

Распределенные алгоритмы и системы

mk.cs.msu.ru → Лекционные курсы → Распределенные алгоритмы и системы

Блок 23

Примеры волновых алгоритмов:
кольцевой алгоритм,
древесный алгоритм,
алгоритм эха

Лектор:
Подымов Владислав Васильевич
E-mail:
valdus@yandex.ru

Кольцевой алгоритм

Это централизованный волновой алгоритм, предназначенный для топологии кольца

Или для произвольной топологии, в которой выделено кольцо, проходящее через все узлы

В алгоритме достаточно будет отправлять сообщения по каналам только в одну сторону (*однонаправленное кольцо*)

В связи с этим полагается, что каждому узлу p известен узел $next_p$, следующий в кольце

Этот алгоритм устроен так: сообщение-маркер (**фишка**) **tok**

- ▶ отправляется в кольцо
- ▶ проходит полный круг и
- ▶ возвращается к исходному отправителю, после чего он принимает решение

Кольцевой алгоритм

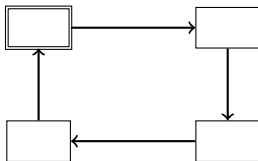
Код инициатора p :

1. $send(\mathbf{tok}) \rightarrow next_p$
2. $receive(\mathbf{tok})$
3. $decide$

Код последователя p :

1. $receive(\mathbf{tok})$
2. $send(\mathbf{tok}) \rightarrow next_p$

Пример для четырёх узлов:



Кольцевой алгоритм

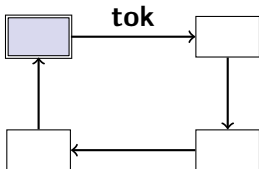
Код инициатора p :

1. $send(\mathbf{tok}) \rightarrow next_p$
2. $receive(\mathbf{tok})$
3. $decide$

Код последователя p :

1. $receive(\mathbf{tok})$
2. $send(\mathbf{tok}) \rightarrow next_p$

Пример для четырёх узлов:



Кольцевой алгоритм

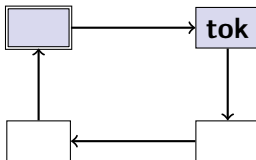
Код инициатора p :

1. $send(\mathbf{tok}) \rightarrow next_p$
2. $receive(\mathbf{tok})$
3. $decide$

Код последователя p :

1. $receive(\mathbf{tok})$
2. $send(\mathbf{tok}) \rightarrow next_p$

Пример для четырёх узлов:



Кольцевой алгоритм

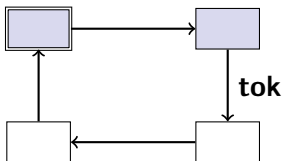
Код инициатора p :

1. $send(\mathbf{tok}) \rightarrow next_p$
2. $receive(\mathbf{tok})$
3. $decide$

Код последователя p :

1. $receive(\mathbf{tok})$
2. $send(\mathbf{tok}) \rightarrow next_p$

Пример для четырёх узлов:



Кольцевой алгоритм

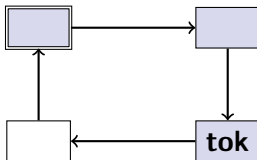
Код инициатора p :

1. $send(\mathbf{tok}) \rightarrow next_p$
2. $receive(\mathbf{tok})$
3. $decide$

Код последователя p :

1. $receive(\mathbf{tok})$
2. $send(\mathbf{tok}) \rightarrow next_p$

Пример для четырёх узлов:



Кольцевой алгоритм

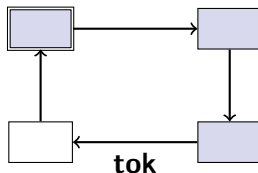
Код инициатора p :

1. $send(\mathbf{tok}) \rightarrow next_p$
2. $receive(\mathbf{tok})$
3. $decide$

Код последователя p :

1. $receive(\mathbf{tok})$
2. $send(\mathbf{tok}) \rightarrow next_p$

Пример для четырёх узлов:



Кольцевой алгоритм

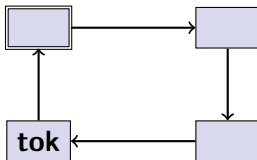
Код инициатора p :

1. $send(\mathbf{tok}) \rightarrow next_p$
2. $receive(\mathbf{tok})$
3. $decide$

Код последователя p :

1. $receive(\mathbf{tok})$
2. $send(\mathbf{tok}) \rightarrow next_p$

Пример для четырёх узлов:



Кольцевой алгоритм

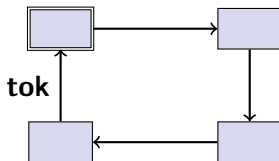
Код инициатора p :

1. $send(\mathbf{tok}) \rightarrow next_p$
2. $receive(\mathbf{tok})$
3. $decide$

Код последователя p :

1. $receive(\mathbf{tok})$
2. $send(\mathbf{tok}) \rightarrow next_p$

Пример для четырёх узлов:



Кольцевой алгоритм

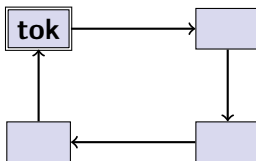
Код инициатора p :

1. $send(\mathbf{tok}) \rightarrow next_p$
2. $receive(\mathbf{tok})$
3. $decide$

Код последователя p :

1. $receive(\mathbf{tok})$
2. $send(\mathbf{tok}) \rightarrow next_p$

Пример для четырёх узлов:



Кольцевой алгоритм

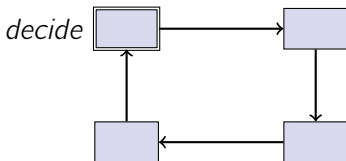
Код инициатора p :

1. $send(\mathbf{tok}) \rightarrow next_p$
2. $receive(\mathbf{tok})$
3. $decide$

Код последователя p :

1. $receive(\mathbf{tok})$
2. $send(\mathbf{tok}) \rightarrow next_p$

Пример для четырёх узлов:



Кольцевой алгоритм

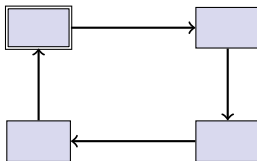
Код инициатора p :

1. $send(\mathbf{tok}) \rightarrow next_p$
2. $receive(\mathbf{tok})$
3. $decide$

Код последователя p :

1. $receive(\mathbf{tok})$
2. $send(\mathbf{tok}) \rightarrow next_p$

Пример для четырёх узлов:



Утверждение (Задача 1). Кольцевой алгоритм является волновым

Древесный алгоритм

Этот волновой алгоритм предназначен для топологии (неориентированного некорневого) дерева

Или для произвольной топологии, в которой выделено остовное дерево

В связи с этим полагается, что каждому узлу p известно множество соседей $Neigh_p$ этого узла в дереве

Алгоритм устроен так:

1. Каждый лист отправляет фишку в единственный доступный ему канал
2. Каждая внутренняя вершина
 - 2.1 принимает фишки от всех соседей, кроме одного,
 - 2.2 отправляет фишку оставшемуся соседу и
 - 2.3 если получает фишку от оставшегося соседа, то принимает решение

Древесный алгоритм

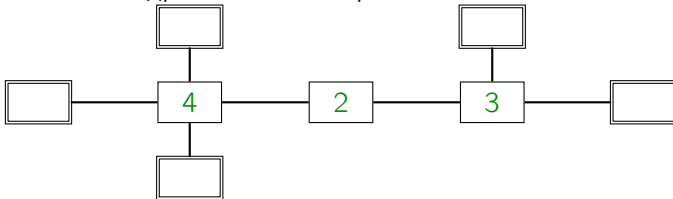
Код инициатора p :

