Алгоритмическая теория игр
Лекция 8

М.Н. Вялый

Лекции в Computer Science club (Санкт-Петербург), 2019

Квазиполиномиальные алгоритмы для игр чётности

Время работы $2^{\mathrm{poly}(\log n)} = n^{\mathrm{polylog}(n)}$, где n — длина входа.

Квазиполиномиальные алгоритмы для игр чётности

Время работы $2^{\mathrm{poly}(\log n)} = n^{\mathrm{polylog}(n)}$, где n — длина входа.

Хронология:

2016 C. S. Calude, S. Jain, B. Khoussainov, W. Li, F. Stephan.

2017 M. Jurdzinski, R. Lazic.

2017 J. Fearnley, S. Jain, S. Schewe, F. Stephan, D. Wojtczak.

2018 K. Lehtinen.

2019 P. Parys.

...

Квазиполиномиальные алгоритмы для игр чётности

Время работы $2^{\mathrm{poly}(\log n)} = n^{\mathrm{polylog}(n)}$, где n — длина входа.

Подход разделяющих автоматов:

$\surd$ 2016 C. S. Calude, S. Jain B. Khoussainov, W. Li, F. Stephan.

$\surd$ 2017 M. Jurdzinski, R. Lazic.

$\surd$ 2017 J. Fearnley, S. Jain, S. Schewe, F. Stephan, D. Wojtczak.

$\surd$ 2018 K. Lehtinen.

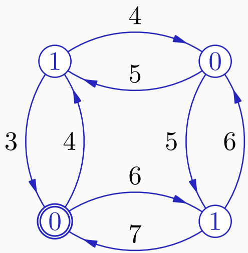
нет 2019 P. Parys.

? ...

Игры с судьёй

Основная идея

Чёт и Нечет играют себе на графе игры чётности, а за их игрой следит Судья.



Нечет выиграл!

Нечет ведёт

Нечет отыграл

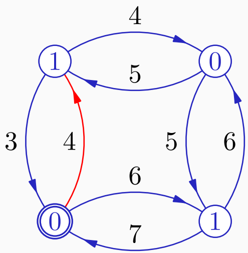
Чёт ведёт

√ Начали!

Судья может присудить победу Нечету. Если он этого не сделал, то Нечет проиграл.

Основная идея

Чёт и Нечет играют себе на графе игры чётности, а за их игрой следит Судья.



Нечет выиграл!

Нечет ведёт

Нечет отыграл

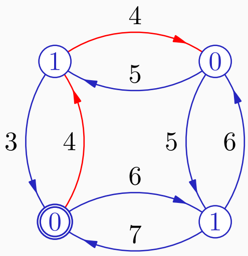
$\surd$ Чёт ведёт

Начали!

Судья может присудить победу Нечету. Если он этого не сделал, то Нечет проиграл.

Основная идея

Чёт и Нечет играют себе на графе игры чётности, а за их игрой следит Судья.



Нечет выиграл!

Нечет ведёт

$\surd$ Нечет отыграл

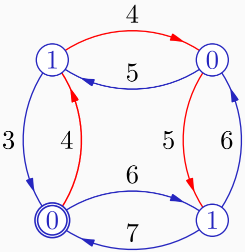
Чёт ведёт

Начали!

Судья может присудить победу Нечету. Если он этого не сделал, то Нечет проиграл.

Основная идея

Чёт и Нечет играют себе на графе игры чётности, а за их игрой следит Судья.



Нечет выиграл!

√ Нечет ведёт

Нечет отыграл

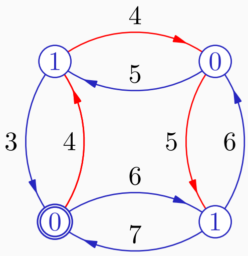
Чёт ведёт

Начали!

Судья может присудить победу Нечету. Если он этого не сделал, то Нечет проиграл.

Основная идея

Чёт и Нечет играют себе на графе игры чётности, а за их игрой следит Судья.



Нечет выиграл!

$\surd$ Нечет ведёт

Нечет отыграл

Чёт ведёт

Начали!

Судья может присудить победу Нечету. Если он этого не сделал, то Нечет проиграл.

Судьёй будет автомат

Память конечная, действия детерминированные. Автомат $\mathcal{A}$:

- Алфавит (сообщения на каждом ходе): $\Sigma = E \times \{0, 1, \dots, d-1\}$ (пары «ребро, приоритет»).
- Состояния $Q(\mathcal{A})$: конечное множество.
- Функция переходов: $\delta_{\mathcal{A}} \colon Q \times \Sigma \to Q$ (по текущему состоянию и текущему сообщению однозначно определено следующее состояние).
- Принимающее состояние $q_{\mathrm{odd}} \in Q$: Судья присуждает победу Нечету. Начальное состояние q_0 («Начали!»).

Соглашение

$d-1$ максимальный приоритет, 0 минимальный приоритет, d чётно.

Судьёй будет автомат

Память конечная, действия детерминированные. Автомат $\mathcal{A}$:

- Алфавит (сообщения на каждом ходе): $\Sigma = E \times \{0, 1, \dots, d-1\}$ (пары «ребро, приоритет»).
- Состояния $Q(\mathcal{A})$: конечное множество.
- Функция переходов: $\delta_{\mathcal{A}} \colon Q \times \Sigma \to Q$ (по текущему состоянию и текущему сообщению однозначно определено следующее состояние).
- Принимающее состояние $q_{\mathrm{odd}} \in Q$: Судья присуждает победу Нечету. Начальное состояние q_0 («Начали!»).

Соглашение

$d-1$ максимальный приоритет, 0 минимальный приоритет, d чётно.

Судьёй будет автомат

Память конечная, действия детерминированные. Автомат $\mathcal{A}$:

- Алфавит (сообщения на каждом ходе): $\Sigma = E \times \{0, 1, \ldots, d-1\}$ (пары «ребро, приоритет»).
- Состояния $Q(\mathcal{A})$: конечное множество.
- Функция переходов: $\delta_{\mathcal{A}}\colon Q \times \Sigma \to Q$ (по текущему состоянию и текущему сообщению однозначно определено следующее состояние).
- Принимающее состояние $q_{\mathrm{odd}} \in Q$: Судья присуждает победу Нечету. Начальное состояние q_0 («Начали!»).

