Алгоритмическая теория игр
Лекция 1

М.Н. Вялый

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

Игры на выигрыш на ациклическом графе

Игры 2 игроков с нулевой суммой, с полной информацией

Примеры: шашки, го.

1. Игроков всего два, обозначаем 0 и 1 (или по смыслу игры).
2. В каждой позиции делает ход один из игроков.
3. Указана начальная позиция, терминальные позиции.
4. В терминальных позициях указан результат (пока это указание на то, кто из игроков выиграл).
5. Пока требуем, чтобы каждая партия длилась конечное количество ходов.

Игры 2 игроков с нулевой суммой, с полной информацией

Примеры: шашки, го.

1. Игроков всего два, обозначаем 0 и 1 (или по смыслу игры).
2. В каждой позиции делает ход один из игроков.
3. Указана начальная позиция, терминальные позиции.
4. В терминальных позициях указан результат (пока это указание на то, кто из игроков выиграл).
5. Пока требуем, чтобы каждая партия длилась конечное количество ходов.

Игры и графы

Игровое поле (arena)

Ориентированный граф $G = (V, E)$ (граф игры); вершины обозначают позиции, рёбра — возможные ходы.

Разбиение позиций $V = V_0 \sqcup V_1$: в позициях из V_0 ход делает 0, в позициях из V_1 ходит 1. Начальная позиция v_0. Каждой терминальной позиции (вершина исходящей степени 0) приписан результат.

Партия игры

Последовательность вершин $v_0, v_1, \ldots$; $(v_i, v_{i+1}) \in E$.

В ациклическом конечном графе всегда заканчивается в терминальной позиции (победитель в партии указан в терминальной позиции).

Игры и графы

Игровое поле (arena)

Ориентированный граф $G = (V, E)$ (граф игры); вершины обозначают позиции, рёбра — возможные ходы.

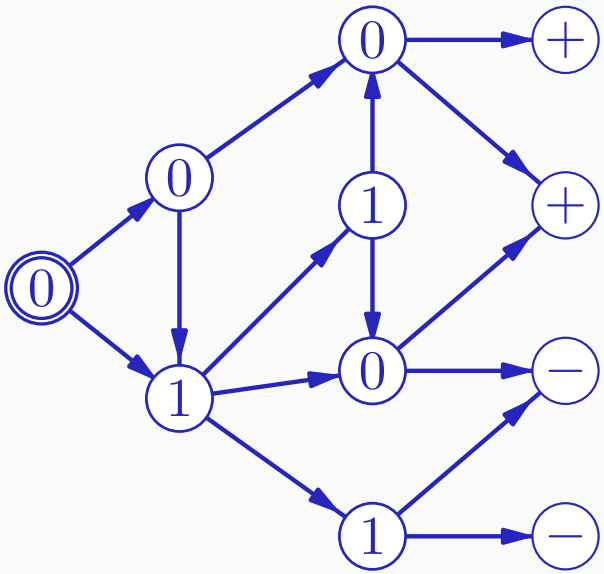
Разбиение позиций $V = V_0 \sqcup V_1$: в позициях из V_0 ход делает 0, в позициях из V_1 ходит 1. Начальная позиция v_0. Каждой терминальной позиции (вершина исходящей степени 0) приписан результат.

Партия игры

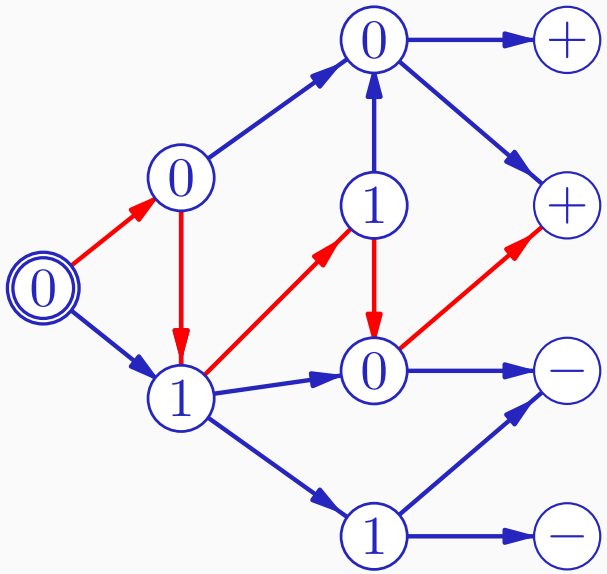
Последовательность вершин v_0, v_1, $\ldots$; $(v_i, v_{i+1}) \in E$.

В ациклическом конечном графе всегда заканчивается в терминальной позиции (победитель в партии указан в терминальной позиции).

Пример



Игровое поле



Партия игры

Стратегии

Определения

Стратегия — это правило, определяющее поведение игроков.

Стратегия s игрока i сопоставляет началу партии τ, ведущему в позицию $v \in V_i$, ход $(v, s(\tau, v)) \in E$.

Стратегия s_i называется выигрывающей для игрока i, если в каждой партии, в которой игрок i придерживается этой стратегии, он выигрывает (как бы ни играл противник).

Позиционная (memoryless) стратегия зависит только от текущей позиции, от предыстории не зависит.

Стратегии

Определения

Стратегия — это правило, определяющее поведение игроков.

Стратегия s игрока i сопоставляет началу партии τ, ведущему в позицию $v \in V_i$, ход $(v, s(\tau, v)) \in E$.

Стратегия s_i называется выигрывающей для игрока i, если в каждой партии, в которой игрок i придерживается этой стратегии, он выигрывает (как бы ни играл противник).

Позиционная (memoryless) стратегия зависит только от текущей позиции, от предыстории не зависит.

Стратегии

Определения

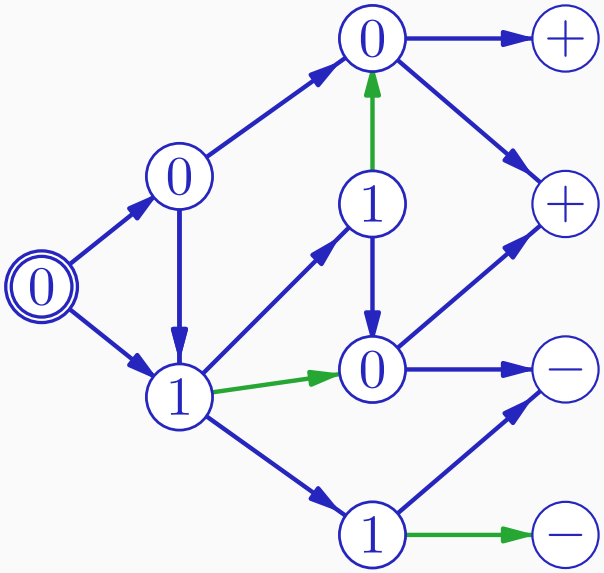
Стратегия — это правило, определяющее поведение игроков.

Стратегия s игрока i сопоставляет началу партии τ, ведущему в позицию $v \in V_i$, ход $(v, s(\tau, v)) \in E$.

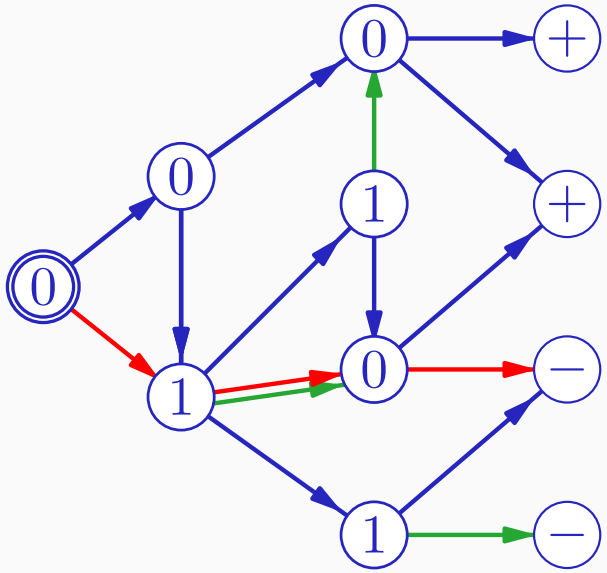
Стратегия s_i называется выигрывающей для игрока i, если в каждой партии, в которой игрок i придерживается этой стратегии, он выигрывает (как бы ни играл противник).

Позиционная (memoryless) стратегия зависит только от текущей позиции, от предыстории не зависит.

Примеры



Пример позиционной стратегии игрока 1



Партия, в которой игрок 1 придерживается этой стратегии

Вывод: эта стратегия не является выигрывающей.

Фундаментальная теорема

Теорема

Если граф игры ациклический, то у одного из игроков есть выигрывающая позиционная стратегия.

Метод доказательства: «разбор с конца», он же «обратная индукция». Основан на простом и хорошо известном факте:

Топологическая сортировка или линейное продолжение

Вершины ациклического графа можно занумеровать так, чтобы каждое ребро вело из вершины с меньшим номером в вершину с большим номером.

Фундаментальная теорема

Теорема

Если граф игры ациклический, то у одного из игроков есть выигрывающая позиционная стратегия.

Метод доказательства: «разбор с конца», он же «обратная индукция». Основан на простом и хорошо известном факте:

Топологическая сортировка или линейное продолжение

Вершины ациклического графа можно занумеровать так, чтобы каждое ребро вело из вершины с меньшим номером в вершину с большим номером.

Доказательство фундаментальной теоремы (начало)

Нумерация $v_1, \ldots, v_n$ согласована с направлением рёбер: если $(v_i, v_j) \in E$, то $i < j$.

Определим пару позиционных стратегий s_0, s_1 в каждой позиции графа игры и припишем каждой позиции результат игры w: 0 или 1.

Определять будем в убывающем порядке позиций и одновременно доказывать по индукции корректность определения.

Доказательство фундаментальной теоремы (окончание)

1. v_n необходимо терминальная; результат в ней известен, стратегии определять не нужно.
2. Пусть стратегии s_0, s_1 и результат w уже определены для позиций с номерами больше t.
   2.1. Если v_t терминальная, то см. выше.
   2.2. Для $v_t \in V_0$:
   2.3. Для $v_t \in V_1$ аналогично.

Доказательство фундаментальной теоремы (окончание)

1. v_n необходимо терминальная; результат в ней известен, стратегии определять не нужно.
2. Пусть стратегии s_0, s_1 и результат w уже определены для позиций с номерами больше t.

   2.1 Если v_t терминальная, то см. выше.

   2.2 Для $v_t \in V_0$:

   2.2.1 если есть ход $(v_t, v_j) \in E$, $w(v_j) = 0$, то $w(v_t) := 0$; $s_0(v_t) := v_j$ (при нескольких возможностях выбор произвольный);

   2.2.2 если все ходы ведут в позиции с $w(v_j) = 1$, то $w(v_t) := 1$; $s_0(v_t)$ произвольна.

   2.3 Для $v_t \in V_1$ аналогично.

Доказательство фундаментальной теоремы (окончание)

1. v_n необходимо терминальная; результат в ней известен, стратегии определять не нужно.
2. Пусть стратегии s_0, s_1 и результат w уже определены для позиций с номерами больше t.

   2.1 Если v_t терминальная, то см. выше.

   2.2 Для $v_t \in V_0$:

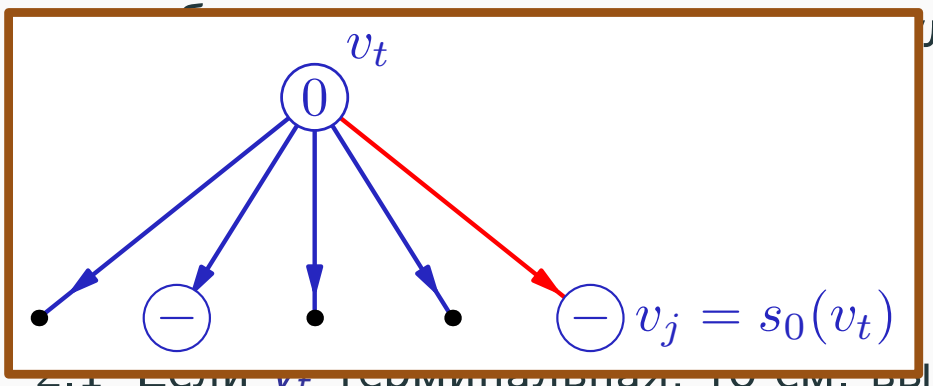
   2.2.1 если есть ход $(v_t, v_j) \in E$, $w(v_j) = 0$, то $w(v_t) := 0$; $s_0(v_t) := v_j$ (при нескольких возможностях выбор произвольный);

   2.2.2 если все ходы ведут в позиции с $w(v_j) = 1$, то $w(v_t) := 1$; $s_0(v_t)$ произвольна.

   2.3 Для $v_t \in V_1$ аналогично.

Доказательство фундаментальной теоремы (окончание)

1. ...льтат в ней известен,
2. ... w уже определены для



2.1 Если v_t терминальная, то см. выше.

2.2 Для $v_t \in V_0$:

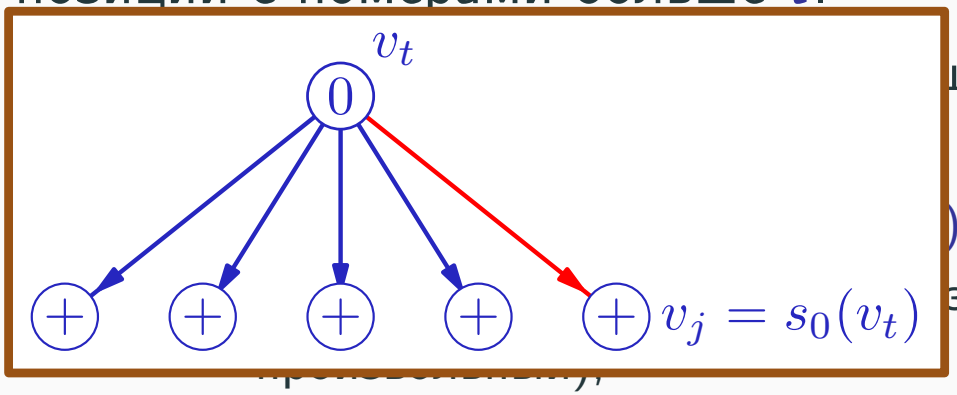
2.2.1 если есть ход $(v_t, v_j) \in E$, $w(v_j) = 0$, то $w(v_t) := 0$; $s_0(v_t) := v_j$ (при нескольких возможностях выбор произвольный);

2.2.2 если все ходы ведут в позиции с $w(v_j) = 1$, то $w(v_t) = 1$, $s_0(v_t)$ произвольна.

