Алгоритмическая теория игр
Лекция 2

М.Н. Вялый

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

Альт-время и обычная память

Теорема

$\mathrm{AltTIME}(T(n) \subseteq \mathrm{DSPACE}(T(n))$ при $T(n) \geqslant n$, $T(n)$ конструируемая по памяти.

Теорема

$\mathrm{DSPACE}(S(n)) \subseteq \mathrm{AltTIME}(S(n)^2)$.

Следствие:

Теорема

$\mathrm{PSPACE} = \mathrm{AltP}$.

Упражнение

Докажите, что функции n^k конструируемые по памяти.

Альт-время и обычная память

Теорема

$\mathrm{AltTIME}(T(n) \subseteq \mathrm{DSPACE}(T(n))$ при $T(n) \geqslant n$, $T(n)$ конструируемая по памяти.

Теорема

$\mathrm{DSPACE}(S(n)) \subseteq \mathrm{AltTIME}(S(n)^2)$.

Следствие:

Теорема

$\mathrm{PSPACE} = \mathrm{AltP}$.

Упражнение

Докажите, что функции n^k конструируемые по памяти.

Класс PSPACE и PSPACE-полные задачи

Базовый пример

Решение АМТ-игры, ATMgame

Даны АМТ M, граница времени T в унарном представлении (то есть слово из T единичек), слово w. Нужно узнать, принимает ли M за время T входное слово w.

Здесь предполагается, что если какое-то вычисление длится дольше T тактов, оно обрывается в отвергающем состоянии.

Утверждение

Решение АМТ-игры является PSPACE-полной задачей.

Базовый пример

Решение АМТ-игры, ATMgame

Даны АМТ M, граница времени T в унарном представлении (то есть слово из T единичек), слово w. Нужно узнать, принимает ли M за время T входное слово w.

Здесь предполагается, что если какое-то вычисление длится дольше T тактов, оно обрывается в отвергающем состоянии.

Утверждение

Решение АМТ-игры является PSPACE-полной задачей.

ATMgame: PSPACE-трудность

$L \in \mathrm{PSPACE}$

Обычная МТ M распознаёт L на памяти $S(n) = \mathrm{poly}(n)$

Из теоремы о превращении памяти в альтернирующее время: существует АМТ M', которая распознаёт L за альтернирующее время $T = O(S(n)^2) = \mathrm{poly}(n)$.

Искомая сводимость $L \leq_p$ ATMgame:

$$w \mapsto (M', T, w).$$

ATMgame: PSPACE-трудность

$L \in \mathrm{PSPACE}$

Обычная МТ M распознаёт L на памяти $S(n) = \mathrm{poly}(n)$

Из теоремы о превращении памяти в альтернирующее время: существует АМТ M', которая распознаёт L за альтернирующее время $T = O(S(n)^2) = \mathrm{poly}(n)$.

Искомая сводимость $L \leq_p$ ATMgame:

$$w \mapsto (M', T, w).$$

ATMgame: PSPACE-трудность

$L \in \mathrm{PSPACE}$

Обычная МТ M распознаёт L на памяти $S(n) = \mathrm{poly}(n)$

Из теоремы о превращении памяти в альтернирующее время: существует АМТ M', которая распознаёт L за альтернирующее время $T = O(S(n)^2) = \mathrm{poly}(n)$.

Искомая сводимость $L \leq_p$ ATMgame:

$$w \mapsto (M', T, w).$$

ATMgame в PSPACE

ATMgame $\in$ PSPACE = AltP

АМТ, которая принимает ATMgame за альтернирующее полиномиальное время, на входе (M, T, w) моделирует работу M на входе w за время T обычным образом (универсальная машина Тьюринга; счётчик ходов, разделение состояний согласно M).

Важнейший пример

TQBF

Нужно вычислить значение формулы

$$Q_1 Q_2 \dots Q_n \varphi(x_1, \dots, x_n),$$

где φ — булева формула от переменных x_1, …, x_n, а $Q_1 Q_2 \dots Q_n$ — последовательность кванторов, $Q_i \in \{\forall, \exists\}$.

Игра на формуле

Игроки последовательно присваивают значения булевым переменным x_1, …, x_n: игрок 0 присваивает значения переменным x_i, для которых $Q_i = \forall$; игрок 1 — переменным x_i, для которых $Q_i = \exists$.

Игра заканчивается, когда всем переменным присвоены значения. Критерий выигрыша игрока 1: $\varphi(x_1, \dots, x_n) = 1$.

Важнейший пример

TQBF

Нужно вычислить значение формулы

$$Q_1 Q_2 \dots Q_n \varphi(x_1, \dots, x_n),$$

где φ — булева формула от переменных x_1, $\dots$, x_n, а $Q_1 Q_2 \dots Q_n$ — последовательность кванторов, $Q_i \in \{\forall, \exists\}$.

Игра на формуле

Игроки последовательно присваивают значения булевым переменным x_1, $\dots$, x_n: игрок 0 присваивает значения переменным x_i, для которых $Q_i = \forall$; игрок 1 — переменным x_i, для которых $Q_i = \exists$.

Игра заканчивается, когда всем переменным присвоены значения. Критерий выигрыша игрока 1: $\varphi(x_1, \dots, x_n) = 1$.

Принадлежность PSPACE

Теорема

TQBF PSPACE-полна.

TQBF $\in$ PSPACE

Проще доказать равносильное TQBF $\in$ AltP.

Чтобы вычислить значение квантифицированной формулы, АМТ присваивает значения переменным по очереди в состояниях, соответствующих кванторам; затем происходит (детерминированное) вычисление значения формулы.

Принадлежность PSPACE

Теорема

TQBF PSPACE-полна.

TQBF $\in$ PSPACE

Проще доказать равносильное TQBF $\in$ AltP.

Чтобы вычислить значение квантифицированной формулы, АМТ присваивает значения переменным по очереди в состояниях, соответствующих кванторам; затем происходит (детерминированное) вычисление значения формулы.

ATMgame $\leq_p$ TQBF: предварительные шаги

Пусть $(M', 1^T, w)$ — вход задачи ATMgame.

По машине M' построим эквивалентную машину M, в которой $\exists$ и $\forall$ состояния чередуются на каждом такте работы.

Q' — состояния M'. Тогда $Q = Q' \cup \tilde{Q}'$ — состояния M'.
Помимо $q \in Q'$, в Q есть ленивый двойник $\tilde{q}$,
противоположного q класса ($\exists$ или $\forall$).

Отношение переходов δ машины M:

- каждый переход $(q_1, a_1, q_2, a_2, d) \in \delta$, в котором классы q_1 и q_2 разные, добавляется в δ без изменений;
- каждый переход $(q_1, a_1, q_2, a_2, d) \in \delta$, в котором классы q_1 и q_2 одинаковые, добавляет в δ два перехода

$$(q_1, a_1, \tilde{q}_2, a_2, d),\ (\tilde{q}_2, a_2, q_2, a_2, 0).$$

ATMgame $\leq_p$ TQBF: предварительные шаги

Пусть $(M', 1^T, w)$ — вход задачи ATMgame.

По машине M' построим эквивалентную машину M, в которой $\exists$ и $\forall$ состояния чередуются на каждом такте работы.

Q' — состояния M'. Тогда $Q = Q' \cup \tilde{Q}'$ — состояния M'.
Помимо $q \in Q'$, в Q есть ленивый двойник $\tilde{q}$,
противоположного q класса ($\exists$ или $\forall$).

Отношение переходов δ машины M:

- каждый переход $(q_1, a_1, q_2, a_2, d) \in \delta$, в котором классы q_1 и q_2 разные, добавляется в δ без изменений;
- каждый переход $(q_1, a_1, q_2, a_2, d) \in \delta$, в котором классы q_1 и q_2 одинаковые, добавляет в δ два перехода

$$(q_1, a_1, \tilde{q}_2, a_2, d),\ (\tilde{q}_2, a_2, q_2, a_2, 0).$$

ATMgame $\leq_p$ TQBF: предварительные шаги

Пусть $(M', 1^T, w)$ — вход задачи ATMgame.

По машине M' построим эквивалентную машину M, в которой $\exists$ и $\forall$ состояния чередуются на каждом такте работы.

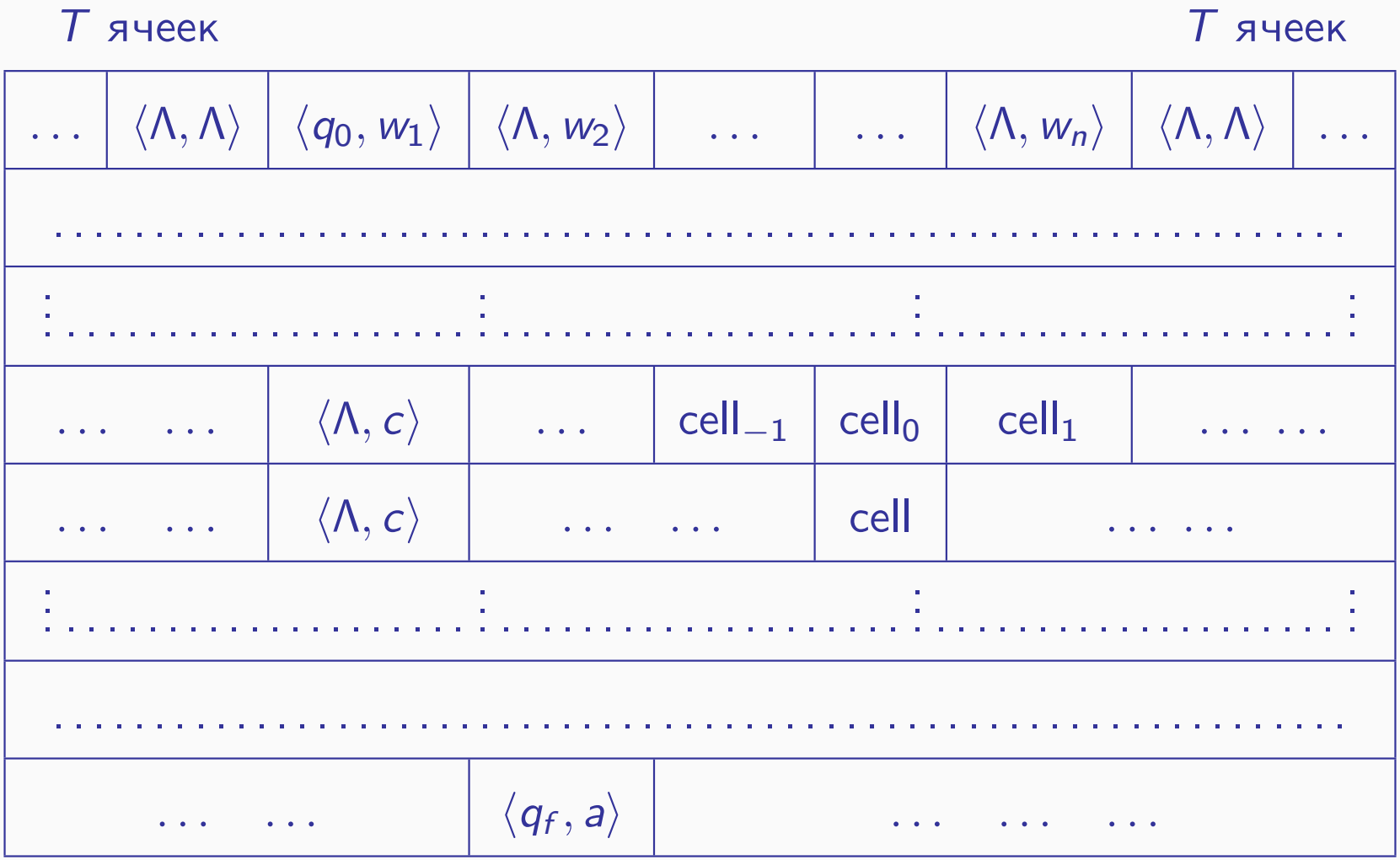
Q' — состояния M'. Тогда $Q = Q' \cup \tilde{Q}'$ — состояния M'.
Помимо $q \in Q'$, в Q есть ленивый двойник $\tilde{q}$,
противоположного q класса ($\exists$ или $\forall$).

