Распределенные алгоритмы и системы

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

Блок 33

Избрание лидера:
алгоритм Галладжера-Хамблета-Спиры (GHS)

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

ВМК МГУ, 2022/2023, весенний семестр

Выборы и построение остовного дерева

Алгоритмы избрания лидера тесно связаны с алгоритмами построения остовного дерева

Пусть C_E и C_T — коммуникационные сложности задачи избрания лидера и построения остовного дерева соответственно

Для избрания лидера в произвольной сети можно построить её остовное дерево и избрать лидера в дереве со сложностью $O(|V|)$: $C_E \leq C_T + O(|V|)$

Построить остовное дерево, имея выделенную вершину (лидера), можно с помощью алгоритма эха со сложностью $2|E|$: $C_T \leq C_E + 2|E|$

Известно, что алгоритм избрания лидера в произвольной топологии (V, E) имеет коммуникационную сложность $\Omega(|V| \log |V| + |E|)$

Значит, для избрания лидера с лучшей сложностью $O(|V| \log |V| + |E|)$ необходимо и достаточно научиться строить дерево с этой сложностью

Алгоритм Галладжера-Хамблета-Спиры (GHS) — это распределённый алгоритм построения остовного дерева, имеющий коммуникационную сложность $O(|V| \log |V| + |E|)$ в худшем случае

Допущения

В GHS используются следующие допущения и соглашения:

1. $\Gamma = (V, E)$ — неориентированный связный граф топологии

2. Каждому ребру e из E приписан вес $\omega(e)$ — число ($\mathbb{N}_0$, $\mathbb{Z}$ или $\mathbb{R}$)

   Разным рёбрам приписаны разные веса

3. Каждый узел p знает веса инцидентных ему каналов и множество смежных узлов ($Neigh_p$)

4. Все узлы являются инициаторами

   То есть системы переходов узлов одинаковы (*с поправкой на начальные знания*), и все узлы начинают выполнять действия, не дожидаясь приёма сообщения

Минимальные остовные деревья

Весом графа назовём сумму весов рёбер этого графа

Минимальное остовное дерево (MST) — это остовное дерево, имеющее наименьший вес среди всех всех остовных деревьев

Утверждение 1. **Если все веса рёбер графа попарно различны, то существует единственное MST этого графа**

Доказательство.

Предположим от противного, что существуют два различных MST: $T_1 = (V, E_1)$ и $T_2 = (V, E_2)$

Рассмотрим ребро e наименьшего веса, содержащееся в одном из этих деревьев и не содержащееся в другом

Без ограничения общности положим, что $e \in E_1$ и $e \notin E_2$

Тогда в графе $T_2' = (V, E_2 \cup \{e\})$ содержится цикл $\mathcal{C}$

Так как в T_1 нет циклов, то в $\mathcal{C}$ содержится ребро e', не содержащееся в T_1

По выбору ребра e, верно $\omega(e) < \omega(e')$

Значит, $(V, (E_2 \setminus \{e'\}) \cup \{e\})$ — дерево, имеющее меньший вес, чем T_2 (*противоречие*) ▼

Минимальные остовные деревья

Фрагментами будем называть поддеревья MST

Ребро e будем называть внутренним ребром фрагмента F, если обе его вершины принадлежат F, внешним, если обе не принадлежат, и граничным, если она принадлежит и одна не принадлежит

Минимальным граничным ребром фрагмента будем называть граничное ребро этого фрагмента, имеющее наименьший вес среди всех граничных рёбер этого фрагмента

Утверждение 2 (Задача 1). Если $F = (V_F, E_F)$ — фрагмент и $e = (v, w)$ — минимальное граничное ребро фрагмента F, то $(V_F \cup \{v, w\}, E_F \cup \{e\})$ — фрагмент

GHS: общее описание

Каждый узел p на каждом этапе выполнения алгоритма принадлежит некоторому фрагменту, вычисленному алгоритмом (текущему фрагменту)

В начале выполнения узел p принадлежит фрагменту $(\{p\}, \emptyset)$

Непересекающиеся фрагменты (V_1, E_1) и (V_2, E_2) при выполнении алгоритма могут быть объединены по ребру e, граничному для этих фрагментов, и результат такого объединения — фрагмент $(V_1 \cup V_2, E_1 \cup E_2 \cup \{e\})$

