РЕШЕНИЕ НЕКОТОРЫХ ЗАДАЧ НА ГРАФАХ С ПОМОЩЬЮ МЕТОДОВ РЕШЕНИЯ СИСТЕМ ЛИНЕЙНЫХ УРАВНЕНИЙ

Александр Вячеславович Пролубников

a.v.prolubnikov@mail.ru

29.02.24

ЗАДАЧИ НА ГРАФАХ И МЕТОДЫ ЛИНЕЙНОЙ АЛГЕБРЫ

ЗАДАЧА НА ГРАФАХ $\rightarrow$ **ЗАДАЧА ЛИНЕЙНОЙ АЛГЕБРЫ**
(возможно, вспомогательная задача, не эквивалентная исходной)

$\rightarrow$ **ВЫЧИСЛИТЕЛЬНАЯ ЗАДАЧА ЛИНЕЙНОЙ АЛГЕБРЫ**

$\rightarrow$ **ЕЁ РЕШЕНИЕ С ПОМОЩЬЮ РЕШЕНИЙ СЛАУ**

МЕТОДЫ РЕШЕНИЯ СЛАУ

ЧИСЛЕННЫЕ МЕТОДЫ ЛИНЕЙНОЙ АЛГЕБРЫ:

- позволяют/не позволяют оценивать вычислительную сложность алгоритма решения **задачи на графах** по времени и по памяти;
- могут гарантировать/не гарантировать точность вычислений, необходимую для решения **задачи на графах**;
- допускают/не допускают эффективные параллельные реализации;
- различные методы могут давать различные алгоритмы решения **задачи на графах** в терминах теории графов.

(например, метод Якоби решения СЛАУ задаёт один обход графа, метод Гаусса-Зейделя — другой)

ЗАДАЧА ПРОВЕРКИ ИЗОМОРФИЗМА ГРАФОВ

ЗАДАЧА ПРОВЕРКИ ИЗОМОРФИЗМА ГРАФОВ

Постановка задачи на графах:

Дано: Два обыкновенных графа G и H.

- $V(G)$, $V(H)$ — множества вершин;
- $E(G)$, $E(H)$ — множества рёбер;

$$V(G) = V(H) = [n].$$

Проверить: Существует ли такая биекция $\varphi : V(G) \to V(H)$, что $\forall i, j \in V(G)$

$$(i, j) \in E(G) \Leftrightarrow (\varphi(i), \varphi(j)) \in E(H).$$

φ — изоморфизм G и H

- если φ существует, то $G \simeq H$;
- иначе — $G \not\simeq H$.

ЗАДАЧА ПРОВЕРКИ ИЗОМОРФИЗМА ГРАФОВ

Матрица смежности графа G:

$$(A(G))_{ij} = \begin{cases} 1, & \text{if } (i,j) \in E(G), \\ 0, & \text{else.} \end{cases}$$

Матричная постановка:

Дано: $A(G)$ и $A(H)$ — матрицы смежности графов G и H.

$$G \simeq H \Leftrightarrow \exists \varphi \in S_n : \; A(H) = P_\varphi A(G) P_\varphi^\top,$$

$$(P_\varphi)_{ij} = \begin{cases} 1, & \text{if } j = \varphi(i), \\ 0, & \text{else.} \end{cases}$$

- если φ существует, то $G \simeq H$;
- иначе — $G \not\simeq H$.

АЛГОРИТМЫ ПРОВЕРКИ ИЗОМОРФИЗМА ГРАФОВ

— рекурсивные (переборные) алгоритмы проверки равенства некоторых *инвариантных* для изоморфных графов характеристик.

до 2015:

Неизвестны полиномиальные алгоритмы проверки изоморфизма графов.

Например, *каноническая разметка графа* — **полный** инвариант графа — требует экспоненциального времени для вычисления.

2015 — László Babai. *Graph isomorphism in quasi-polynomial time*.

АЛГОРИТМ СО СЛОЖНОСТЬЮ $2^{O((\log n)^c)}$

Задачи с квази-полиномиальной сложностью — кандидаты попасть в **промежуточный уровень задач**: ни полиномиально не разрешимы, но и, скорее всего, NP-трудны.

ПРОВЕРКА ИЗОМОРФИЗМА ЗА КВАЗИ-ПОЛИНОМИАЛЬНОЕ ВРЕМЯ

https://en.wikipedia.org/wiki/Graph_isomorphism:

In November 2015, László Babai, a mathematician and computer scientist at the University of Chicago, claimed to have proven that the graph isomorphism problem is solvable in quasi-polynomial time. He published preliminary versions of these results in the proceedings of the 2016 Symposium on Theory of Computing, and of the 2018 International Congress of Mathematicians. In January 2017, Babai briefly retracted the quasi-polynomiality claim and stated a sub-exponential time complexity bound instead. He restored the original claim five days later.

As of 2020, the full journal version of Babai's paper has not yet been published.

ПРОВЕРКА ИЗОМОРФИЗМА ЗА КВАЗИ-ПОЛИНОМИАЛЬНОЕ ВРЕМЯ

https://cstheory.stackexchange.com/questions/40353/whats-the-status-of-babais-graph-isomorphism-result:

The correction (and hence the quasi-polynomial result) was immediately supported by Harald Andrés Helfgott. His expository paper (arxiv.org/abs/1701.04372) and its translation (arxiv.org/abs/1710.04574) are all the support that is needed for the quasi-polynomial results. Helfgott's exposition appeared as Exposé 1125 in Astérisque 407 (2019), Séminaire Bourbaki 2016/2017, pp. 135–182.

There are no current known issues with Babai's proof, and it has gone through extensive peer review. That said, even published peer-reviewed papers have later been withdrawn as incorrect.

The support by Harald Andrés Helfgott along with the lack of other issues should be sufficient for us to accept that graph isomorphism is indeed solvable in quasi-polynomial time.

