Распределённые алгоритмы

mk.cs.msu.ru → Лекционные курсы → Распределённые алгоритмы

Блок 29

Алгоритм Тарри

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

ВМК МГУ, 2023/2024, весенний семестр

Алгоритм Тарри (далее — $\mathfrak{T}$) — это РАО, предназначенный для произвольного неориентированного связного графа топологии, не использующий сообщения кроме фишек и устроенный произвольно в рамках следующих ограничений (правил):

O1. Фишку нельзя отправить одному соседу дважды

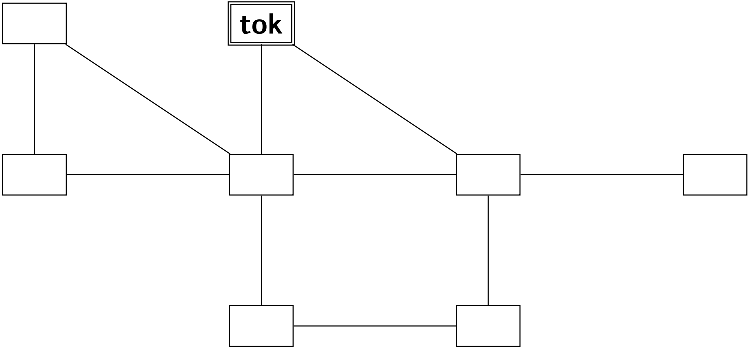
O2. Последователь не может отправить фишку тому, от кого её принял первым действием, пока не отправит её остальным соседям

O3. Узел принимает решение тогда и только тогда, когда не может отправить фишку соседу

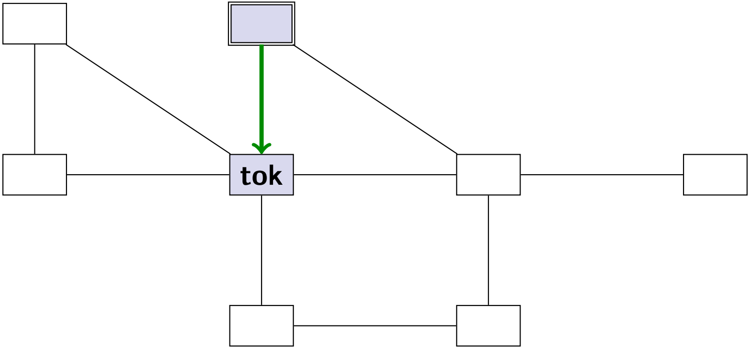
Узел, от которого p принял фишку самым первым своим действием, будем называть родителем узла p (у инициатора нет родителя)

Согласно одному из утверждений в блоке 24, если признать, что алгоритм Тарри — волновой, то граф, составленный из всех вершин и дуг (p, q), где p — последователь и q — его родитель, является остовным деревом сети