2.3 Для $v_t \in V_1$ аналогично.

Доказательство фундаментальной теоремы (окончание)

1. v_n необходимо терминальная; результат в ней известен, стратегии определять не нужно.
2. Пусть стратегии s_0, s_1 и результат w уже определены для позиций с номерами больше t.



ше.

$) = 0$, то $w(v_t) := 0$;

зможностях выбор

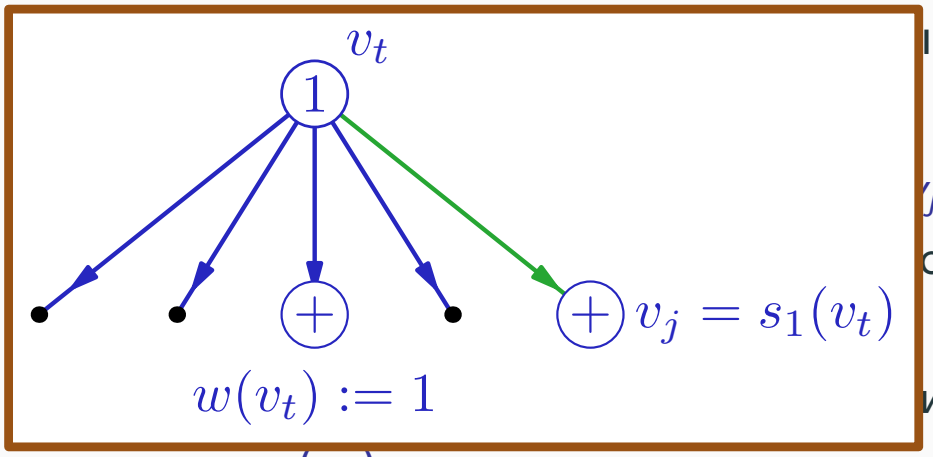
произвольный);

2.2.2 если все ходы ведут в позиции с $w(v_j) = 1$, то $w(v_t) = 1$, $s_0(v_t)$ произвольна.

2.3 Для $v_t \in V_1$ аналогично.

Доказательство фундаментальной теоремы (окончание)

1. v_n необходимо терминальная; результат в ней известен, стратегии определять не нужно.
2. Пусть стратегии s_0, s_1 и результат w уже определены для позиций с номерами больше t.



ше.

$_j) = 0$, то $w(v_t) := 0$;

озможностях выбор

и с $w(v_j) = 1$, то $w(v_t) = 1$,

$s_0(v_t)$ произвольна.

2.3 Для $v_t \in V_1$ аналогично.

Доказательство фундаментальной теоремы (окончание)

1. v_n необходимо терминальная; результат в ней известен, стратегии определять не нужно.
2. Пусть стратегии s_0, s_1 и результат w уже определены для позиций с номерами больше t.

   2.1 Если v_t

   2.2 Для $v_t \in$

   2.2.1 если := 0;

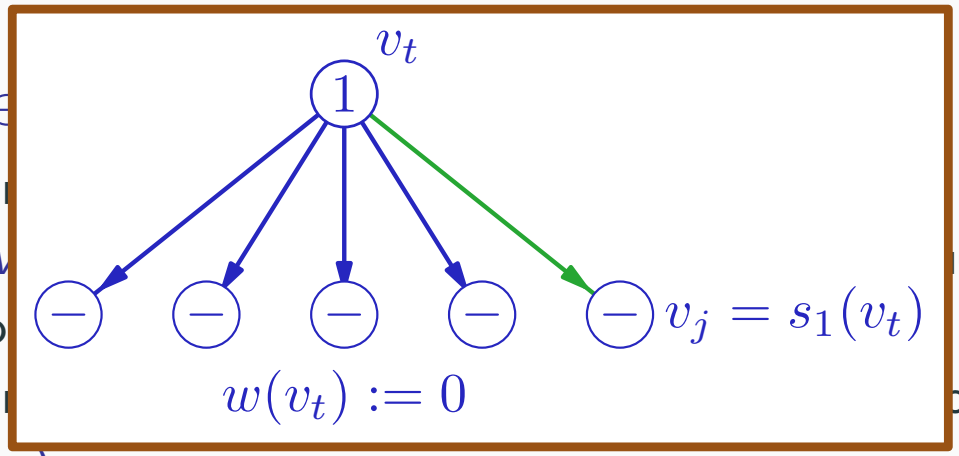
   $s_0(v$ бор

   про

   2.2.2 если о $w(v_t) = 1$,

   $s_0(v_t)$ произвольна.

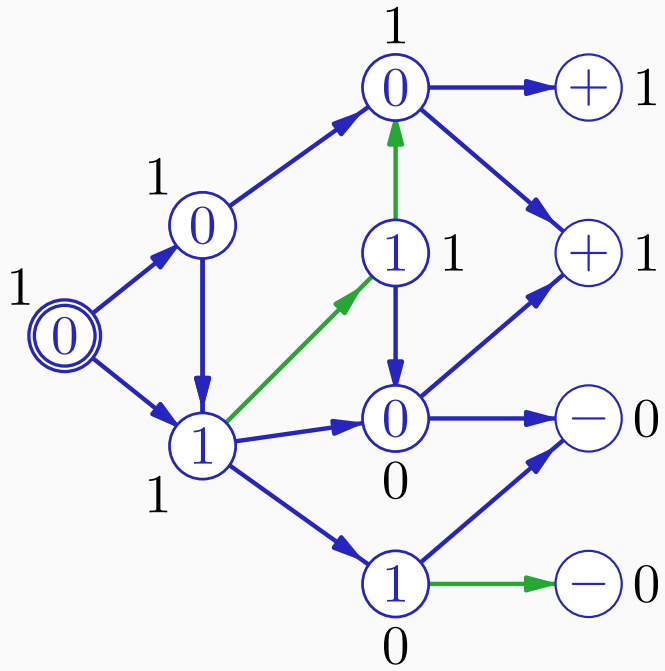
   2.3 Для $v_t \in V_1$ аналогично.



Доказательство фундаментальной теоремы (окончание)

1. v_n необходимо терминальная; результат в ней известен, стратегии определять не нужно.
2. Пусть стратегии s_0, s_1 и результат w уже определены для позиций с номерами больше t.
   - 2.1 Если v_t терминальная, то см. выше.
   - 2.2 Для $v_t \in V_0$:
     - 2.2.1 если есть ход $(v_t, v_j) \in E$, $w(v_j) = 0$, то $w(v_t) := 0$; $s_0(v_t) := v_j$ (при нескольких возможностях выбор произвольный);
     - 2.2.2 если все ходы ведут в позиции с $w(v_j) = 1$, то $w(v_t) = 1$, $s_0(v_t)$ произвольна.
   - 2.3 Для $v_t \in V_1$ аналогично.

Возвращение к примеру игры



Выигрывающая позиционная стратегия игрока 1 и результаты игры для всех позиций

Алгоритм решения игры

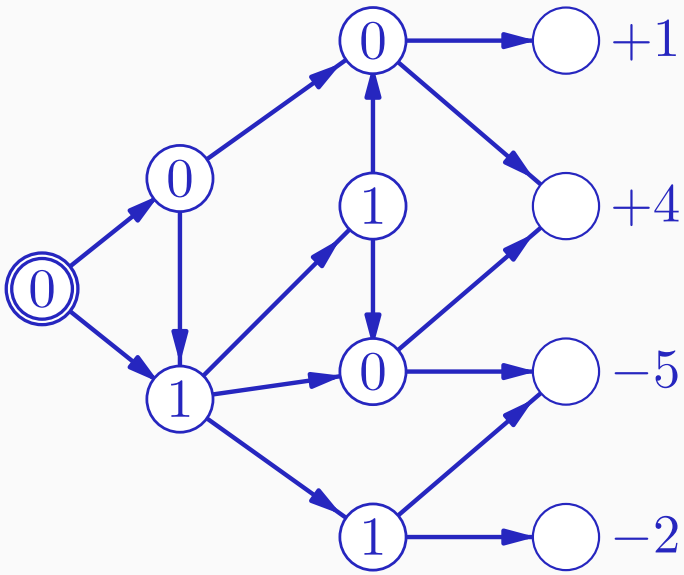
Под решением игры мы понимаем ответ на вопрос «у кого из игроков есть выигрывающая стратегия в игре»?

Доказательство теоремы даёт эффективный алгоритм решения игры. Под эффективным алгоритмом мы понимаем алгоритм, работающий за полиномиальное время от длины входа.
В сущности, единственный нетривиальный шаг состоит в топологической сортировке, все дальнейшие действия, описанные в доказательстве, требуют времени $O(|E|)$.
Полиномиальные алгоритмы топологической сортировки хорошо известны.

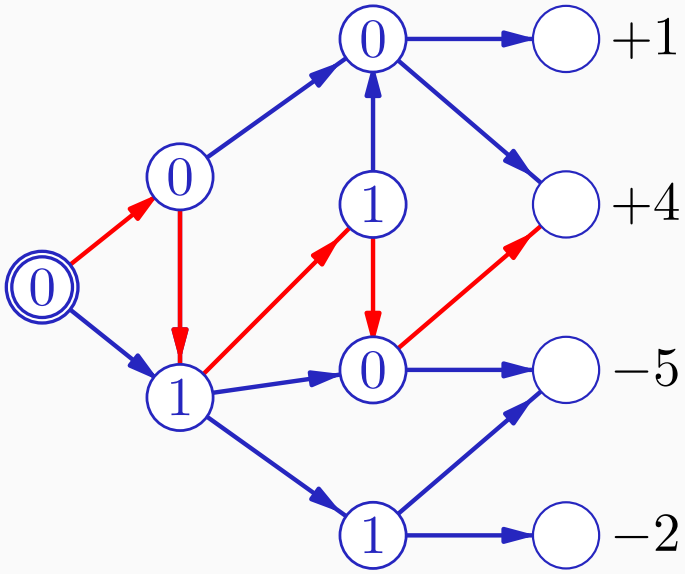
Игры на результат на ациклическом графе

Терминальные платежи

Результат — число (выигрыш игрока 1). Определяется терминальной позицией



Пример игрового поля игры на результат



Партия игры с результатом $+4$ (выигрыш игрока 1)

Гарантирующие стратегии

Определение

Стратегия s игрока 1 (Максимизатора) гарантирует результат a, если в любой партии, в которой 1 придерживается s, результат не меньше a.

Аналогично, стратегия s игрока 0 (Минимизатора) гарантирует a, если в любой партии, в которой 0 придерживается s, результат не больше a.

Цена игры

Число a называется ценой игры, если у обоих игроков есть стратегии, гарантирующие a.

Гарантирующие стратегии

Определение

Стратегия s игрока 1 (Максимизатора) гарантирует результат a, если в любой партии, в которой 1 придерживается s, результат не меньше a.

Аналогично, стратегия s игрока 0 (Минимизатора) гарантирует a, если в любой партии, в которой 0 придерживается s, результат не больше a.

Цена игры

Число a называется ценой игры, если у обоих игроков есть стратегии, гарантирующие a.

Существование цены для игр на ациклическом графе (1)

Теорема

Если граф игры ациклический и платежи терминальные, то в игре существует цена, а гарантирующие её стратегии можно выбрать позиционными.

Аналогично предыдущему случаю: разбор позиций с конца.

Различия. Цена в терминальной — результат в ней.

Цена $c(v)$ и стратегия в вершине $v \in V_0$:

$$c(v) = \min_{(vu)\in E} c(u), \quad s_0(v) = \arg\min_{(vu)\in E} c(u).$$

Цена в вершине $v \in V_1$:

$$c(v) = \max_{(vu)\in E} c(u) \quad s_1(v) = \arg\max_{(vu)\in E} c(u).$$

Существование цены для игр на ациклическом графе (1)

Теорема

Если граф игры ациклический и платежи терминальные, то в игре существует цена, а гарантирующие её стратегии можно выбрать позиционными.

Аналогично предыдущему случаю: разбор позиций с конца.

Различия. Цена в терминальной — результат в ней.

Цена $c(v)$ и стратегия в вершине $v \in V_0$:

$$c(v) = \min_{(vu)\in E} c(u), \quad s_0(v) = \arg\min_{(vu)\in E} c(u).$$

Цена в вершине $v \in V_1$:

$$c(v) = \max_{(vu)\in E} c(u) \quad s_1(v) = \arg\max_{(vu)\in E} c(u).$$

Существование цены для игр на ациклическом графе (1)

Теорема

Если граф игры ациклический и платежи терминальные, то в игре существует цена, а гарантирующие её стратегии можно выбрать позиционными.

Аналогично предыдущему случаю: разбор позиций с конца.

Различия. Цена в терминальной — результат в ней.