АЛГОРИТМ СПЕКТРАЛЬНОГО РАСЩЕПЛЕНИЯ

R. Faizullin, A. Prolubnikov. An Algorithm of the Spectral Splitting for (breaking) the Double Permutation Cipher (code)// Pattern Recognition and Image Analysis. MAIK, Nauka. 2002. Vol. 12, PP. 365-375. 2002.

Спектр графа $\mathrm{spec}(G) = \{\lambda_1, \ldots, \lambda_n\}$ — НЕ полный вариант.

Возмущая матрицу можно задать изменения спектра, которые будут одни и те же (с точностью доо перестановки) для изоморфных графов.

Расчёт спектра: нельзя гарантировать достижение необходимой точности за заданное количество итераций, то есть при заданной вычислительной сложности вычислений $\rightarrow$ *расчёт элементов обратной матрицы:*

$$(A^{-1})_{ii} = \frac{\prod_{i=1}^{n-1} \tilde{\lambda}_i}{\prod_{i=1}^{n} \lambda_i}$$

$\lambda_i \in \mathrm{spec}(G)$ и $\tilde{\lambda}_i \in \mathrm{spec}(G \setminus i)$

ХАРАКТЕРИСТИЧЕСКИЙ ПОЛИНОМ ГРАФА И ЕГО МОДИФИКАЦИИ

- характеристический полином матрицы смежности графа

$$\chi_G(x) = \det(A(G) - xE),$$

x — переменная, E единичная матрица.

- Лапласиан графа

$$L(G) = D(G) - A(G),$$

где $D(G) = \mathrm{diag}(d_1, \ldots, d_n)$ — диагональная матрица, d_i — степень вершины $i \in V(G)$.

- беззнаковый Лапласиан графа

$$|L(G)| = D(G) + A(G),$$

- обобщение $\xi_G(x, y)$:

$$\xi_G(x, y) = \det(xE - (A(G) - yD(G))),$$

ХАРАКТЕРИСТИЧЕСКИЙ ПОЛИНОМ ГРАФА И ЕГО МОДИФИКАЦИИ

- полином Зейделя $\zeta_G(x)$:

$$\zeta_G(x) = \det(xE - (F - E - 2A(G))),$$

where $(F)_{ij} = 1$ for $i, j = \overline{1, n}$;

- полином

$$\psi_G(x, y, \lambda) = \det(A(x, y) - \lambda E),$$

где $A(x, y)$ — матрица, получаемая из A, заменой единиц на x и нулей (недиагональных) на y.

ни один из этих полиномов не является полным инвариантом графа

МОДИФИЦИРОВАННЫЙ ХАРАКТЕРИСТИЧЕСКИЙ ПОЛИНОМ ГРАФА

Переменные в полиномах выше не связана с вершинами графа.

$\to$ *свяжем переменные* x_i *с вершинами* $i \in V(G)$.

$x_1, \ldots, x_n$ — независимые переменные, $X = \mathrm{diag}(x_1, \ldots, x_n)$.

Модифициованный характеристический полином $\eta_G(x_1, \ldots, x_n)$

для графа G, $|V(G)| = n$, $\eta_G(x_1, \ldots, x_n)$ — это полином вида

$$\eta_G(x_1, \ldots, x_n) = \det(A(G) + X). \tag{1}$$

Полиномы вида (1) для графов на $n = 1, 2, 3$ вершинах.

1) $n = 1$: x_1;
2) $n = 2$: x_1x_2, $x_1x_2 - 1$;
3) $n = 3$: $x_1x_2x_3$, $x_1x_2x_3 - x_1$, $x_1x_2x_3 - x_1 - x_3$, $x_1x_2x_3 - x_1 - x_2 - x_3 + 2$.

ПРОВЕРКА ИЗОМОРФИЗМА ГРАФОВ КАК ПРОВЕРКА ИЗОМОРФИЗМА ПОЛИНОМОВ

- $c \subseteq V(G)$,
- $x_c = \prod_{i \in c} x_i$,
- $A(G)_c$ определитель c-подматрицы

$A(G)$ — коэффициент перед $x_c = \prod_{i \in c} x_i$ в полиноме $\eta_G(x_1, \ldots, x_n)$.

Теорема.

$G \simeq H$ и $\varphi : V(G) \to V(H)$ — их изоморфизм $\Leftrightarrow$

$$\forall x \in \mathbb{R}^n \quad \eta_G(x_1, \ldots, x_n) = \eta_H(x_{\varphi(1)}, \ldots, x_{\varphi(n)}).$$

полиномы η_G и η_H *изоморфны* $\Leftrightarrow$ коэффициент перед x_c равен коэффициенту перед $x_{\varphi(c)}$ для всех $c \subset V(G)$

$\Leftrightarrow$ для всех $c \subset V(G)$ имеем: $A(G)_c = A(H)_{\varphi(c)}$.

ПРОВЕРКА ИЗОМОРФИЗМА ПОЛИНОМОВ ГРАФОВ

$$\forall x \in \mathbb{R}^n \quad \eta_G(x_1, \ldots, x_n) = \eta_H(x_{\varphi(1)}, \ldots, x_{\varphi(n)})$$

$\rightarrow$ пытаемся установить φ по результатам проверки значений полиномов в заданных точках.

2^n коэффициентов, которые должны быть попарно равны φ:

$$A(G)_c = A(H)_{\varphi(c)}, \ c \subset V$$

Если существует не экспоненциальный алгоритм решения задачи ИГ

$\Rightarrow$ возможна проверка равенства полиномов от n переменных с 2^n коэффициентами (при неизвестном φ)

за *не экспоненциальное время*

ПОЛИНОМИАЛЬНАЯ РАЗРЕШИМОСТЬ ИГ ДЛЯ ДЕРЕВЬЕВ

Утв.

Если T — дерево, то полином $\eta_T(x_1, \ldots, x_n)$ имеет не более n^2 не равных нулю коэффициентов.