Соглашение

$d-1$ максимальный приоритет, 0 минимальный приоритет, d чётно.

Судьёй будет автомат

Память конечная, действия детерминированные. Автомат $\mathcal{A}$:

- Алфавит (сообщения на каждом ходе): $\Sigma = E \times \{0, 1, \dots, d-1\}$ (пары «ребро, приоритет»).
- Состояния $Q(\mathcal{A})$: конечное множество.
- Функция переходов: $\delta_{\mathcal{A}} \colon Q \times \Sigma \to Q$ (по текущему состоянию и текущему сообщению однозначно определено следующее состояние).
- Принимающее состояние $q_{\mathrm{odd}} \in Q$: Судья присуждает победу Нечету. Начальное состояние q_0 («Начали!»).

Соглашение

$d-1$ максимальный приоритет, 0 минимальный приоритет, d чётно.

Судьёй будет автомат

Память конечная, действия детерминированные. Автомат $\mathcal{A}$:

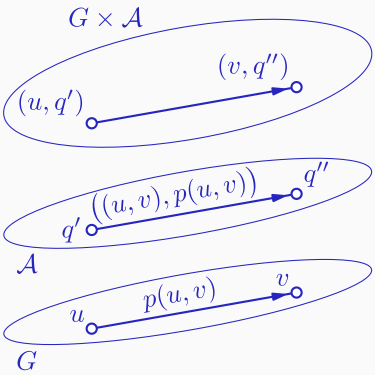
- Алфавит (сообщения на каждом ходе): $\Sigma = E \times \{0, 1, \dots, d-1\}$ (пары «ребро, приоритет»).
- Состояния $Q(\mathcal{A})$: конечное множество.
- Функция переходов: $\delta_{\mathcal{A}} \colon Q \times \Sigma \to Q$ (по текущему состоянию и текущему сообщению однозначно определено следующее состояние).
- Принимающее состояние $q_{\mathrm{odd}} \in Q$: Судья присуждает победу Нечету. Начальное состояние q_0 («Начали!»).

Соглашение

$d-1$ максимальный приоритет, 0 минимальный приоритет, d чётно.

Игры с Судьёй — это игры достижимости

Игра G, судья $\mathcal{A}$. Граф игры достижимости: $G \times \mathcal{A}$.



Вершины:

$$\{(v, q) : v \in V(G), q \in Q(\mathcal{A})\}.$$

Рёбра:

$$\Big\{((u, q'), (v, q'')) :$$

$$(u, v) \in E(G),\ q'' = \delta_{\mathcal{A}}(q', (e, p(u, v)))\Big\}.$$

Целевое множество Нечета:

$V(G) \times \{q_{\mathrm{odd}}\}$.

Игры с Судьёй — это игры достижимости (продолжение)

Утверждение

Игра на графе G, стартовая вершина v_0, с Судьёй $\mathcal{A}$ совпадает с игрой достижимости на расширенном графе $G \times \mathcal{A}$ со стартовой вершиной (v_0, q_0).

Проверка

Ход (u, v) в игре G, состояние Судьи q', меняет состояние Судьи на $q'' = \delta_{\mathcal{A}}(q', ((u, v), p(u, v)))$. Это и есть ход в игре достижимости.

Победа Нечету присуждается, когда состояние Судьи q_{odd}. Это равносильно достижению множества $V \times \{q_{\mathrm{odd}}\}$ в расширенном графе.

Игры с Судьёй — это игры достижимости (продолжение)

Утверждение

Игра на графе G, стартовая вершина v_0, с Судьёй $\mathcal{A}$ совпадает с игрой достижимости на расширенном графе $G \times \mathcal{A}$ со стартовой вершиной (v_0, q_0).

Проверка

Ход (u, v) в игре G, состояние Судьи q', меняет состояние Судьи на $q'' = \delta_{\mathcal{A}}(q', ((u, v), p(u, v)))$. Это и есть ход в игре достижимости.

Победа Нечету присуждается, когда состояние Судьи q_{odd}. Это равносильно достижению множества $V \times \{q_{\text{odd}}\}$ в расширенном графе.

Игры с Судьёй — это игры достижимости (продолжение)

Утверждение

Игра на графе G, стартовая вершина v_0, с Судьёй $\mathcal{A}$ совпадает с игрой достижимости на расширенном графе $G \times \mathcal{A}$ со стартовой вершиной (v_0, q_0).

Проверка

Ход (u, v) в игре G, состояние Судьи q', меняет состояние Судьи на $q'' = \delta_{\mathcal{A}}\big(q', ((u, v), p(u, v))\big)$. Это и есть ход в игре достижимости.

Победа Нечету присуждается, когда состояние Судьи q_{odd}. Это равносильно достижению множества $V \times \{q_{\mathrm{odd}}\}$ в расширенном графе.

Связь с играми чётности

Корректность Судьи

Если у Нечета есть выигрывающая стратегия в игре чётности, то у него есть выигрывающая стратегия в игре с Судьёй.

Полнота Судьи

Если у Чёта есть выигрывающая стратегия в игре чётности, то у него есть выигрывающая стратегия в игре с судьёй.

Утверждение

Если судья-автомат корректный и полный, то решение игры чётности сводится к решению игры достижимости на расширенном графе (решается за время, полиномиальное от размера графа и размера автомата).

Связь с играми чётности

Корректность Судьи

Если у Нечета есть выигрывающая стратегия в игре чётности, то у него есть выигрывающая стратегия в игре с Судьёй.

Полнота Судьи

Если у Чёта есть выигрывающая стратегия в игре чётности, то у него есть выигрывающая стратегия в игре с судьёй.

Утверждение

Если судья-автомат корректный и полный, то решение игры чётности сводится к решению игры достижимости на расширенном графе (решается за время, полиномиальное от размера графа и размера автомата).

Связь с играми чётности

Корректность Судьи

Если у Нечета есть выигрывающая стратегия в игре чётности, то у него есть выигрывающая стратегия в игре с Судьёй.

Полнота Судьи

Если у Чёта есть выигрывающая стратегия в игре чётности, то у него есть выигрывающая стратегия в игре с судьёй.

Утверждение

Если судья-автомат корректный и полный, то решение игры чётности сводится к решению игры достижимости на расширенном графе (решается за время, полиномиальное от размера графа и размера автомата).