1. $send(\mathbf{tok})$ (в единственный доступный канал)

Код последователя p :

1. **var** $X_p : 2^{Neigh_p} = Neigh_p$
2. Пока $|X_p| > 1$:
 - 2.1 $receive(\mathbf{tok}) \leftarrow q$ для какого-либо $q \in X$
 - 2.2 $X_p := X_p \setminus \{q\}$;
3. Присвоить в q единственный элемент X_p
4. $send(\mathbf{tok}) \rightarrow q$
5. $receive(\mathbf{tok}) \leftarrow q$
6. $decide$

Пример выполнения древесного алгоритма



Древесный алгоритм

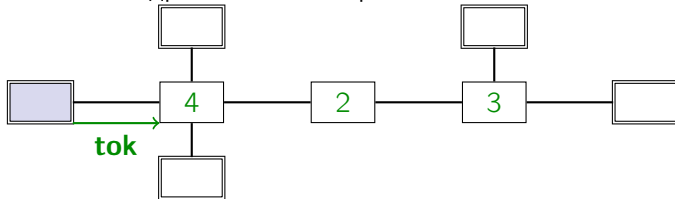
Код инициатора p :

1. $\text{send}(\mathbf{tok})$ (в единственный доступный канал)

Код последователя p :

1. **var** $X_p : 2^{Neigh_p} = Neigh_p$
2. Пока $|X_p| > 1$:
 - 2.1 $\text{receive}(\mathbf{tok}) \leftarrow q$ для какого-либо $q \in X$
 - 2.2 $X_p := X_p \setminus \{q\}$;
3. Присвоить в q единственный элемент X_p
4. $\text{send}(\mathbf{tok}) \rightarrow q$
5. $\text{receive}(\mathbf{tok}) \leftarrow q$
6. decide

Пример выполнения древесного алгоритма



Древесный алгоритм

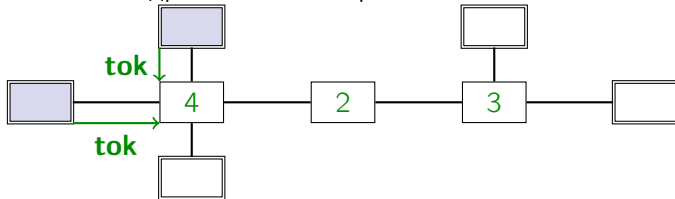
Код инициатора p :

1. $send(\mathbf{tok})$ (в единственный доступный канал)

Код последователя p :

1. **var** $X_p : 2^{Neigh_p} = Neigh_p$
2. Пока $|X_p| > 1$:
 - 2.1 $receive(\mathbf{tok}) \leftarrow q$ для какого-либо $q \in X$
 - 2.2 $X_p := X_p \setminus \{q\}$;
3. Присвоить в q единственный элемент X_p
4. $send(\mathbf{tok}) \rightarrow q$
5. $receive(\mathbf{tok}) \leftarrow q$
6. $decide$

Пример выполнения древесного алгоритма



Древесный алгоритм

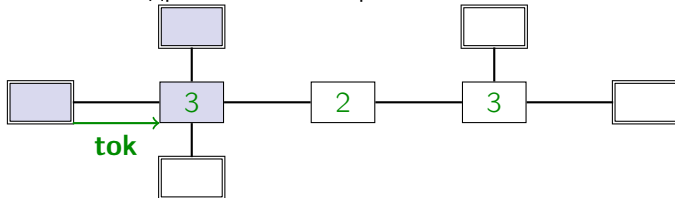
Код инициатора p :

1. $send(\mathbf{tok})$ (в единственный доступный канал)

Код последователя p :

1. **var** $X_p : 2^{Neigh_p} = Neigh_p$
2. Пока $|X_p| > 1$:
 - 2.1 $receive(\mathbf{tok}) \leftarrow q$ для какого-либо $q \in X$
 - 2.2 $X_p := X_p \setminus \{q\}$;
3. Присвоить в q единственный элемент X_p
4. $send(\mathbf{tok}) \rightarrow q$
5. $receive(\mathbf{tok}) \leftarrow q$
6. $decide$

Пример выполнения древесного алгоритма



Древесный алгоритм

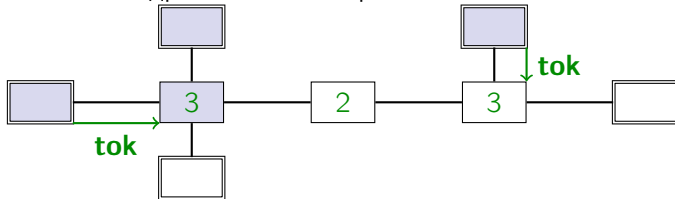
Код инициатора p :

1. $send(\mathbf{tok})$ (в единственный доступный канал)

Код последователя p :

1. **var** $X_p : 2^{Neigh_p} = Neigh_p$
2. Пока $|X_p| > 1$:
 - 2.1 $receive(\mathbf{tok}) \leftarrow q$ для какого-либо $q \in X$
 - 2.2 $X_p := X_p \setminus \{q\}$;
3. Присвоить в q единственный элемент X_p
4. $send(\mathbf{tok}) \rightarrow q$
5. $receive(\mathbf{tok}) \leftarrow q$
6. $decide$

Пример выполнения древесного алгоритма



Древесный алгоритм

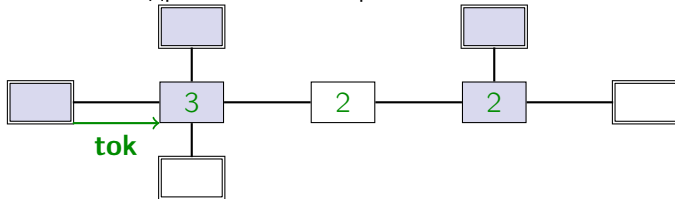
Код инициатора p :

1. $\text{send}(\mathbf{tok})$ (в единственный доступный канал)

Код последователя p :

1. **var** $X_p : 2^{Neigh_p} = Neigh_p$
2. Пока $|X_p| > 1$:
 - 2.1 $\text{receive}(\mathbf{tok}) \leftarrow q$ для какого-либо $q \in X$
 - 2.2 $X_p := X_p \setminus \{q\}$;
3. Присвоить в q единственный элемент X_p
4. $\text{send}(\mathbf{tok}) \rightarrow q$
5. $\text{receive}(\mathbf{tok}) \leftarrow q$
6. decide

Пример выполнения древесного алгоритма



Древесный алгоритм

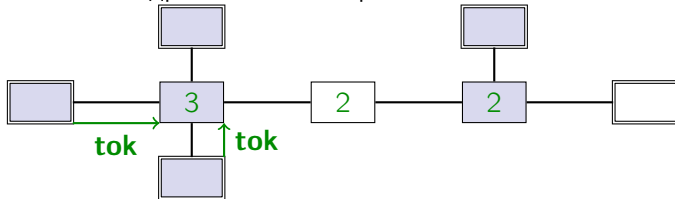
Код инициатора p :

1. $send(\mathbf{tok})$ (в единственный доступный канал)

Код последователя p :