Можно показать, что для дерева

$$\eta_T(x_1, \ldots, x_n) = \sum_{M \in \mathcal{M}} x_{i_1} \cdot \ldots \cdot x_{i_1},$$

для паросочетания M: $\{i_1, \ldots, i_k\} = V \setminus M$.

- где $\mathcal{M}$ — ножество паросочетаний в дереве;
- знак всех мономов «+».

Так как для дерева $|\mathcal{M}| < n^2$, то и среди коэффициентов не более n^2 не равных нулю. Значит задача простая, и возможно полиномиальное решение.

ПРОВЕРКА РАВЕНСТВА ПОЛИНОМОВ ГРАФОВ С ПОСЛЕДОВАТЕЛЬНЫМ ПОСТРОЕНИЕМ φ

определяем φ последовательно:

1-я итерация: определяем равенство $\eta_G(x)$ и $\eta_H(x_\varphi)$ для

$$\varphi = \begin{pmatrix} 1 & 2 & \dots & n \\ \varphi(1) & \cdot & \dots & \cdot \end{pmatrix}$$

2-я итерация: определяем равенство $\eta_G(x)$ и $\eta_H(x_\varphi)$ для

$$\varphi = \begin{pmatrix} 1 & 2 & \dots & n \\ \varphi(1) & \varphi(2) & \dots & \cdot \end{pmatrix}$$

и т.д.

$\rightarrow$ проверка равенства $\eta_G(x)$ и $\eta_H(x_\varphi)$ при текущем φ в точках

- $(\varepsilon_1, d, \dots, d)$,
- $(\varepsilon_1, \varepsilon_2, \dots, d)$,
- $\dots$,
- $(\varepsilon_1, \varepsilon_3, \dots, \varepsilon_n)$.

ПРЯМОЙ АЛГОРИТМ ПРОВЕРКИ ИГ

АЛГОРИТМ 1 (G, H);

1 $J \leftarrow V(H)$;
2 **for** $i \leftarrow 1$ **to** n
3 случайно выбрать $\varepsilon_i \in S$;
4 **if** $(\exists j \in J \ : \ \eta_G(\varepsilon_\varphi^{(i)}) = \eta_H(\varepsilon^{(i-1)} + \varepsilon_i e_j))$
5 $\varepsilon^{(i)} := \varepsilon^{(i-1)} + \varepsilon_i e_i$;
6 $\varphi(i) \leftarrow j$;
7 $J \leftarrow J \setminus \{j\}$;
8 **else print** "$G \not\simeq H$".
9 **print** "$G \simeq H$, φ — изоморфизм G и H".

— рандомизированный тест на изоморфизм.

Не позволяет решать задачу ИГ, в частности, для *сильно-регулярных графов*.

РЕКУРСИВНЫЙ ВАРИАНТ АЛГОРИТМА 1

АЛГОРИТМ 2 (G, H)

1 $J \leftarrow V(H)$;
2 $flag \leftarrow false$;
3 $\forall i \in V(G) : \varphi(i)$ не определено.
4 Установить соответствие (1);
5 **if** *flag*
6 **print** "$G \simeq H$, φ — изоморфизм G и H";
7 **else**
8 **print** "$G \not\simeq H$".

— рекурсивный запуск АЛГОРИТМА 1,
с проверкой (потенциально) всех возможностей
установления изоморфизма графов G и H.

РЕКУРСИВНЫЙ ВАРИАНТ АЛГОРИТМА 1

Установить соответствие $(i \in V(G))$

1 **if** $i = n$
2 $\quad$ $flag \leftarrow true$; $\varphi(n) \leftarrow k$, где k такое, что $J = \{k\}$;
3 $\quad$ **exit**;
4 **else**
5 $\quad$ **for** $j \leftarrow 1$ **to** n
6 $\quad\quad$ выбрать случайно $\varepsilon_i \in S$;
7 $\quad\quad$ **if** $(j \in J$ and $\eta_G(\varepsilon_\varphi^{(i)}) = \eta_H(\varepsilon^{(i-1)} + \varepsilon_i e_j))$
8 $\quad\quad\quad$ $\varepsilon^{(i)} := \varepsilon^{(i-1)} + \varepsilon_i e_i$;
9 $\quad\quad\quad$ $\varphi(i) \leftarrow j$;
10 $\quad\quad\quad$ $J \leftarrow J \setminus \{j\}$;
11 $\quad\quad\quad$ Установить соответствие $(i+1)$;
12 $\quad\quad\quad$ **if** $flag = false$
13 $\quad\quad\quad\quad$ $J \leftarrow J \cup \{j\}$; $\varphi(i)$ не определено;
14 **exit**.

ПРОВЕРКА ИГ СРАВНЕНИЕМ ЗНАЧЕНИЙ ПОЛИНОМОВ η_G и η_H

Если $\varepsilon_i \neq \varepsilon_j \Rightarrow$ — веса петель инцидентных вершинам графов *разные*, то G и H графы с тривиальной группой автоморфизмов (у всех вершин петли различного веса) $\Rightarrow$ проверка

$$\exists \varphi \in S_n \text{ т.ч. } A(H) = P_\varphi A(G) P_\varphi^\top$$

может быть произведена АЛГОРИТМОМ 1.

$\Rightarrow$ производя *согласованные возмущения* диагональных элементов матриц графов, *дерегуляризуем графы*. В результате: упрощается решение здачи проверки изоморфизма графов.

$\Rightarrow$ *последовательная дерегуляризация графов в ходе работы алгоритма*

КАК ВЫЧИСЛЯТЬ ЗНАЧЕНИЯ $\eta_G(x)$ и $\eta_H(x)$?

ПРОБЛЕМЫ:

- экспоненциальное количество коэффициентов у полиномов — алгоритм со сложностью $O(2^n)$;
- вычисление определителя — $O(n^3)$;
- точное вычисление определителей — — экспоненциальное количество элементарных машинных операций.

ПОДХОД: не вычислять сами значения $\eta_G(x)$ и $\eta_H(x)$, а сравнивать их (равны/не равны).