Отношение переходов δ машины M:

- каждый переход $(q_1, a_1, q_2, a_2, d) \in \delta$, в котором классы q_1 и q_2 разные, добавляется в δ без изменений;
- каждый переход $(q_1, a_1, q_2, a_2, d) \in \delta$, в котором классы q_1 и q_2 одинаковые, добавляет в δ два перехода

$$(q_1, a_1, \tilde{q}_2, a_2, d),\ (\tilde{q}_2, a_2, q_2, a_2, 0).$$

Пространственно-временна́я диаграмма



Кодировка диаграммы булевыми переменными

Количество строк $T + 1$.

Количество столбцов $n + 2T$.

Длина кода ячейки $N = O(\log |Q| + \log |A|)$.

Всего нужно $(T + 1)(n + 2T)N$ битов.

Первые $(n + 2T)N$ битов константы.

Остальные $m = T(n + 2T)N$ битов: переменные x_i, $1 \leqslant i \leqslant m$.

Формула $\mathsf{Accept}(x_1, \ldots, x_m)$

$\mathsf{Accept}(x_1, \ldots, x_m) = 1 \quad \Leftrightarrow \quad$ диаграмма описывает корректное принимающее вычисление: все переходы согласованы с отношением переходов и последнее состояние принимающее.

Утверждение

Accept представляется конъюнкцией T формул, которые проверяют

- согласованность четвёрок $\mathrm{cell}_{-1}, \mathrm{cell}_0, \mathrm{cell}_1, \mathrm{cell}$ с отношением переходов, если над нижней ячейкой стоит головка (код вида $\langle q, a \rangle$);
- совпадение кодов в тех парах ячеек, которые соседствуют по вертикали и над которыми не стоит головка;

и формулы, проверяющей $q_f \in Q_f$.

Формула $\mathsf{Accept}(x_1, \ldots, x_m)$

$\mathsf{Accept}(x_1, \ldots, x_m) = 1 \quad \Leftrightarrow \quad$ диаграмма описывает корректное принимающее вычисление: все переходы согласованы с отношением переходов и последнее состояние принимающее.

Утверждение

Accept представляется конъюнкцией T формул, которые проверяют

- согласованность четвёрок cell_{-1}, cell_0, cell_1, cell с отношением переходов, если над нижней ячейкой стоит головка (код вида $\langle q, a \rangle$);
- совпадение кодов в тех парах ячеек, которые соседствуют по вертикали и над которыми не стоит головка;

и формулы, проверяющей $q_f \in Q_y$.

Формула $\mathrm{Accept}(x_1, \ldots, x_m)$

$\mathrm{Accept}(x_1, \ldots, x_m) = 1 \quad \Leftrightarrow \quad$ диаграмма описывает корректное принимающее вычисление: все переходы согласованы с отношением переходов и последнее состояние принимающее.

Утверждение

Accept представляется конъюнкцией T формул, которые проверяют

- согласованность четвёрок cell_{-1}, cell_0, cell_1, cell с отношением переходов, если над нижней ячейкой стоит головка (код вида $\langle q, a \rangle$);
- совпадение кодов в тех парах ячеек, которые соседствуют по вертикали и над которыми не стоит головка;

и формулы, проверяющей $q_f \in Q_y$.

Формула $\mathsf{Accept}(x_1, \ldots, x_m)$

$\mathsf{Accept}(x_1, \ldots, x_m) = 1 \quad \Leftrightarrow \quad$ диаграмма описывает корректное принимающее вычисление: все переходы согласованы с отношением переходов и последнее состояние принимающее.

Утверждение

Accept представляется конъюнкцией T формул, которые проверяют

- согласованность четвёрок cell_{-1}, cell_0, cell_1, cell с отношением переходов, если над нижней ячейкой стоит головка (код вида $\langle q, a \rangle$);
- совпадение кодов в тех парах ячеек, которые соседствуют по вертикали и над которыми не стоит головка;

и формулы, проверяющей $q_f \in Q_y$.

Формула $\mathsf{Accept}(x_1, \ldots, x_m)$

$\mathsf{Accept}(x_1, \ldots, x_m) = 1 \quad \Leftrightarrow \quad$ диаграмма описывает корректное принимающее вычисление: все переходы согласованы с отношением переходов и последнее состояние принимающее.

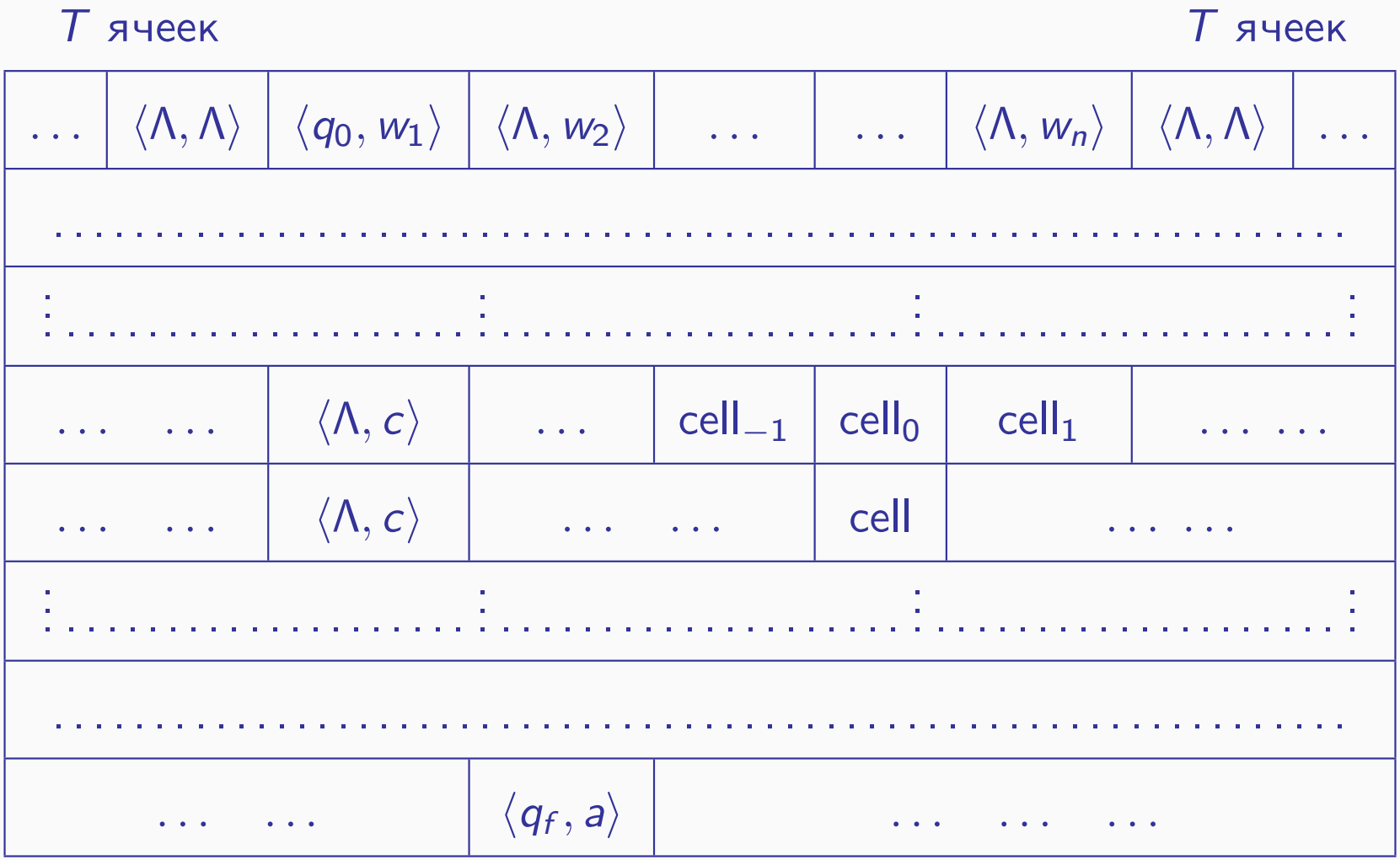
Утверждение

Accept представляется конъюнкцией T формул, которые проверяют

- согласованность четвёрок cell_{-1}, cell_0, cell_1, cell с отношением переходов, если над нижней ячейкой стоит головка (код вида $\langle q, a \rangle$);
- совпадение кодов в тех парах ячеек, которые соседствуют по вертикали и над которыми не стоит головка;

и формулы, проверяющей $q_f \in Q_y$.

Посмотрим ещё раз на диаграмму



Сводимость

$(M', 1^T, w) \mapsto \mathcal{Q}_1 x_1 \ldots \mathcal{Q}_m x_m$ Accept$(x_1, \ldots, x_m)$. Кванторы чередуются по строкам, начиная с квантора, отвечающего q_0.

Оценка размера формулы

Всего в конъюнкцию входит $2T + 1 = O(\text{size})$ формул, каждая зависит от $\leqslant 4N$ переменных,
$N = O(\log |Q| + \log |A|) = O(\log \text{size})$, где size — размер входа задачи ATMgame. Размер каждого члена конъюнкции не больше $\text{poly}(N)2^{4N} = \text{poly}(\text{size})$.

Утверждение

Сводящая функция вычисляется за полиномиальное время.

Сводимость

$(M', 1^T, w) \mapsto \mathcal{Q}_1 x_1 \ldots \mathcal{Q}_m x_m$ Accept$(x_1, \ldots, x_m)$. Кванторы чередуются по строкам, начиная с квантора, отвечающего q_0.

Оценка размера формулы

Всего в конъюнкцию входит $2T + 1 = O(\mathrm{size})$ формул, каждая зависит от $\leqslant 4N$ переменных,
$N = O(\log |Q| + \log |A|) = O(\log \mathrm{size})$, где size — размер входа задачи ATMgame. Размер каждого члена конъюнкции не больше $\mathrm{poly}(N)2^{4N} = \mathrm{poly}(\mathrm{size})$.