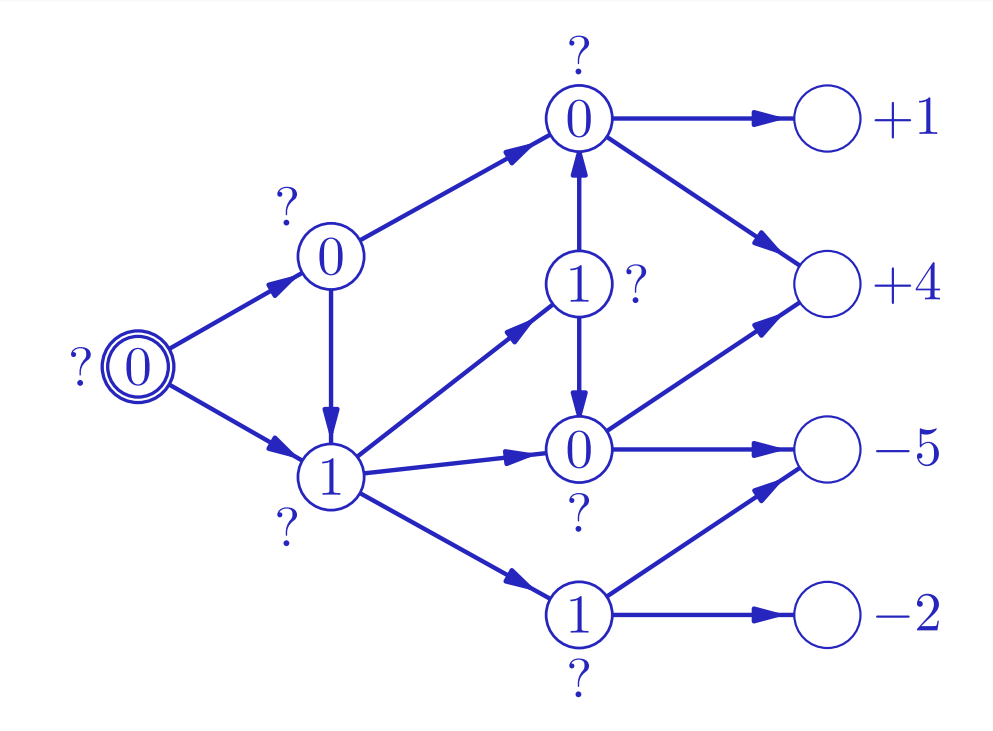
Цена $c(v)$ и стратегия в вершине $v \in V_0$:

$$c(v) = \min_{(vu)\in E} c(u), \quad s_0(v) = \arg\min_{(vu)\in E} c(u).$$

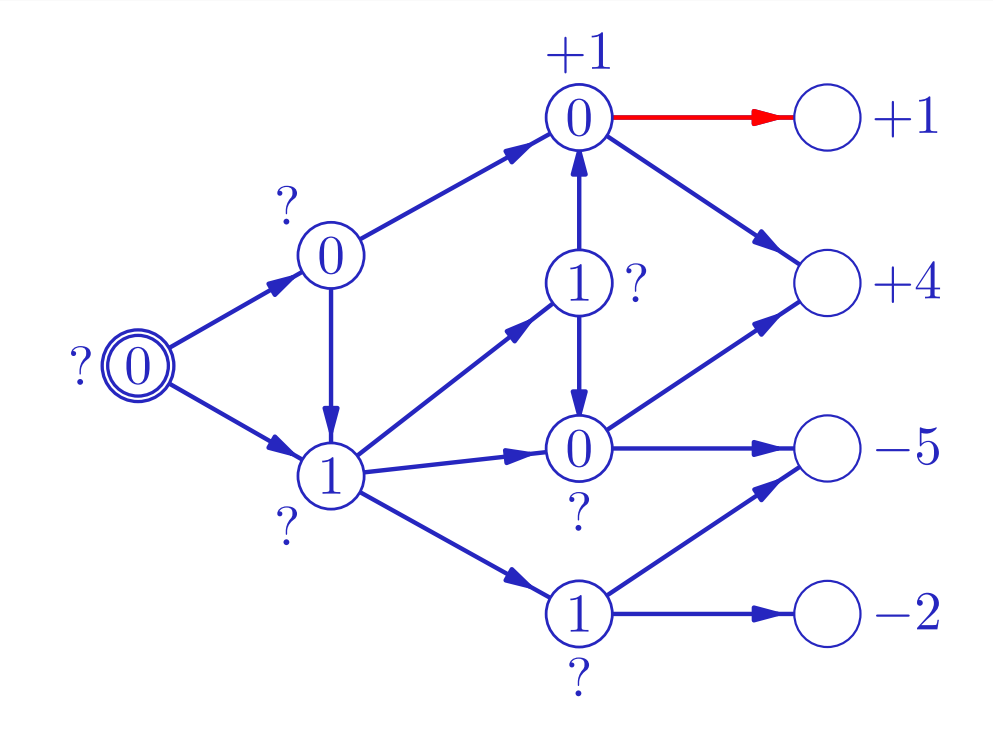
Цена в вершине $v \in V_1$:

$$c(v) = \max_{(vu)\in E} c(u) \quad s_1(v) = \arg\max_{(vu)\in E} c(u).$$

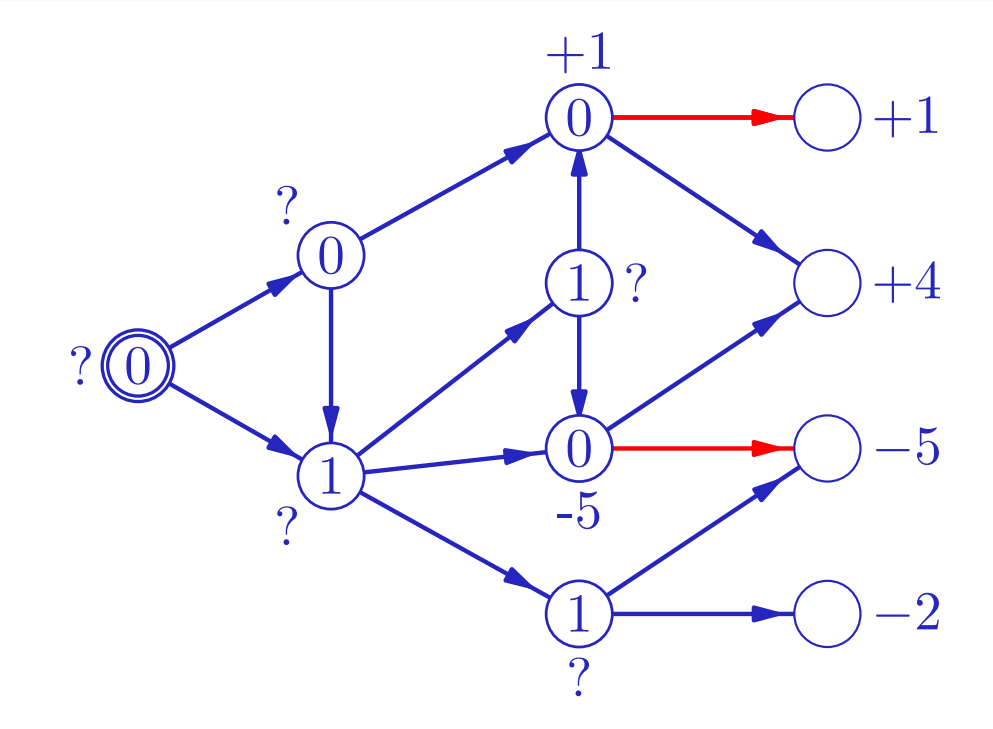
Пример анализа игры



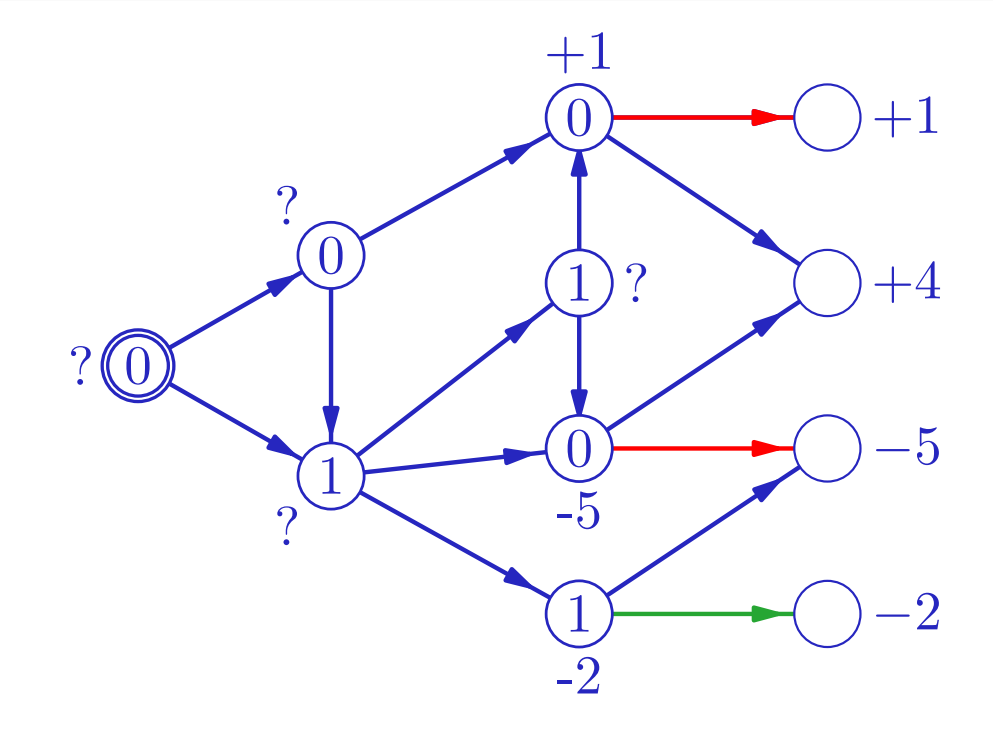
Пример анализа игры



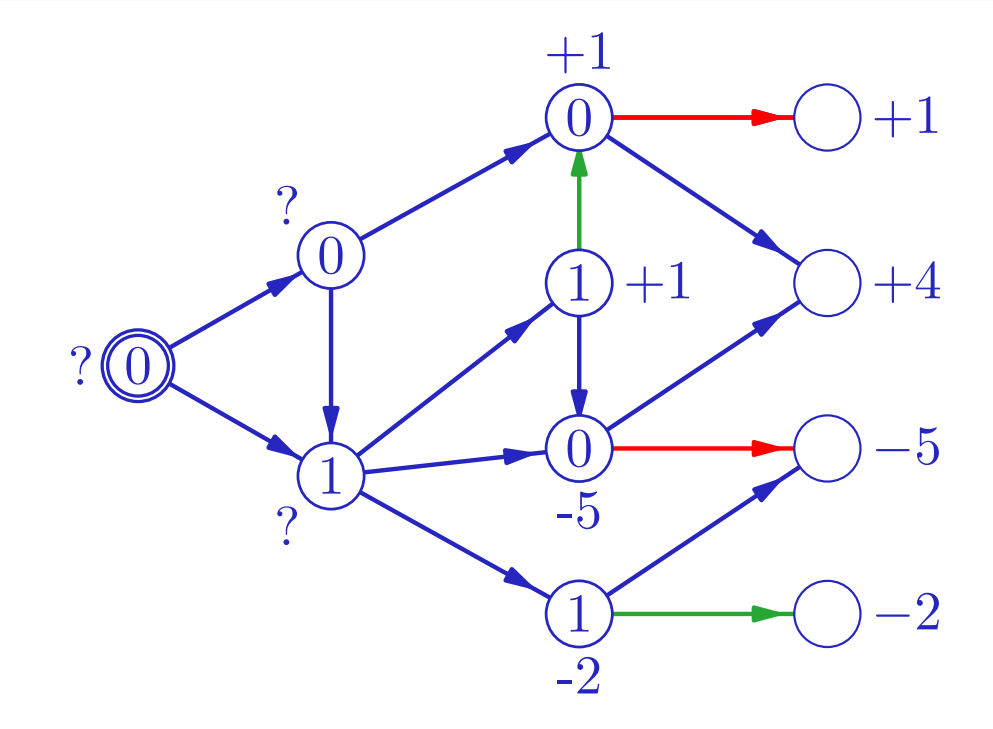
Пример анализа игры



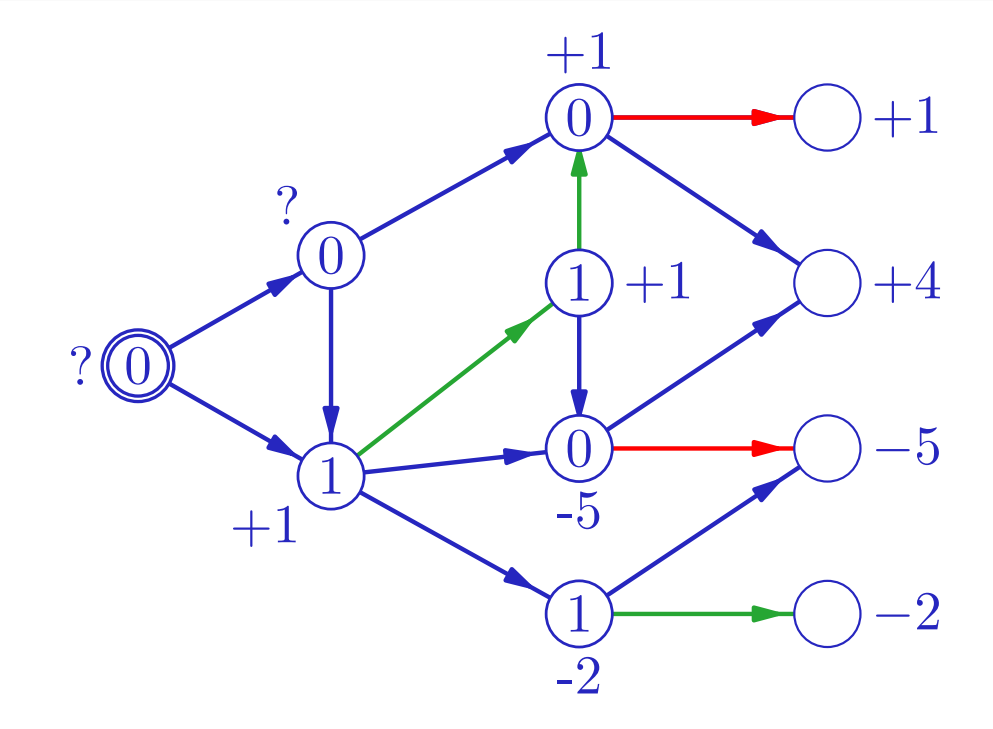
Пример анализа игры



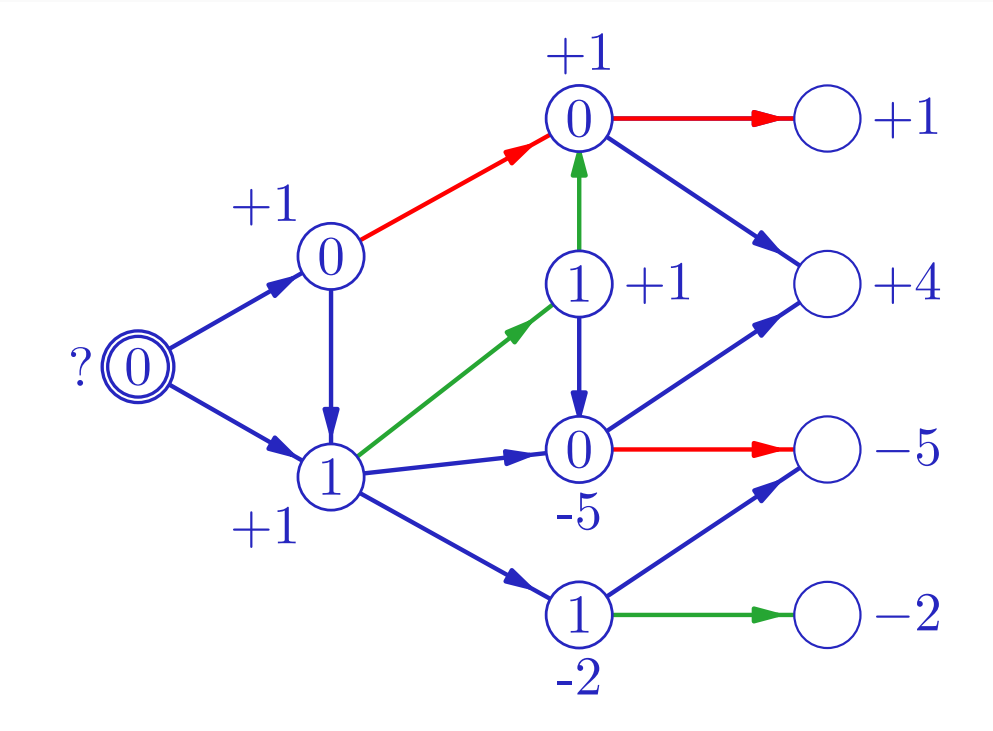
Пример анализа игры



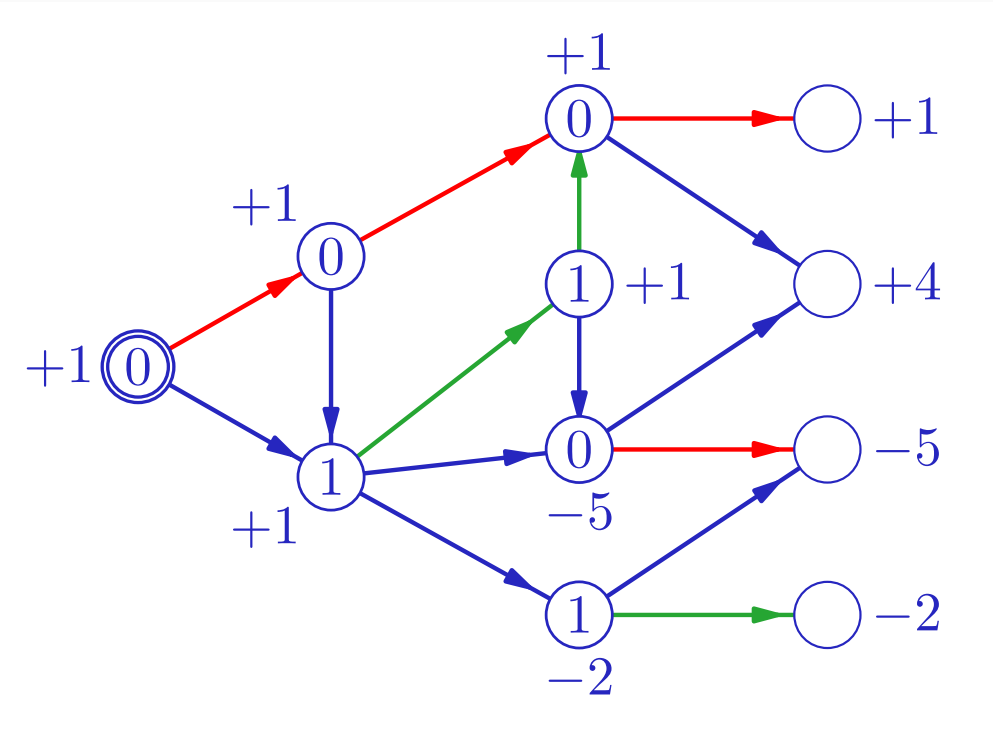
Пример анализа игры



Пример анализа игры



Пример анализа игры



Аддитивные платежи

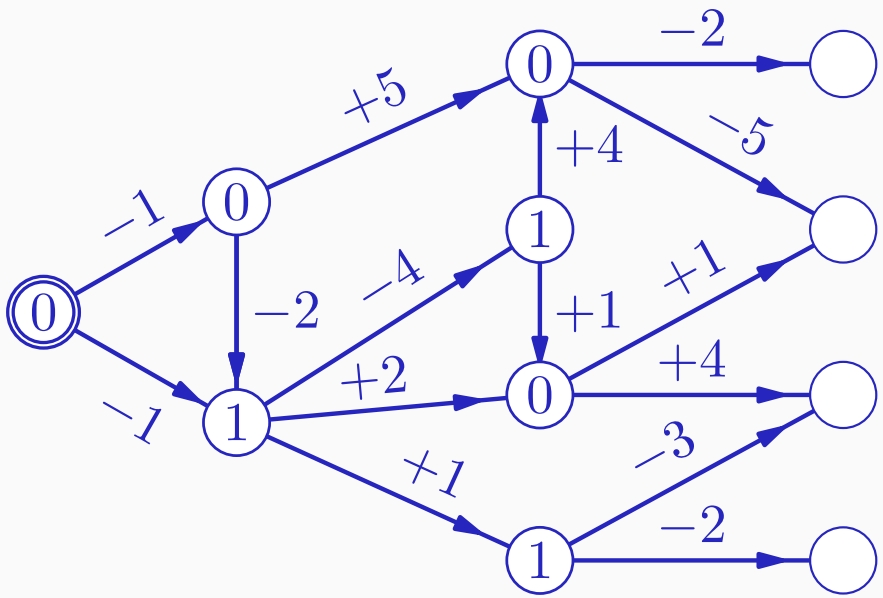
Каждый ход (u, v) имеет стоимость или платёж $w(u, v)$ (прибавка к выигрышу, она по-прежнему противоположна для двух игроков).

Результат партии равен сумме стоимостей по всем рёбрам партии.

Аддитивные платежи

Каждый ход (u, v) имеет стоимость или платёж $w(u, v)$ (прибавка к выигрышу, она по-прежнему противоположна для двух игроков).

Результат партии равен сумме стоимостей по всем рёбрам партии.

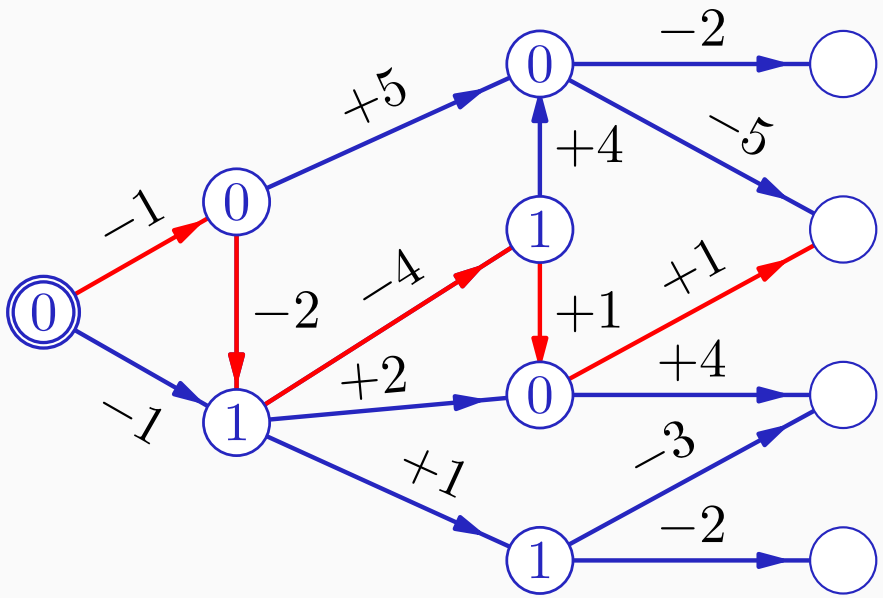


Стоимости указаны на рёбрах

Аддитивные платежи

Каждый ход (u, v) имеет стоимость или платёж $w(u, v)$ (прибавка к выигрышу, она по-прежнему противоположна для двух игроков).

Результат партии равен сумме стоимостей по всем рёбрам партии.



Партия игры с результатом -5 (проигрыш игрока 1)

Существование цены для игр на ациклическом графе (2)

Теорема

Если граф игры ациклический, а платежи аддитивные, то в игре существует цена, а гарантирующие её стратегии можно выбрать позиционными.

Аналогично предыдущему случаю: разбор позиций с конца.

Различия. Цена в позиции v_t вычисляется по формулам

$$c(v_t) = 0, \quad \text{если } v_t \text{ терминальная;}$$

$$c(v_t) = \min_{v_j:(v_t,v_j)\in E} \big(w(v_t, v_j) + c(v_j)\big), \quad \text{если } v_t \in V_0;$$

$$c(v_t) = \max_{v_j:(v_t,v_j)\in E} \big(w(v_t, v_j) + c(v_j)\big), \quad \text{если } v_t \in V_1.$$

Существование цены для игр на ациклическом графе (2)

Теорема

Если граф игры ациклический, а платежи аддитивные, то в игре существует цена, а гарантирующие её стратегии можно выбрать позиционными.

Аналогично предыдущему случаю: разбор позиций с конца.

Различия. Цена в позиции v_t вычисляется по формулам

$$c(v_t) = 0, \quad \text{если } v_t \text{ терминальная;}$$

$$c(v_t) = \min_{v_j:(v_t,v_j)\in E} \big(w(v_t, v_j) + c(v_j)\big), \quad \text{если } v_t \in V_0;$$

$$c(v_t) = \max_{v_j:(v_t,v_j)\in E} \big(w(v_t, v_j) + c(v_j)\big), \quad \text{если } v_t \in V_1.$$

Существование цены для игр на ациклическом графе (2)

Теорема

Если граф игры ациклический, а платежи аддитивные, то в игре существует цена, а гарантирующие её стратегии можно выбрать позиционными.

Аналогично предыдущему случаю: разбор позиций с конца.

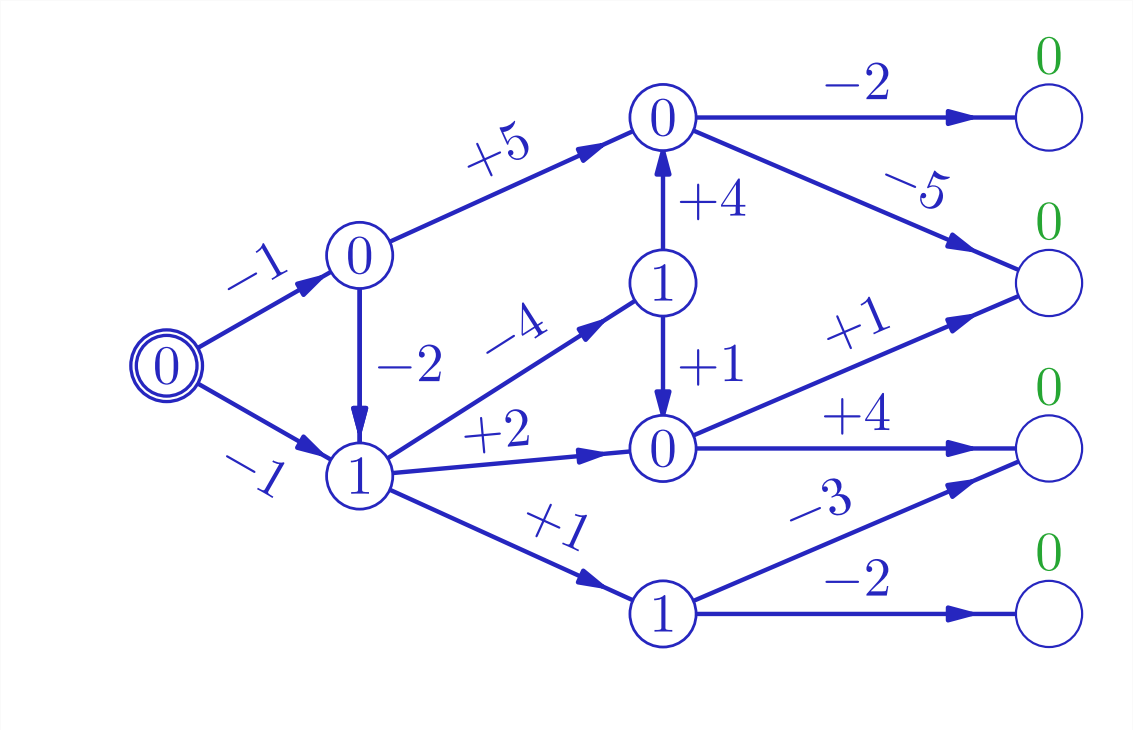
Различия. Цена в позиции v_t вычисляется по формулам