— используя **численные методы решения СЛАУ** с матрицами $A(G)$ и $A(H)$ графов можно сравнивать значения $\eta_G(x_\varphi)$ и $\eta_H(x)$.

— за полиномиальное время при использовании *машинных чисел* может быть достигнута необходимая точность **для решения задачи на графах**.

СРАВНЕНИЕ ЗНАЧЕНИЙ $\eta_G(x)$ и $\eta_H(x)$

Матрица графа — модифицированная матрица смежности с диагональным преобладанием: $\det A > 0$

Обращение A:

$$Ax = e_i, \ \ i = \overline{1, n}$$

— решение i-й СЛАУ — i-й столбец (строка) A^{-1}

Утверждение 1.

$\eta_G(\varepsilon_\varphi^{(i)}) = \eta_H(\varepsilon_\varphi^{(i-1)} + \varepsilon_i e_j)$ тогда и только тогда, когда

$$\left(A_G\left(\varepsilon_\varphi^{(i)}\right)\right)^{-1}_{\varphi(i)\varphi(i)} = \frac{\eta_{G\setminus\{i\}}\left(\varepsilon_\varphi^{(i)}\right)}{\eta_G\left(\varepsilon_\varphi^{(i)}\right)} = \frac{\eta_{H\setminus\{j\}}\left(\varepsilon_\varphi^{(i-1)} + \varepsilon_i e_j\right)}{\eta_H\left(\varepsilon_\varphi^{(i-1)} + \varepsilon_i e_j\right)} = \left(A_H\left(\varepsilon^{(i)}\right)\right)^{-1}_{ii}.$$

— то есть достаточно сравнивать диагональные элементы обратных матриц.

СРАВНЕНИЕ ЗНАЧЕНИЙ $\eta_G(x)$ и $\eta_H(x)$

Итерационные методы решения СЛАУ:

- гарантируют за полиномиальное от n время достижение точности, достаточной для численной реализации решения **задачи на графах**.

Утверждение 2.

Если для некоторого $\varphi \in S_n$ имеем $\eta_G(\varepsilon^{(i)}) = \eta_H(\varepsilon_\varphi^{(i)})$ для $i < k$ и $\eta_G(\varepsilon^{(k)}) \neq \eta_H(\varepsilon_\varphi^{(k)})$, то

$$\frac{\eta_{G\setminus\{k\}}(\varepsilon^{(k)})}{\eta_G(\varepsilon^{(k)})} \neq \frac{\eta_{H\setminus\{j\}}(\varepsilon_\varphi^{(k)})}{\eta_H(\varepsilon_\varphi^{(k)})}, \tag{2}$$

и это неравенство может быть проверено за $O(n^4)$ элементарных машинных операций с машинными числами с длиной мантиссы $O(n^2)$.

(методы Якоби и Гаусса-Зейделя решения СЛАУ)

ПРОВЕРКА ИЗОМОРФИЗМА ГРАФОВ

- А.В. Пролубников. Точность и сложность вычислений, необходимые для проверки изоморфизма графов сравнением полиномов // Вычислительные технологии. 2016. Т. 21, № 6. С. 71–88.

- А.В. Пролубников. Сведение задачи проверки изоморфизма графов к задаче проверки равенства полиномов от n переменных // Труды института математики и механики УрО РАН. 2016. Т. 22, №1. С. 235-240.

- A.V. Prolubnikov. Reduction of the graph isomorphism problem to equality checking of n-variables polynomials and the algorithms that use the reduction // arXiv.org, 2016

ГИПОТЕЗА О ВООСТАНОВИМОСТИ ГРАФА

ГИПОТЕЗА (Келли, Улам)

Граф G более чем на 3 вершинах восстановим по *колоде* своих подграфов, полученных удалением одной вершины.

Колода G: $\mathcal{D}G = \{G \setminus i\}_{i=1}^{n}$ — непомеченные графы.

Обязательно ли изоморфны графы с одинаковыми колодами?

Колода *полиномов* G: $\mathcal{D}\eta_G(x) = \{\eta_{G \setminus i}(x)\}_{i=1}^{n}$.

$$\eta_{G \setminus i}(x) = \frac{\partial \eta_G}{\partial x_i}(x)$$

$\mathcal{D}\eta_G(x)$ — набор частных производных функций (функции от $n-1$ переменной).

Однозначно ли восстановим полином графа по своим частным производным ПЕРВОГО порядка?

ОБХОДЫ ГРАФОВ

ЗАДАЧИ О СВЯЗНОСТИ НА ГРАФАХ

ЗАДАЧИ О НАДЁЖНОСТИ ИНФРАСТРУКТУРЫ:

- транспортные сети,
- сети передачи данных,
- большие интегральные схемы
  (до миллиона и более компонентов: диодов, транзисторов, резисторов, конденсаторов, соединенных между собой),
- сенсорные сети

и др. $(n \geq 10^5)$

$\Rightarrow$ ЗАДАЧИ О СВЯЗНОСТИ НА ГРАФАХ:

- проверка связности графа,
- нахождение компонент связности графа,
- нахождение точек сочленения,
- нахождение мостов

и др.

ОБХОДЫ ГРАФОВ

G — обыкновенный (неориентированный, без кратных рёбер и петель) помеченный граф

- V, $|V| = n$ — множество вершин,
- E, $|E| = m$ — множество рёбер.

Вершины графа занумерованы от 1 до n.

Обход графа — это итерационный алгоритм: переходы по рёбрам графа, начиная с некоторой стартовой вершины $s \in V$,

Обычные подходы к реализации обхода графа:

- поиск в глубину *(Depth-First search, DFS)*
- *поиск в ширину (Breadth-First search, BFS)*
  (алгоритм Беллмана-Форда).

В результате: посещаются все вершины из компоненты связности G.