1. **var** $X_p : 2^{Neigh_p} = Neigh_p$
2. Пока $|X_p| > 1$:
 - 2.1 $receive(\mathbf{tok}) \leftarrow q$ для какого-либо $q \in X$
 - 2.2 $X_p := X_p \setminus \{q\}$;
3. Присвоить в q единственный элемент X_p
4. $send(\mathbf{tok}) \rightarrow q$
5. $receive(\mathbf{tok}) \leftarrow q$
6. $decide$

Пример выполнения древесного алгоритма



Древесный алгоритм

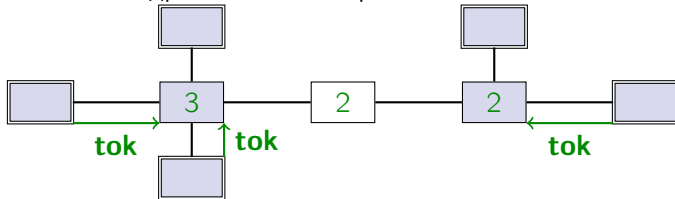
Код инициатора p :

1. $send(\mathbf{tok})$ (в единственный доступный канал)

Код последователя p :

1. **var** $X_p : 2^{Neigh_p} = Neigh_p$
2. Пока $|X_p| > 1$:
 - 2.1 $receive(\mathbf{tok}) \leftarrow q$ для какого-либо $q \in X$
 - 2.2 $X_p := X_p \setminus \{q\}$;
3. Присвоить в q единственный элемент X_p
4. $send(\mathbf{tok}) \rightarrow q$
5. $receive(\mathbf{tok}) \leftarrow q$
6. $decide$

Пример выполнения древесного алгоритма



Древесный алгоритм

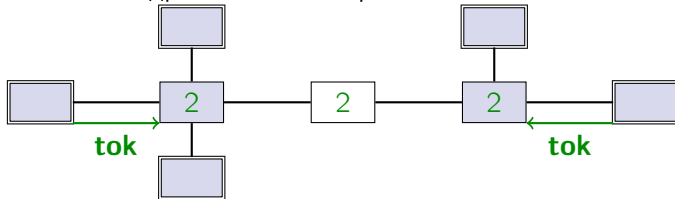
Код инициатора p :

1. $send(\mathbf{tok})$ (в единственный доступный канал)

Код последователя p :

1. **var** $X_p : 2^{Neigh_p} = Neigh_p$
2. Пока $|X_p| > 1$:
 - 2.1 $receive(\mathbf{tok}) \leftarrow q$ для какого-либо $q \in X$
 - 2.2 $X_p := X_p \setminus \{q\}$;
3. Присвоить в q единственный элемент X_p
4. $send(\mathbf{tok}) \rightarrow q$
5. $receive(\mathbf{tok}) \leftarrow q$
6. $decide$

Пример выполнения древесного алгоритма



Древесный алгоритм

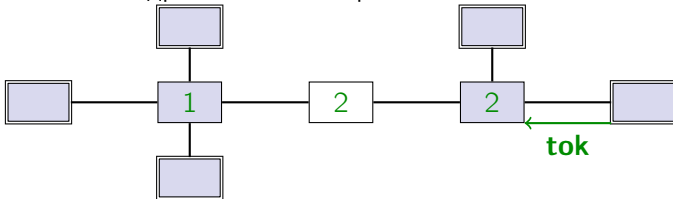
Код инициатора p :

1. $send(\mathbf{tok})$ (в единственный доступный канал)

Код последователя p :

1. **var** $X_p : 2^{Neigh_p} = Neigh_p$
2. Пока $|X_p| > 1$:
 - 2.1 $receive(\mathbf{tok}) \leftarrow q$ для какого-либо $q \in X$
 - 2.2 $X_p := X_p \setminus \{q\}$;
3. Присвоить в q единственный элемент X_p
4. $send(\mathbf{tok}) \rightarrow q$
5. $receive(\mathbf{tok}) \leftarrow q$
6. $decide$

Пример выполнения древесного алгоритма



Древесный алгоритм

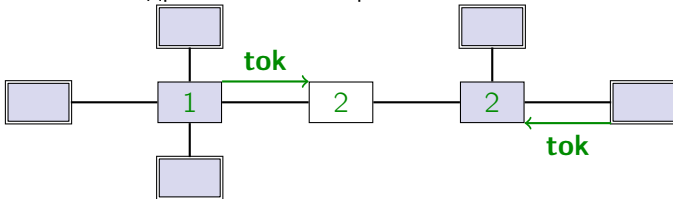
Код инициатора p :

1. $send(\mathbf{tok})$ (в единственный доступный канал)

Код последователя p :

1. **var** $X_p : 2^{Neigh_p} = Neigh_p$
2. Пока $|X_p| > 1$:
 - 2.1 $receive(\mathbf{tok}) \leftarrow q$ для какого-либо $q \in X$
 - 2.2 $X_p := X_p \setminus \{q\}$;
3. Присвоить в q единственный элемент X_p
4. $send(\mathbf{tok}) \rightarrow q$
5. $receive(\mathbf{tok}) \leftarrow q$
6. $decide$

Пример выполнения древесного алгоритма



Древесный алгоритм

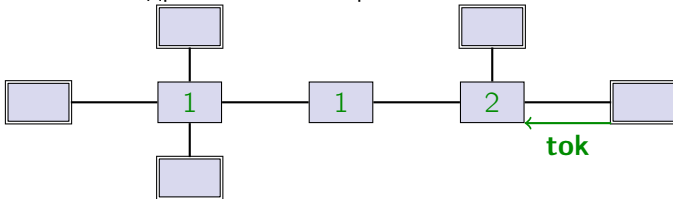
Код инициатора p :

1. $send(\mathbf{tok})$ (в единственный доступный канал)

Код последователя p :

1. **var** $X_p : 2^{Neigh_p} = Neigh_p$
2. Пока $|X_p| > 1$:
 - 2.1 $receive(\mathbf{tok}) \leftarrow q$ для какого-либо $q \in X$
 - 2.2 $X_p := X_p \setminus \{q\}$;
3. Присвоить в q единственный элемент X_p
4. $send(\mathbf{tok}) \rightarrow q$
5. $receive(\mathbf{tok}) \leftarrow q$
6. $decide$

Пример выполнения древесного алгоритма



Древесный алгоритм

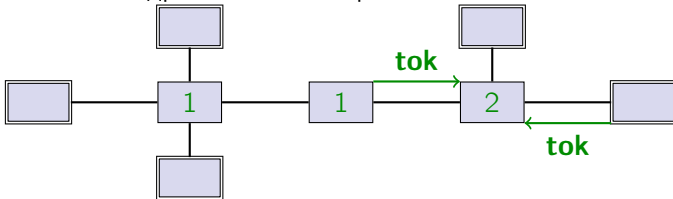
Код инициатора p :

1. $send(\mathbf{tok})$ (в единственный доступный канал)

Код последователя p :

1. **var** $X_p : 2^{Neigh_p} = Neigh_p$
2. Пока $|X_p| > 1$:
 - 2.1 $receive(\mathbf{tok}) \leftarrow q$ для какого-либо $q \in X$
 - 2.2 $X_p := X_p \setminus \{q\}$;
3. Присвоить в q единственный элемент X_p
4. $send(\mathbf{tok}) \rightarrow q$
5. $receive(\mathbf{tok}) \leftarrow q$
6. $decide$