$$c(v_t) = 0, \qquad \text{если } v_t \text{ терминальная;}$$

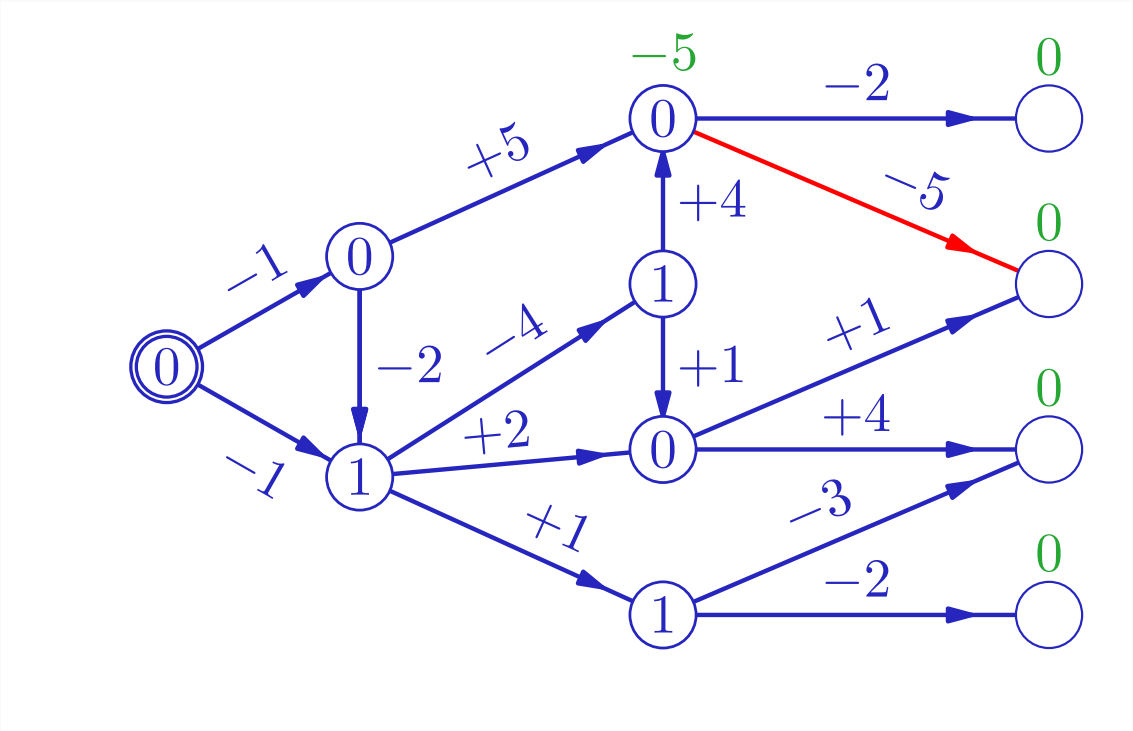
$$c(v_t) = \min_{v_j:(v_t,v_j)\in E} \big(w(v_t, v_j) + c(v_j)\big), \qquad \text{если } v_t \in V_0;$$

$$c(v_t) = \max_{v_j:(v_t,v_j)\in E} \big(w(v_t, v_j) + c(v_j)\big), \qquad \text{если } v_t \in V_1.$$

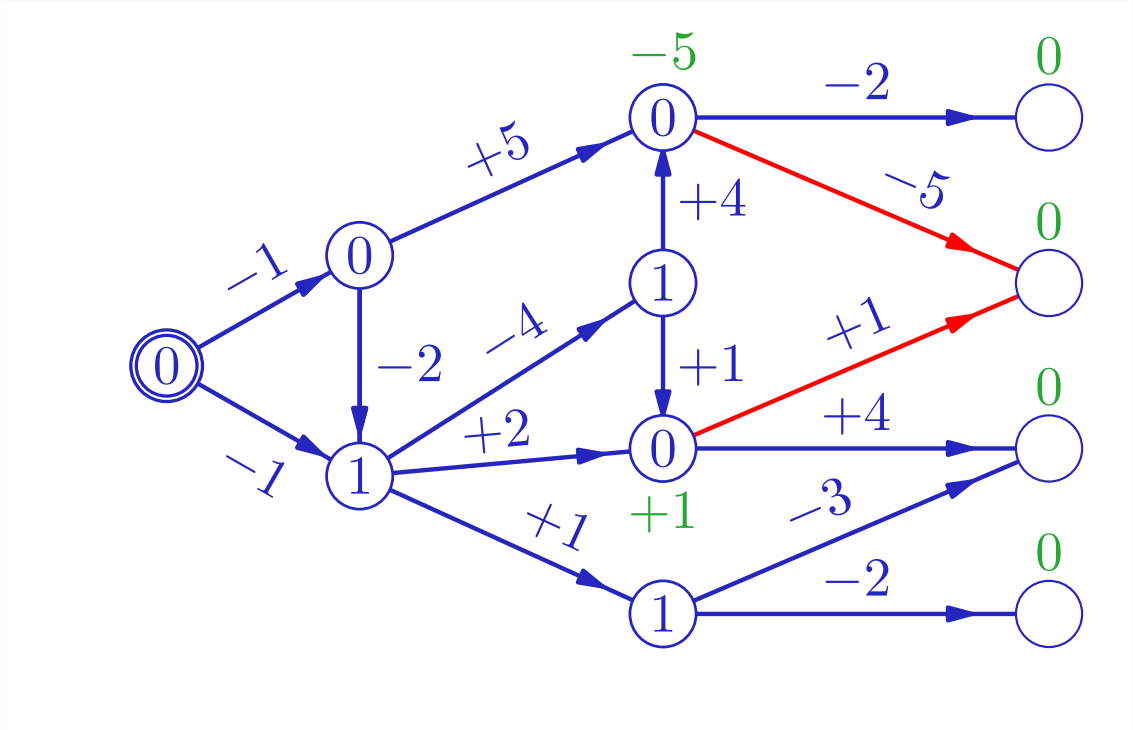
Пример анализа игры с аддитивными платежами



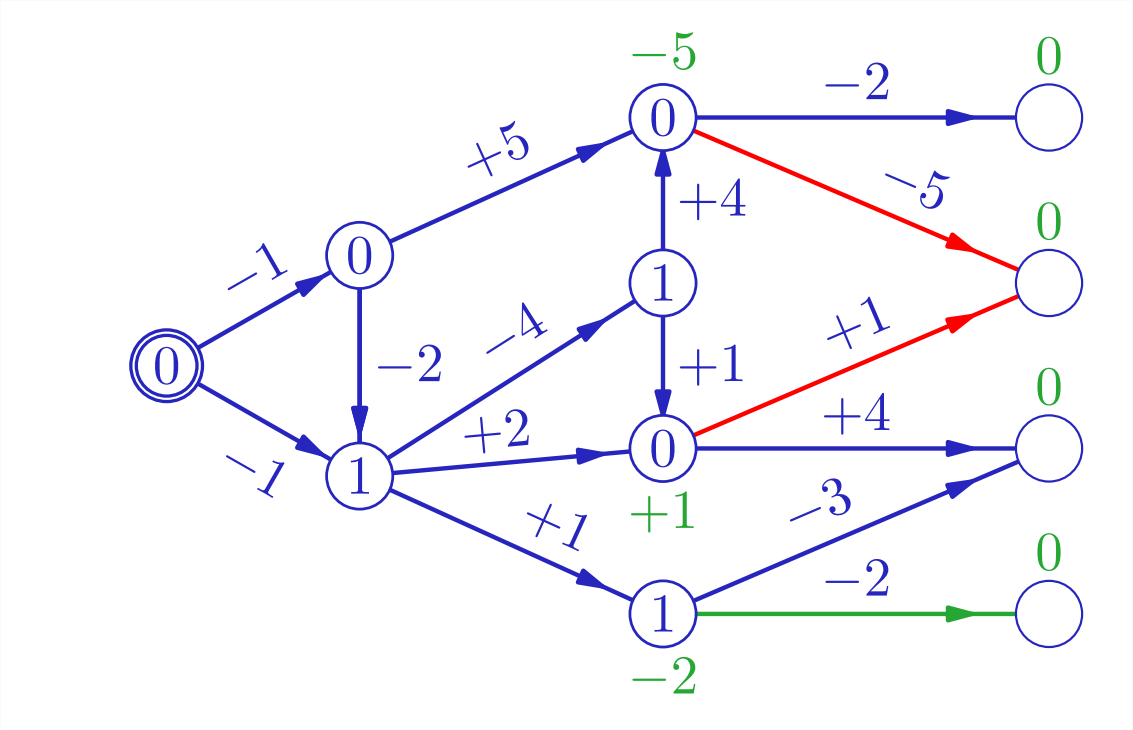
Пример анализа игры с аддитивными платежами



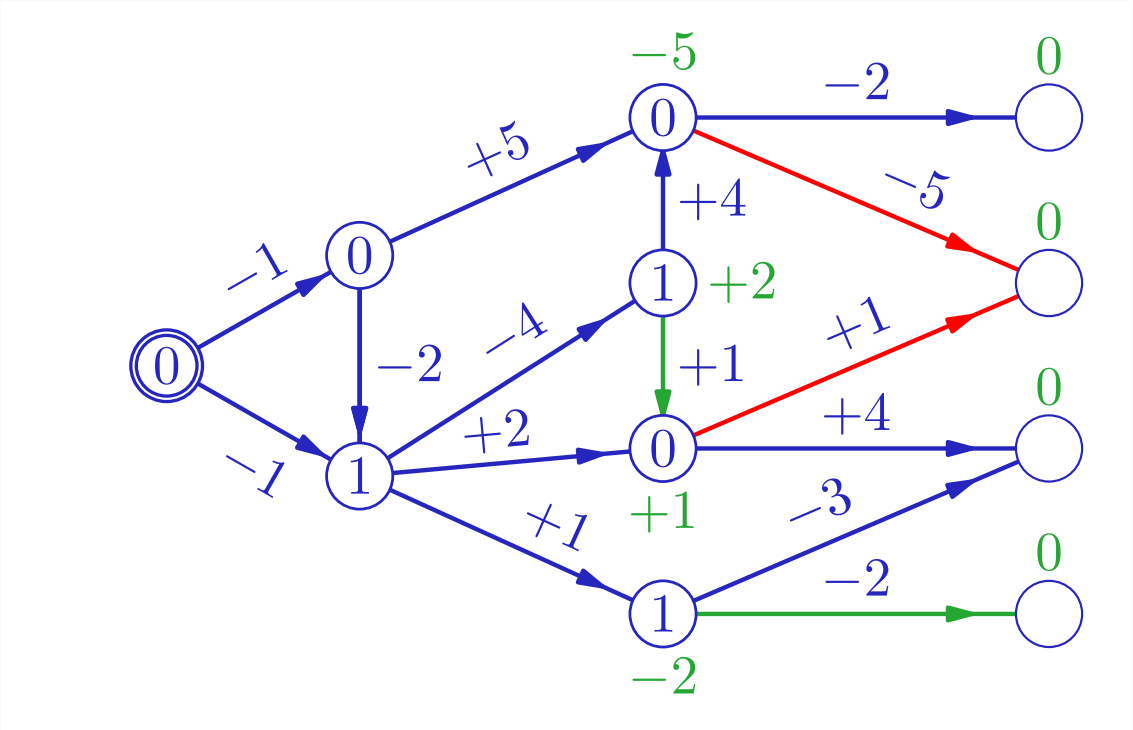
Пример анализа игры с аддитивными платежами



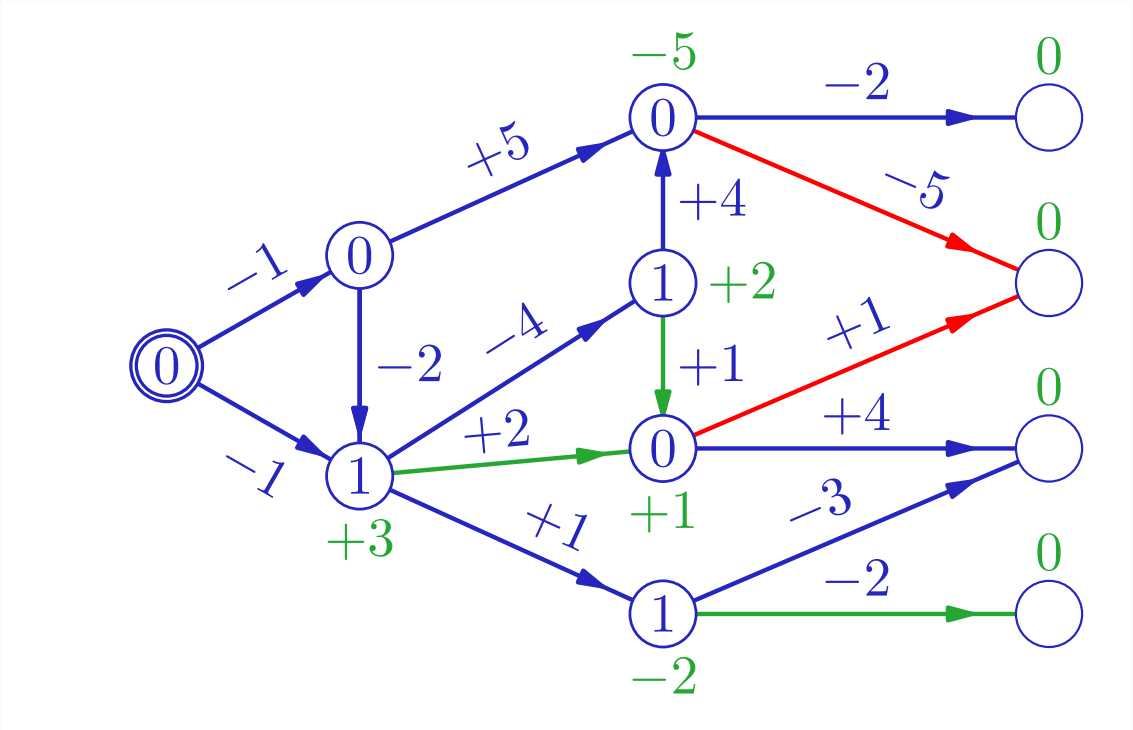
Пример анализа игры с аддитивными платежами