ОБХОДЫ ГРАФОВ

Поиск в глубину (DFS)

идём «вглубь» графа, насколько это возможно:

- перебираем все исходящие из рассматриваемой вершины рёбра;
- если ребро ведёт в вершину, которая не была рассмотрена ранее, то запускаем алгоритм от этой нерассмотренной вершины, после возвращаемся и продолжаем перебирать рёбра;
- возврат: если в рассматриваемой вершине не осталось рёбер, которые ведут в нерассмотренную вершину.

Поиск в ширину (BFS), построение *дерева достижимости*:

- на каждой итерации производим переходы в вершины смежные достигнутым к текущей итерации

  — обычный выбор при построении эффективных алгоритмов нахождения компонент связности

АЛГЕБРАИЧЕСКИЙ BFS

- A — матрица смежности G;
- $s \in V$ — *стартовая вершина*, e_s — s-ый вектор ст. базиса $\mathbb{R}^n$;
- $x^{(0)} = e_s$.

Итерации BFS: $x^{(k)} = Ax^{(k-1)}$

если $x_i^{(k)} \neq 0 \Rightarrow i \in V$ достигнута в процессе обхода.

Алгебраический BFS: максимально эффективное использование возможностей компьютерных архитектур

— *параллельные вычисления, работа с памятью* (GraphBLAS).

Вычислительная сложность: $O(m+n)$.

Разреженные графы $(m = O(n))$: $O(n)$

— минимальная возможная выч. сложность BFS.

ИТЕРАЦИОННЫЕ МЕТОДЫ РЕШЕНИЯ СЛАУ

СЛАУ:

$$Ax = e_s,$$

где $s \in V$ — *стартовая вершина*, e_s — s-ый вектор ст. базиса $\mathbb{R}^n$;

метод Якоби:

$$x_i^{(k+1)} = \frac{1}{a_{ii}} \left(b_i - \sum_{j \neq i} a_{ij} x_j^{(k)} \right)$$

метод Гаусса-Зейделя:

$$x_i^{(k+1)} = \frac{1}{a_{ii}} \left(b_i - \sum_{l=1}^{i-1} a_{ij} x_j^{(k+1)} - \sum_{j=i+1}^{n} a_{ij} x_j^{(k)} \right)$$

$b_i = 1$, если $i = j$, and $b_i = 0$ иначе.

ИТЕРАЦИОННЫЕ МЕТОДЫ РЕШЕНИЯ СЛАУ

Если производить относительно небольшое количество итераций,

не достигая сходимости к точному решению СЛАУ,

то получаем последовательность приближённых решений СЛАУ

$$\{x^{(0)}, x^{(1)}, \ldots, x^{(k)}, \ldots\}$$

РЕАЛИЗУЕМЫЕ ПРИ ЭТОМ ОБХОДЫ ГРАФА:

На $(k+1)$-ой итерации производится переход в вершину $i \in V$, если

$$(x_i^{(k)} = 0) \text{ и } (x_i^{(k+1)} \neq 0)$$

— вершина i не достигнута к k-ой итерации,
но достигается на $(k+1)$-ой итерации.

Нахождение компоненты связности $(G, s \in V(G)) : C$;

1 $x^{(0)} = (0, \ldots, 0)$, $x^{(1)} = (0, \ldots, 0)$;
2 $k \leftarrow 0$;
3 **while** $\exists i \in V(G) : \left((x_i^{(k)} = 0) \wedge (x_i^{(k+1)} \neq 0)\right)$
4 $\quad x^{(k+1)} = \mathbf{F}(x^{(k)})$;
5 $\quad C \leftarrow C \cup \{i : x_i^{(k)} \neq 0\}$;
6 $\quad k \leftarrow k+1$;
7 **result** $\leftarrow C$.

Варианты алгоритма обхода:

- **F** — итерация метода Якоби — *Jacobi search, JS*;
- **F** — итерация метода Гаусса-Зейделя — *Gauss-Seidel search, GSS*.

РАЗНИЦА МЕЖДУ ОБХОДАМИ JS (BFS) и GSS

Цепь — конечная последовательность смежных вершин в графе.

Цепь с правильным порядком вершин (правильная цепь), начинающаяся из вершины i — это последовательность вершин $i_1, \ldots, i_l$, $i_j \in V$ такая, что $i_1 = i$, $i_j < i_{j+1}$, $j = \overline{1, l-1}$.

Разница между обходами JS и GSS:

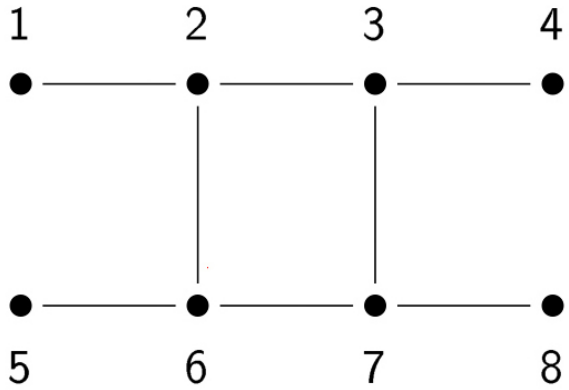
- на итерациях JS и BFS:

только: переходы в вершины смежные достигнутым к текущей итерации;

- на итерациях GSS:

1) переходы в вершины смежные достигнутым к текущей итерации;
2) **переходы через все правильные цепи, начинающиеся из вершин, достигнутых к текущей итерации.**