Каждому фрагменту присваивается особое число — ранг

Фрагмент большего ранга (по сравнению с заданным) будем называть старшим, меньшего — младшим, равного — ровесником

Фрагмент вида $(\{p\}, \emptyset)$ имеет ранг 0

Ранг объединения фрагментов F_1 и F_2 с рангами r_1 и r_2 равен

- рангу старшего фрагмента, если $r_1 \neq r_2$
- $r_1 + 1$, если $r_1 = r_2$

GHS: общее описание

Каждому фрагменту с хотя бы одним ребром присваивается имя

Каждый узел знает имя своего текущего фрагмента

Объединению фрагментов разных рангов присваивается имя старшего фрагмента

Имя объединения ровесников — это вес соединяющего их ребра

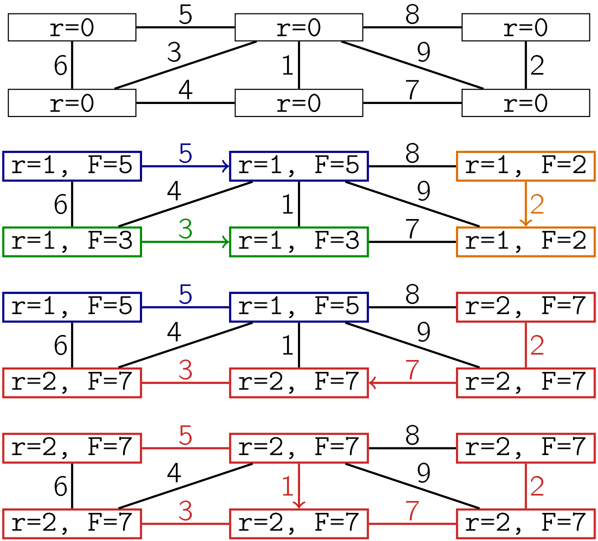
Ребро, вес которого является именем фрагмента, будем называть стержнем, и узлы стержня будем называть стержневыми

Так как веса всех рёбер попарно различны, то имена различных фрагментов различны, и проверка граничности ребра равносильна проверке неравенства имён узлов этого ребра

После объединения фрагментов ранг и имя рассылаются всем узлам фрагмента в направлении от стержня

GHS: общее описание

Пример построения MST из фрагментов (`r` — ранг, `F` — имя):



GHS: общее описание

Утверждение 3. **В любом фрагменте любого ранга $r \in \mathbb{N}_0$ содержится не менее 2^r узлов**

Доказательство (индукцией по способам объединения фрагментов).

В каждом фрагменте ранга 0 содержится ровно $2^0 = 1$ узел

Если объединяются фрагменты разных рангов, старший F имеет ранг r и содержит не менее 2^r узлов, то ранг объединения равен r, и в объединении содержатся по крайней мере все узлы F, и их не менее 2^r

Если объединяются фрагменты одного ранга r и каждый содержит не менее 2^r узлов, то объединение имеет ранг $(r+1)$ и содержит не менее $(2^r + 2^r) = 2^{r+1}$ узел ▼

GHS: общее описание

Утверждение 4. При описанном построении MST значение ранга в каждом узле изменяется не более $\log_2|V|$ раз

Доказательство.

По утверждению 3, значение ранга фрагмента не может быть больше $\log_2|V|$

При изменении ранга в узле он увеличивается хотя бы на 1

Значит, изменение может произойти не более $\log_2|V|$ раз ▼

Следствие. Суммарное число изменений рангов в узлах сети не превосходит $|V|\log_2|V|$

GHS: правила присоединения фрагмента

Объединение фрагментов в GHS носит «полуодносторонний» характер: либо младший фрагмент присоединяется к старшему фрагменту, либо каждый из ровесников присоединяется к другому ровеснику

Такое присоединение фрагмента F с именем $name_F$ и рангом r_F согласно наименьшему граничному ребру e_F фрагмента F подчиняется двум правилам