Пример выполнения древесного алгоритма



Древесный алгоритм

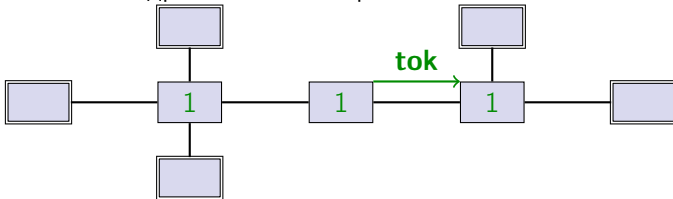
Код инициатора p :

1. $send(\mathbf{tok})$ (в единственный доступный канал)

Код последователя p :

1. **var** $X_p : 2^{Neigh_p} = Neigh_p$
2. Пока $|X_p| > 1$:
 - 2.1 $receive(\mathbf{tok}) \leftarrow q$ для какого-либо $q \in X$
 - 2.2 $X_p := X_p \setminus \{q\}$;
3. Присвоить в q единственный элемент X_p
4. $send(\mathbf{tok}) \rightarrow q$
5. $receive(\mathbf{tok}) \leftarrow q$
6. $decide$

Пример выполнения древесного алгоритма



Древесный алгоритм

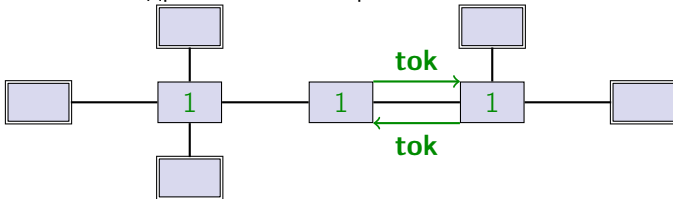
Код инициатора p :

1. $\text{send}(\mathbf{tok})$ (в единственный доступный канал)

Код последователя p :

1. **var** $X_p : 2^{Neigh_p} = Neigh_p$
2. Пока $|X_p| > 1$:
 - 2.1 $\text{receive}(\mathbf{tok}) \leftarrow q$ для какого-либо $q \in X$
 - 2.2 $X_p := X_p \setminus \{q\}$;
3. Присвоить в q единственный элемент X_p
4. $\text{send}(\mathbf{tok}) \rightarrow q$
5. $\text{receive}(\mathbf{tok}) \leftarrow q$
6. decide

Пример выполнения древесного алгоритма



Древесный алгоритм

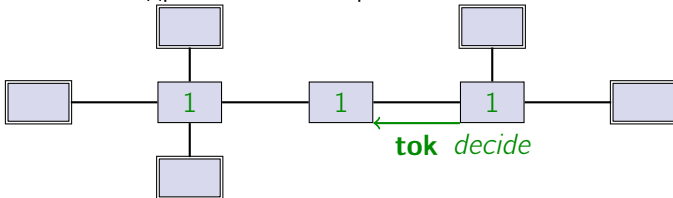
Код инициатора p :

1. $send(\mathbf{tok})$ (в единственный доступный канал)

Код последователя p :

1. **var** $X_p : 2^{Neigh_p} = Neigh_p$
2. Пока $|X_p| > 1$:
 - 2.1 $receive(\mathbf{tok}) \leftarrow q$ для какого-либо $q \in X$
 - 2.2 $X_p := X_p \setminus \{q\}$;
3. Присвоить в q единственный элемент X_p
4. $send(\mathbf{tok}) \rightarrow q$
5. $receive(\mathbf{tok}) \leftarrow q$
6. $decide$

Пример выполнения древесного алгоритма



Древесный алгоритм

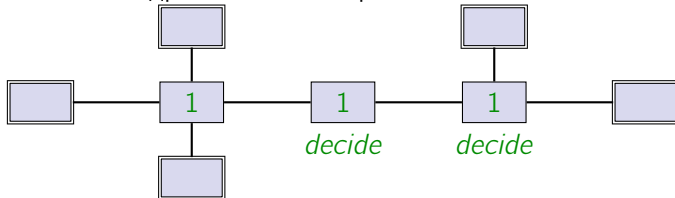
Код инициатора p :

1. $send(\mathbf{tok})$ (в единственный доступный канал)

Код последователя p :

1. **var** $X_p : 2^{Neigh_p} = Neigh_p$
2. Пока $|X_p| > 1$:
 - 2.1 $receive(\mathbf{tok}) \leftarrow q$ для какого-либо $q \in X$
 - 2.2 $X_p := X_p \setminus \{q\}$;
3. Присвоить в q единственный элемент X_p
4. $send(\mathbf{tok}) \rightarrow q$
5. $receive(\mathbf{tok}) \leftarrow q$
6. $decide$

Пример выполнения древесного алгоритма



Древесный алгоритм

Утверждение. Древесный алгоритм для деревьев с хотя бы двумя внутренними узлами является волновым

Доказательство.

Завершаемость следует из того, что

- ▶ инициатор выполняет ровно одну команду,
- ▶ а последователь —
 - ▶ конечное число витков цикла (столько, сколько всего существует узлов),
 - ▶ в каждом витке выполняет 2 команды и
 - ▶ вне цикла выполняет 4 команды

Принятие решения можно обосновать так

Пусть V и E — множество всех вершин и всех рёбер дерева топологии соответственно

Древесный алгоритм

Доказательство.

Наблюдение 1: каждый узел отправляет не более одного сообщения

Наблюдение 2:

1. **var** $X_p : 2^{Neigh_p} = Neigh_p$
2. Пока $|X_p| > 1$:
 - 2.1 $receive(\mathbf{tok}) \leftarrow q$ для какого-либо $q \in X$
 - 2.2 $X_p := X_p \setminus \{q\}$;

Перед началом выполнения цикла (2) и после каждого его витка $|X_p|$ — это количество соседей p , от которых сообщение ещё не принято:

- ▶ Перед началом $|X_p| = |Neigh_p|$ и сообщений нет от всех соседей
- ▶ После очередного витка количество соседей, от которых не получено сообщение, и мощность X_p уменьшаются на 1

Древесный алгоритм

Доказательство.

Наблюдение 3

Положим $|X_p| = 1$ для каждого листа

Тогда перед выполнением цикла (2) и после каждого его витка верно $\sum_{p \in V} |X_p| = 2|E| - K$, где K — суммарное число сообщений, принятых в цикле

Наблюдение 4: в заключительной конфигурации все отправленные сообщения приняты (то есть все каналы пусты)

Это следует из того, что

- ▶ каждый узел отправляет не более одного сообщения в каждый доступный канал и
- ▶ каждый последователь способен принять по одному сообщению из каждого доступного канала (команды 2.1 и 5)

Древесный алгоритм

Доказательство. $(\sum_{p \in V} |X_p| = 2|E| - K)$

Предположим от противного, что по завершении алгоритма ни один последователь не принял решение

Тогда ни один последователь не выполнил *receive* (5) после цикла (2) — иначе после этой команды можно немедленно принять решение (6)

Значит, обозначенное выше K в заключительной конфигурации — это суммарное число как принятых, так и отправленных сообщений

Так как каждый узел отправляет не более одного сообщения, то $(|V| - K)$ узлов q не отправили сообщения, и для них по условию цикла (2) верно $|X_q| \geq 2$

При этом для каждого узла верно $|X_p| \geq 1$ (команды 1, 2, 2.2)

Значит, $2|E| - K = \sum_{p \in V} |X_p| \geq |V| + (|V| - K) = 2|V| - K$

Следовательно, $|E| \geq |V|$, что **НЕВОЗМОЖНО** для дерева

Древесный алгоритм

Доказательство.