Пример корректного и полного автомата

1. Судья следит, чтобы игроки придерживались позиционных стратегий. Если из одной и той же позиции игрок сделал два разных хода, Судья объявляет его проигравшим. (Нечет проигрывает при переходе в поглощающее состояние $\neq q_{\mathrm{odd}}$.)
2. Судья отсчитывает n ходов (n количество вершин графа игры).
3. Затем Судья определяет максимальный приоритет p^* за следующие n ходов. Если p^* нечётный, присуждает победу Нечету. Иначе — Чёту.

Пример корректного и полного автомата

1. Судья следит, чтобы игроки придерживались позиционных стратегий. Если из одной и той же позиции игрок сделал два разных хода, Судья объявляет его проигравшим. (Нечет проигрывает при переходе в поглощающее состояние $\neq q_{\mathrm{odd}}$.)
2. Судья отсчитывает n ходов (n количество вершин графа игры).
3. Затем Судья определяет максимальный приоритет p^* за следующие n ходов. Если p^* нечётный, присуждает победу Нечету. Иначе — Чёту.

Пример корректного и полного автомата

1. Судья следит, чтобы игроки придерживались позиционных стратегий. Если из одной и той же позиции игрок сделал два разных хода, Судья объявляет его проигравшим. (Нечет проигрывает при переходе в поглощающее состояние $\neq q_{\mathrm{odd}}$.)
2. Судья отсчитывает n ходов (n количество вершин графа игры).
3. Затем Судья определяет максимальный приоритет p^* за следующие n ходов. Если p^* нечётный, присуждает победу Нечету. Иначе — Чёту.

Корректность и полнота примера

1. Отклонение от позиционной стратегии карается немедленным проигрышем. Поэтому каждый из игроков вынужден придерживаться позиционной стратегии.
2. После $\leqslant n$ ходов партия выйдет на цикл.
3. Если Чёт придерживается выигрывающей стратегии, максимальный приоритет за следующие n ходов (рёбра цикла будут пройдены) чётное и Чёт выиграет (полнота).
4. Если Нечет придерживается выигрывающей позиционной стратегии, максимальный приоритет будет нечётным и Нечет выиграет (корректность).

Корректность и полнота примера

1. Отклонение от позиционной стратегии карается немедленным проигрышем. Поэтому каждый из игроков вынужден придерживаться позиционной стратегии.
2. После $\leqslant n$ ходов партия выйдет на цикл.
3. Если Чёт придерживается выигрывающей стратегии, максимальный приоритет за следующие n ходов (рёбра цикла будут пройдены) чётное и Чёт выиграет (полнота).
4. Если Нечет придерживается выигрывающей позиционной стратегии, максимальный приоритет будет нечётным и Нечет выиграет (корректность).

Корректность и полнота примера

1. Отклонение от позиционной стратегии карается немедленным проигрышем. Поэтому каждый из игроков вынужден придерживаться позиционной стратегии.
2. После $\leqslant n$ ходов партия выйдет на цикл.
3. Если Чёт придерживается выигрывающей стратегии, максимальный приоритет за следующие n ходов (рёбра цикла будут пройдены) чётное и Чёт выиграет (полнота).
4. Если Нечет придерживается выигрывающей позиционной стратегии, максимальный приоритет будет нечётным и Нечет выиграет (корректность).

Корректность и полнота примера

1. Отклонение от позиционной стратегии карается немедленным проигрышем. Поэтому каждый из игроков вынужден придерживаться позиционной стратегии.
2. После $\leqslant n$ ходов партия выйдет на цикл.
3. Если Чёт придерживается выигрывающей стратегии, максимальный приоритет за следующие n ходов (рёбра цикла будут пройдены) чётное и Чёт выиграет (полнота).
4. Если Нечет придерживается выигрывающей позиционной стратегии, максимальный приоритет будет нечётным и Нечет выиграет (корректность).

Количество состояний автомата из примера

Если Судья помнит ходы во всех вершинах, ему нужно порядка n^n состояний только для этого.

Подсчёт числа ходов и определение максимума требуют полиномиального по n количества состояний.

Количество состояний автомата из примера

Если Судья помнит ходы во всех вершинах, ему нужно порядка n^n состояний только для этого.

Подсчёт числа ходов и определение максимума требуют полиномиального по n количества состояний.

Автомат с несколькими счётчиками

Конструкция

У Судьи есть $d/2$ помощников $\mathcal{A}_p$, $0 < p < d$ нечётное.

Действия помощника $\mathcal{A}_p$:

1. Подсчитывает, сколько раз встретился приоритет p с тех пор, как не встречалось приоритетов выше p.
2. Если значение счётчика становится равным n, наступает переполнение счётчика и $\mathcal{A}_p$ переходит в состояние q_{odd} (Судья присуждает выигрыш Нечету).

Конструкция

У Судьи есть $d/2$ помощников $\mathcal{A}_p$, $0 < p < d$ нечётное.

Действия помощника $\mathcal{A}_p$:

1. Подсчитывает, сколько раз встретился приоритет p с тех пор, как не встречалось приоритетов выше p.
2. Если значение счётчика становится равным n, наступает переполнение счётчика и $\mathcal{A}_p$ переходит в состояние q_{odd} (Судья присуждает выигрыш Нечету).

Конструкция

У Судьи есть $d/2$ помощников $\mathcal{A}_p$, $0 < p < d$ нечётное.

Действия помощника $\mathcal{A}_p$:

1. Подсчитывает, сколько раз встретился приоритет p с тех пор, как не встречалось приоритетов выше p.
2. Если значение счётчика становится равным n, наступает переполнение счётчика и $\mathcal{A}_p$ переходит в состояние q_{odd} (Судья присуждает выигрыш Нечету).

Корректность

Пусть Нечет придерживается в игре достижимости выигрывающей стратегии в игре чётности.

Тогда $p = \limsup p(v_t, v_{t+1})$ нечётное.

Значит, счётчик автомата $\mathcal{A}_p$ рано или поздно переполнится.

Полнота

Пусть Чёт придерживается позиционной выигрывающей стратегии в игре чётности.

Утверждение

Ни один счётчик не переполнится.

Доказательство

Если счётчик $\mathcal{A}_p$ переполнился, то случилось n ходов с приоритетом p, остальные приоритеты на этом отрезке партии меньше.