Утверждение

Сводящая функция вычисляется за полиномиальное время.

Сводимость

$(M', 1^T, w) \mapsto \mathcal{Q}_1 x_1 \ldots \mathcal{Q}_m x_m$ Accept$(x_1, \ldots, x_m)$. Кванторы чередуются по строкам, начиная с квантора, отвечающего q_0.

Оценка размера формулы

Всего в конъюнкцию входит $2T + 1 = O(\mathrm{size})$ формул, каждая зависит от $\leqslant 4N$ переменных,
$N = O(\log |Q| + \log |A|) = O(\log \mathrm{size})$, где size — размер входа задачи ATMgame. Размер каждого члена конъюнкции не больше $\mathrm{poly}(N)2^{4N} = \mathrm{poly}(\mathrm{size})$.

Утверждение

Сводящая функция вычисляется за полиномиальное время.

Частный случай

TQCNF

Формула под кванторами — КНФ (конъюнкция дизъюнкций литералов, то есть переменных или отрицаний переменных).

Утверждение

TQCNF PSPACE-полна.

Идея доказательства

Для любой булевой формулы есть эквивалентная вида

$$\varphi(x) = \exists y C(x, y),$$

где $C(x, y)$ — КНФ, а количество переменных y равно размеру формулы φ плюс 1 (размер формулы — количество пропозициональных связок в ней).

Частный случай

TQCNF

Формула под кванторами — КНФ (конъюнкция дизъюнкций литералов, то есть переменных или отрицаний переменных).

Утверждение

TQCNF PSPACE-полна.

Идея доказательства

Для любой булевой формулы есть эквивалентная вида

$$\varphi(x) = \exists y C(x, y),$$

где $C(x, y)$ — КНФ, а количество переменных y равно размеру формулы φ плюс 1 (размер формулы — количество пропозициональных связок в ней).

Частный случай

TQCNF

Формула под кванторами — КНФ (конъюнкция дизъюнкций литералов, то есть переменных или отрицаний переменных).

Утверждение

TQCNF PSPACE-полна.

Идея доказательства

Для любой булевой формулы есть эквивалентная вида

$$\varphi(x) = \exists y C(x, y),$$

где $C(x, y)$ — КНФ, а количество переменных y равно размеру формулы φ плюс 1 (размер формулы — количество пропозициональных связок в ней).

PSPACE-полные игры: «география»

Игра «география»

Задана конечным ориентированным графом G и начальной вершиной v_0. В начале игры в v_0 помещается фишка. Игроки по очереди двигают фишку в соседнюю по графу вершину. Вершины не должны повторяться. Кто не может сделать ход, тот проиграл.

Утверждение

«География» PSPACE-полна.

Этот пример хорошо изложен в учебной литературе. См., например, учебник Сипсера или многие другие.

PSPACE-полные игры: «география»

Игра «география»

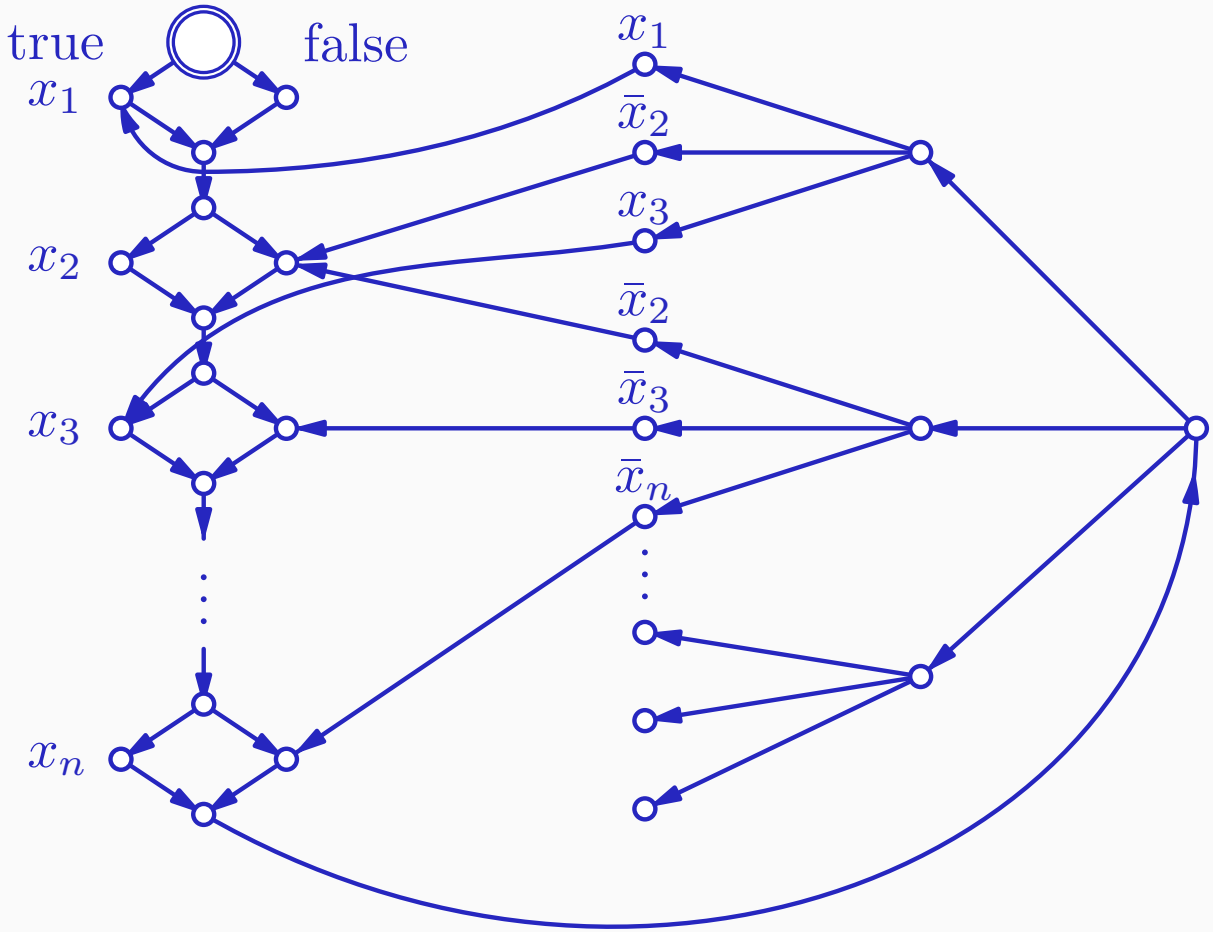
Задана конечным ориентированным графом G и начальной вершиной v_0. В начале игры в v_0 помещается фишка. Игроки по очереди двигают фишку в соседнюю по графу вершину. Вершины не должны повторяться. Кто не может сделать ход, тот проиграл.

Утверждение

«География» PSPACE-полна.

Этот пример хорошо изложен в учебной литературе. См., например, учебник Сипсера или многие другие.

TQCNF $\leq_p$ «география» в одной картинке



$$\varphi(x_1, \ldots, x_n) = \exists x_1 \forall x_2 \ldots \exists x_n \mathrm{CNF}(x_1, \ldots, x_n)$$

Вершинные кегли, NODE KAYLES

Правила игры

Игра задана конечным неориентированным графом G. Игроки по очереди закрашивают вершины. На каждом ходе игрок должен выбрать незакрашенную вершину, у которой все соседи также не закрашены, и покрасить. Кто не может сделать ход, тот проиграл.

По-другому: игроки строят независимое множество в графе, добавляя на каждом ходе по одной вершине. Тот, кому не удаётся расширить независимое множество, проиграл.

По-третьему: в вершинах графа стоят кегли. Игрок может выбить кегли в вершине и всех её соседях. Требуется на каждом ходе выбивать хотя бы одну кеглю.

Вершинные кегли, NODE KAYLES

Правила игры

Игра задана конечным неориентированным графом G. Игроки по очереди закрашивают вершины. На каждом ходе игрок должен выбрать незакрашенную вершину, у которой все соседи также не закрашены, и покрасить. Кто не может сделать ход, тот проиграл.

По-другому: игроки строят независимое множество в графе, добавляя на каждом ходе по одной вершине. Тот, кому не удаётся расширить независимое множество, проиграл.

По-третьему: в вершинах графа стоят кегли. Игрок может выбить кегли в вершине и всех её соседях. Требуется на каждом ходе выбивать хотя бы одну кеглю.

Вершинные кегли, NODE KAYLES

Правила игры

Игра задана конечным неориентированным графом G. Игроки по очереди закрашивают вершины. На каждом ходе игрок должен выбрать незакрашенную вершину, у которой все соседи также не закрашены, и покрасить. Кто не может сделать ход, тот проиграл.

По-другому: игроки строят независимое множество в графе, добавляя на каждом ходе по одной вершине. Тот, кому не удаётся расширить независимое множество, проиграл.

По-третьему: в вершинах графа стоят кегли. Игрок может выбить кегли в вершине и всех её соседях. Требуется на каждом ходе выбивать хотя бы одну кеглю.

Решение игры «вершинные кегли» PSPACE-полно

Принадлежность классу AltP почти очевидна.

TQCNF $\leq_p$ «вершинные кегли»: первый шаг — приведение формулы к удобному виду

$$\Phi = \exists x_n \forall x_{n-1} \ldots \exists x_1 \bigwedge_{j=1}^{m} D_j,$$

D_j — дизъюнкты (дизъюнкции литералов), $D_1 = x_1 \vee \neg x_1$.

Легко добиться, добавляя при необходимости фиктивную переменную. От D_1 значение формулы не зависит.

Решение игры «вершинные кегли» PSPACE-полно

Принадлежность классу AltP почти очевидна.

TQCNF $\leq_p$ «вершинные кегли»: первый шаг — приведение формулы к удобному виду

$$\Phi = \exists x_n \forall x_{n-1} \ldots \exists x_1 \bigwedge_{j=1}^{m} D_j,$$

D_j — дизъюнкты (дизъюнкции литералов), $D_1 = x_1 \vee \neg x_1$.

Легко добиться, добавляя при необходимости фиктивную переменную. От D_1 значение формулы не зависит.

Построение сводимости $\Phi \mapsto G_\Phi$

Вершины G_Φ разбиты на блоки X_i, $0 \leqslant i \leqslant n$.

Блок $X_0 = \{d_1, \ldots, d_m\}$ (блок дизъюнктов).

Блок X_i, $1 \leqslant i \leqslant n$: пара вершин-литералов x_i, $\bar{x}_i$ и вспомогательные вершины $y_{i,0}, \ldots, y_{i,i-1}$.

Рёбра G_Φ:

1. каждый блок — клика;
2. вершина d_j соединена в точности с литералами, входящими в D_j;
3. вершина $y_{i,j}$ соединена со всеми вершинами из блоков X_k при $0 \leqslant k < i$, $k \neq j$, и только с ними.