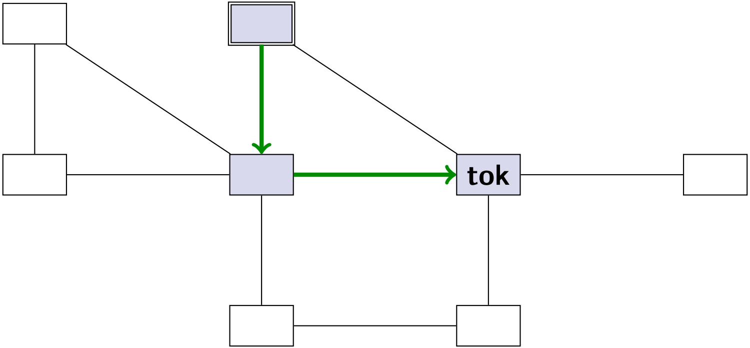
Пример выполнения $\mathfrak{T}$



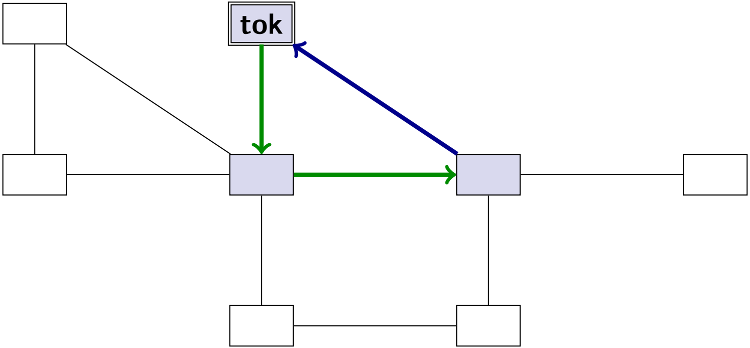
Пример выполнения $\mathfrak{T}$



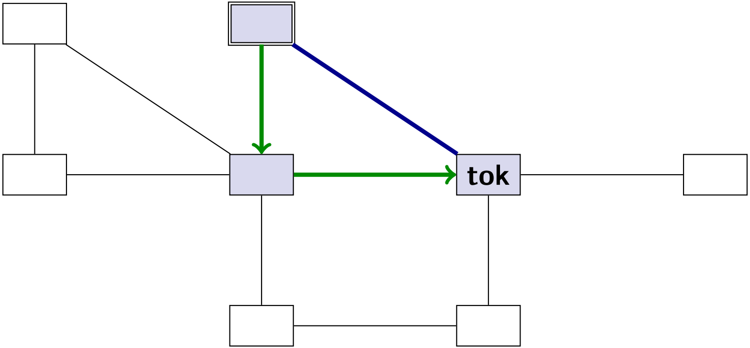
Пример выполнения $\mathfrak{T}$



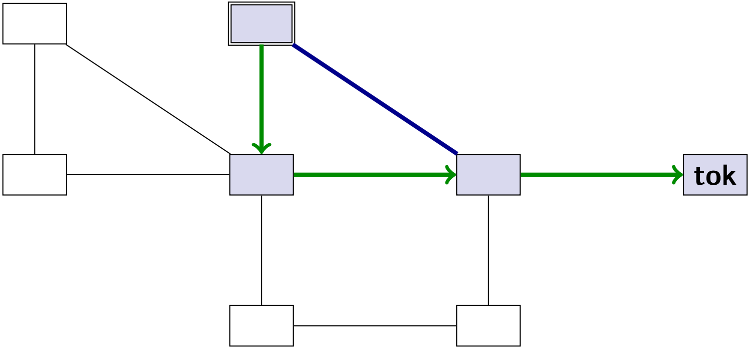
Пример выполнения $\mathfrak{T}$



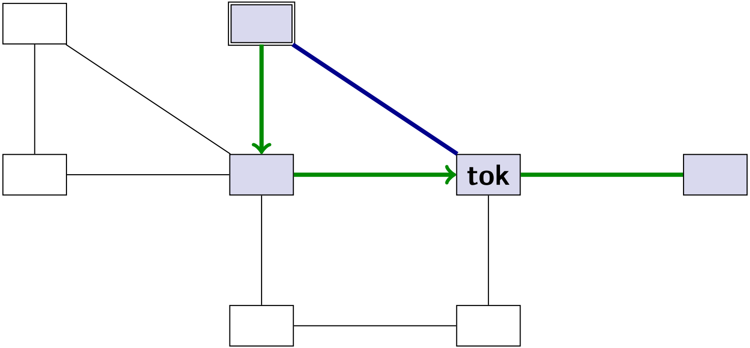
Пример выполнения $\mathfrak{T}$



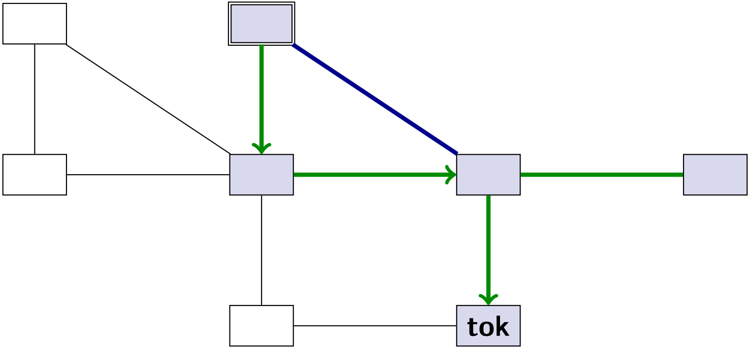