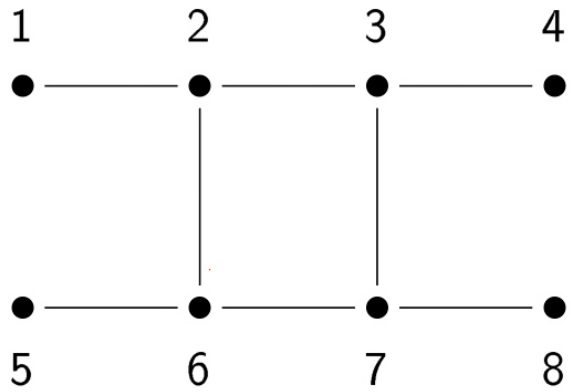
ПРИМЕРЫ ОБХОДОВ JS (BFS) И GSS



JS (BFS):

$$\begin{cases} x_1^{(k+1)} = 1 - x_2^{(k)}; \\ x_2^{(k+1)} = -x_1^{(k)} - x_3^{(k)} - x_6^{(k)}; \\ x_3^{(k+1)} = -x_2^{(k)} - x_4^{(k)} - x_7^{(k)}; \\ x_4^{(k+1)} = -x_3^{(k)}; \\ x_5^{(k+1)} = -x_6^{(k)}; \\ x_6^{(k+1)} = -x_2^{(k)} - x_5^{(k)} - x_7^{(k)}; \\ x_7^{(k+1)} = -x_3^{(k)} - x_6^{(k)} - x_8^{(k)}; \\ x_8^{(k+1)} = -x_7^{(k)}. \end{cases}$$

ПРИМЕРЫ ОБХОДОВ JS (BFS) И GSS

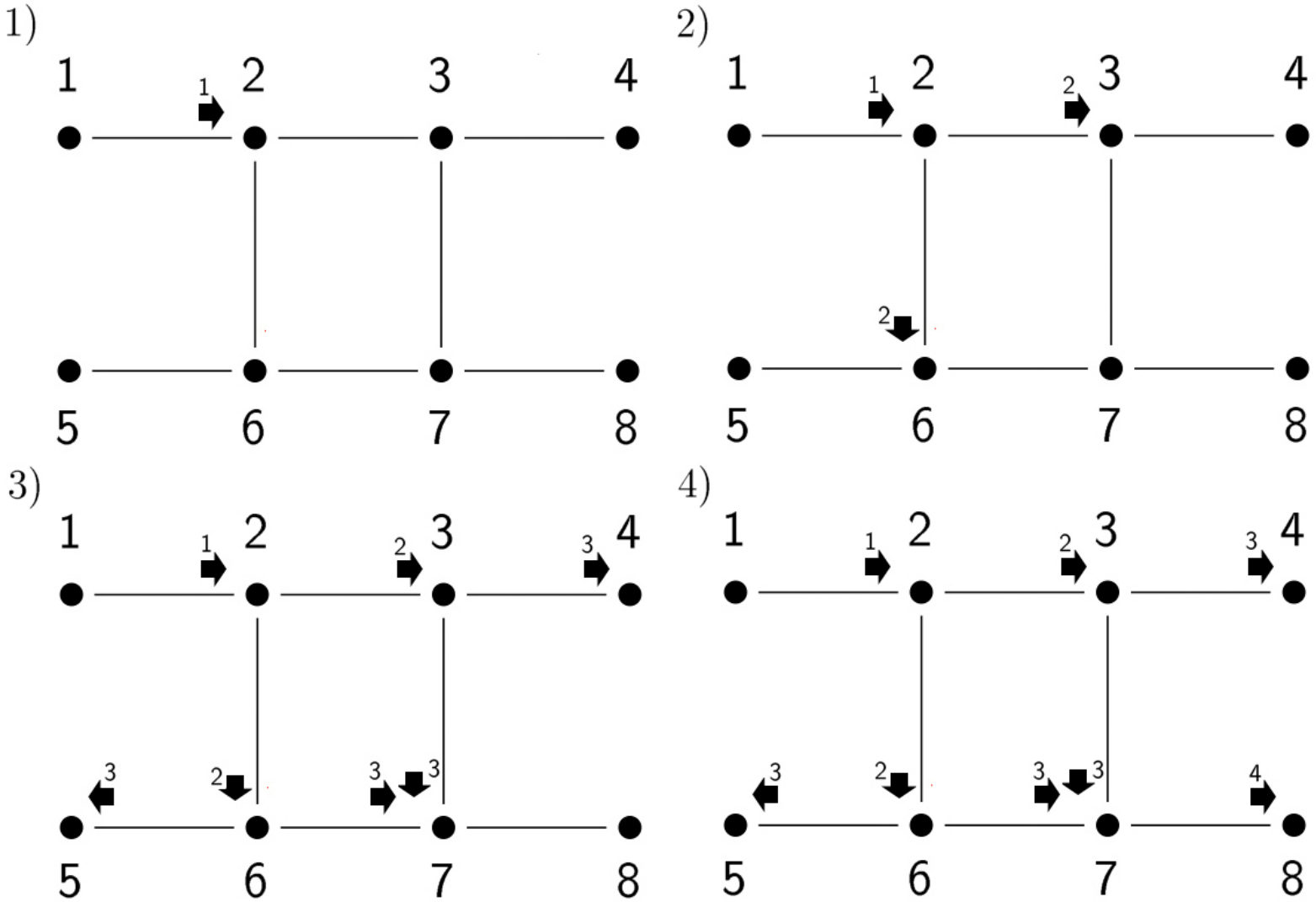


GSS:

$$\begin{cases} x_1^{(k+1)} = 1 - x_2^{(k)}; \\ x_2^{(k+1)} = -x_1^{(k+1)} - x_3^{(k)} - x_6^{(k)}; \\ x_3^{(k+1)} = -x_2^{(k+1)} - x_4^{(k)} - x_7^{(k)}; \\ x_4^{(k+1)} = -x_3^{(k+1)}; \\ x_5^{(k+1)} = -x_6^{(k)}; \\ x_6^{(k+1)} = -x_2^{(k+1)} - x_5^{(k+1)} - x_7^{(k)}; \\ x_7^{(k+1)} = -x_3^{(k+1)} - x_6^{(k+1)} - x_8^{(k)}; \\ x_8^{(k+1)} = -x_7^{(k+1)}. \end{cases}$$

ОБХОД JS (BFS)