Для обоснования **полноты покрытия** покажем индукцией по размеру поддерева, что

- ▶ если действие $\vec{\alpha}[i]$ последовательности действий вычисления алгоритма — это отправка сообщения $p \rightarrow q$,
- ▶ и если $T_{pq} = (V_{pq}, E_{pq})$ — компонента связности вершины p после удаления ребра (p, q) ,
- ▶ то $\forall x \in V_{pq} : \exists j \in \mathcal{I}_{\vec{\alpha}}, j \preceq i : \vec{\alpha}[j] \in \mathcal{A}_x$

Полнота покрытия непосредственно следует из этого факта (*когда он будет доказан*), того, что принятию решения (б) узлом r предшествует приём сообщений $s \rightarrow r$ для всех соседей s (2.1, 4), и **свойств отношения \preceq**

Древесный алгоритм

Доказательство. $(\forall x \in V_{pq} : \exists j \in \mathcal{I}_\pi, j \preceq i : \vec{\alpha}[j] \in \mathcal{A}_x)$

База: если p — лист, то $V_{pq} = \{p\}$, и $j = i$

Индуктивный переход

Если p не лист (то есть последователь), то **по устройству алгоритма** им отправляется сообщение $p \rightarrow q$ (4) только после предшествующего приёма сообщений $r \rightarrow p$ от всех остальных соседей r (2.1) и предшествующей взаимосвязанной отправки этих сообщений соседями r

По **предположению индукции**, упомянутому действию отправки $r \rightarrow p$ причинно-следственно предшествует хотя бы одно событие каждой вершины дерева T_{rp}

При этом $V_{pq} = (\cup_{r \in \text{Neigh}_p \setminus \{q\}} V_{rp}) \cup \{p\}$

Следовательно, по **свойствам отношения** \preceq , отправке $p \rightarrow q$ нестрого предшествуют действия во всех узлах V_{pq} ▼

Задача 2. Докажите, что в древесном алгоритме решение всегда принимают ровно два узла

Алгоритм эха

Это централизованный волновой алгоритм, предназначенный для произвольных топологий

Полагается, что каждому узлу p известно множество его соседей $Neigh_p$ (как и в древесном алгоритме)

Алгоритм устроен так:

- ▶ Инициатор

- ▶ Рассылает фишки по всем доступным ему каналам («кричит»)
- ▶ Собирает фишки из всех каналов («дожидается эха»)
- ▶ Принимает решение

- ▶ Последователь

- ▶ После получения первой фишки рассылает фишки по всем доступным ему каналам, кроме того, по которому получил первую фишку
- ▶ Собирает фишки из всех каналов
- ▶ Отправляет фишку в тот канал, из которого была получена первая фишка

Алгоритм эха

Код инициатора p :

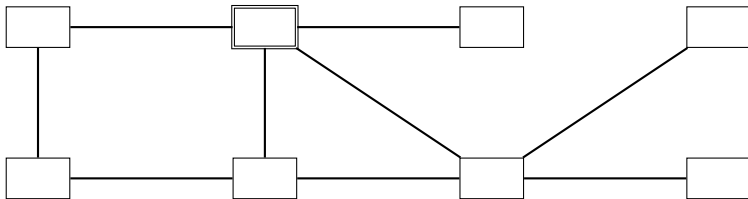
1. Для всех $q \in Neigh_p$: $send(\mathbf{tok}) \rightarrow q$
2. $N_p := 0$;
3. Пока $N_p < |Neigh_p|$:
 - 3.1 $receive(\mathbf{tok})$ (от любого соседа)
 - 3.2 $N_p := N_p + 1$;
4. $decide$

Код последователя p :

1. $receive(\mathbf{tok}) \leftarrow q$ для какого-либо $q \in Neigh_p$
2. Для всех $r \in Neigh_p \setminus \{q\}$: $send(\mathbf{tok}) \rightarrow r$
3. $N_p := 0$;
4. Пока $N_p < |Neigh_p| - 1$:
 - 4.1 $receive(\mathbf{tok})$ (от любого соседа)
 - 4.2 $N_p := N_p + 1$;
5. $send(\mathbf{tok}) \rightarrow q$

Алгоритм эха

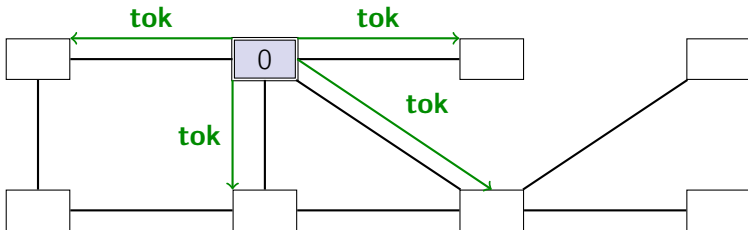
Пример выполнения алгоритма эха



Утверждение (Задача 3). Для любого связного графа топологии с хотя бы двумя узлами алгоритм эха является волновым

Алгоритм эха

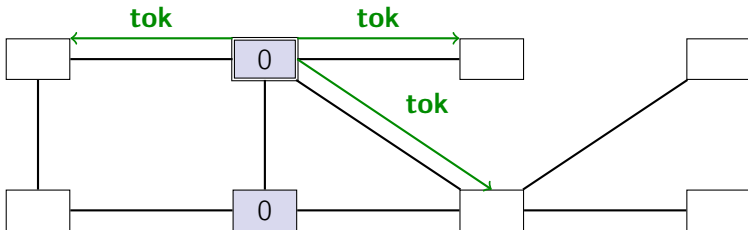
Пример выполнения алгоритма эха



Утверждение (Задача 3). Для любого связного графа топологии с хотя бы двумя узлами алгоритм эха является волновым

Алгоритм эха

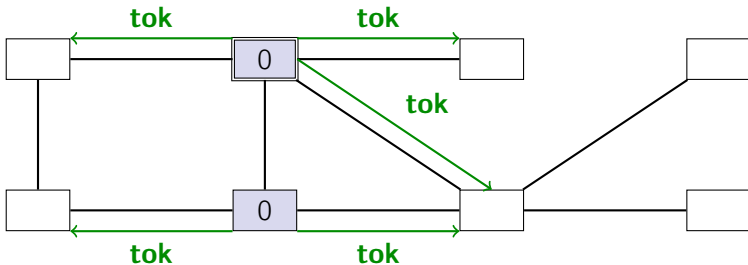
Пример выполнения алгоритма эха



Утверждение (Задача 3). Для любого связного графа топологии с хотя бы двумя узлами алгоритм эха является волновым

Алгоритм эха

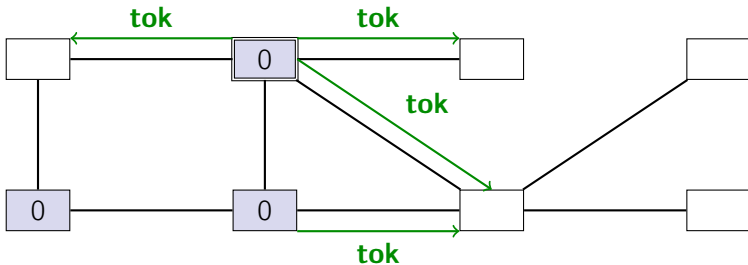
Пример выполнения алгоритма эха



Утверждение (Задача 3). Для любого связного графа топологии с хотя бы двумя узлами алгоритм эха является волновым