Пример выполнения $\mathfrak{T}$



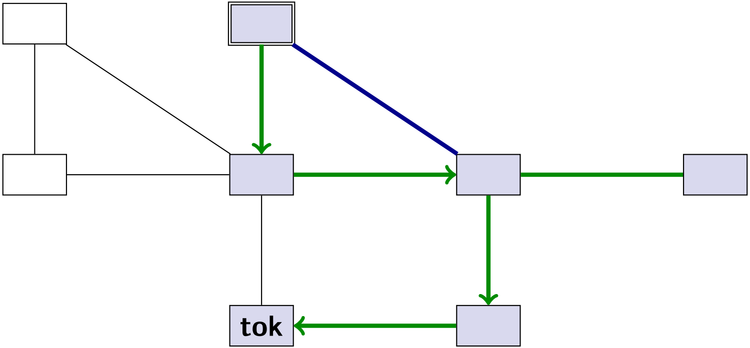
Пример выполнения $\mathfrak{T}$



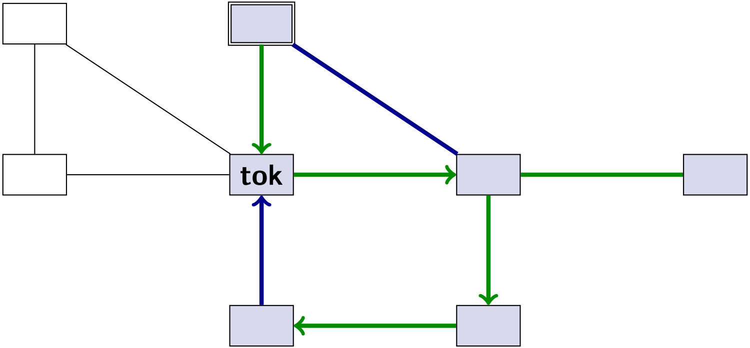
Пример выполнения $\mathfrak{T}$



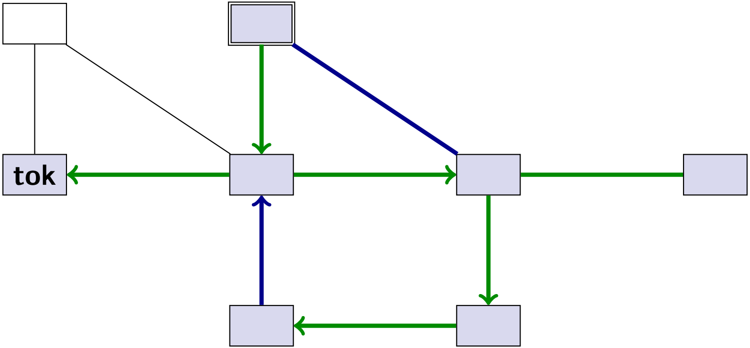
Пример выполнения $\mathfrak{T}$



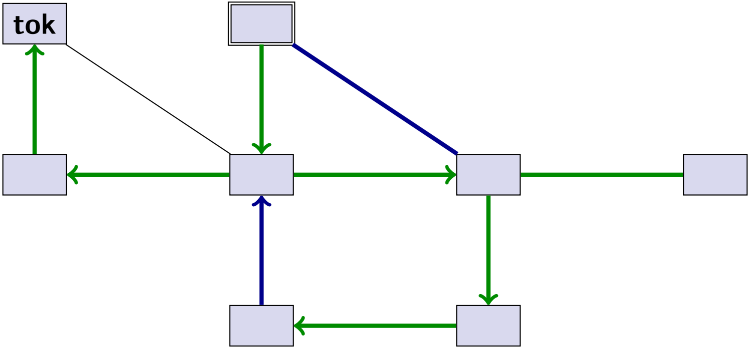
Пример выполнения $\mathfrak{T}$



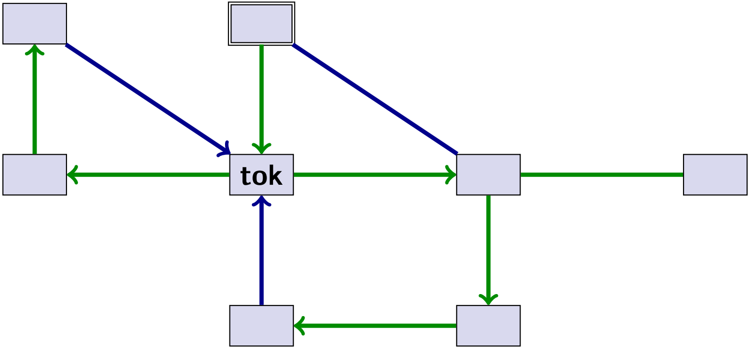