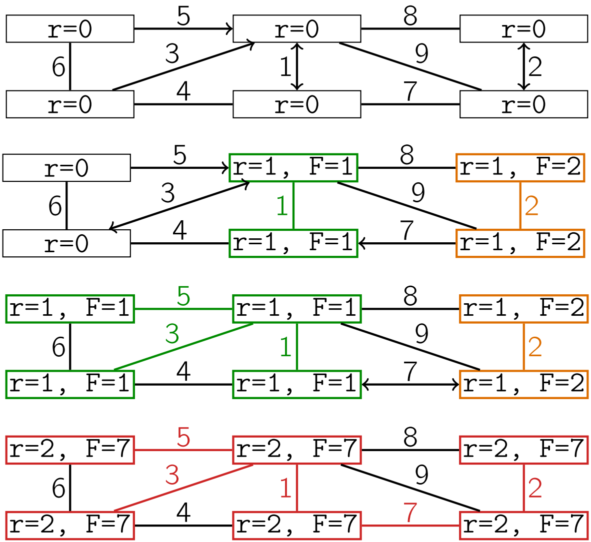
Правило A. Если e_F соединяет F с фрагментом F' большего ранга, то узлы F присоединяются к фрагменту F': изменяют свои имя и ранг на имя и ранг F'

Правило B. Если e_F соединяет F с фрагментом F' того же ранга и является минимальным граничным ребром фрагмента F', то F присоединяется к фрагменту F' (и F' присоединяется к фрагменту F)

В остальных случаях F ожидает возможности применения правила A или B

GHS: правила присоединения фрагмента

Пример объединения фрагментов в GHS по правилам A, B



GHS: переменные узла p

- $state_p : \{find, found\} = found$ — режим работы узла
  - *find* означает поиск минимального граничного ребра
  - *found* означает, что минимальное граничное ребро найдено
- $ch\_stat_p[q] : \{basic, branch, reject\} = basic$ для каждого $q \in Neigh_p$ — статус канала связи
  - *branch* означает, что канал $p - q$ входит в MST
  - *reject* означает, что канал не входит в MST
  - *basic* означает, что принадлежность канала MST неясна
- $name_p = \perp$ — имя текущего фрагмента
- $best\omega_p = \perp$ — вес наилучшего известного узлу граничного ребра
- $rank_p = 0$ — ранг узла
- $parent_p = \perp$ — сосед в направлении стержня фрагмента (родитель)
- $test\_ch_p = \perp, best\_ch_p = \perp$ — вероятные направления к минимальному граничному ребру
- $N\_ch_p = 0$ — счётчик исследованных каналов

GHS: виды сообщений

- (**connect**, *rank*) — отправляется в минимальное граничное ребро для обозначения готовности фрагмента присоединиться
- (**initiate**, (*rank*, *name*, *state*)) — принимается по минимальному граничному ребру для присоединения и пересылается узлам фрагмента
- (**test**, (*rank*, *name*)) — отправляется в ребро типа *basic* наименьшего веса, инцидентному узлу, для поиска минимального граничного ребра
- **reject** — отправляется в ответ на **test** во внутреннее ребро
- **accept** — отправляется в ответ на **test** в граничное ребро
- (**report**, *bestω*) — отправляется по направлению к стержню с наименьшим обнаруженным весом граничного ребра
- **changeroot** — принимается из канала, показывающего направление к стержню

GHS: инициализация узла p

Первое однократно выполняющееся действие каждого узла:

1. $q = \operatorname*{Argmin}_{q \in Neigh_p} \omega_{(p,q)}$
2. $ch\_stat_p[q] := branch;$
3. $send(\textbf{connect}, 0) \to q$

Узел в тривиальном фрагменте ранга 0 очевидным образом находит минимальное граничное ребро и объявляет о готовности присоединиться по этому ребру

GHS: приём connect узлом p

Предусловие: в канале хотя бы от одного соседа q есть сообщение $(\textbf{connect}, rank)$, где $rank \leq rank_p$, и если $rank = rank_p$, то $ch\_stat_p[q] \neq basic$ (*т.е. можно применить правило A или B*)

1. $receive(\textbf{connect}, rank) \leftarrow q$ для сообщения и соседа, упомянутых в предусловии
2. Если $rank < rank_p$ (*правило A*):
   2.1 $ch\_stat_p[q] := branch$;
   2.2 $send(\textbf{initiate}, (rank_p, name_p, state_p)) \rightarrow q$
3. Иначе (*правило B*): $send(\textbf{initiate}, (rank_p + 1, \omega_{(p,q)}, find))$

Узлу q отправляется приказ присоединиться, если он

- младший или
- ровесник, и при этом узел p уверен, что (p, q) — минимальное граничное ребро его текущего фрагмента

Вопрос: а что происходит с сообщением, если у него бóльший ранг, либо равный, но канал не типа basic?