Алгоритм эха

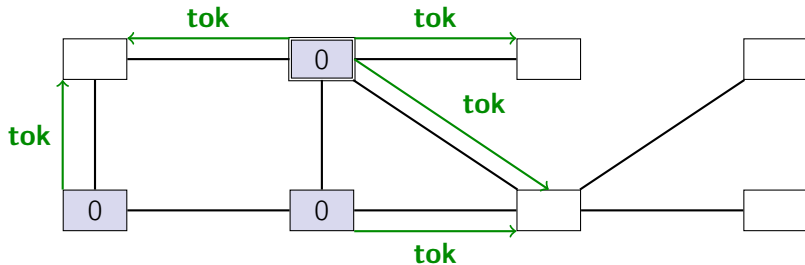
Пример выполнения алгоритма эха



Утверждение (Задача 3). Для любого связного графа топологии с хотя бы двумя узлами алгоритм эха является волновым

Алгоритм эха

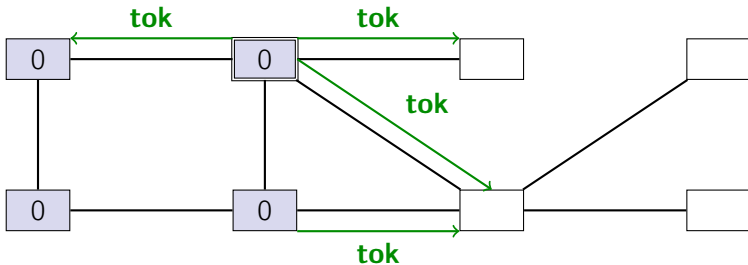
Пример выполнения алгоритма эха



Утверждение (Задача 3). Для любого связного графа топологии с хотя бы двумя узлами алгоритм эха является волновым

Алгоритм эха

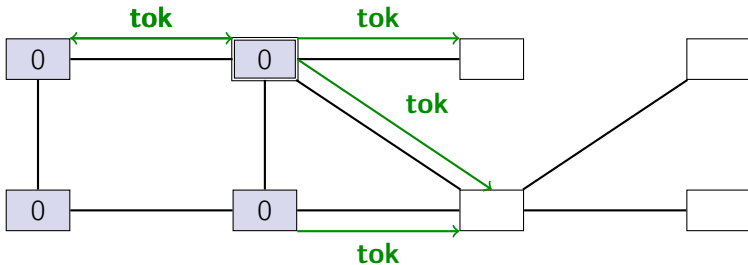
Пример выполнения алгоритма эха



Утверждение (Задача 3). Для любого связного графа топологии с хотя бы двумя узлами алгоритм эха является волновым

Алгоритм эха

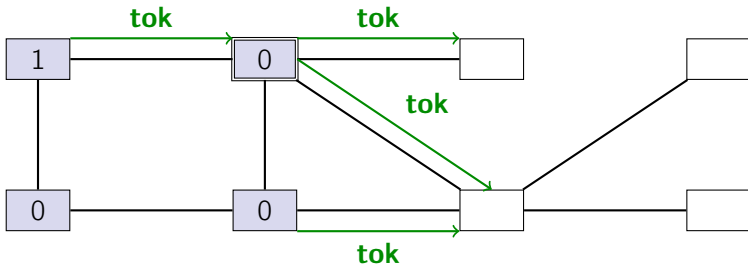
Пример выполнения алгоритма эха



Утверждение (Задача 3). Для любого связного графа топологии с хотя бы двумя узлами алгоритм эха является волновым

Алгоритм эха

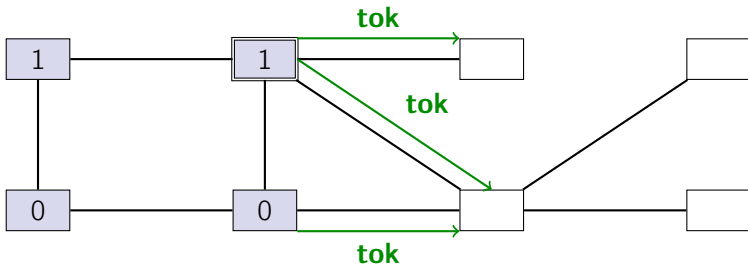
Пример выполнения алгоритма эха



Утверждение (Задача 3). Для любого связного графа топологии с хотя бы двумя узлами алгоритм эха является волновым

Алгоритм эха

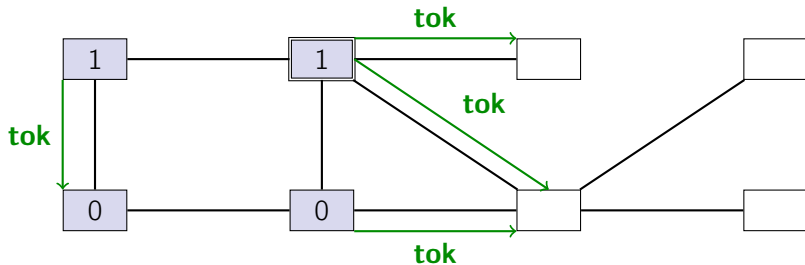
Пример выполнения алгоритма эха



Утверждение (Задача 3). Для любого связного графа топологии с хотя бы двумя узлами алгоритм эха является волновым

Алгоритм эха

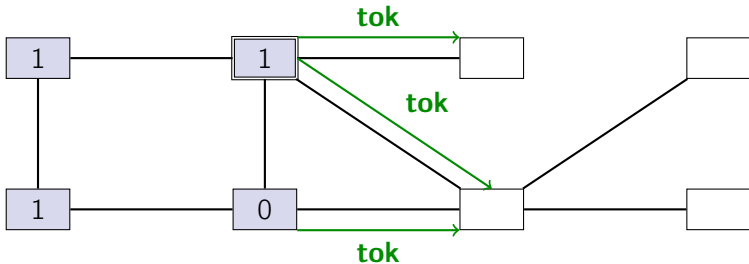
Пример выполнения алгоритма эха



Утверждение (Задача 3). Для любого связного графа топологии с хотя бы двумя узлами алгоритм эха является волновым

Алгоритм эха

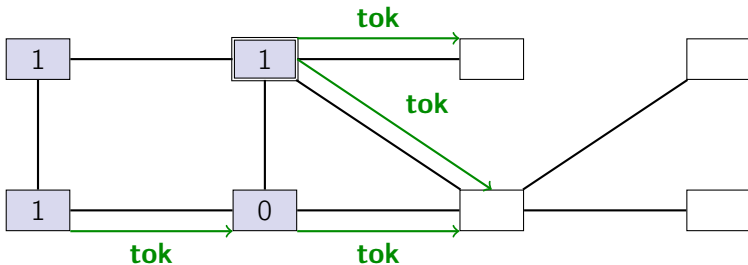
Пример выполнения алгоритма эха



Утверждение (Задача 3). Для любого связного графа топологии с хотя бы двумя узлами алгоритм эха является волновым

Алгоритм эха

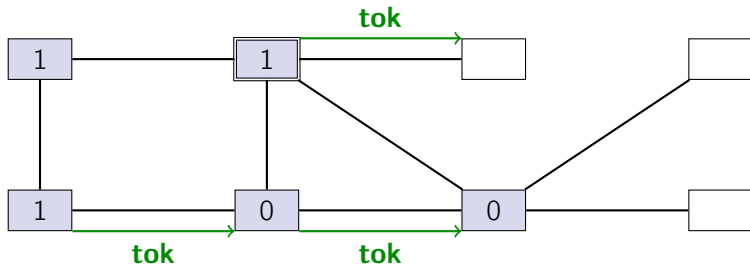
Пример выполнения алгоритма эха



Утверждение (Задача 3). Для любого связного графа топологии с хотя бы двумя узлами алгоритм эха является волновым