Построение сводимости $\Phi \mapsto G_\Phi$

Вершины G_Φ разбиты на блоки X_i, $0 \leqslant i \leqslant n$.

Блок $X_0 = \{d_1, \ldots, d_m\}$ (блок дизъюнктов).

Блок X_i, $1 \leqslant i \leqslant n$: пара вершин-литералов x_i, $\bar{x}_i$ и вспомогательные вершины $y_{i,0}, \ldots, y_{i,i-1}$.

Рёбра G_Φ:

1. каждый блок — клика;
2. вершина d_j соединена в точности с литералами, входящими в D_j;
3. вершина $y_{i,j}$ соединена со всеми вершинами из блоков X_k при $0 \leqslant k < i$, $k \neq j$, и только с ними.

Построение сводимости $\Phi \mapsto G_\Phi$

Вершины G_Φ разбиты на блоки X_i, $0 \leqslant i \leqslant n$.

Блок $X_0 = \{d_1, \ldots, d_m\}$ (блок дизъюнктов).

Блок X_i, $1 \leqslant i \leqslant n$: пара вершин-литералов x_i, $\bar{x}_i$ и вспомогательные вершины $y_{i,0}, \ldots, y_{i,i-1}$.

Рёбра G_Φ:

1. каждый блок — клика;
2. вершина d_j соединена в точности с литералами, входящими в D_j;
3. вершина $y_{i,j}$ соединена со всеми вершинами из блоков X_k при $0 \leqslant k < i$, $k \neq j$, и только с ними.

Построение сводимости $\Phi \mapsto G_\Phi$

Вершины G_Φ разбиты на блоки X_i, $0 \leqslant i \leqslant n$.

Блок $X_0 = \{d_1, \ldots, d_m\}$ (блок дизъюнктов).

Блок X_i, $1 \leqslant i \leqslant n$: пара вершин-литералов x_i, $\bar{x}_i$ и вспомогательные вершины $y_{i,0}, \ldots, y_{i,i-1}$.

Рёбра G_Φ:

1. каждый блок — клика;
2. вершина d_j соединена в точности с литералами, входящими в D_j;
3. вершина $y_{i,j}$ соединена со всеми вершинами из блоков X_k при $0 \leqslant k < i$, $k \neq j$, и только с ними.

Построение сводимости $\Phi \mapsto G_\Phi$

Вершины G_Φ разбиты на блоки X_i, $0 \leqslant i \leqslant n$.

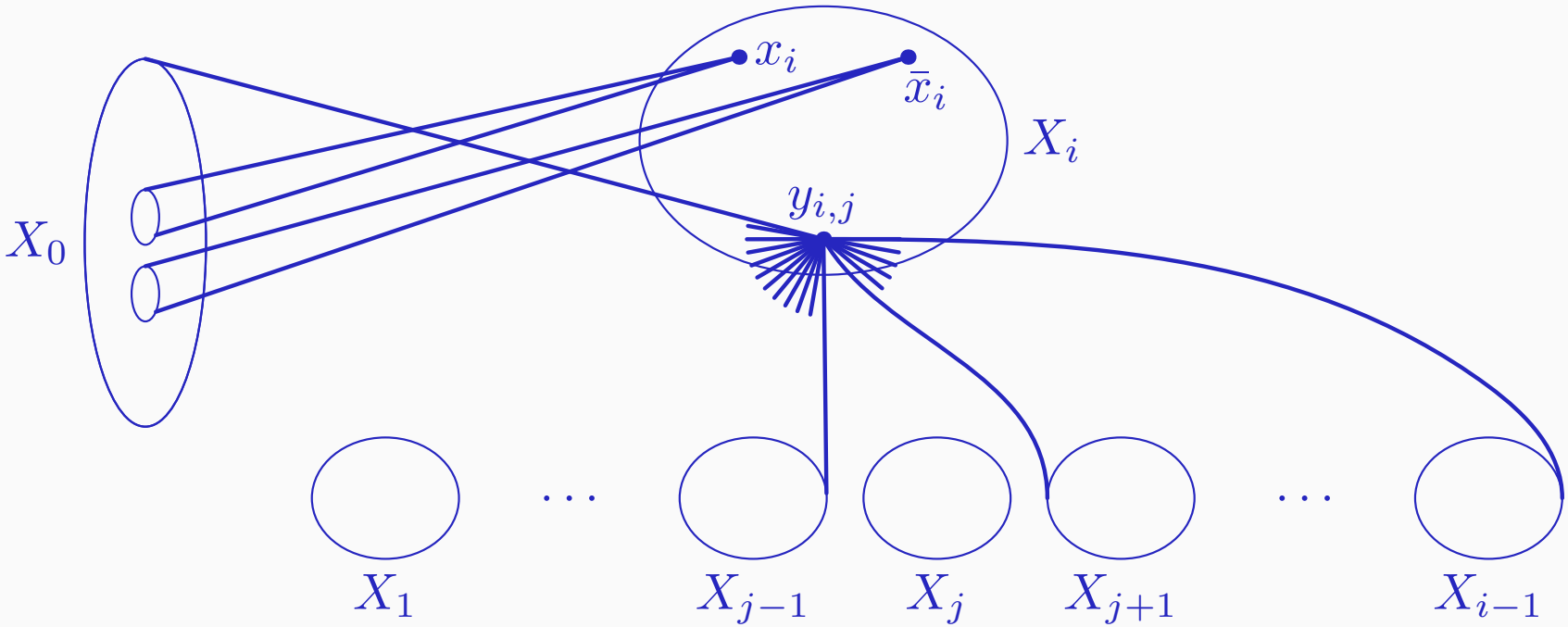
Блок $X_0 = \{d_1, \ldots, d_m\}$ (блок дизъюнктов).

Блок X_i, $1 \leqslant i \leqslant n$: пара вершин-литералов x_i, $\bar{x}_i$ и вспомогательные вершины $y_{i,0}, \ldots, y_{i,i-1}$.

Рёбра G_Φ:

1. каждый блок — клика;
2. вершина d_j соединена в точности с литералами, входящими в D_j;
3. вершина $y_{i,j}$ соединена со всеми вершинами из блоков X_k при $0 \leqslant k < i$, $k \neq j$, и только с ними.

Граф G_Φ на картинке



Наблюдения за игрой на графе G_Φ

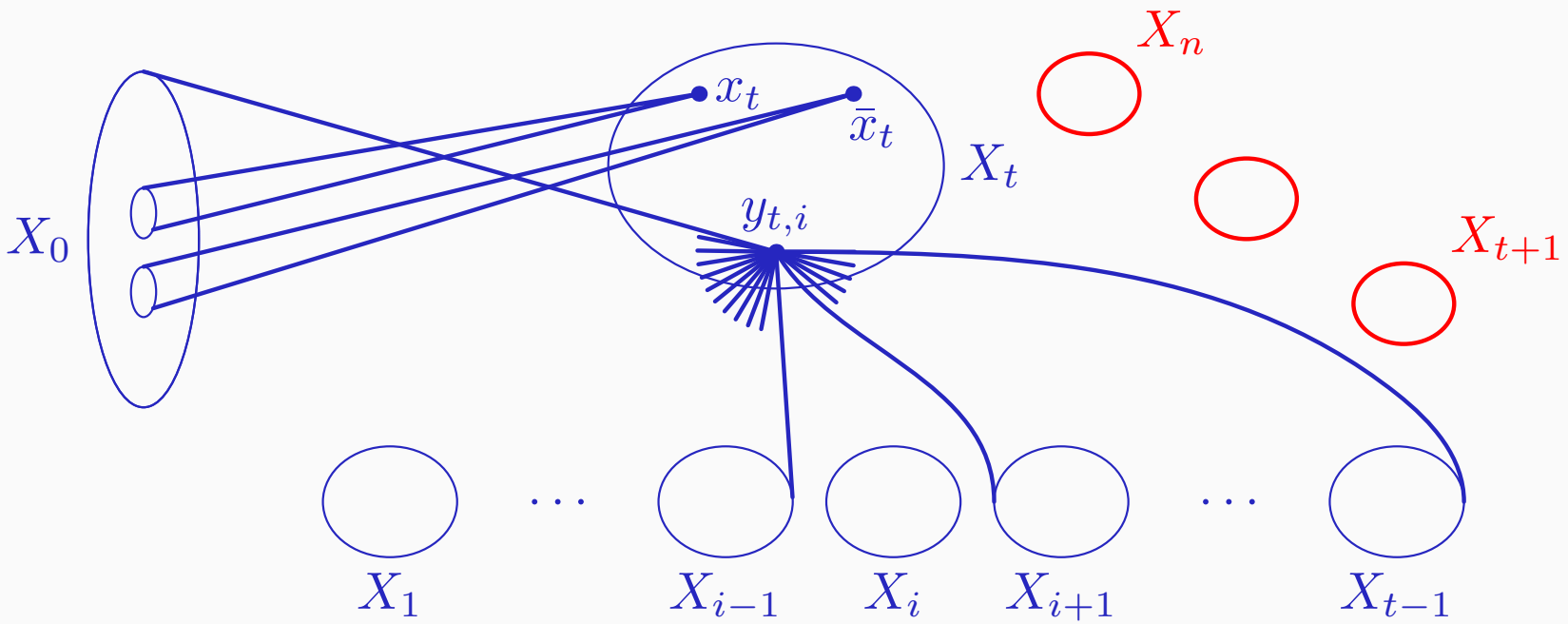
1. Блок — клика. Поэтому в каждой партии не более одного хода в каждый блок.
2. Партия правильная, если игроки закрашивают вершины-литералы из блоков X_n, X_{n-1} и так далее в нисходящем порядке. Отклонение от правильной партии карается проигрышем. Проверка разбором случаев.

Наблюдения за игрой на графе G_Φ

1. Блок — клика. Поэтому в каждой партии не более одного хода в каждый блок.
2. Партия правильная, если игроки закрашивают вершины-литералы из блоков X_n, X_{n-1} и так далее в нисходящем порядке. Отклонение от правильной партии карается проигрышем. Проверка разбором случаев.