Концов рёбер $\geqslant n+1$, какие-то совпадают. Получаем нечётный цикл. Нечет выигрывает в игре чётности, двигаясь по нему.

Полнота

Пусть Чёт придерживается позиционной выигрывающей стратегии в игре чётности.

Утверждение

Ни один счётчик не переполнится.

Доказательство

Если счётчик $\mathcal{A}_p$ переполнился, то случилось n ходов с приоритетом p, остальные приоритеты на этом отрезке партии меньше.
Концов рёбер $\geqslant n+1$, какие-то совпадают.
Получаем нечётный цикл. Нечет выигрывает в игре чётности, двигаясь по нему.

Полнота

Пусть Чёт придерживается позиционной выигрывающей стратегии в игре чётности.

Утверждение

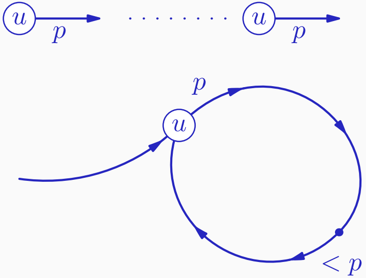
Ни один счётчик не переполнится.

Доказательство

Если счётчик $\mathcal{A}_p$ переполнился, то случилось n ходов с приоритетом p, остальные приоритеты на этом отрезке партии меньше.

Концов рёбер $\geqslant n + 1$, какие-то совпадают.

Получаем нечётный цикл. Нечет выигрывает в игре чётности, двигаясь по нему.



Количество состояний

У каждого $\mathcal{A}_p$ количество состояний n.

И ещё одно общее состояние q_{odd}.

С таким количеством помощников Судье не нужно ничего помнить.

Итого: $1 + n^{d/2}$ состояний.

Замечание

Построили алгоритм решения игр чётности, который лучше алгоритма, получающегося из псевдополиномиального алгоритма решения циклических игр.

Количество состояний

У каждого $\mathcal{A}_p$ количество состояний n.

И ещё одно общее состояние q_{odd}.

С таким количеством помощников Судье не нужно ничего помнить.

Итого: $1 + n^{d/2}$ состояний.

Замечание

Построили алгоритм решения игр чётности, который лучше алгоритма, получающегося из псевдополиномиального алгоритма решения циклических игр.

Количество состояний

У каждого $\mathcal{A}_p$ количество состояний n.

И ещё одно общее состояние q_{odd}.

С таким количеством помощников Судье не нужно ничего помнить.

Итого: $1 + n^{d/2}$ состояний.

Замечание

Построили алгоритм решения игр чётности, который лучше алгоритма, получающегося из псевдополиномиального алгоритма решения циклических игр.

Абстрактные счётчики на упорядоченных деревьях

Один счётчик, но правила изменения сложнее

1. Автомат по-прежнему обращает внимание только на приоритеты.
2. Состояния автомата линейно упорядочены, начальное состояние q_0 минимальное, целевое для Нечета состояние q_{odd} максимальное.
3. Изменения состояний управляются корневым упорядоченным однородным деревом T глубины $d/2$.

Один счётчик, но правила изменения сложнее

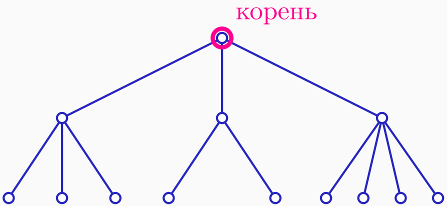
1. Автомат по-прежнему обращает внимание только на приоритеты.
2. Состояния автомата линейно упорядочены, начальное состояние q_0 минимальное, целевое для Нечета состояние q_{odd} максимальное.
3. Изменения состояний управляются корневым упорядоченным однородным деревом T глубины $d/2$.

Один счётчик, но правила изменения сложнее

1. Автомат по-прежнему обращает внимание только на приоритеты.
2. Состояния автомата линейно упорядочены, начальное состояние q_0 минимальное, целевое для Нечета состояние q_{odd} максимальное.
3. Изменения состояний управляются корневым упорядоченным однородным деревом T глубины $d/2$.

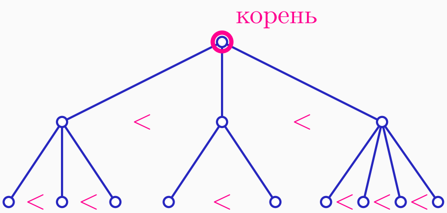
Упорядоченные деревья: терминология

Корневое дерево: выделена особая вершина — корень.



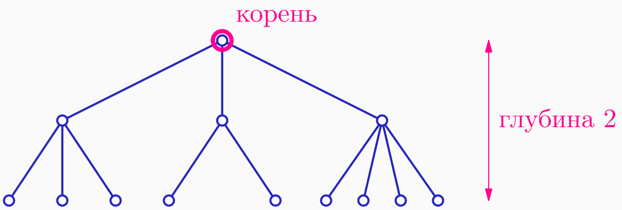
Упорядоченные деревья: терминология

Упорядоченное дерево: на дочках каждой вершины задан линейный порядок.



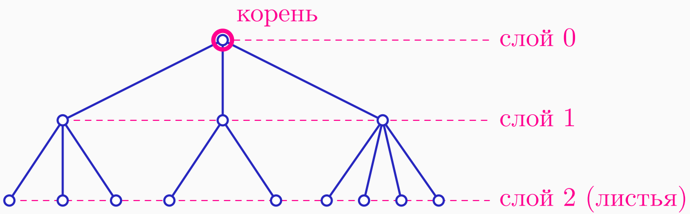
Упорядоченные деревья: терминология

Однородное дерево: длина пути из корня в лист одна и та же для всех листьев (глубина дерева).

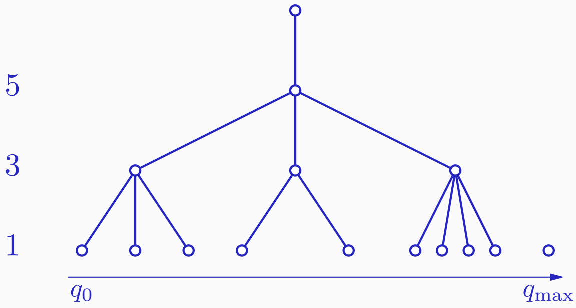


Упорядоченные деревья: терминология

Слой: вершины на одинаковом расстоянии от корня.