Алгоритм эха

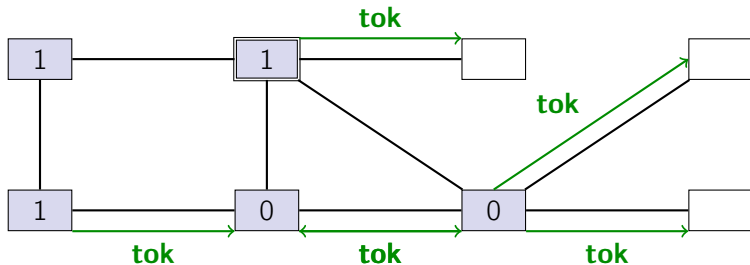
Пример выполнения алгоритма эха



Утверждение (Задача 3). Для любого связного графа топологии с хотя бы двумя узлами алгоритм эха является волновым

Алгоритм эха

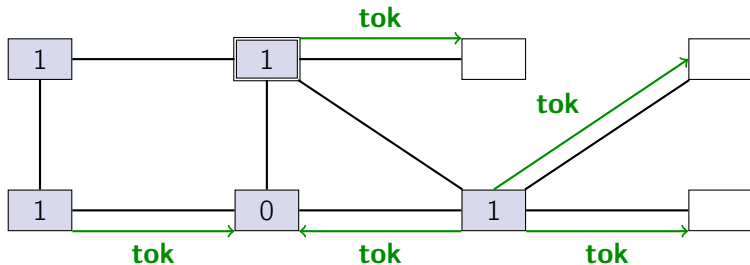
Пример выполнения алгоритма эха



Утверждение (Задача 3). Для любого связного графа топологии с хотя бы двумя узлами алгоритм эха является волновым

Алгоритм эха

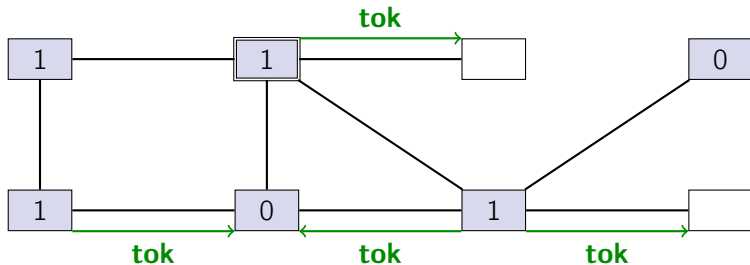
Пример выполнения алгоритма эха



Утверждение (Задача 3). Для любого связного графа топологии с хотя бы двумя узлами алгоритм эха является волновым

Алгоритм эха

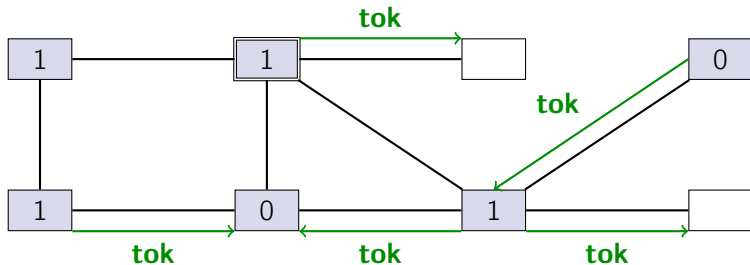
Пример выполнения алгоритма эха



Утверждение (Задача 3). Для любого связного графа топологии с хотя бы двумя узлами алгоритм эха является волновым

Алгоритм эха

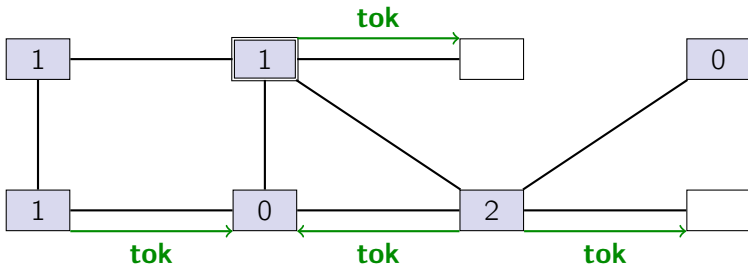
Пример выполнения алгоритма эха



Утверждение (Задача 3). Для любого связного графа топологии с хотя бы двумя узлами алгоритм эха является волновым

Алгоритм эха

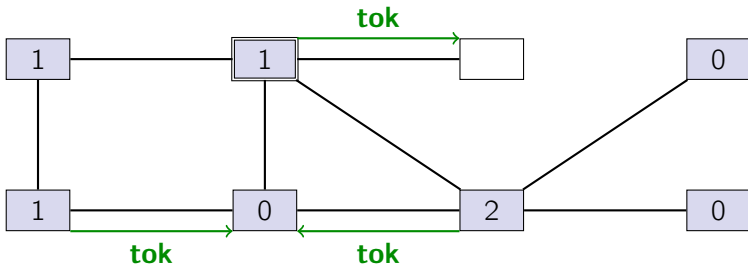
Пример выполнения алгоритма эха



Утверждение (Задача 3). Для любого связного графа топологии с хотя бы двумя узлами алгоритм эха является волновым

Алгоритм эха

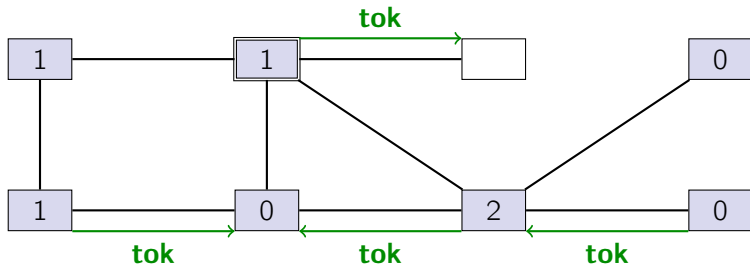
Пример выполнения алгоритма эха



Утверждение (Задача 3). Для любого связного графа топологии с хотя бы двумя узлами алгоритм эха является волновым

Алгоритм эха

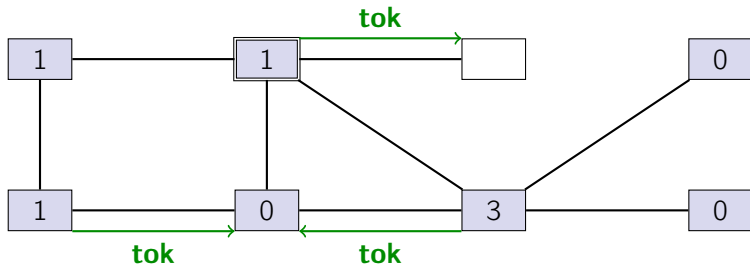
Пример выполнения алгоритма эха



Утверждение (Задача 3). Для любого связного графа топологии с хотя бы двумя узлами алгоритм эха является волновым

Алгоритм эха

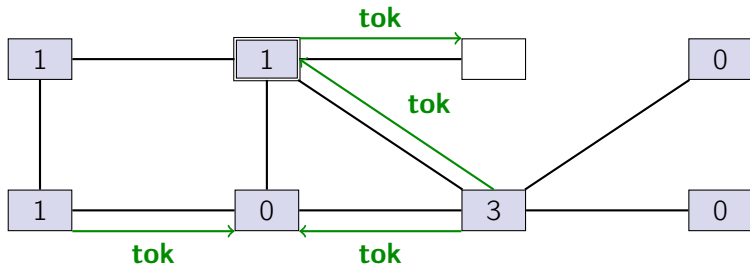
Пример выполнения алгоритма эха



Утверждение (Задача 3). Для любого связного графа топологии с хотя бы двумя узлами алгоритм эха является волновым