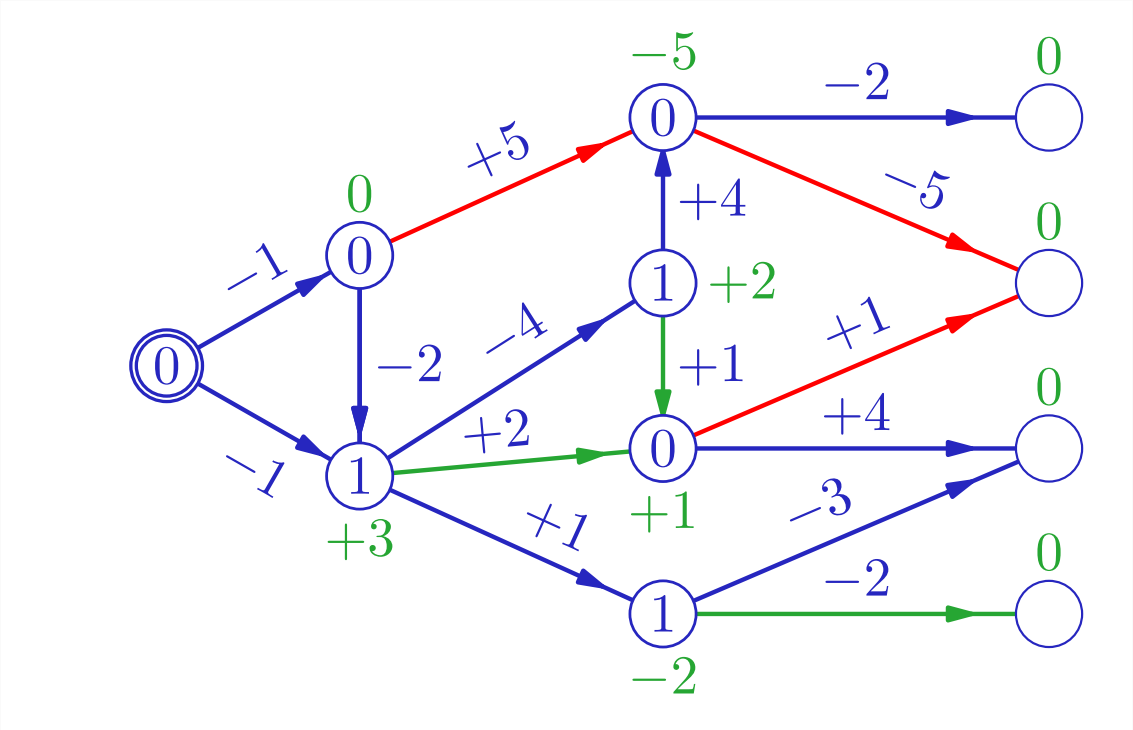
Пример анализа игры с аддитивными платежами



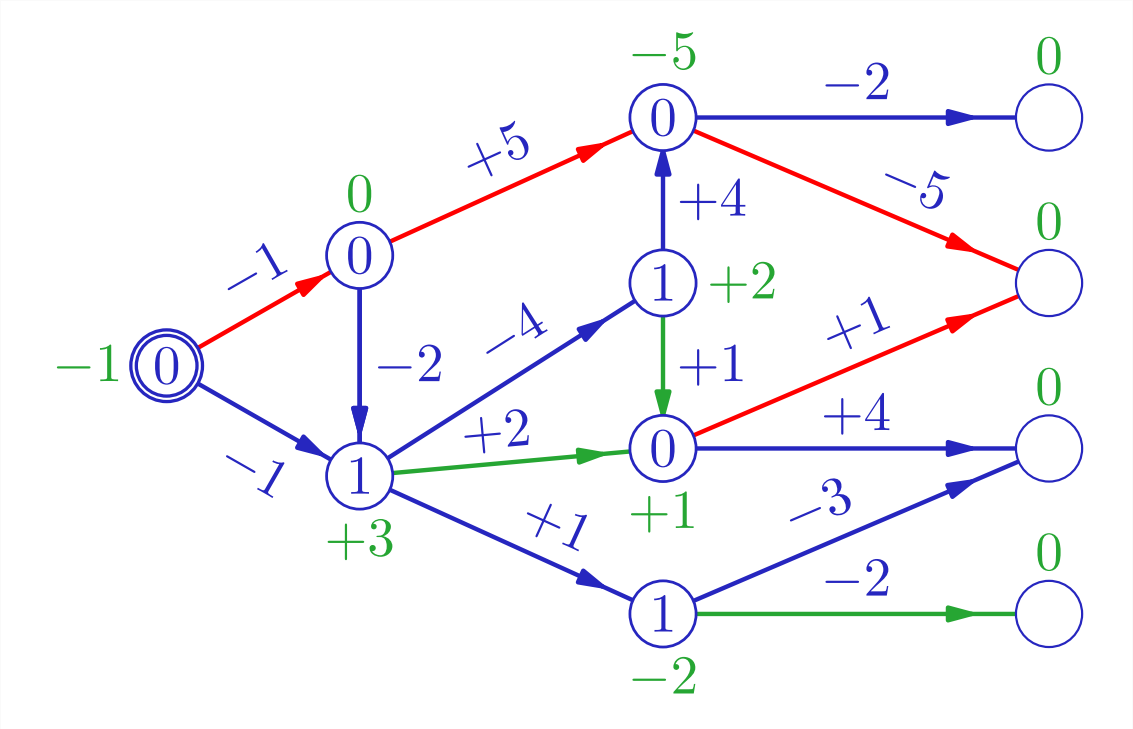
Пример анализа игры с аддитивными платежами



Пример анализа игры с аддитивными платежами



Пример анализа игры с аддитивными платежами



Игры с коротким описанием

Пример

Игроки по очереди присваивают значения булевым переменным $x_0, \ldots, x_{100}$: игрок 0 присваивает значения переменным с чётными номерами, а игрок 1 — переменным с нечётными номерами. Первым ходит игрок 0. Игра заканчивается, когда всем переменным присвоены значения. Если больше половины переменных имеют значение 1, то выиграл игрок 1; в противном случае выиграл его противник.

Утверждение

В этой игре есть выигрывающая стратегия у игрока 0.

Игры с коротким описанием

Пример

Игроки по очереди присваивают значения булевым переменным $x_0, \ldots, x_{100}$: игрок 0 присваивает значения переменным с чётными номерами, а игрок 1 — переменным с нечётными номерами. Первым ходит игрок 0. Игра заканчивается, когда всем переменным присвоены значения. Если больше половины переменных имеют значение 1, то выиграл игрок 1; в противном случае выиграл его противник.

Утверждение

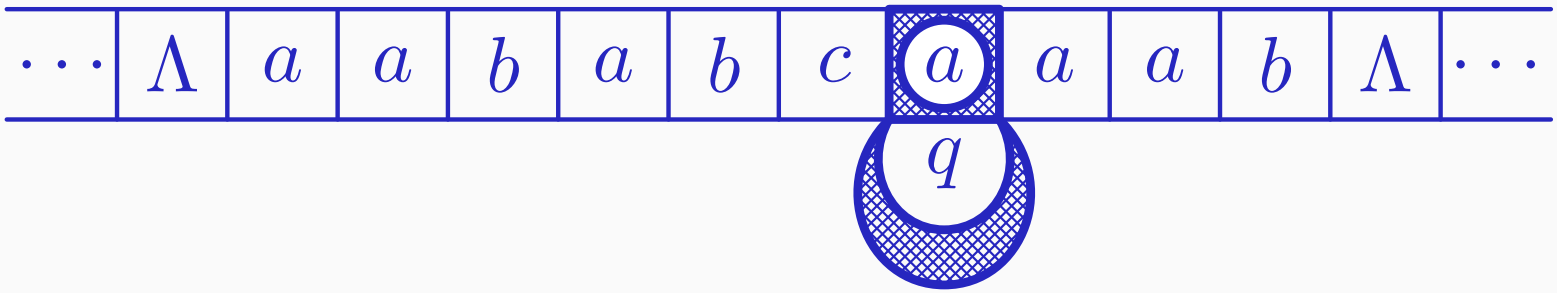
В этой игре есть выигрывающая стратегия у игрока 0.

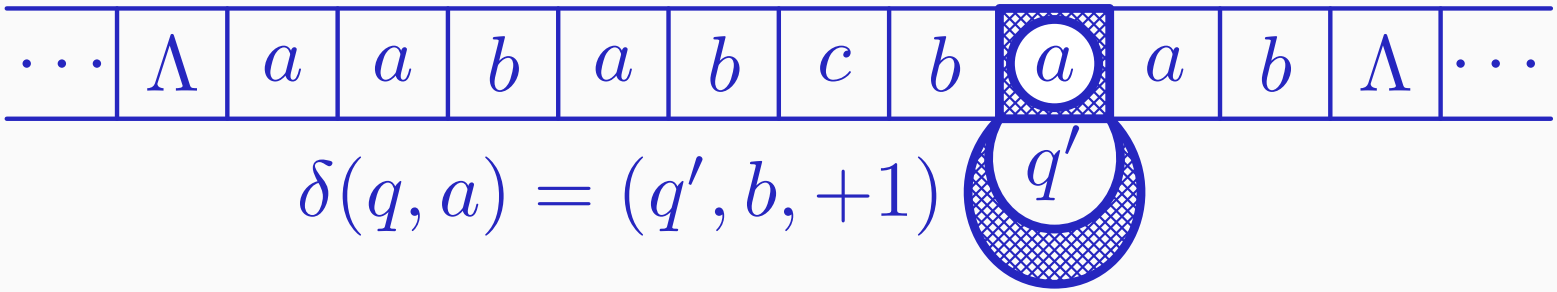
Модели вычислений и сложностные классы

Машины Тьюринга

МТ задаётся набором $(A, \Lambda, Q, q_0, Q_f, \delta)$.

$|A| < \infty$; $|Q| < \infty$; $\Lambda \in A$; $q_0 \in Q$; $Q_f \subseteq Q$;
$\delta\colon Q \times A \to Q \times A \times \{-1, 0, +1\}$.



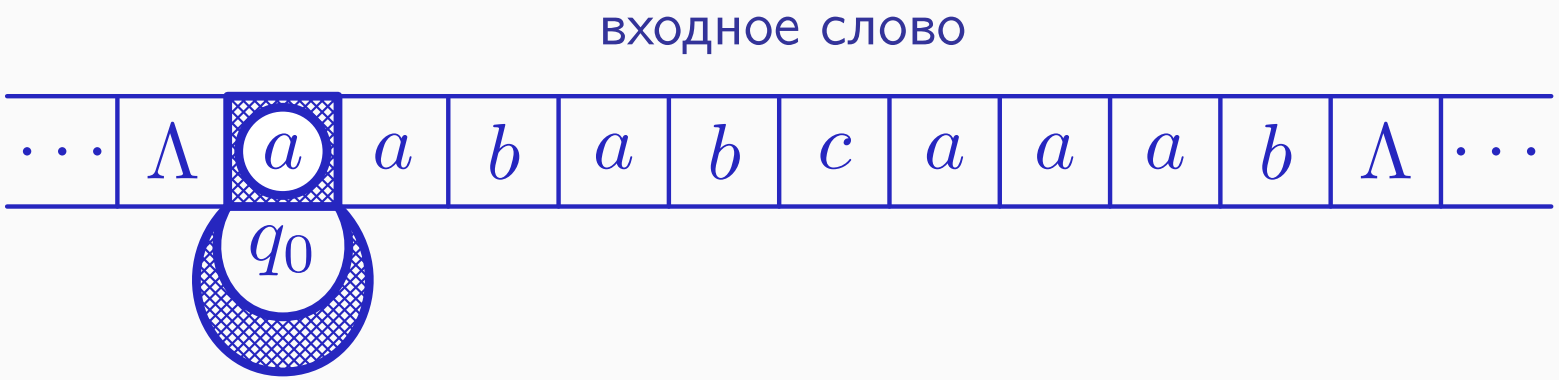


Начальная конфигурация и результат работы

Определение

Конфигурация МТ это тройка (состояние, положение головки, записанное на ленте слово).

Машина начинает работу в конфигурации вида



и останавливается, когда текущее состояние принадлежит Q_f.

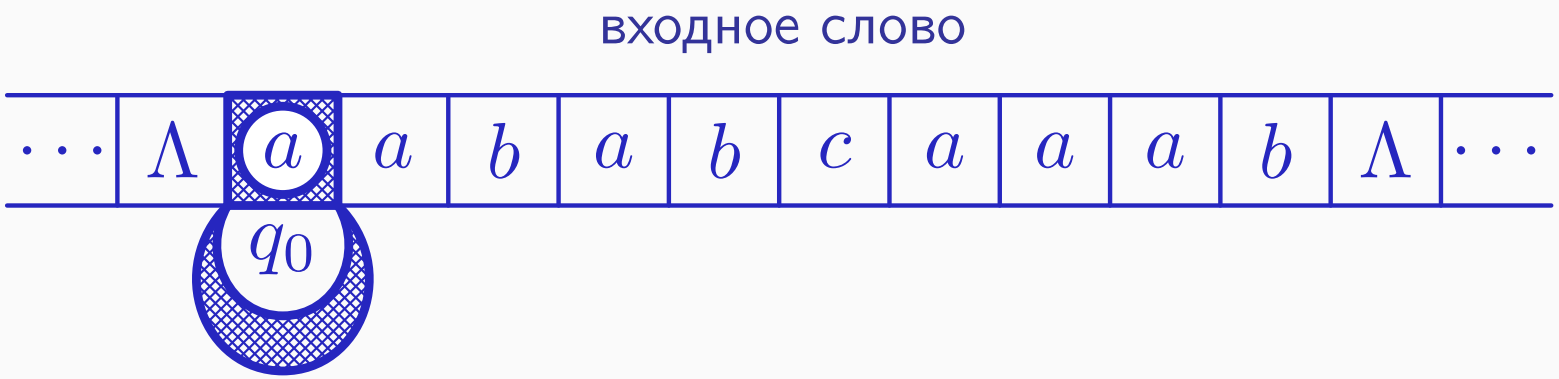
Результат работы МТ — это слово в алфавите $(A \setminus \{\Lambda\})^*$ от текущего положения головки до ближайшего справа Λ.