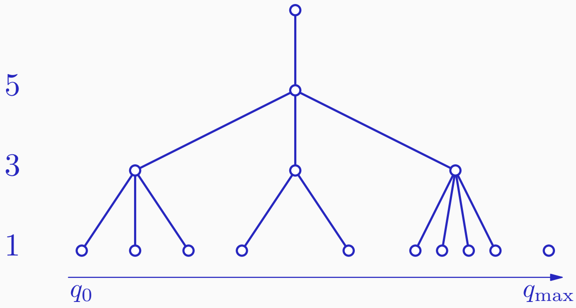


Автомат $\mathcal{A}_T$ из дерева T



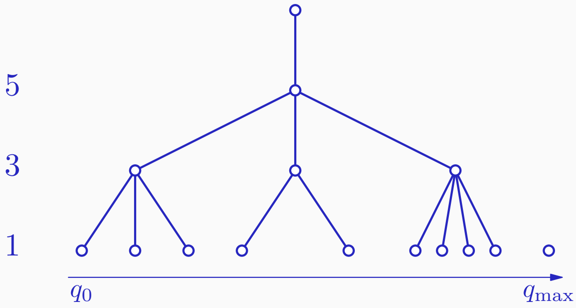
Состояния автомата: листья $L(T)$ дерева T и особое состояние $q_{\mathsf{max}} = q_{\mathrm{odd}}$.

Автомат $\mathcal{A}_T$ из дерева T



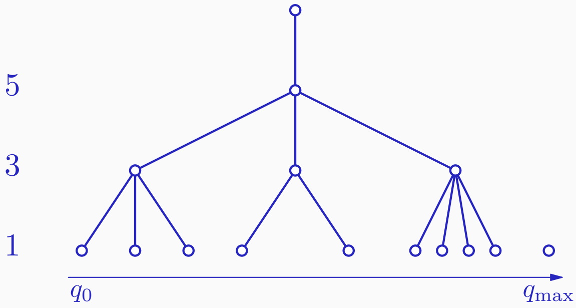
Начальное состояние q_0 — минимальный лист дерева T.

Автомат $\mathcal{A}_T$ из дерева T



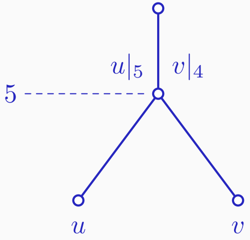
Слои T, не считая корня, нумеруем нечётными числами от $d-1$ до 1, начиная от сыновей корня.

Автомат $\mathcal{A}_T$ из дерева T



Для определения функции переходов нужна обрезка листьев в дереве.

Обрезка листьев

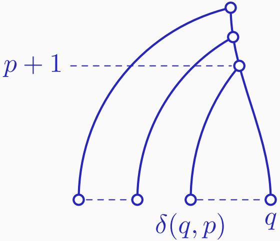


Определение

Обрезка $v|_p$ листа v — это вершина на пути от корня к листу v на уровне p, если p нечётно, на уровне $p+1$, если p чётно.

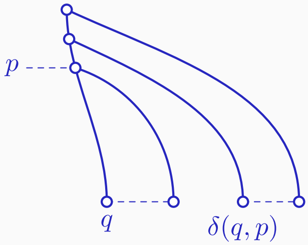
Функция переходов автомата $\mathcal{A}_T$

$$\delta(q,p)=\begin{cases}\min(q' : q'|_p = q|_p), & (p \text{ чётно}),\\ \min(q' : q'|_p > q|_p), & \\ (p \text{ нечётно и } q|_p \text{ не максимально на слое } p), & \\ q_{\mathrm{odd}}, & (p \text{ нечётно и } q|_p \text{ максимально на слое } p).\end{cases}$$



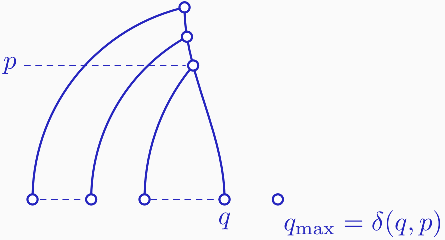
Функция переходов автомата $\mathcal{A}_T$

$$\delta(q,p) = \begin{cases} \min(q' : q'|_p = q|_p), & (p \text{ чётно}), \\ \min(q' : q'|_p > q|_p), & \\ (p \text{ нечётно и } q|_p \text{ не максимально на слое } p), & \\ q_{\text{odd}}, & (p \text{ нечётно и } q|_p \text{ максимально на слое } p). \end{cases}$$



Функция переходов автомата $\mathcal{A}_T$

$$\delta(q,p) = \begin{cases} \min(q' : q'|_p = q|_p), & (p \text{ чётно}), \\ \min(q' : q'|_p > q|_p), & \\ \multicolumn{2}{l}{(p \text{ нечётно и } q|_p \text{ не максимально на слое } p),} \\ q_{\mathrm{odd}}, & (p \text{ нечётно и } q|_p \text{ максимально на слое } p). \end{cases}$$



Монотонность автомата

Утверждение

Для любого p и любых двух состояний $q_1 \leqslant q_2$ выполняется неравенство $\delta(q_1, p) \leqslant \delta(q_2, p)$.

Доказательство

1. Обрезки монотонны: $q_1 \leqslant q_2 \quad \Rightarrow \quad q_1|_p \leqslant q_2|_p$.
   Обратно слабее: $q_1|_p < q_2|_p \quad \Rightarrow \quad q_1 < q_2$.
2. $\min(q' : q'|_p = q_1|_p) \leqslant \min(q' : q'|_p = q_2|_p)$, так как при $q_1|_p < q_2|_p$, все элементы слева меньше всех элементов справа. При $q_1|_p = q_2|_p$ множества совпадают.
3. $\min(q' : q'|_p > q_1|_p) \leqslant \min(q' : q'|_p > q_2|_p)$, так как при $q_1|_p \leqslant q_2|_p$ минимум слева берётся по бо́льшему множеству.

Монотонность автомата

Утверждение

Для любого p и любых двух состояний $q_1 \leqslant q_2$ выполняется неравенство $\delta(q_1, p) \leqslant \delta(q_2, p)$.