Алгоритм эха

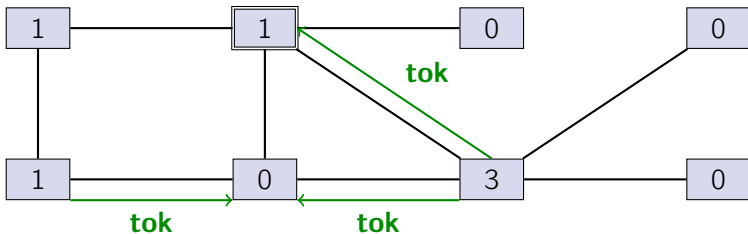
Пример выполнения алгоритма эха



Утверждение (Задача 3). Для любого связного графа топологии с хотя бы двумя узлами алгоритм эха является волновым

Алгоритм эха

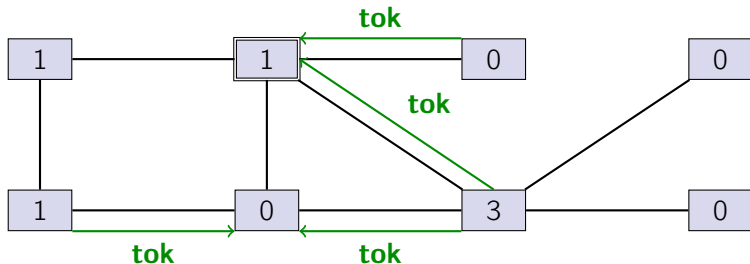
Пример выполнения алгоритма эха



Утверждение (Задача 3). Для любого связного графа топологии с хотя бы двумя узлами алгоритм эха является волновым

Алгоритм эха

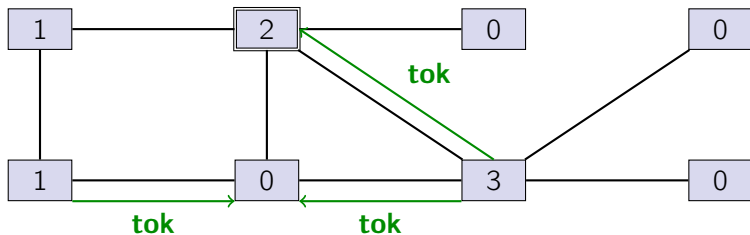
Пример выполнения алгоритма эха



Утверждение (Задача 3). Для любого связного графа топологии с хотя бы двумя узлами алгоритм эха является волновым

Алгоритм эха

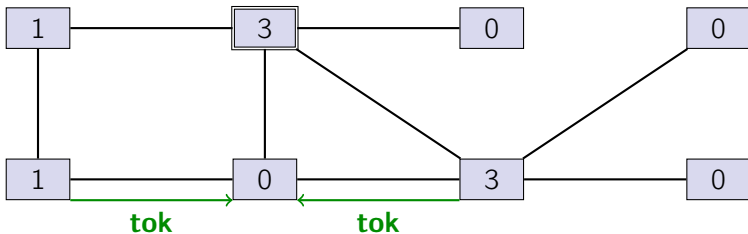
Пример выполнения алгоритма эха



Утверждение (Задача 3). Для любого связного графа топологии с хотя бы двумя узлами алгоритм эха является волновым

Алгоритм эха

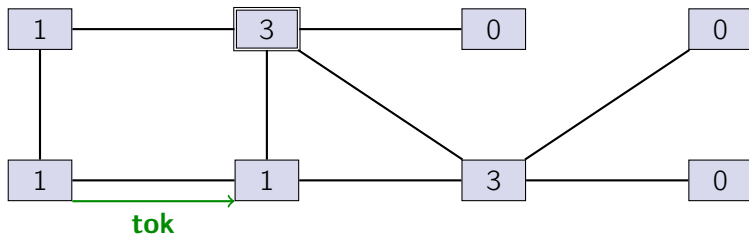
Пример выполнения алгоритма эха



Утверждение (Задача 3). Для любого связного графа топологии с хотя бы двумя узлами алгоритм эха является волновым

Алгоритм эха

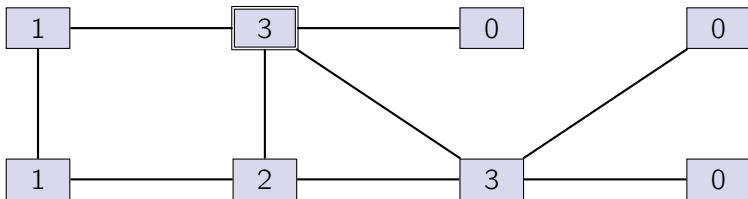
Пример выполнения алгоритма эха



Утверждение (Задача 3). Для любого связного графа топологии с хотя бы двумя узлами алгоритм эха является волновым

Алгоритм эха

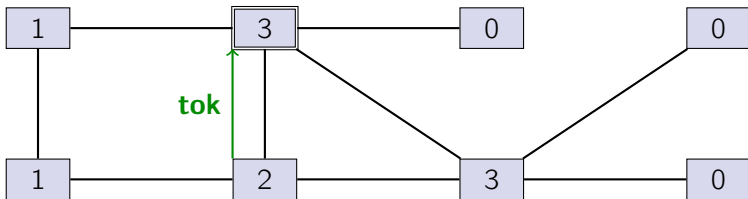
Пример выполнения алгоритма эха



Утверждение (Задача 3). Для любого связного графа топологии с хотя бы двумя узлами алгоритм эха является волновым

Алгоритм эха

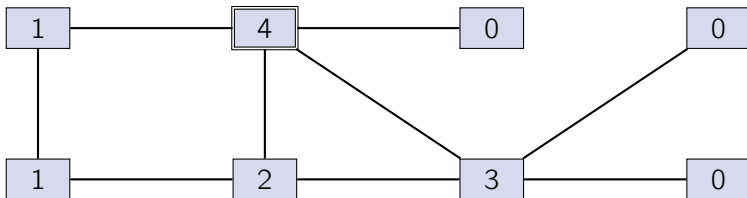
Пример выполнения алгоритма эха



Утверждение (Задача 3). Для любого связного графа топологии с хотя бы двумя узлами алгоритм эха является волновым

Алгоритм эха

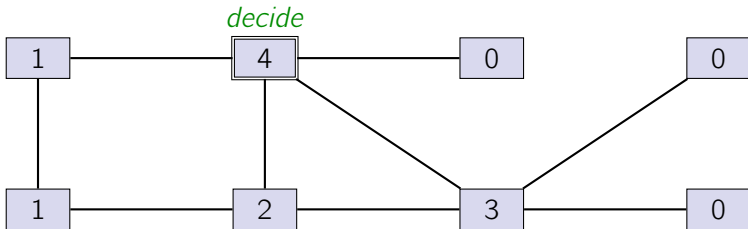
Пример выполнения алгоритма эха



Утверждение (Задача 3). Для любого связного графа топологии с хотя бы двумя узлами алгоритм эха является волновым

Алгоритм эха

Пример выполнения алгоритма эха



Утверждение (Задача 3). Для любого связного графа топологии с хотя бы двумя узлами алгоритм эха является волновым