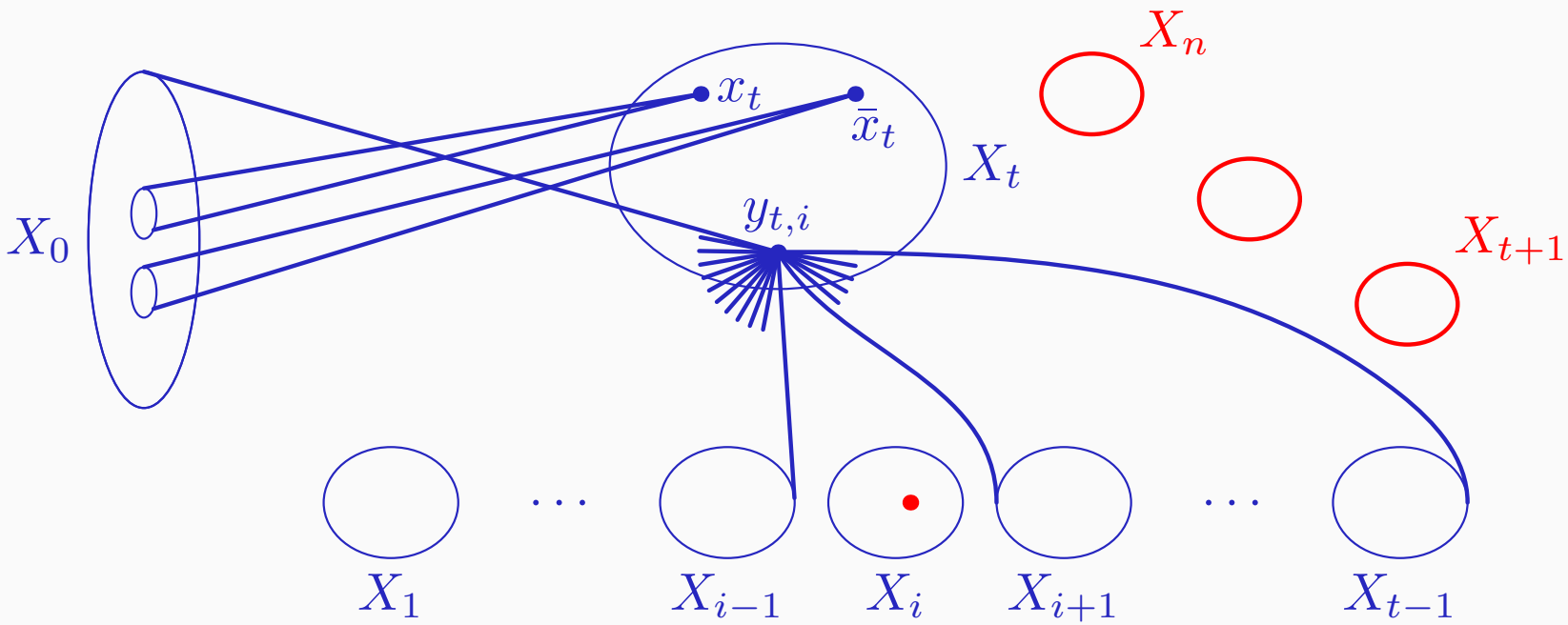
Ход в младший блок

Начало партии правильное, закрашены вершины-литералы из блоков $X_n, \ldots, X_{t+1}$. Затем игрок A закрасил вершину в блоке X_i, $i < t$. Вершина $y_{t,i}$ не выбита. Противник закрашивает $y_{t,i}$ и выбивает все оставшиеся вершины.



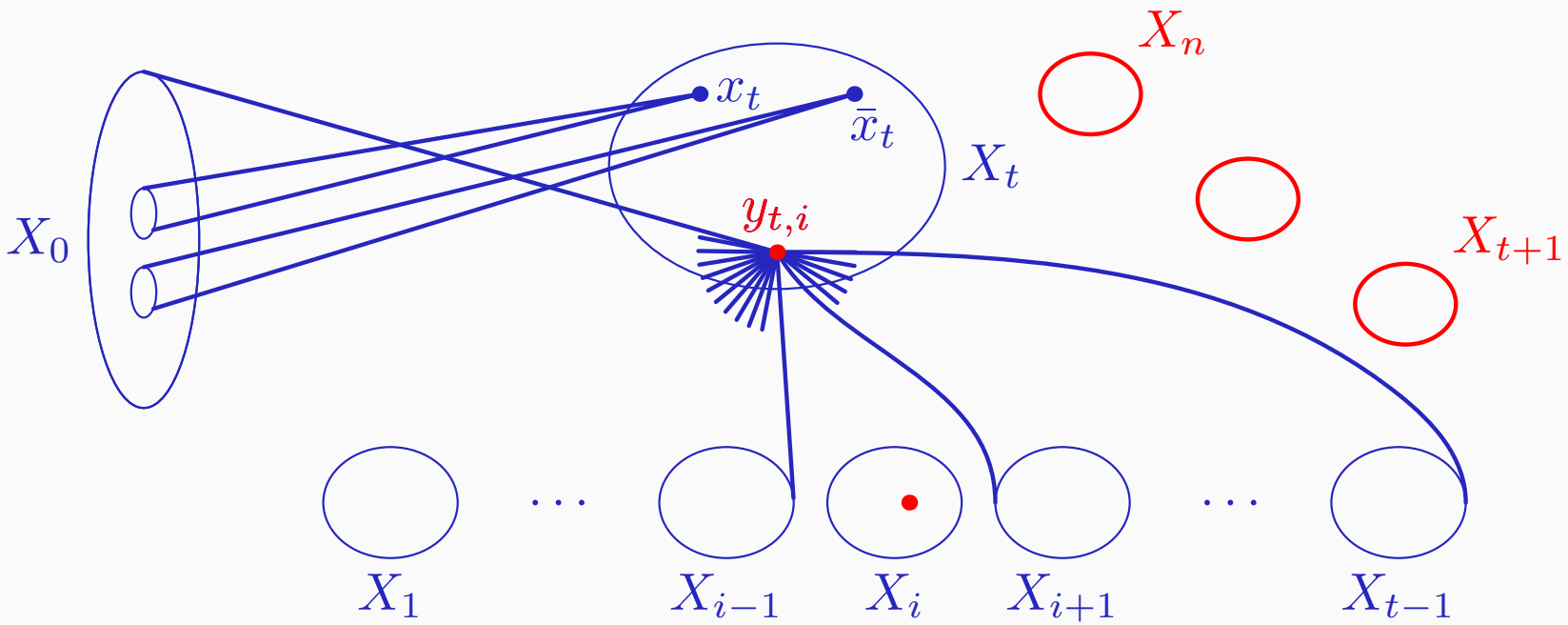
Ход в младший блок

Начало партии правильное, закрашены вершины-литералы из блоков $X_n, \ldots, X_{t+1}$. Затем игрок A закрасил вершину в блоке X_i, $i < t$. Вершина $y_{t,i}$ не выбита. Противник закрашивает $y_{t,i}$ и выбивает все оставшиеся вершины.



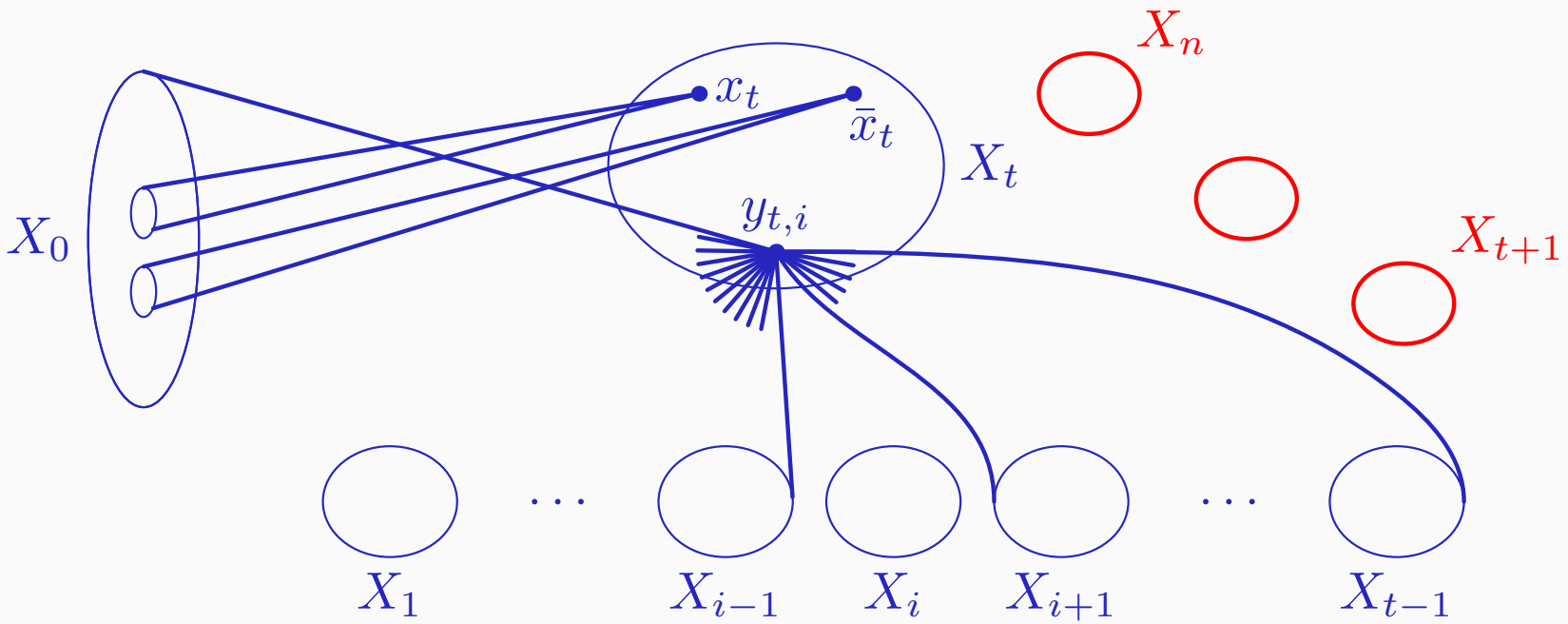
Ход в младший блок

Начало партии правильное, закрашены вершины-литералы из блоков $X_n, \ldots, X_{t+1}$. Затем игрок A закрасил вершину в блоке X_i, $i < t$. Вершина $y_{t,i}$ не выбита. Противник закрашивает $y_{t,i}$ и выбивает все оставшиеся вершины.