Пример выполнения $\mathfrak{T}$



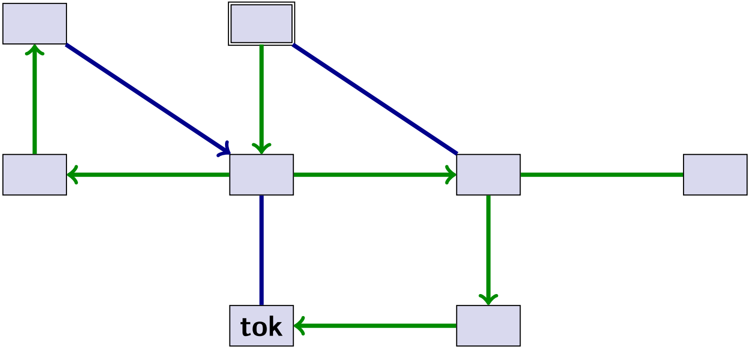
Пример выполнения $\mathfrak{T}$



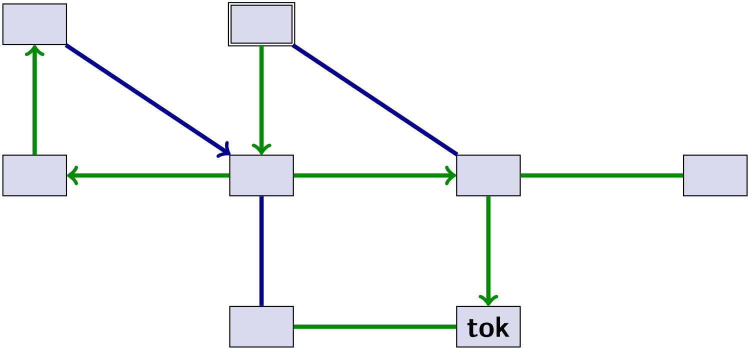
Пример выполнения $\mathfrak{T}$



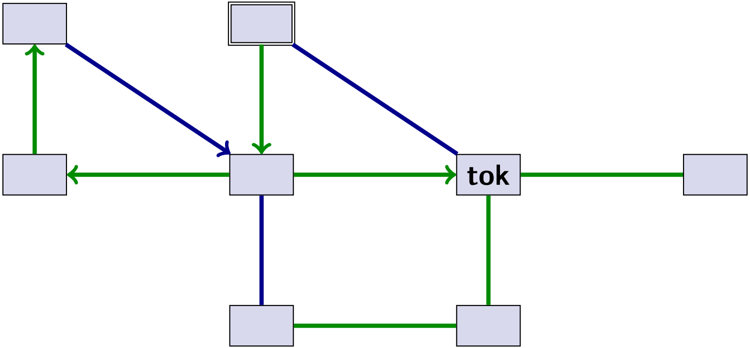
Пример выполнения $\mathfrak{T}$



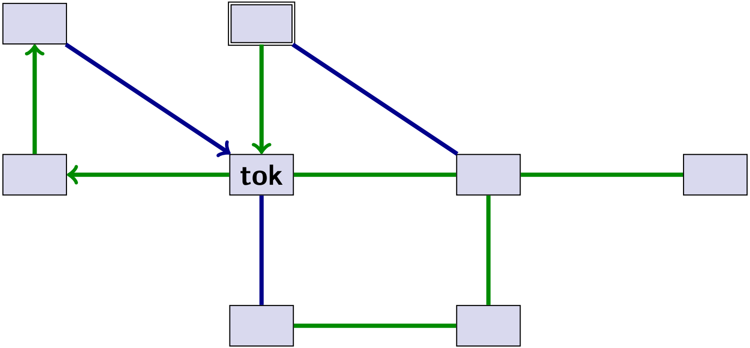
Пример выполнения $\mathfrak{T}$



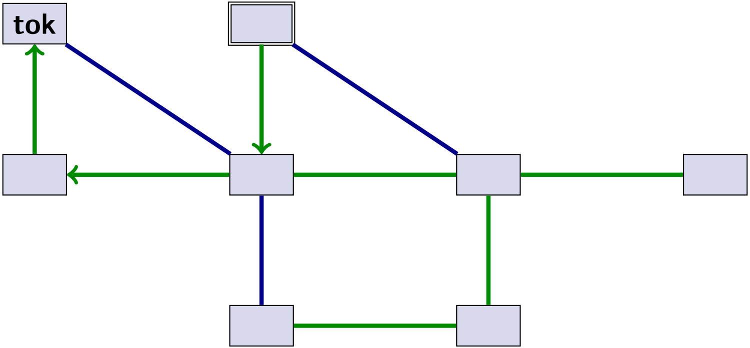
Пример выполнения $\mathfrak{T}$