Доказательство

1. Обрезки монотонны: $q_1 \leqslant q_2 \quad \Rightarrow \quad q_1|_p \leqslant q_2|_p$.
   Обратно слабее: $q_1|_p < q_2|_p \quad \Rightarrow \quad q_1 < q_2$.
2. $\min(q' : q'|_p = q_1|_p) \leqslant \min(q' : q'|_p = q_2|_p)$, так как при $q_1|_p < q_2|_p$, все элементы слева меньше всех элементов справа. При $q_1|_p = q_2|_p$ множества совпадают.
3. $\min(q' : q'|_p > q_1|_p) \leqslant \min(q' : q'|_p > q_2|_p)$, так как при $q_1|_p \leqslant q_2|_p$ минимум слева берётся по бо́льшему множеству.

Монотонность автомата

Утверждение

Для любого p и любых двух состояний $q_1 \leqslant q_2$ выполняется неравенство $\delta(q_1, p) \leqslant \delta(q_2, p)$.

Доказательство

1. Обрезки монотонны: $q_1 \leqslant q_2 \quad \Rightarrow \quad q_1|_p \leqslant q_2|_p$.
   Обратно слабее: $q_1|_p < q_2|_p \quad \Rightarrow \quad q_1 < q_2$.
2. $\min(q' : q'|_p = q_1|_p) \leqslant \min(q' : q'|_p = q_2|_p)$, так как при $q_1|_p < q_2|_p$, все элементы слева меньше всех элементов справа. При $q_1|_p = q_2|_p$ множества совпадают.
3. $\min(q' : q'|_p > q_1|_p) \leqslant \min(q' : q'|_p > q_2|_p)$, так как при $q_1|_p \leqslant q_2|_p$ минимум слева берётся по бо́льшему множеству.

Монотонность автомата

Утверждение

Для любого p и любых двух состояний $q_1 \leqslant q_2$ выполняется неравенство $\delta(q_1, p) \leqslant \delta(q_2, p)$.

Доказательство

1. Обрезки монотонны: $q_1 \leqslant q_2 \quad \Rightarrow \quad q_1|_p \leqslant q_2|_p$.
   Обратно слабее: $q_1|_p < q_2|_p \quad \Rightarrow \quad q_1 < q_2$.
2. $\min(q' : q'|_p = q_1|_p) \leqslant \min(q' : q'|_p = q_2|_p)$, так как при $q_1|_p < q_2|_p$, все элементы слева меньше всех элементов справа. При $q_1|_p = q_2|_p$ множества совпадают.
3. $\min(q' : q'|_p > q_1|_p) \leqslant \min(q' : q'|_p > q_2|_p)$, так как при $q_1|_p \leqslant q_2|_p$ минимум слева берётся по бо́льшему множеству.

Корректность автомата из дерева

Лемма

Любой автомат $\mathcal{A}_T$ корректен для любой игры G.

Доказательство

Корректность автомата из дерева

Лемма

Любой автомат $\mathcal{A}_T$ корректен для любой игры G.

Доказательство

Пусть Нечет играет по выигрывающей позиционной стратегии в игре чётности, когда играет в игру с автоматом $\mathcal{A}_T$.

Верхний предел приоритетов p в такой партии нечётен.

Корректность автомата из дерева

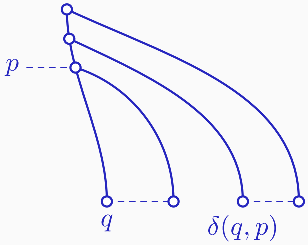
Лемма

Любой автомат $\mathcal{A}_T$ корректен для любой игры G.

Доказательство

Рассмотрим партию с того момента, когда приоритеты выше p не встречаются.

Переход по ребру с приоритетом p: обрезка состояния на уровне p увеличивается.



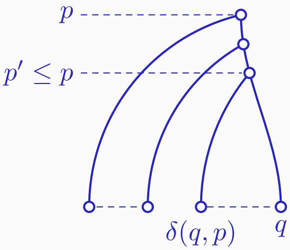
Лемма

Любой автомат $\mathcal{A}_T$ корректен для любой игры G.

Доказательство

Рассмотрим партию с того момента, когда приоритеты выше p не встречаются.

Переход по ребру с чётным приоритетом $< p$: обрезка на уровне p не изменяется.



Корректность автомата из дерева

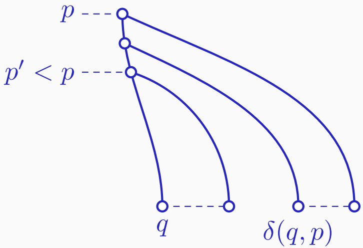
Лемма

Любой автомат $\mathcal{A}_T$ корректен для любой игры G.

Доказательство

Рассмотрим партию с того момента, когда приоритеты выше p не встречаются.

Переход по ребру с нечётным приоритетом $< p$: обрезка состояния на уровне p не уменьшается.



Корректность автомата из дерева

Лемма

Любой автомат $\mathcal{A}_T$ корректен для любой игры G.

Доказательство

Рассмотрим партию с того момента, когда приоритеты выше p не встречаются. Разобранные случаи показывают, что обрезка на уровне p никогда не уменьшается, а увеличивается бесконечное количество раз. Значит, рано или поздно автомат окажется в состоянии q_{odd}.

Полнота не всегда выполняется

Пусть дерево — это путь из корня в лист. У автомата всего два состояния.

Нечет выигрывает в игре с судьёй при любом ходе с нечётным приоритетом.

Упражнение

Приведите пример такой игры, в которой есть выигрывающая стратегия у Чёта, но Нечет может добиться прохождения по ребру с нечётным приоритетом, играя против этой стратегии.

Полнота не всегда выполняется

Пусть дерево — это путь из корня в лист. У автомата всего два состояния.

Нечет выигрывает в игре с судьёй при любом ходе с нечётным приоритетом.

Упражнение

Приведите пример такой игры, в которой есть выигрывающая стратегия у Чёта, но Нечет может добиться прохождения по ребру с нечётным приоритетом, играя против этой стратегии.

Полнота автомата на дереве сильно связных компонент стратегического графа

Цель

Теорема

Для любой игры чётности G на графе с n позициями и d приоритетами существует такое дерево T размера $O(nd)$, что автомат $\mathcal{A}_T$ полон для этой игры.