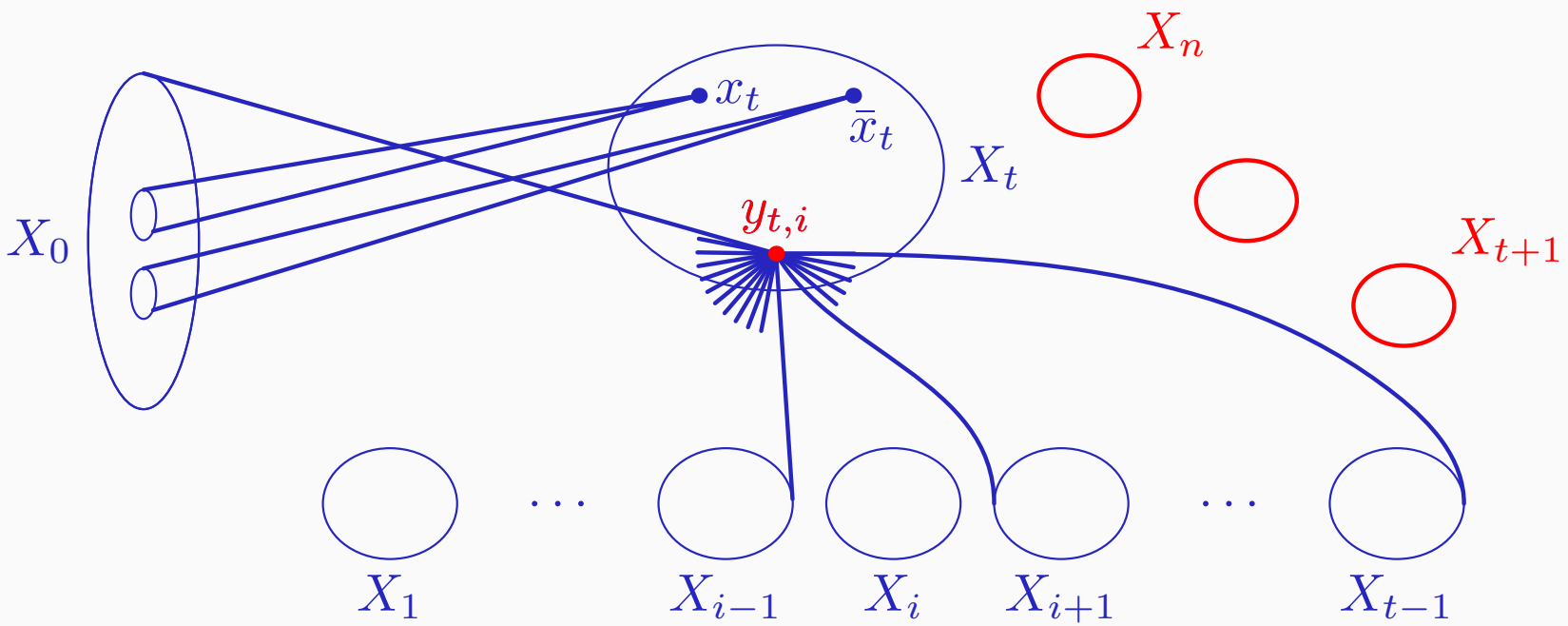
Ход во вспомогательную вершину (1)

Начало партии правильное, закрашены вершины-литералы из блоков $X_n, \ldots, X_{t+1}$. Затем игрок A закрасил вершину $y_{t,i}$, $0 < i < t$. Блок X_i не выбит. Противник красит вершину в X_i и выбивает все оставшиеся вершины.



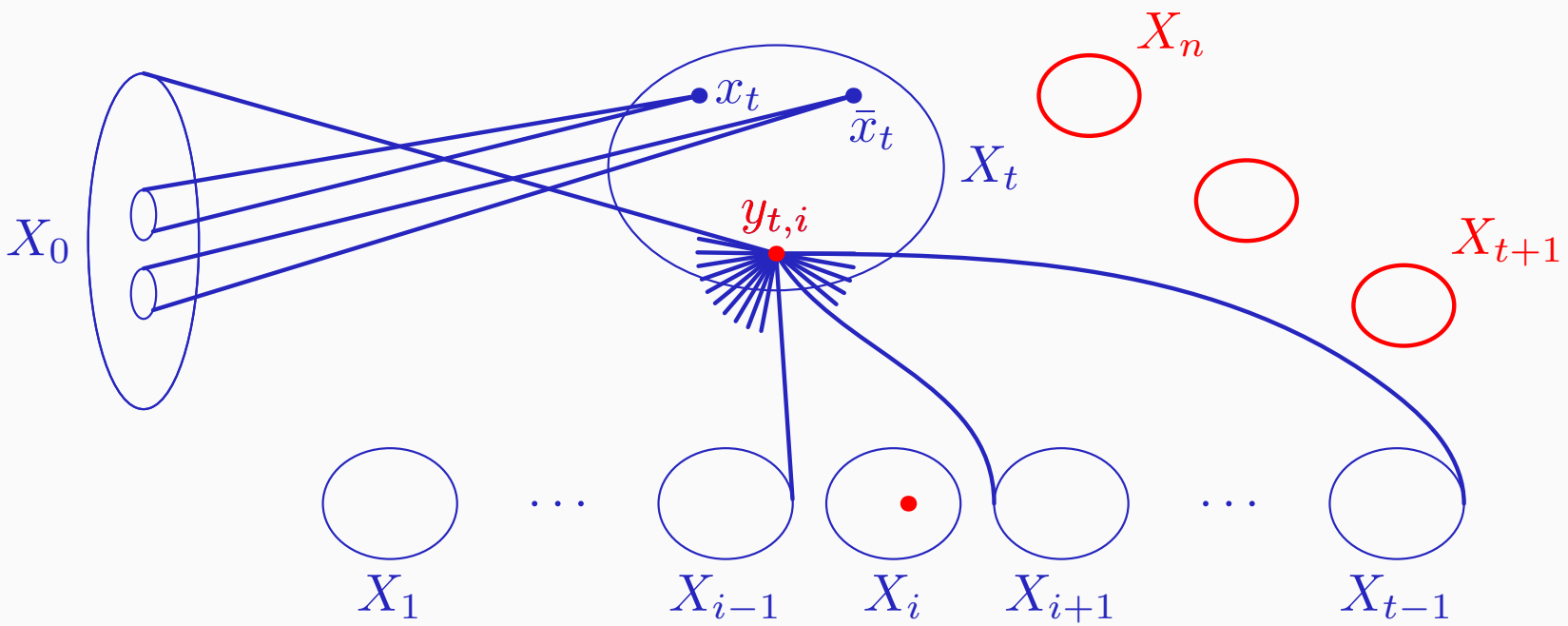
Ход во вспомогательную вершину (1)

Начало партии правильное, закрашены вершины-литералы из блоков $X_n, \ldots, X_{t+1}$. Затем игрок A закрасил вершину $y_{t,i}$, $0 < i < t$. Блок X_i не выбит. Противник красит вершину в X_i и выбивает все оставшиеся вершины.



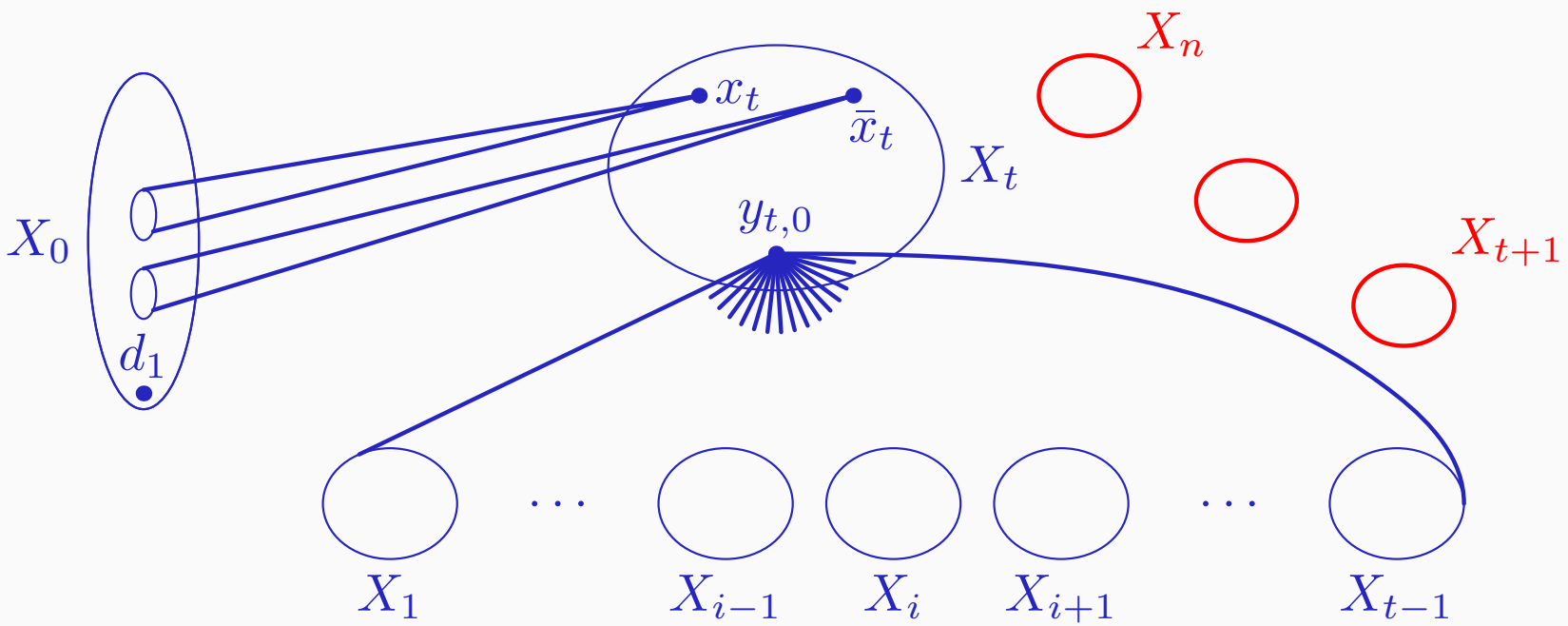
Ход во вспомогательную вершину (1)

Начало партии правильное, закрашены вершины-литералы из блоков $X_n, \ldots, X_{t+1}$. Затем игрок A закрасил вершину $y_{t,i}$, $0 < i < t$. Блок X_i не выбит. Противник красит вершину в X_i и выбивает все оставшиеся вершины.



Ход во вспомогательную вершину (2)

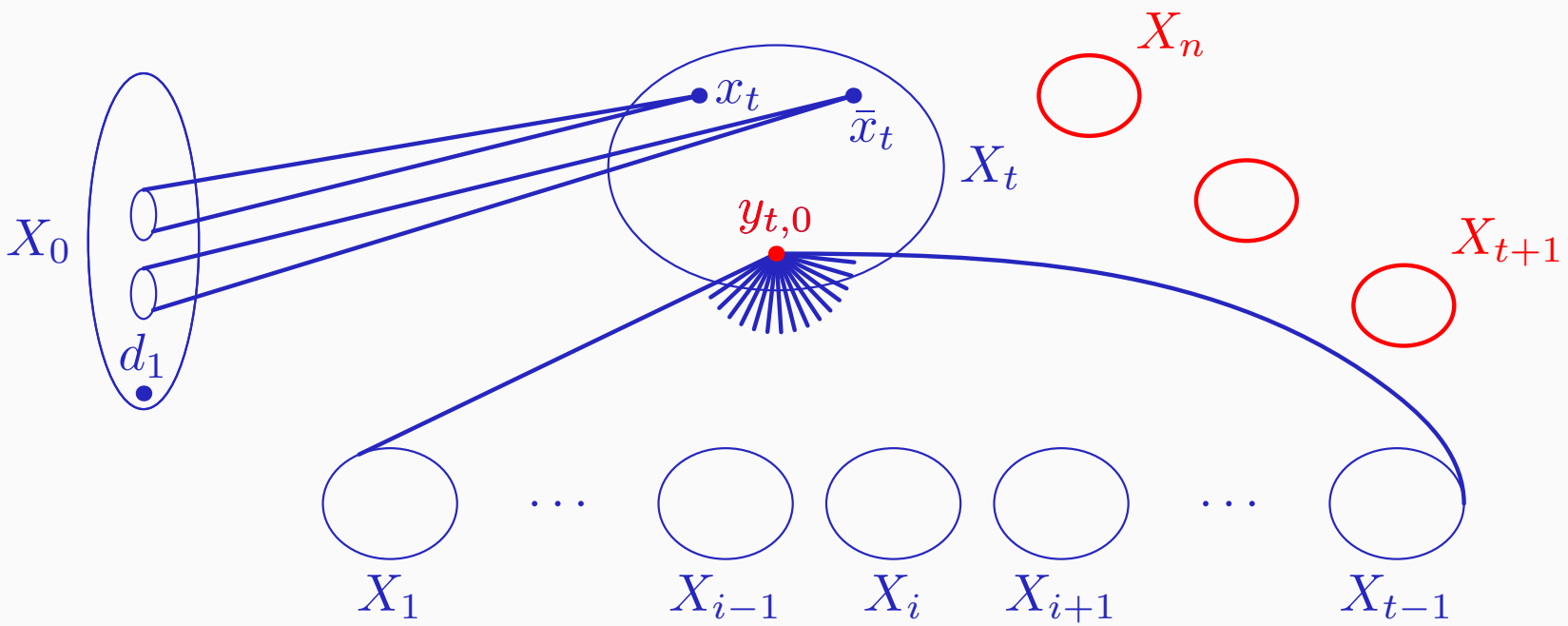
Начало партии правильное, закрашены вершины-литералы из блоков $X_n, \ldots, X_{t+1}$. Затем игрок A закрасил вершину $y_{t,0}$. В X_0 не выбита вершина d_1 (так как $D_1 = x_1 \vee \neg x_1$). Противник красит d_1 и выбивает все оставшиеся вершины.