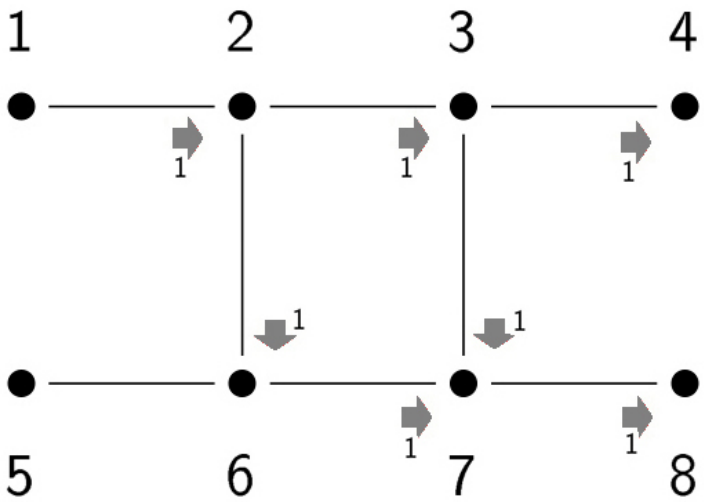
➧ — итерация JS (BFS)



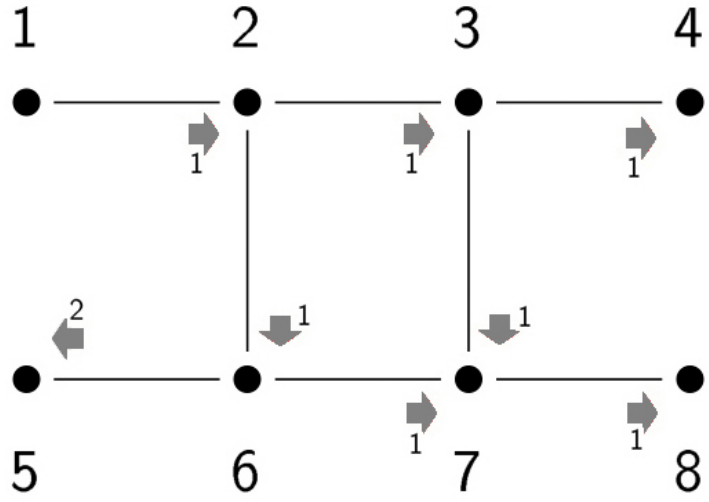
ОБХОД GSS

— итерация GSS

1)



2)



ПОСЛЕДОВАТЕЛЬНОСТИ $X^{(k)}$

$V^{(k)}$ — множество вершин, достигнутых к k-ой итерации.

JS:

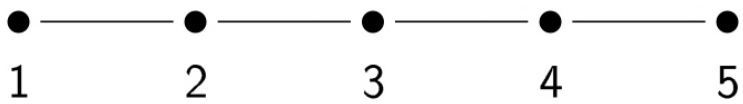
0) $x^{(0)} = (1,0,0,0,0,0,0,0)$, $V^{(0)} = \{1\}$;
1) $x^{(1)} = (1,-1,0,0,0,0,0,0)$, $V^{(1)} = \{1,2\}$;
2) $x^{(2)} = (2,-1,1,0,0,1,0,0)$, $V^{(2)} = \{1,2,3,6\}$;
3) $x^{(3)} = (2,-1,1,0,0,1,0,0)$, $V^{(3)} = \{1,2,3,4,5,6,7\}$;
4) $x^{(4)} = (5,-4,7,-1,-1,7,-2,2)$, $V^{(4)} = \{1,2,3,4,5,6,7,8\}$.

GSS:

0) $x^{(0)} = (1,0,0,0,0,0,0,0)$, $V^{(0)} = \{1\}$;
1) $x^{(1)} = (1,-1,1,-1,0,1,-2,0)$, $V^{(1)} = \{1,2,3,4,6,7,8\}$;
2) $x^{(2)} = (2,-4,7,-7,-1,7,-16,2)$, $V^{(2)} = \{1,2,3,4,5,6,7,8\}$.

ПРИМЕРЫ ОБХОДОВ JS И GSS

Правильная цепь:



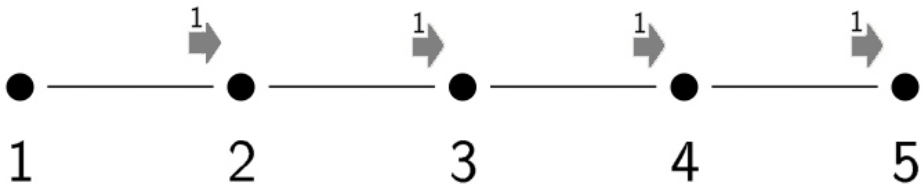
$$\text{JS:} \begin{cases} x_1^{(k+1)} = 1 - x_2^{(k)}; \\ x_2^{(k+1)} = -x_1^{(k)} - x_3^{(k)}; \\ x_3^{(k+1)} = -x_2^{(k)} - x_4^{(k)}; \\ x_4^{(k+1)} = -x_3^{(k)} - x_5^{(k)}; \\ x_5^{(k+1)} = -x_4^{(k)}. \end{cases}$$

0) $x^{(0)} = (1,0,0,0,0),\ V^{(0)} = \{1\};$
1) $x^{(1)} = (1,-1,0,0,0),\ V^{(1)} = \{1,2\};$
2) $x^{(2)} = (2,-1,1,0,0),\ V^{(2)} = \{1,2,3\};$
3) $x^{(3)} = (2,-3,1,-1,0),\ V^{(3)} = \{1,2,3,4\};$
4) $x^{(4)} = (4,-3,4,-1,1),\ V^{(4)} = \{1,2,3,4,5\}.$

ПРИМЕРЫ ОБХОДОВ JS И GSS

Правильная цепь:

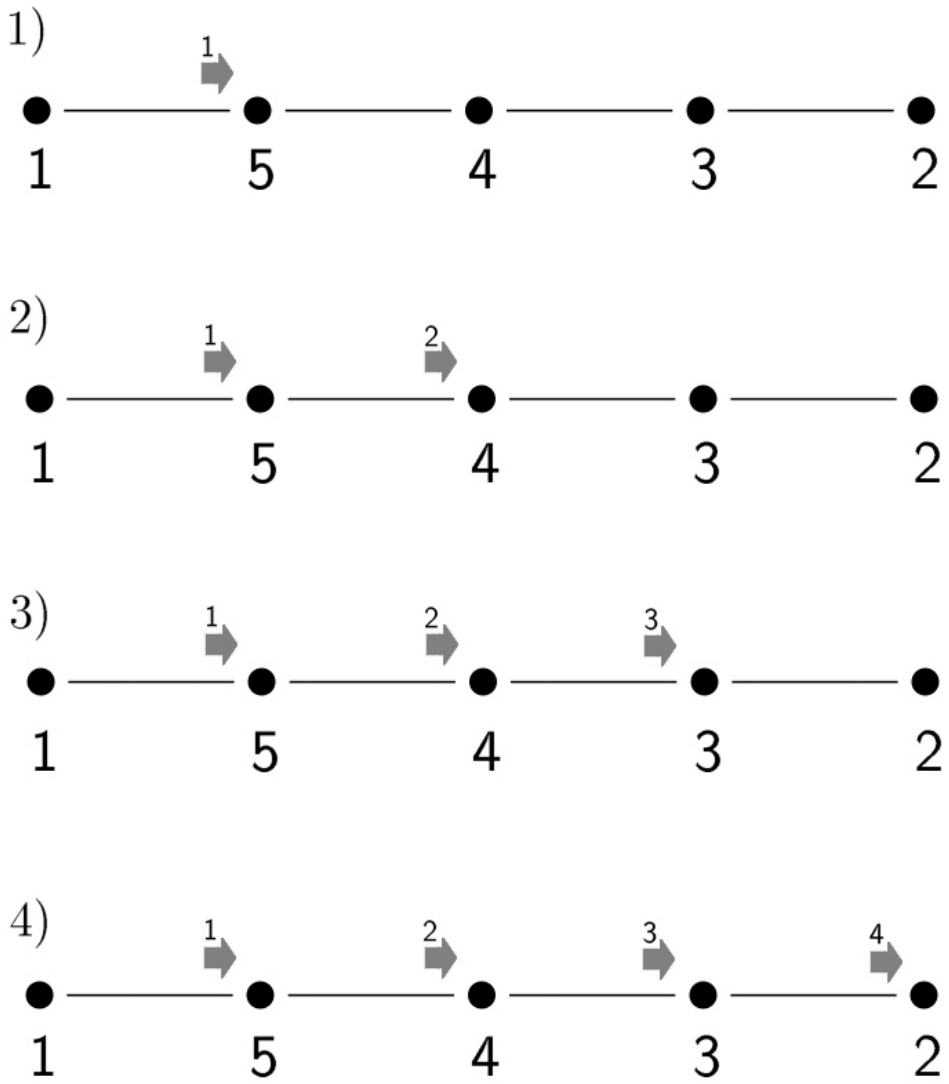
$$\text{GSS:} \begin{cases} x_1^{(k+1)} = 1 - x_2^{(k)}; \\ x_2^{(k+1)} = -x_1^{(k+1)} - x_3^{(k)}; \\ x_3^{(k+1)} = -x_2^{(k+1)} - x_4^{(k)}; \\ x_4^{(k+1)} = -x_3^{(k+1)} - x_5^{(k)}; \\ x_5^{(k+1)} = -x_4^{(k+1)}. \end{cases}$$



0) $x^{(0)} = (1, 0, 0, 0, 0)$, $V^{(0)} = \{1\}$;
1) $x^{(1)} = (1, -1, 1, -1, 1)$, $V^{(1)} = \{1, 2, 3, 4, 5\}$.

ПРИМЕРЫ ОБХОДОВ JS И GSS

Не правильная цепь:



ТРЕБУЕМОЕ КОЛИЧЕСТВО ИТЕРАЦИЙ

Количество итераций обхода заданного графа определяется:

- нумерацией его вершин,
- выбором стартовой вершины.

Поскольку

- на итерациях JS и BFS:

только: переходы в вершины смежные достигнутым к текущей итерации;

- на итерациях GSS:
  1) переходы в вершины смежные достигнутым к текущей итерации;
  2) **переходы через все правильные цепи, начинающиеся из вершин, достигнутых к текущей итерации**;

Утверждение 3.

Если задана одна и та же стартовая вершина, то для обхода GSS требуется столько же или меньше итераций, чем для BFS.

ТРЕБУЕМОЕ КОЛИЧЕСТВО ИТЕРАЦИЙ

Пусть

- $\mathcal{L} = \{i_1, \ldots, i_l\}$ — самая длинная простая цепь, исходящая из стартовой вершины: $i_1 = s$;
- $\ell(c)$ — длина цепи c;
- $C_R = \{c \subset \mathcal{L} \mid c$ — правильная цепь макс. по включению$\}$

Тогда количество итераций составит:

- N_{BFS} для BFS: $N_{\text{BFS}} = \ell(\mathcal{L})$;
- N_{GSS} для GSS:

$$N_{\text{GSS}} = \ell(\mathcal{L}) - \sum_{c \in C_R} \ell(c) \leq \ell(\mathcal{L}) = N_{\text{BFS}}$$

N_{GSS} = длина максимальной простой цепи из стартовой вершины минус сумма длин её правильных подцепей.

ОБХОДЫ ГРАФОВ:

- варианты метода простой итерации решения СЛАУ с матрицей смежности графа и специальным образом выбранной правой частью могут быть рассмотрены как обходы графов;
- обход, реализуемый в ходе проведения итераций GSS **не эквивалентен** обходам в ширину и глубину;
- если в графе присутствуют правильные цепи, исходящие из достигаемых на итерациях этого обхода вершин, то количество итераций, которые требуются для проведения итераций GSS, будет меньше, чем требуется для BFS; в худшем случае количество итераций то же.