Другими словами:

Пусть в игре чётности G у Чёта есть выигрывающая стратегия $x\colon V_0 \to V$.

Мы хотим построить такое небольшое дерево, что оно задаёт полный (и корректный) автомат $\mathcal{A}_T$: в игре с судьёй $\mathcal{A}_T$ Чёт имеет выигрывающую стратегию.

Теорема

Для любой игры чётности G на графе с n позициями и d приоритетами существует такое дерево T размера $O(nd)$, что автомат $\mathcal{A}_T$ полон для этой игры.

Другими словами:

Пусть в игре чётности G у Чёта есть выигрывающая стратегия $x\colon V_0 \to V$.

Мы хотим построить такое небольшое дерево, что оно задаёт полный (и корректный) автомат $\mathcal{A}_T$: в игре с судьёй $\mathcal{A}_T$ Чёт имеет выигрывающую стратегию.

Стратегический граф и его подграфы

Граф G_x получается удалением всех ходов Чёта, не входящих в стратегию x всех позиций и рёбер, недостижимых из начальной.

Граф $G_x(p)$ получается из графа G_x удалением всех рёбер с приоритетами выше p.

Утверждение

В стратегическом графе Чёта нет нечётных циклов.

Доказательство

Если в стратегическом графе есть нечётный цикл, то Нечет может выйти на этот цикл и двигаться по нему неограниченно долго. В такой партии Нечет выигрывает, что противоречит выбору стратегии Чёта.

Стратегический граф и его подграфы

Граф G_x получается удалением всех ходов Чёта, не входящих в стратегию x всех позиций и рёбер, недостижимых из начальной.

Граф $G_x(p)$ получается из графа G_x удалением всех рёбер с приоритетами выше p.

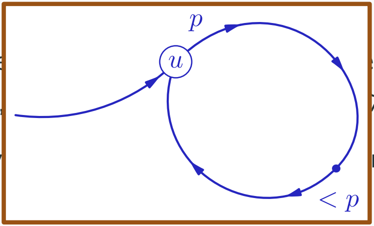
Утверждение

В стратегическом графе Чёта нет нечётных циклов.

Доказательство

Если в стратегическом графе есть нечётный цикл, то Нечет может выйти на этот цикл и двигаться по нему неограниченно долго. В такой партии Нечет выигрывает, что противоречит выбору стратегии Чёта.

Граф G_x получае



та, не входящих в стратегию x всех позиций и рёбер, й.

Граф $G_x(p)$ полу м всех рёбер с приоритетами выше p.

Утверждение

В стратегическом графе Чёта нет нечётных циклов.

Доказательство

Если в стратегическом графе есть нечётный цикл, то Нечет может выйти на этот цикл и двигаться по нему неограниченно долго. В такой партии Нечет выигрывает, что противоречит выбору стратегии Чёта.

Компоненты сильной связности стратегического графа

Компоненты сильной связности (КСС)

$u \sim v \quad \Leftrightarrow \quad$ из u достижима v и наоборот.

Классы эквивалентности этого отношения называются компонентами сильной связности.

Утверждение

Рёбра с нечётным приоритетом p не лежат в компонентах сильной связности графа $G_x(p)$.

Доказательство

Если (u, v) — ребро с нечётным приоритетом p, то v достижима из u. Но u недостижима из v: иначе был бы нечётный цикл.

Компоненты сильной связности стратегического графа

Компоненты сильной связности (КСС)

$u \sim v \quad \Leftrightarrow \quad$ из u достижима v и наоборот.

Классы эквивалентности этого отношения называются компонентами сильной связности.

Утверждение

Рёбра с нечётным приоритетом p не лежат в компонентах сильной связности графа $G_x(p)$.

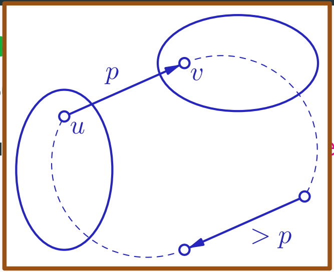
Доказательство

Если (u, v) — ребро с нечётным приоритетом p, то v достижима из u. Но u недостижима из v: иначе был бы нечётный цикл.

Компоненты сильной связности стратегического графа

Компоненты сильной связности (

$u \sim v \quad \Leftrightarrow \quad$ из u достижима v и нао



Классы эквивалентности этого отнош ентами сильной связности.

Утверждение

Рёбра с нечётным приоритетом p не лежат в компонентах сильной связности графа $G_x(p)$.

Доказательство

Если (u, v) — ребро с нечётным приоритетом p, то v достижима из u. Но u недостижима из v: иначе был бы нечётный цикл.

Построение дерева

Вершины дерева T_x — подмножества вершин стратегического графа. Более точно, это все вершины графа $V(G_x)$ и КСС его подграфов $G_x(p)$, p нечётное.

Отношение «быть потомком» в дереве T_x совпадает с отношением включения на этих множествах.

Более явно:

Потомками корня являются КСС графа G_x.

Если C — КСС $G_x(p+2)$, то её дочки — те КСС $G_x(p)$, которые лежат в C.

Построение дерева

Вершины дерева T_x — подмножества вершин стратегического графа. Более точно, это все вершины графа $V(G_x)$ и КСС его подграфов $G_x(p)$, p нечётное.

Отношение «быть потомком» в дереве T_x совпадает с отношением включения на этих множествах.

Более явно:

Потомками корня являются КСС графа G_x.

Если C — КСС $G_x(p+2)$, то её дочки — те КСС $G_x(p)$, которые лежат в C.

Порядок на дочках

Конденсация графа G — граф сильно связных компонент. Вершины — КСС.
Ребра — такие пары (C_1, C_2), что есть ребро в G с началом в C_1 и концом в C_2.

Утверждение (очевидное)

Конденсация — это ациклический граф.

Лемма (топологическая сортировка)

Вершины ациклического графа можно упорядочить так, чтобы начало каждого ребра графа имело меньший номер, чем конец.

Называем такое упорядочение линейным продолжением.

Определение порядка

Выберем линейное продолжение для конденсации графа $G_x(p)$ и упорядочим КСС в соответствии с ним.

Порядок на дочках

Конденсация графа G — граф сильно связных компонент. Вершины — КСС. Ребра — такие пары (C_1, C_2), что есть ребро в G с началом в C_1 и концом в C_2.