Ход во вспомогательную вершину (2)

Начало партии правильное, закрашены вершины-литералы из блоков $X_n, \ldots, X_{t+1}$. Затем игрок A закрасил вершину $y_{t,0}$. В X_0 не выбита вершина d_1 (так как $D_1 = x_1 \vee \neg x_1$).

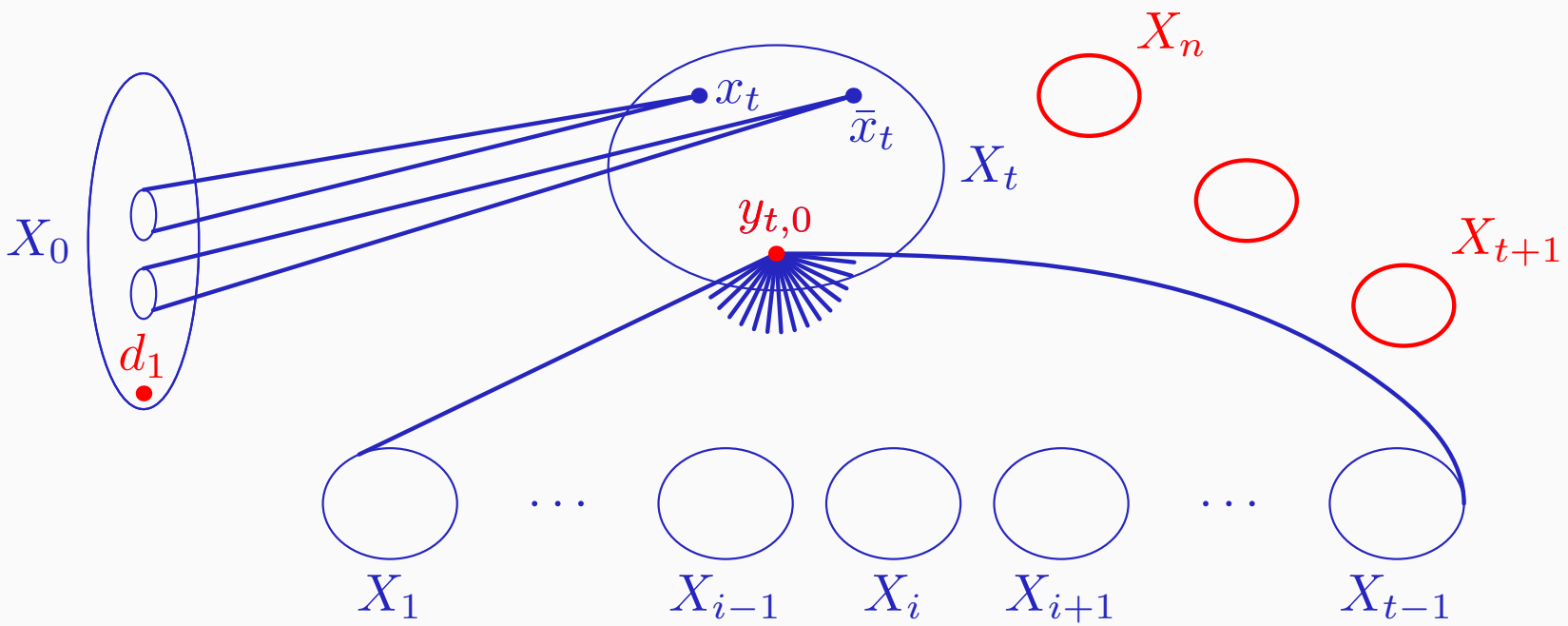
Противник красит d_1 и выбивает все оставшиеся вершины.



Ход во вспомогательную вершину (2)

Начало партии правильное, закрашены вершины-литералы из блоков $X_n, \ldots, X_{t+1}$. Затем игрок A закрасил вершину $y_{t,0}$. В X_0 не выбита вершина d_1 (так как $D_1 = x_1 \vee \neg x_1$).

Противник красит d_1 и выбивает все оставшиеся вершины.



Корректность сводимости с точки зрения игрока 1

Утверждение

Если у игрока 1 в игре на формуле Φ есть выигрывающая стратегия, то она есть и у игрока F, который ходит первым в вершинных кеглях.

Доказательство

Корректность сводимости с точки зрения игрока 1

Утверждение

Если у игрока 1 в игре на формуле Φ есть выигрывающая стратегия, то она есть и у игрока F, который ходит первым в вершинных кеглях.

Доказательство

F следует стратегии игрока 1, разыгрывая правильную партию. Если переменной x_t присваивается значение 1, то игрок F выбирает литерал x_t; в противном случае — литерал $\bar{x}_t$.

Корректность сводимости с точки зрения игрока 1

Утверждение

Если у игрока 1 в игре на формуле Φ есть выигрывающая стратегия, то она есть и у игрока F, который ходит первым в вершинных кеглях.

Доказательство

Если противник отклоняется от правильной партии, игрок F его наказывает и выигрывает.

Корректность сводимости с точки зрения игрока 1

Утверждение

Если у игрока 1 в игре на формуле Φ есть выигрывающая стратегия, то она есть и у игрока F, который ходит первым в вершинных кеглях.

Доказательство

Иначе игра продолжается, пока не будут выбиты все блоки $X_n, \ldots, X_1$. Последним ходит F (по предположению о формуле Φ).

Корректность сводимости с точки зрения игрока 1

Утверждение

Если у игрока 1 в игре на формуле Φ есть выигрывающая стратегия, то она есть и у игрока F, который ходит первым в вершинных кеглях.

Доказательство

Стратегия игрока 1 в игре на формуле выигрывающая. Поэтому при выбранных значениях переменных каждый дизъюнкт в КНФ обращается в 1. Но это означает, что в вершинных кеглях в блоке дизъюнктов выбиты все вершины. Противнику некуда ходить, игрок F выиграл.

Корректность сводимости с точки зрения игрока 0

Утверждение

Если у игрока 0 в игре на формуле Φ есть выигрывающая стратегия, то она есть и у игрока S, который ходит вторым в вершинных кеглях.

Доказательство

Корректность сводимости с точки зрения игрока 0

Утверждение

Если у игрока 0 в игре на формуле Φ есть выигрывающая стратегия, то она есть и у игрока S, который ходит вторым в вершинных кеглях.

Доказательство

S следует стратегии игрока 0, разыгрывая правильную партию.

Корректность сводимости с точки зрения игрока 0

Утверждение

Если у игрока 0 в игре на формуле Φ есть выигрывающая стратегия, то она есть и у игрока S, который ходит вторым в вершинных кеглях.

Доказательство

Если противник отклоняется от правильной партии, игрок S его наказывает и выигрывает.

Корректность сводимости с точки зрения игрока 0

Утверждение

Если у игрока 0 в игре на формуле Φ есть выигрывающая стратегия, то она есть и у игрока S, который ходит вторым в вершинных кеглях.

Доказательство

Иначе игра продолжается, пока не будут выбиты все блоки $X_n, \ldots, X_1$. Последним ходит F (по предположению о формуле Φ).

Корректность сводимости с точки зрения игрока 0

Утверждение

Если у игрока 0 в игре на формуле Φ есть выигрывающая стратегия, то она есть и у игрока S, который ходит вторым в вершинных кеглях.

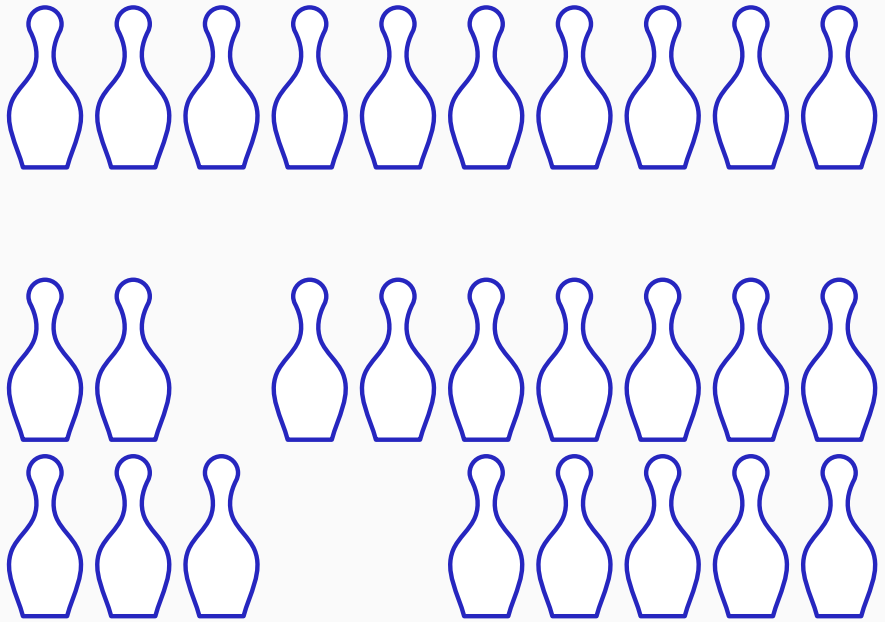
Доказательство

Поскольку игрок S следовал выигрывающей стратегии игрока 0 в игре на формуле, при выбранных значениях переменных хотя бы один дизъюнкт D_j обращается в 0. То есть, вершина d_j в блоке X_0 ещё не выбита. Игрок S закрашивает её и выигрывает.