Начальная конфигурация и результат работы

Определение

Конфигурация МТ это тройка (состояние, положение головки, записанное на ленте слово).

Машина начинает работу в конфигурации вида



и останавливается, когда текущее состояние принадлежит Q_f.
Результат работы МТ — это слово в алфавите $(A \setminus \{\Lambda\})^*$ от текущего положения головки до ближайшего справа Λ.

Решающие МТ

$Q_f = Q_y \sqcup Q_n$;

Q_y: принимающие состояния; Q_n: отвергающие состояния.

Определение

Решающая МТ M принимает слово w, если при работе на входном слове w она останавливается в принимающем состоянии.

$L(M)$ — то множество слов, принимаемых M.

Решающая МТ M распознаёт язык $L \subseteq \{0,1\}^*$, если $L = L(M)$.

Основные ресурсы МТ

Время: количество тактов работы.

Память: максимальный размер непустой части ленты за всё время работы.

Сложностные классы

$\mathrm{DTIME}(T(n))$: $L \in \mathrm{DTIME}(T(n)$, если $\exists M$: $L(M) = L$ и M работает за время $O(T(n))$, n — длина входа.

P: класс полиномиального времени,

$$\mathrm{P} = \bigcup_{k=1}^{\infty} \mathrm{DTIME}(n^k).$$

$\mathrm{DSPACE}(S(n))$: $L \in \mathrm{DSPACE}(S(n)$, если $\exists M$: $L(M) = L$ и M работает на памяти $O(S(n))$, n — длина входа.

PSPACE: класс полиномиальной памяти,

$$\mathrm{PSPACE} = \bigcup_{k=1}^{\infty} \mathrm{DSPACE}(n^k).$$

Сложностные классы

$\mathrm{DTIME}(T(n))$: $L \in \mathrm{DTIME}(T(n)$, если $\exists M$: $L(M) = L$ и M работает за время $O(T(n))$, n — длина входа.

P: класс полиномиального времени,

$$\mathrm{P} = \bigcup_{k=1}^{\infty} \mathrm{DTIME}(n^k).$$

$\mathrm{DSPACE}(S(n))$: $L \in \mathrm{DSPACE}(S(n)$, если $\exists M$: $L(M) = L$ и M работает на памяти $O(S(n))$, n — длина входа.

PSPACE: класс полиномиальной памяти,

$$\mathrm{PSPACE} = \bigcup_{k=1}^{\infty} \mathrm{DSPACE}(n^k).$$

Сложностные классы

$\mathrm{DTIME}(T(n))$: $L \in \mathrm{DTIME}(T(n)$, если $\exists M$: $L(M) = L$ и M работает за время $O(T(n))$, n — длина входа.

P: класс полиномиального времени,

$$\mathrm{P} = \bigcup_{k=1}^{\infty} \mathrm{DTIME}(n^k).$$

$\mathrm{DSPACE}(S(n))$: $L \in \mathrm{DSPACE}(S(n)$, если $\exists M$: $L(M) = L$ и M работает на памяти $O(S(n))$, n — длина входа.

PSPACE: класс полиномиальной памяти,

$$\mathrm{PSPACE} = \bigcup_{k=1}^{\infty} \mathrm{DSPACE}(n^k).$$

Сложностные классы

$\mathrm{DTIME}(T(n))$: $L \in \mathrm{DTIME}(T(n)$, если $\exists M$: $L(M) = L$ и M работает за время $O(T(n))$, n — длина входа.

P: класс полиномиального времени,

$$\mathrm{P} = \bigcup_{k=1}^{\infty} \mathrm{DTIME}(n^k).$$

$\mathrm{DSPACE}(S(n))$: $L \in \mathrm{DSPACE}(S(n)$, если $\exists M$: $L(M) = L$ и M работает на памяти $O(S(n))$, n — длина входа.

PSPACE: класс полиномиальной памяти,

$$\mathrm{PSPACE} = \bigcup_{k=1}^{\infty} \mathrm{DSPACE}(n^k).$$

Сводимости и полные языки

Полиномиальная сводимость

$L_1 \leq_p L_2$, если существует такая всюду определённая и вычислимая за полиномиальное время функция $f : \{0,1\}^* \to \{0,1\}^*$ (сводящая функция), что $x \in L_1$ равносильно $f(x) \in L_2$.

$\mathcal{C}$-трудные и $\mathcal{C}$-полные языки

$\mathcal{C}$ — сложностной класс. Язык L является $\mathcal{C}$-трудным, если $K \leq_p L$ для любого $K \in \mathcal{C}$; и является $\mathcal{C}$-полным, если дополнительно $L \in C$.

Сводимости и полные языки

Полиномиальная сводимость

$L_1 \leq_p L_2$, если существует такая всюду определённая и вычислимая за полиномиальное время функция $f : \{0,1\}^* \to \{0,1\}^*$ (сводящая функция), что $x \in L_1$ равносильно $f(x) \in L_2$.

$\mathcal{C}$-трудные и $\mathcal{C}$-полные языки

$\mathcal{C}$ — сложностной класс. Язык L является $\mathcal{C}$-трудным, если $K \leq_p L$ для любого $K \in \mathcal{C}$; и является $\mathcal{C}$-полным, если дополнительно $L \in C$.

Альтернирующие машины Тьюринга

Определение

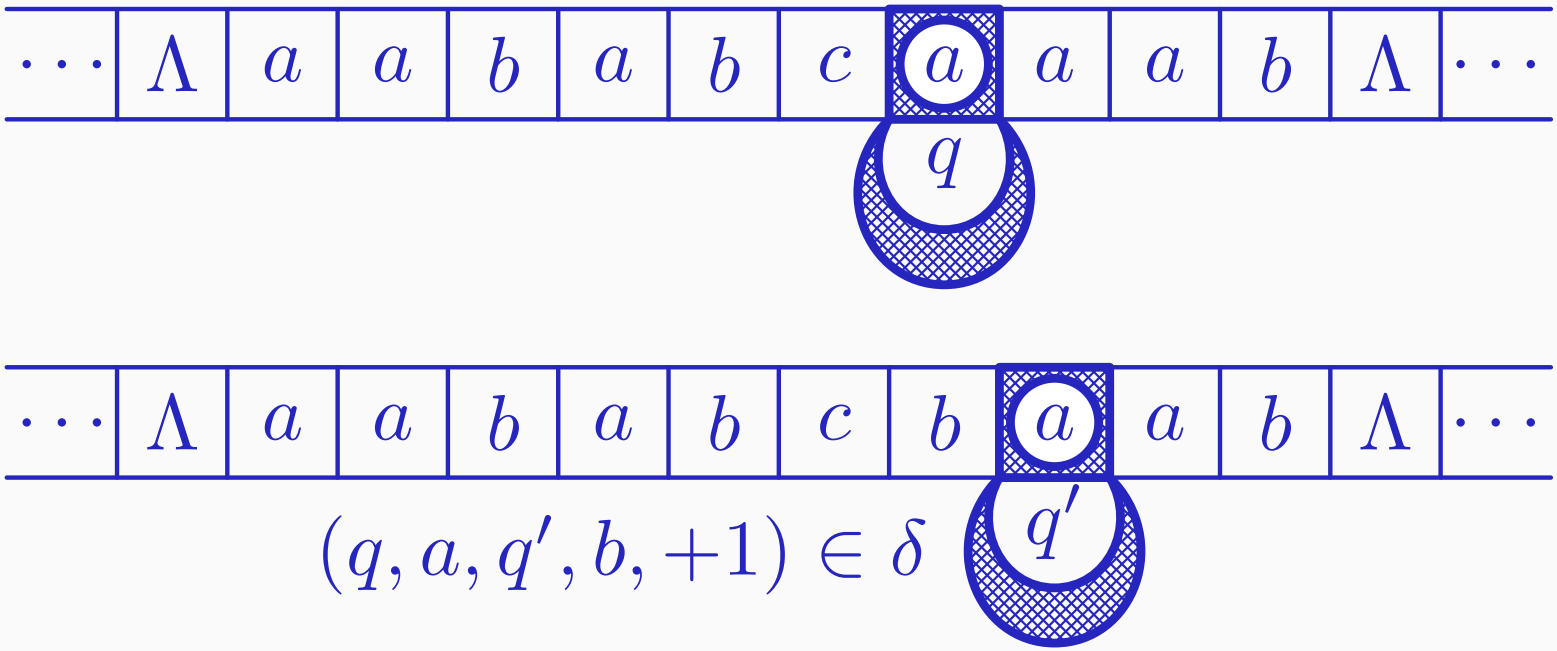
АМТ определяется почти так же, как и обычная МТ. Два важных отличия:

1. Все состояния АМТ разбиты на два множества: $Q_\vee$ (экзистенциальные состояния) и $Q_\wedge$ (универсальные состояния).
2. Вместо функции переходов для АМТ определено отношение переходов

$$\delta \subseteq Q \times A \times Q \times A \times \{-1, 0, +1\}.$$

Вычисление AMT

Вычисление такой машины не определено однозначно. Теперь это любая последовательность конфигураций, в которой каждые две последовательные конфигурации согласованы с отношением переходов. Вычисление либо заканчивается в финальном состоянии, либо продолжается вечно.



Игра на решающей АМТ

Граф конфигураций: конфигурации соединены ребром, если из первой можно перейти во вторую согласованным с отношением переходов способом. Финальные конфигурации тупиковые.

Определение игры

В состояниях из $Q_\vee$ ходит игрок 1. В состояниях из $Q_\wedge$ ходит игрок 0. Игра начинается из начальной конфигурации и заканчивается в конфигурации, состояние которой финальное.

Партия такой игры и есть вычисление АМТ. Игрок 1 выигрывает, если партия завершается в принимающем состоянии.

Игра на решающей АМТ

Граф конфигураций: конфигурации соединены ребром, если из первой можно перейти во вторую согласованным с отношением переходов способом. Финальные конфигурации тупиковые.

Определение игры

В состояниях из $Q_\vee$ ходит игрок 1. В состояниях из $Q_\wedge$ ходит игрок 0. Игра начинается из начальной конфигурации и заканчивается в конфигурации, состояние которой финальное.

Партия такой игры и есть вычисление АМТ. Игрок 1 выигрывает, если партия завершается в принимающем состоянии.

АМТ принимает слово за время T

АМТ M принимает слово w

В игре на M с входным словом w у игрока 1 есть выигрывающая стратегия.

Время работы: максимальная длина партии, в которой игрок 1 придерживается выбранной стратегии.

Замечание

Граф конфигураций не обязательно ациклический.

Игра не обязательно конечная.

На словах, отвергаемых АМТ, время может быть сколь угодно велико, машина может не останавливаться.

АМТ принимает слово за время T

АМТ M принимает слово w

В игре на M с входным словом w у игрока 1 есть выигрывающая стратегия.

Время работы: максимальная длина партии, в которой игрок 1 придерживается выбранной стратегии.

Замечание

Граф конфигураций не обязательно ациклический.

Игра не обязательно конечная.

На словах, отвергаемых АМТ, время может быть сколь угодно велико, машина может не останавливаться.

АМТ принимает слово за время T

АМТ M принимает слово w

В игре на M с входным словом w у игрока 1 есть выигрывающая стратегия.

Время работы: максимальная длина партии, в которой игрок 1 придерживается выбранной стратегии.

Замечание

Граф конфигураций не обязательно ациклический.

Игра не обязательно конечная.

На словах, отвергаемых АМТ, время может быть сколь угодно велико, машина может не останавливаться.

Классы альтернирующего времени

$\mathrm{AltTIME}(T(n))$: $L \in \mathrm{AltTIME}(T(n)$, если существует такая АМТ M, что $L(M) = L$ и M работает за время $O(T(n))$.

AltP:

$$\mathrm{AltP} = \bigcup_{k=1}^{\infty} \mathrm{AltTIME}(n^k).$$

Классы альтернирующего времени

$\mathrm{AltTIME}(T(n))$: $L \in \mathrm{AltTIME}(T(n)$, если существует такая АМТ M, что $L(M) = L$ и M работает за время $O(T(n))$.

AltP:

$$\mathrm{AltP} = \bigcup_{k=1}^{\infty} \mathrm{AltTIME}(n^k).$$

Когда в игре только один игрок

Недетерминированная МТ

Это АМТ, у которой $Q = Q_\vee$. Сложностные классы $\mathrm{NTIME}(T(n))$ и NP определяются как и раньше, но с заменой АМТ на НМТ.

Знаменитая гипотеза

Классы P и NP различаются.

Когда в игре только один игрок

Недетерминированная МТ

Это АМТ, у которой $Q = Q_\vee$. Сложностные классы $\mathrm{NTIME}(T(n))$ и NP определяются как и раньше, но с заменой АМТ на НМТ.

Знаменитая гипотеза

Классы P и NP различаются.

Альтернация превращает время в память

Альтернирующее время не больше обычной памяти

Теорема

Пусть $T(n)$ — такая конструируемая по памяти функция, что $T(n) \geqslant n$. Тогда $\mathrm{AltTIME}(T(n)) \subseteq \mathrm{DSPACE}(T(n))$.

Определение

Функция $f : \mathbb{N} \to \mathbb{N}$ называется конструируемой по памяти, если существует МТ (обычная, детерминированная), которая по входу 1^n вычисляет $f(n)$ на памяти $O(f(n))$.

