doi:10.1109/TKDE.2020.3025588
6. *Гэри М., Джонсон Д.* Вычислительные машины и труднорешаемые задачи / пер. с англ. Левнера Е. В., Фрумкина М. А. М.: Мир, 1982.
7. *Косовская Т. М., Косовский Н. Н.* Выделение общих свойств объектов для создания логических онтологий // Вестник Санкт-Петербургского университета. Прикладная математика. Информатика. Процессы управления. 2022. Т. 18, № 1. С. 37–51. doi: 10.21638/11701/spbu10.2022.103
8. *Косовская Т. М., Косовский Н. Н.* Полиномиальная эквивалентность задач изоморфизм предикатных формул и изоморфизм графов // Вестник Санкт-Петербургского университета. Математика. Механика. Астрономия. 2019. Т. 6 (64), № 3. С. 430–439. doi: 10.21638/11701/spbu01.2019.308
9. *Чжоу Ц., Косовская Т. М.* Алгоритм выделения общих свойств объектов, описанных на языке исчисления предикатов с одним предикатным символом // Вестник Санкт-Петербургского университета. Математика. Механика. Астрономия. 2024. Т. 11 (69). № 4. С. 733–744. doi: 10.21638/spbu01.2024.409
10. *Matthes E.* Python crash course: A hands-on, project-based introduction to programming, 3rd Edition. San Francisco: no starch press, 2023.
11. *Wu Z. et al.* A comprehensive survey on graph neural networks // IEEE Transactions on Neural Networks and Learning Systems. 2021. Т. 32, № 1. С. 4–24. doi:10.1109/tnnls.2020.2978386
12. *Xu K. et al.* How powerful are graph neural networks? // arxiv:1810.00826, 2018.
13. *Zheng K. et al.* NASMDR: a framework for miRNA-drug resistance prediction using efficient neural architecture search and graph isomorphism networks // Briefings in Bioinformatics. 2022. Т. 23, № 5. С. 1–9. doi: 10.1093/bib/bbac338
14. *Su J. et al.* $M^2$DC: A meta-learning framework for generalizable diagnostic classification of major depressive disorder // IEEE Transactions on Medical Imaging. 2024. doi: 10.1109/TMI.2024.3461312

15. *Feng Y. et al.* Hypergraph isomorphism computation // IEEE Transactions on Pattern Analysis and Machine Intelligence. 2024. Т. 46, № 5. С. 3880–3896. doi: 10.1109/TPAMI.2024.3353199

*Поступила в редакцию 26.09.2024, окончательный вариант — 15.10.2024.*

**Чжоу Цзюань, аспирант, кафедра информатики, математико-механический факультет, СПбГУ, ✉ st103098@student.spbu.ru**

Computer tools in education, 2024
№ 3: 5–13
http://cte.eltech.ru
doi:10.32603/2071-2340-2024-3-5-13

# An Algorithm for Identifying the Maximal Common Subgraph in Oriented Graphs

Zhou J.[1], Postgraduate, ✉ st103098@student.spbu.ru, orcid.org/0009-0007-2350-987X

[1]Saint Petersburg State University, 7-9, Universitetskaya nab., Saint Petersburg, 199034, Russia

**Abstract**

Graph isomorphism checking is highly significant within both graph theory and its applications, particularly regarding oriented graph isomorphism, where identifying a bijection is an NP-hard problem.

The problem of finding the maximal common subgraph between two oriented graphs represents a specific instance of predicate formula isomorphism, occurring when an elementary conjunction includes only a single binary predicate. Initially, the algorithm was developed to determine the maximal common subformula of two elementary conjunctions. This paper provides pseudocode for algorithms that extract the maximal common subgraph in oriented graphs, accompanied by an illustrative application example.