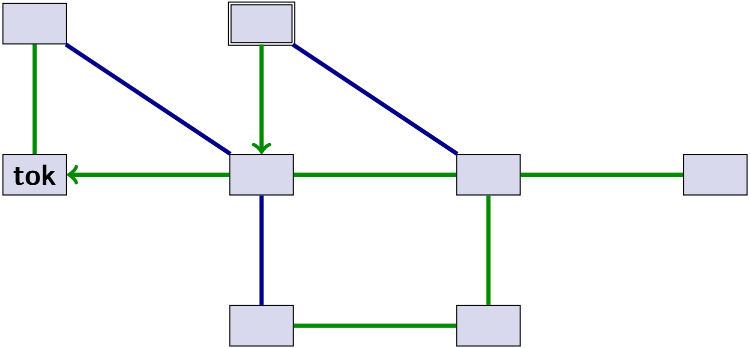
Пример выполнения $\mathfrak{T}$



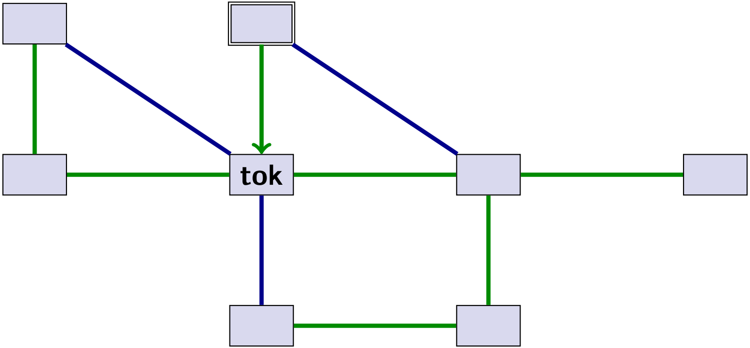
Пример выполнения $\mathfrak{T}$



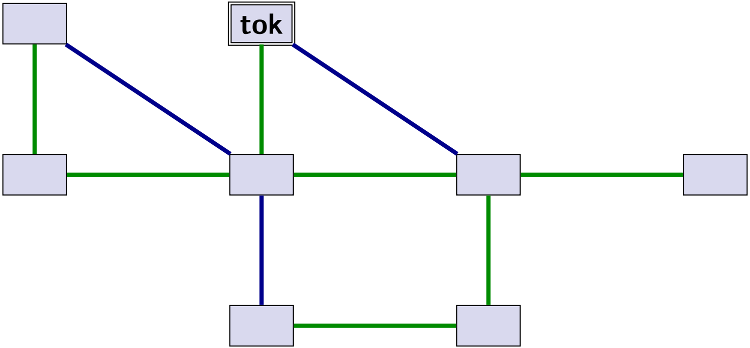
Пример выполнения $\mathfrak{T}$



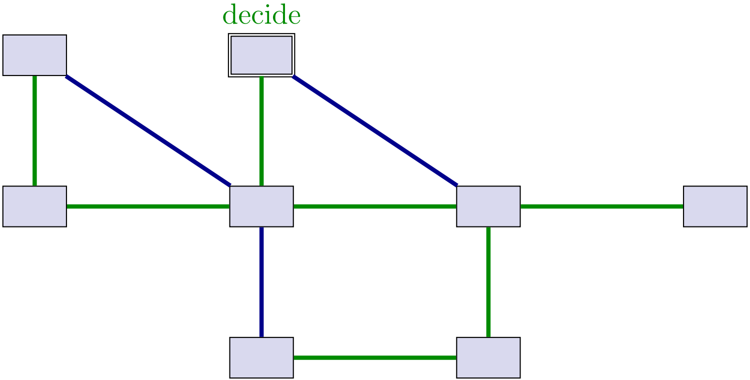
Пример выполнения $\mathfrak{T}$



Пример выполнения $\mathfrak{T}$



Пример выполнения $\mathfrak{T}$



Переменные в узле p:

- $used_p[q] : bool = \mathbb{f}$; для каждого $q \in Neigh_p$
- $parent_p : Neigh_p \cup \{\perp\} = \perp$

Процедура Get_p приёма и обработки сообщения узлом p:

1. $\mathrm{receive}_{q_0}(\underline{\textbf{tok}})$ для произвольного $q_0 \in Neigh_p$
2. Если $parent_p = \perp$, то: $parent_p := q_0$;
3. Если $\forall q \in Neigh_p : used_p[q]$, то: $decide$
4. Иначе (относительно (3)), если $\exists q \in Neigh_p \setminus \{parent_p\} : \neg used_p[q]$, то:
   - 4.1 Произвольно выбрать q такого вида, как написно в (4)
   - 4.2 $used_p[q] := \mathbb{t}$;
   - 4.3 $\mathrm{send}_q(\textbf{tok})$
5. Иначе (относительно (3) и (4)):
   - 5.1 $used_p[parent_p] := \mathbb{t}$;
   - 5.2 $\mathrm{send}_{parent_p}(\textbf{tok})$

Далее (4.1)–(4.3) считаются одним действием (4.X), а (5.1)–(5.2) — одним действием (5.X)

Код последователя p:

1. В бесконечном цикле:
   1.1 Get_p

Код инициатора p:

1. $parent_p := p$;
2. Произвольно выбрать $q \in Neigh_p$
3. $used_p[q] := \mathbb{t}$;
4. $\mathrm{send}_q(\mathbf{tok})$
5. В бесконечном цикле:
   5.1 Get_p

Утверждение (смысл $used_p$). В любой конфигурации γ любого вычисления $\mathcal{E}$ р.с. р.а. $\mathfrak{T}$ верно:

$$used_p[q] = \mathbb{t} \text{ в } \gamma$$
$$\Leftrightarrow$$
в начале E до γ узел p отправил хотя бы одну фишку узлу q

Утверждение (уникальность отправки). В любом вычислении р.с. р.а. $\mathfrak{T}$ для любых узла p и его соседа q узел p

- отправляет фишку узлу q не более одного раза и
- принимает фишку от узла q не более одного раза

Утверждение (завершаемость). Все вычисления р.с. р.а. $\mathfrak{T}$ конечны

Утверждение (баланс фишек). В любой конфигурации любого вычисления р.с. р.а. $\mathfrak{T}$ для любого узла p непосредственно перед началом и непосредственно после завершения выполнения действий Get_p для количества фишек, отправленных узлом p (S_p) и принятых им (R_p), верно следующее:

- Если p — последователь, то $S_p = R_p$
- Если p — инициатор, не принявший решение, то $S_p = R_p + 1$

Утверждение (возможность переотправки последователем). В любом вычислении р.с. р.а. $\mathfrak{T}$ любой последователь p непосредственно после выполнения приёма (1) в Get_p имеет значение $used_p[q] = \mathbb{f}$ хотя бы для одного соседа q

Утверждение (принятие решения инициатором). В любом вычислении р.с. р.а. $\mathfrak{T}$ рано или поздно принимается решение, и оно принимается только инициатором

Утверждение (покрытие соседей). **В любом вычислении р.с. р.а. $\mathfrak{T}$ ко времени принятия решения инициатор и каждый последователь, принимающий хотя бы одну фишку,**

- **отправляет каждому соседу ровно одну фишку и**
- **принимает от каждого соседа ровно одну фишку**

Утверждение (вовлечение всех узлов). **В любом вычислении р.с. р.а. $\mathfrak{T}$ ко времени принятия решения каждый последователь принимает хотя бы одну фишку**

Утверждение (линейная причинность). **В любом вычислении р.с. р.а. $\mathfrak{T}$ любое действие, выполняющееся хронологически позже, является следствием любого действия, выполняющегося хронологически раньше**

Утверждение (полнота покрытия). **В любом вычислении р.с. р.а. $\mathfrak{T}$ принятие решения является следствием хотя бы одного действия в каждом узле**

Д.з. 1. Доказать по одному утверждению с каждого из слайдов 6–8

Теорема. **Алгоритм Тарри — это распределённый алгоритм обхода**