Утверждение (очевидное)

Конденсация — это ациклический граф.

Лемма (топологическая сортировка)

Вершины ациклического графа можно упорядочить так, чтобы начало каждого ребра графа имело меньший номер, чем конец.

Называем такое упорядочение линейным продолжением.

Определение порядка

Выберем линейное продолжение для конденсации графа $G_x(p)$ и упорядочим КСС в соответствии с ним.

Порядок на дочках

Конденсация графа G — граф сильно связных компонент. Вершины — КСС. Ребра — такие пары (C_1, C_2), что есть ребро в G с началом в C_1 и концом в C_2.

Утверждение (очевидное)

Конденсация — это ациклический граф.

Лемма (топологическая сортировка)

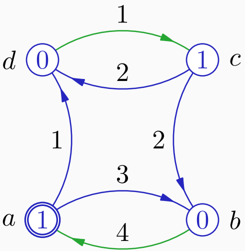
Вершины ациклического графа можно упорядочить так, чтобы начало каждого ребра графа имело меньший номер, чем конец.

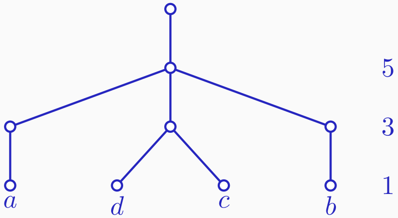
Называем такое упорядочение линейным продолжением.

Определение порядка

Выберем линейное продолжение для конденсации графа $G_x(p)$ и упорядочим КСС в соответствии с ним.

Пример графа КСС





Размер дерева КСС

Утверждение

В дереве T_x не более $1 + dn/2$ вершин, здесь n — количество вершин в графе игры чётности, d — количество приоритетов.

Доказательство

Каждый слой p, всего $d/2$ штук, соответствует разбиению позиций графа G_x (не более n позиций) на компоненты сильной связности, их тоже не больше n. И ещё есть корень.

Размер дерева КСС

Утверждение

В дереве T_x не более $1 + dn/2$ вершин, здесь n — количество вершин в графе игры чётности, d — количество приоритетов.

Доказательство

Каждый слой p, всего $d/2$ штук, соответствует разбиению позиций графа G_x (не более n позиций) на компоненты сильной связности, их тоже не больше n. И ещё есть корень.

Полнота дерева КСС

Лемма

Если x — выигрывающая позиционная стратегия Чёта в игре чётности, то автомат $\mathcal{A}_x$, построенный по дереву T_x, обладает свойством полноты.

Доказательство

Чёт придерживается стратегии x. Докажем, что в такой партии автомат никогда не окажется в состоянии q_{odd}.

Ходы с чётными приоритетами не увеличивают значение счётчика.

Полнота дерева КСС

Лемма

Если x — выигрывающая позиционная стратегия Чёта в игре чётности, то автомат $\mathcal{A}_x$, построенный по дереву T_x, обладает свойством полноты.

Доказательство

Чёт придерживается стратегии x. Докажем, что в такой партии автомат никогда не окажется в состоянии q_{odd}.

Ходы с чётными приоритетами не увеличивают значение счётчика.

Пусть сделан ход (u, v) с нечётным приоритетом p. Позиции u и v лежат в разных компонентах $C_u < C_v$ связности графа $G_x(p)$.

Пусть q' — лист-потомок C_v. Тогда $q|_p = C_u < C_v = q'|_p$.

Полнота дерева КСС

Лемма

Если x — выигрывающая позиционная стратегия Чёта в игре чётности, то автомат $\mathcal{A}_x$, построенный по дереву T_x, обладает свойством полноты.

Доказательство

Чёт придерживается стратегии x. Докажем, что в такой партии автомат никогда не окажется в состоянии q_{odd}.

Ходы с чётными приоритетами не увеличивают значение счётчика.

Пусть сделан ход (u, v) с нечётным приоритетом p. Позиции u и v лежат в разных компонентах $C_u < C_v$ связности графа $G_x(p)$.

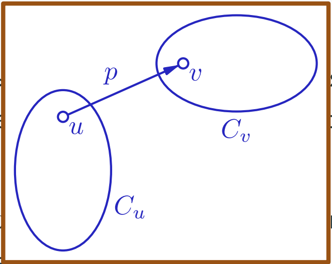
Пусть q' — лист-потомок C_v. Тогда $q|_p = C_u < C_v = q'|_p$.

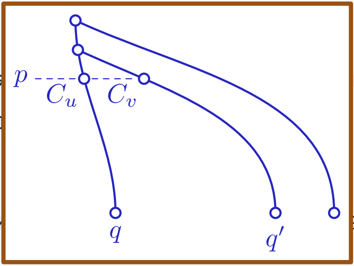
Полнота дерева КСС

Лемма

Если x — вы … ности, то автомат

$\mathcal{A}_x$, построе …





Доказател …

Чёт придер … автомат никогда

не окажется в состоянии q_{odd}.

Ходы с чётными приоритетами не увеличивают значение счётчика.

Пусть сделан ход (u, v) с нечётным приоритетом p. Позиции u и v лежат в разных компонентах $C_u < C_v$ связности графа $G_x(p)$.

Пусть q' — лист-потомок C_v. Тогда $q|_p = C_u < C_v = q'|_p$.

Итог

- Построен корректный полный автомат размера не больше $1 + dn/2$.
- Но при его построении использована выигрывающая стратегия Чёта.
  Если знать выигрывающую стратегию, то автомат уже не нужен.

Ещё одна идея

Нужно брать дерево с запасом, чтобы в нём содержались все возможные деревья КСС стратегических графов.

Итог

- Построен корректный полный автомат размера не больше $1 + dn/2$.
- Но при его построении использована выигрывающая стратегия Чёта. Если знать выигрывающую стратегию, то автомат уже не нужен.

Ещё одна идея

Нужно брать дерево с запасом, чтобы в нём содержались все возможные деревья КСС стратегических графов.

Итог

- Построен корректный полный автомат размера не больше $1 + dn/2$.
- Но при его построении использована выигрывающая стратегия Чёта.
  Если знать выигрывающую стратегию, то автомат уже не нужен.

Ещё одна идея

Нужно брать дерево с запасом, чтобы в нём содержались все возможные деревья КСС стратегических графов.