**Keywords:** *graph isomorphism, maximal common oriented subgraph, graph bijection.*

**Citation:** J. Zhou, "An Algorithm for Identifying the Maximal Common Subgraph in Oriented Graphs", *Computer tools in education*, no. 3, pp. 5–13, 2024 (in Russian); doi:10.32603/2071-2340-2024-3-5-13

**Acknowledgements:** *The author are grateful for the financial support provided by the Program of China Scholarship Council (Grant No. 202108620001).*

## References

1. X. Chen, S. Jia, and Y. Xiang, "A review: Knowledge reasoning over knowledge graph," *Expert Systems with Applications*, vol. 141, pp. 1–21, 2020; doi:10.1016/j.eswa.2019.112948
2. G. Bouritsas et al., "Improving graph neural network expressivity via subgraph isomorphism counting," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 1, pp. 657–668, 2023; doi:10.1109/TPAMI.2022.3154319
3. X. Zeng et al., "GNNGL-PPI: multi-category prediction of protein-protein interactions using graph neural networks based on global graphs and local subgraphs," *BMC Genomics*, vol. 25, no. 1, pp. 1–13, 2024; doi:10.1186/s12864-024-10299-x

4. S. Wu et al., “Graph neural networks in recommender systems: a survey,” *ACM Computing Surveys*, vol. 55, no. 5, pp. 1–37, 2022; doi:10.1145/3535101
5. D. Cheng et al., “Graph neural network for fraud detection via spatial-temporal attention,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 34, no. 8, pp. 3800–3813, 2020; doi:10.1109/TKDE.2020.3025588
6. M. R. Garey and D. S. Johnson, *Computers and intractability: A Guide to the Theory of NP-Completeness*, New York: W. H. Freeman and Comp., 1979.
7. T. M. Kosovskaya and N. N. Kosovskii, “Extraction of common properties of objects for creation of a logic ontology,” *Vestnik of Saint Petersburg University. Applied Mathematics. Computer Science. Control Processes*, vol. 18, no. 1, pp. 37–51, 2022 (in Russian); doi: 10.21638/11701/spbu10.2022.103
8. T. M. Kosovskaya and N. N. Kosovskii, “Polynomial equivalence of the problems predicate formulas isomorphism and graph isomorphism,” *Vestnik of Saint Petersburg University. Mathematics. Mechanics. Astronomy*, vol. 6(64), no. 3, pp. 430–439, 2019 (in Russian); doi:10.21638/11701/spbu01.2019.308
9. J. Zhou and T. M. Kosovskaya, “Algorithm for extraction common properties of objects described in the predicate calculus language with a single predicate symbol,” *Vestnik of Saint Petersburg University. Mathematics. Mechanics. Astronomy*, vol. 11(69), no. 4, pp. 733–744, 2024 (in Russian); doi:10.21638/spbu01.2024.409
10. E. Matthes, *Python crash course: A hands-on, project-based introduction to programming*, 3rd ed., San Francisco, USA: No starch press, 2023.
11. Z. Wu et al., “A comprehensive survey on graph neural networks,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 32, no. 1, pp. 4–24, 2021; doi:10.1109/tnnls.2020.2978386
12. K. Xu et al., “How powerful are graph neural networks?,” in *Arxiv*, 2018. [Online]. Available: https://arxiv.org/abs/1810.00826
13. K. Zheng et al., “NASMDR: a framework for miRNA-drug resistance prediction using efficient neural architecture search and graph isomorphism networks,” *Briefings in Bioinformatics*, vol. 23, no. 5, pp. 1–9, 2022; doi:10.1093/bib/bbac338
14. J. Su et al., “$M^2$DC: A meta-learning framework for generalizable diagnostic classification of major depressive disorder,” *in IEEE Transactions on Medical Imaging*, 2024; doi:10.1109/TMI.2024.3461312
15. Y. Feng et al., “Hypergraph Isomorphism Computation,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 46, no. 5, pp. 3880–3896, 2024; doi:10.1109/TPAMI.2024.3353199

*Received 26-09-2024, the final version — 15-10-2024.*

**Zhou Juan, Postgraduate, the Faculty of Mathematics and Mechanics, SPbU,** ✉ **st103098@student.spbu.ru**