План доказательства

Реализуем рекурсивную процедуру оценки каждой позиции игры на памяти $O(T(n))$.

Альтернирующее время не больше обычной памяти

Теорема

Пусть $T(n)$ — такая конструируемая по памяти функция, что $T(n) \geqslant n$. Тогда $\mathrm{AltTIME}(T(n)) \subseteq \mathrm{DSPACE}(T(n))$.

Определение

Функция $f : \mathbb{N} \to \mathbb{N}$ называется конструируемой по памяти, если существует МТ (обычная, детерминированная), которая по входу 1^n вычисляет $f(n)$ на памяти $O(f(n))$.

План доказательства

Реализуем рекурсивную процедуру оценки каждой позиции игры на памяти $O(T(n))$.

Альтернирующее время не больше обычной памяти

Теорема

Пусть $T(n)$ — такая конструируемая по памяти функция, что $T(n) \geqslant n$. Тогда $\mathrm{AltTIME}(T(n)) \subseteq \mathrm{DSPACE}(T(n))$.

Определение

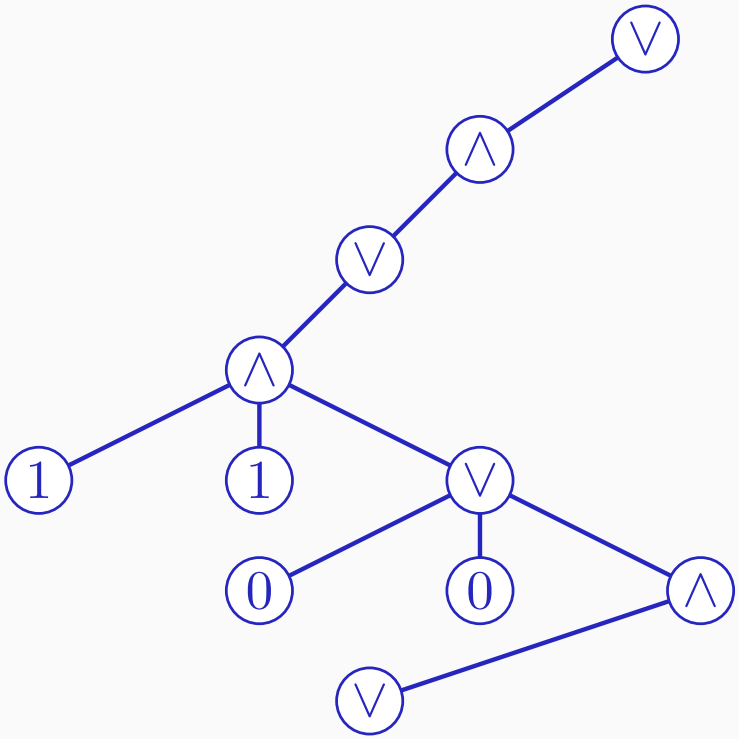
Функция $f : \mathbb{N} \to \mathbb{N}$ называется конструируемой по памяти, если существует МТ (обычная, детерминированная), которая по входу 1^n вычисляет $f(n)$ на памяти $O(f(n))$.

План доказательства

Реализуем рекурсивную процедуру оценки каждой позиции игры на памяти $O(T(n))$.

Оценка памяти

Достаточно помнить в каждый момент времени текущий путь по дереву рекурсии и результаты вычисления по остальным веткам ветвления.



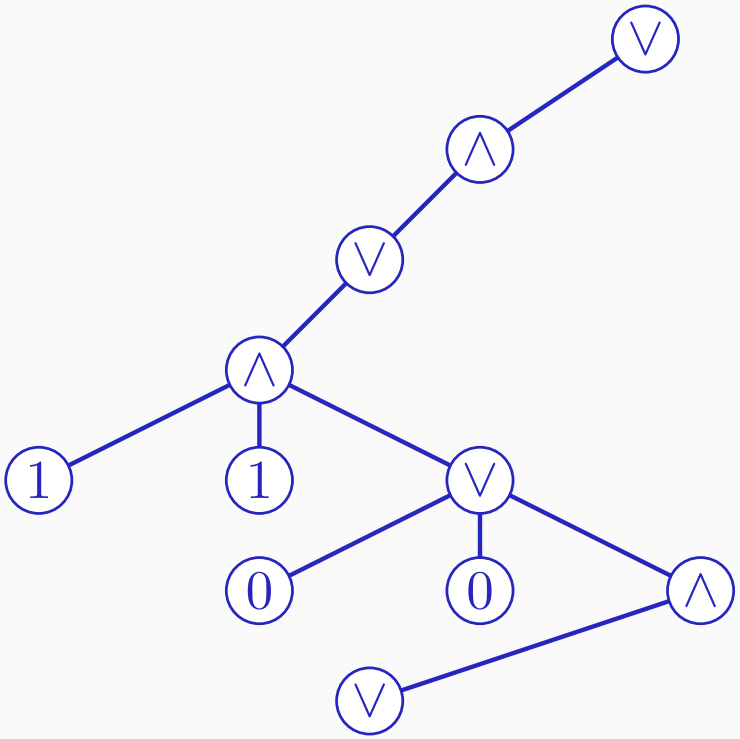
Глубина дерева $\leqslant T(n)$.

Ветвление $O(1)$.

Запоминать все конфигурации на каждом шаге рекурсии не требуется, их можно восстановить.

Оценка памяти

Достаточно помнить в каждый момент времени текущий путь по дереву рекурсии и результаты вычисления по остальным веткам ветвления.



Глубина дерева $\leqslant T(n)$.

Ветвление $O(1)$.

Запоминать все конфигурации на каждом шаге рекурсии не требуется, их можно восстановить.

Вопрос

Где использована конструируемость $T(n)$ по памяти?

Ответ

Для вычисления глубины дерева рекурсии. Если глубина становится слишком большой ($> C \cdot T(n)$, константу C можно «зашить» в конструкцию МТ), то игрок 1 проиграл.

Вопрос

Где использована конструируемость $T(n)$ по памяти?

Ответ

Для вычисления глубины дерева рекурсии. Если глубина становится слишком большой ($> C \cdot T(n)$, константу C можно «зашить» в конструкцию МТ), то игрок 1 проиграл.

В другую сторону чуть хуже

Теорема

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

Оценка числа конфигураций и времени работы

МТ M распознаёт язык $L \in \mathrm{DSPACE}(S(n))$ на памяти $C \cdot S(n)$. Множество состояний Q, алфавит A.

Память $O(S(n))$, поэтому возможных конфигураций МТ не больше $|Q| \cdot O(S(n)) \cdot |A|^{O(S(n))}$.

Время работы МТ не больше количества конфигураций: если какая-то конфигурация повторилась, машина «зацикливается» и никогда не останавливается.

В другую сторону чуть хуже

Теорема

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

Оценка числа конфигураций и времени работы

МТ M распознаёт язык $L \in \mathrm{DSPACE}(S(n))$ на памяти $C \cdot S(n)$. Множество состояний Q, алфавит A.

Память $O(S(n))$, поэтому возможных конфигураций МТ не больше $|Q| \cdot O(S(n)) \cdot |A|^{O(S(n))}$.

Время работы МТ не больше количества конфигураций: если какая-то конфигурация повторилась, машина «зацикливается» и никогда не останавливается.

Функция достижимости конфигураций

Функция $R(\alpha, \beta, t)$

Равна 1, если из конфигурации α машина переходит в конфигурацию β за t тактов работы. Равна 0 в противном случае.

Рекурсивное условие

1. Если $t = 0$, то $R(\alpha, \beta, 0) = 1$ тогда и только тогда, когда $\alpha = \beta$.
2. Если $t = 1$, то $R(\alpha, \beta, 0) = 1$ тогда и только тогда, когда β получается из α за один такт работы МТ.
3. $R(\alpha, \beta, t) = 1 \quad \Leftrightarrow \quad$ существует такая конфигурация γ, что $R(\alpha, \gamma, \lfloor t/2 \rfloor) = 1$ и $R(\alpha, \gamma, t - \lfloor t/2 \rfloor) = 1$.

Игра Оптимиста и Пессимиста

Оптимист старается обосновать, что $R(\alpha, \beta, t) = 1$.

Пессимист ему возражает.

Ход Оптимиста: указывает γ.

Ход Пессимиста: указывает, какой из членов конъюнкции

$$R(\alpha, \gamma, \lfloor t/2 \rfloor) = 1 \ \wedge \ R(\alpha, \gamma, t - \lfloor t/2 \rfloor) = 1$$

будет проверяться дальше.

В терминальных позициях $t = 0$ или $t = 1$, выигрыш Оптимиста определяется значением функции $R(\alpha, \beta, t)$.

Игра и функция достижимости

Утверждение

$R(\alpha, \beta, t) = 1 \quad \Leftrightarrow \quad$ у Оптимиста есть выигрывающая стратегия.

Доказательство

Если есть выигрывающая стратегия у Оптимиста, то по ней восстанавливается путь из α в β.

И наоборот, если такой путь есть, то выигрывающая стратегия Оптимиста состоит в том, чтобы выбирать γ, следуя этому пути.

Игра и функция достижимости

Утверждение

$R(\alpha, \beta, t) = 1 \quad \Leftrightarrow \quad$ у Оптимиста есть выигрывающая стратегия.

Доказательство

Если есть выигрывающая стратегия у Оптимиста, то по ней восстанавливается путь из α в β.

И наоборот, если такой путь есть, то выигрывающая стратегия Оптимиста состоит в том, чтобы выбирать γ, следуя этому пути.

АМТ, вычисляющая результат игры

Начальная конфигурация — α. Оптимист выбирает целевую (принимающую) конфигурацию β и время работы T.

Затем АМТ моделирует партию игры. В каждом раунде γ, отвечающий середине текущего интервала, а Пессимист выбирает, какой из подинтервалов рассматривать далее.

Замечание

Оптимист заведомо не выигрывает, если выбирает слишком большое время T.

Оптимист заведомо не выигрывает, если на каком-то ходе выбирает слишком длинную конфигурацию.

АМТ, вычисляющая результат игры

Начальная конфигурация — α. Оптимист выбирает целевую (принимающую) конфигурацию β и время работы T.

Затем АМТ моделирует партию игры. В каждом раунде γ, отвечающий середине текущего интервала, а Пессимист выбирает, какой из подинтервалов рассматривать далее.

Замечание

Оптимист заведомо не выигрывает, если выбирает слишком большое время T.

Оптимист заведомо не выигрывает, если на каком-то ходе выбирает слишком длинную конфигурацию.

Корректность и оценка времени работы

Если $w \in L$, то выигрывающая стратегия Оптимиста состоит в том, чтобы записывать конфигурации, возникающие на принимающем вычислении машины M.

1. Количество раундов ограничено логарифмом времени работы исходной МТ, то есть $O(S(n))$;
2. На каждом раунде Оптимисту нужно $O(S(n))$ времени, чтобы выбрать γ, а Пессимисту нужно $O(1)$ времени, чтобы выбрать один из двух интервалов.

Итоговая оценка: альтернирующее время $O(S(n)^2)$.

Если $w \notin L$, то Пессимист всегда может указать интервал, на котором $R(\alpha, \beta, t) = 0$.

Корректность и оценка времени работы

Если $w \in L$, то выигрывающая стратегия Оптимиста состоит в том, чтобы записывать конфигурации, возникающие на принимающем вычислении машины M.

1. Количество раундов ограничено логарифмом времени работы исходной МТ, то есть $O(S(n))$;
2. На каждом раунде Оптимисту нужно $O(S(n))$ времени, чтобы выбрать γ, а Пессимисту нужно $O(1)$ времени, чтобы выбрать один из двух интервалов.

Итоговая оценка: альтернирующее время $O(S(n)^2)$.

Если $w \notin L$, то Пессимист всегда может указать интервал, на котором $R(\alpha, \beta, t) = 0$.

Корректность и оценка времени работы

Если $w \in L$, то выигрывающая стратегия Оптимиста состоит в том, чтобы записывать конфигурации, возникающие на принимающем вычислении машины M.

1. Количество раундов ограничено логарифмом времени работы исходной МТ, то есть $O(S(n))$;
2. На каждом раунде Оптимисту нужно $O(S(n))$ времени, чтобы выбрать γ, а Пессимисту нужно $O(1)$ времени, чтобы выбрать один из двух интервалов.

Итоговая оценка: альтернирующее время $O(S(n)^2)$.

Если $w \notin L$, то Пессимист всегда может указать интервал, на котором $R(\alpha, \beta, t) = 0$.

Корректность и оценка времени работы

Если $w \in L$, то выигрывающая стратегия Оптимиста состоит в том, чтобы записывать конфигурации, возникающие на принимающем вычислении машины M.

1. Количество раундов ограничено логарифмом времени работы исходной МТ, то есть $O(S(n))$;
2. На каждом раунде Оптимисту нужно $O(S(n))$ времени, чтобы выбрать γ, а Пессимисту нужно $O(1)$ времени, чтобы выбрать один из двух интервалов.

Итоговая оценка: альтернирующее время $O(S(n)^2)$.

Если $w \notin L$, то Пессимист всегда может указать интервал, на котором $R(\alpha, \beta, t) = 0$.

Корректность и оценка времени работы

Если $w \in L$, то выигрывающая стратегия Оптимиста состоит в том, чтобы записывать конфигурации, возникающие на принимающем вычислении машины M.

1. Количество раундов ограничено логарифмом времени работы исходной МТ, то есть $O(S(n))$;
2. На каждом раунде Оптимисту нужно $O(S(n))$ времени, чтобы выбрать γ, а Пессимисту нужно $O(1)$ времени, чтобы выбрать один из двух интервалов.

Итоговая оценка: альтернирующее время $O(S(n)^2)$.

Если $w \notin L$, то Пессимист всегда может указать интервал, на котором $R(\alpha, \beta, t) = 0$.