Ускорение решения игр

Обычная игра KAYLES

Выставлен в ряд n кеглей. Игроки ходят по очереди. Игрок может выбить либо одну кеглю, либо две, но стоящие рядом. Если игрок не может сделать ход, он проиграл.



Симметричная стратегия

Утверждение

В игре KAYLES с любым положительным количеством кеглей выигрывающая стратегия есть у того игрока, который делает первый ход.

Доказательство

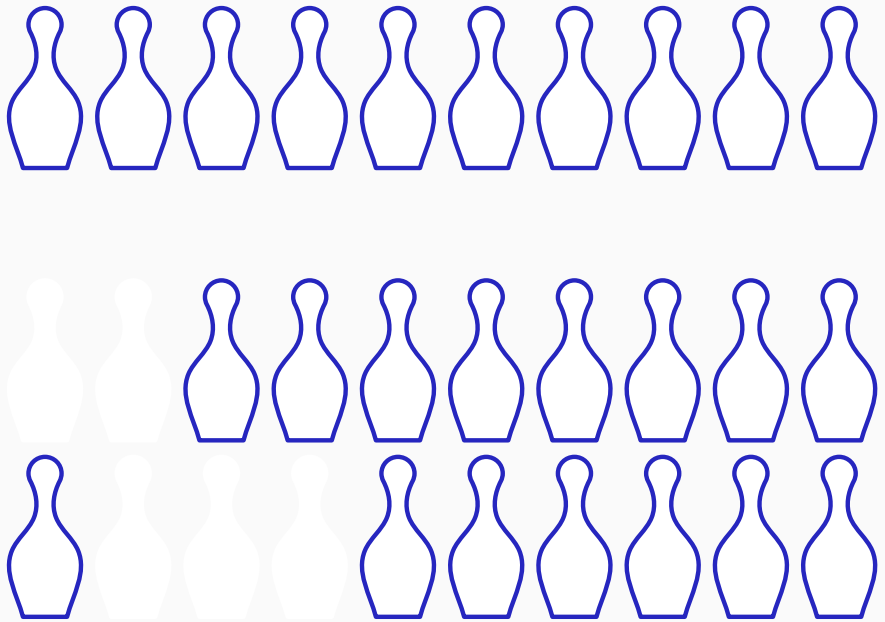
Своим первым ходом игрок делит кегли на две равные группы, в зависимости от чётности числа кеглей выбивая либо одну центральную кеглю, либо две.

Далее повторяет ход противника в другой группе (всегда возможно, если придерживаться симметричной стратегии).

Более сложный пример: NODE KAYLES на графе-пути

Пересказ с кеглями

Выставлен в ряд n кеглей. Игроки ходят по очереди. Игрок может выбить кеглю и её соседей. Если игрок не может сделать ход, он проиграл.



Симметричная стратегия для нечётного числа кеглей

Первый выбивает три центральные и дальше повторяет ходы противника.

Какой ответ для чётного числа кеглей?

Симметричная стратегия не работает.

При $n = 6$ выигрывающая стратегия у первого.

При $n = 8$ — у второго.

Симметричная стратегия для нечётного числа кеглей

Первый выбивает три центральные и дальше повторяет ходы противника.

Какой ответ для чётного числа кеглей?

Симметричная стратегия не работает.

При $n = 6$ выигрывающая стратегия у первого.

При $n = 8$ — у второго.

Беспристрастные игры

Ходы игроков чередуются и в каждой позиции множество возможных ходов одинаково для обоих игроков. Это позволяет уменьшить количество анализируемых позиций вдвое.

Классификация позиций в беспристрастной игре

N-позиция: выигрывающая стратегия у того, кто ходит (булево значение 1).

P-позиция: выигрывающая стратегия у того, кто не ходит (булево значение 0).

Правила оценки позиции в беспристрастной игре

Если есть ход в P-позицию, то первый выигрывает.

Если все ходы ведут в N-позиции, то второй выигрывает.

В булевых значениях

Если из позиции v есть ходы в позиции $u_1, \ldots, u_k$ с оценками $p_1, \ldots, p_k$, то оценка $p(v)$ позиции v вычисляется по формуле

$$p(v) = \neg \bigwedge_{i=1}^{k} p_i = \bigvee_{i=1}^{k} \neg p_i \stackrel{\text{def}}{=} [p_1, \ldots, p_k].$$

Правила оценки позиции в беспристрастной игре

Если есть ход в P-позицию, то первый выигрывает.

Если все ходы ведут в N-позиции, то второй выигрывает.

В булевых значениях

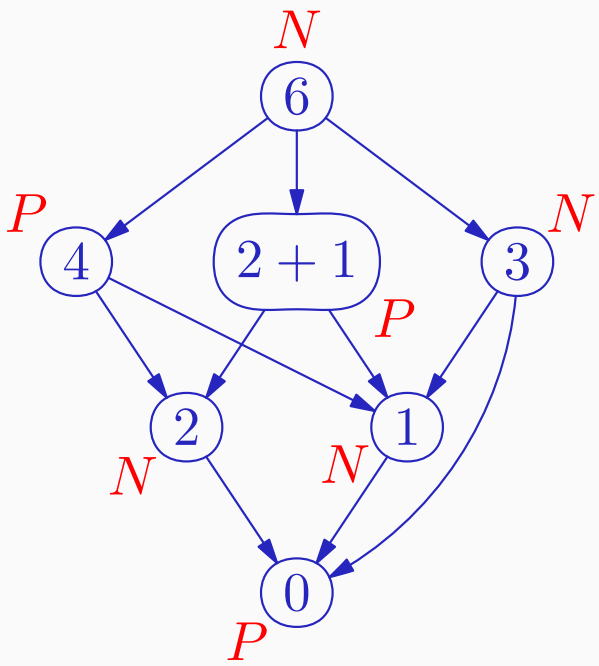
Если из позиции v есть ходы в позиции $u_1, \ldots, u_k$ с оценками $p_1, \ldots, p_k$, то оценка $p(v)$ позиции v вычисляется по формуле

$$p(v) = \neg \bigwedge_{i=1}^{k} p_i = \bigvee_{i=1}^{k} \neg p_i \stackrel{\text{def}}{=} [p_1, \ldots, p_k].$$

Анализ NODE KAYLES для P_6

Позиции задаются разбиениями числа кеглей. Для начальных 6 кеглей возможные позиции

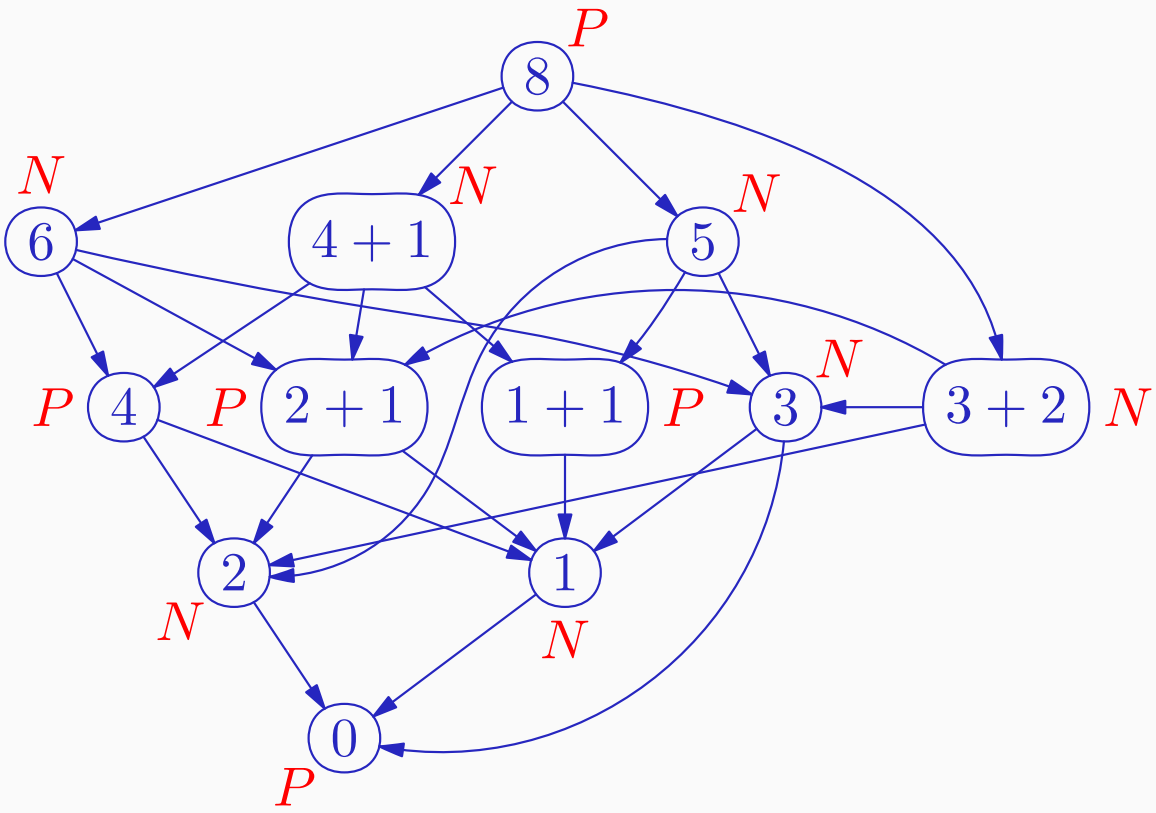
$$6,\ 4,\ 3,\ 2+1,\ 2,\ 1,\ 0.$$



Анализ NODE KAYLES для P_8

Возможные позиции

$$8,\ 6,\ 5,\ 4+1,\ 3+2,\ 4,\ 3,\ 2+1,\ 2,\ 1+1,\ 1,\ 0.$$



NODE KAYLES для P_n

Число позиций $e^{\Omega(\sqrt{n})}$.

Для $n = 2016$ есть шансы прямого разбора графа позиций на компьютере.

(И на таком пути можно узнать, что это N-позиция.)

Для $n = 34402$ шансов на анализ всего графа позиций нет.

Тем не менее, можно выяснить, что это P-позиция.

Помогает ним-функция (функция Шпрага–Гранди).

NODE KAYLES для P_n

Число позиций $e^{\Omega(\sqrt{n})}$.

Для $n = 2016$ есть шансы прямого разбора графа позиций на компьютере.

(И на таком пути можно узнать, что это N-позиция.)

Для $n = 34402$ шансов на анализ всего графа позиций нет.

Тем не менее, можно выяснить, что это P-позиция.

Помогает ним-функция (функция Шпрага–Гранди).

NODE KAYLES для P_n

Число позиций $e^{\Omega(\sqrt{n})}$.

Для $n = 2016$ есть шансы прямого разбора графа позиций на компьютере.

(И на таком пути можно узнать, что это N-позиция.)

Для $n = 34402$ шансов на анализ всего графа позиций нет.

Тем не менее, можно выяснить, что это P-позиция.

Помогает ним-функция (функция Шпрага–Гранди).