GHS: приём initiate узлом p

Предусловие: в канале хотя бы от одного соседа q есть сообщение типа **initiate**

1. $receive(\textbf{initiate}, (rank, name, state)) \leftarrow q$ для любого $q \in Neigh_p$
2. $(rank_p, name_p, state_p) := (rank, name, state)$;
3. $parent_p := q$;
4. $(best\omega_p, best\_ch_p) := (\infty, \perp)$;
5. Для всех $r \in Neigh_p$, таких что $r \neq q$ и $ch\_stat_p[r] = branch$: $send(\textbf{initiate}, (rank, name, state))$
6. Если $state_p = find$:
   - 6.1 $N\_ch_p := 0$;
   - 6.2 $TEST_p$

После получения приказа о присоединении узел

- обновляет информацию о фрагменте (ранг, имя, статус поиска минимального граничного ребра),

GHS: приём initiate узлом p

Предусловие: в канале хотя бы от одного соседа q есть сообщение типа **initiate**

1. $receive(\mathbf{initiate}, (rank, name, state)) \leftarrow q$ для любого $q \in Neigh_p$
2. $(rank_p, name_p, state_p) := (rank, name, state)$;
3. $parent_p := q$;
4. $(best\omega_p, best\_ch_p) := (\infty, \perp)$;
5. Для всех $r \in Neigh_p$, таких что $r \neq q$ и $ch\_stat_p[r] = branch$: $send(\mathbf{initiate}, (rank, name, state))$
6. Если $state_p = find$:
   - 6.1 $N\_ch_p := 0$;
   - 6.2 $TEST_p$

После получения приказа о присоединении узел

- «сбрасывает» информацию о наилучшем известном граничном ребре,

GHS: приём initiate узлом p

Предусловие: в канале хотя бы от одного соседа q есть сообщение типа **initiate**

1. $receive(\textbf{initiate}, (rank, name, state)) \leftarrow q$ для любого $q \in Neigh_p$
2. $(rank_p, name_p, state_p) := (rank, name, state)$;
3. $parent_p := q$;
4. $(best\omega_p, best\_ch_p) := (\infty, \perp)$;
5. Для всех $r \in Neigh_p$, таких что $r \neq q$ и $ch\_stat_p[r] = branch$: $send(\textbf{initiate}, (rank, name, state))$
6. Если $state_p = find$:
   - 6.1 $N\_ch_p := 0$;
   - 6.2 $TEST_p$

После получения приказа о присоединении узел

- пересылает остальным соседям приказ о присоединении и

GHS: приём initiate узлом p

Предусловие: в канале хотя бы от одного соседа q есть сообщение типа **initiate**

1. $receive(\mathbf{initiate}, (rank, name, state)) \leftarrow q$ для любого $q \in Neigh_p$
2. $(rank_p, name_p, state_p) := (rank, name, state)$;
3. $parent_p := q$;
4. $(best\omega_p, best\_ch_p) := (\infty, \perp)$;
5. Для всех $r \in Neigh_p$, таких что $r \neq q$ и $ch\_stat_p[r] = branch$: $send(\mathbf{initiate}, (rank, name, state))$
6. Если $state_p = find$:
   - 6.1 $N\_ch_p := 0$;
   - 6.2 $TEST_p$

После получения приказа о присоединении узел

- если в приказе говорится продолжить поиск минимального граничного ребра, то присоединяется к поиску при помощи (*описанной далее*) процедуры $TEST_p$

GHS: приём initiate узлом p

Предусловие: в канале хотя бы от одного соседа q есть сообщение типа **initiate**

1. $receive(\textbf{initiate}, (rank, name, state)) \leftarrow q$ для любого $q \in Neigh_p$
2. $(rank_p, name_p, state_p) := (rank, name, state)$;
3. $parent_p := q$;
4. $(best\omega_p, best\_ch_p) := (\infty, \perp)$;
5. Для всех $r \in Neigh_p$, таких что $r \neq q$ и $ch\_stat_p[r] = branch$: $send(\textbf{initiate}, (rank, name, state))$
6. Если $state_p = find$:
   6.1 $N\_ch_p := 0$;
   6.2 $TEST_p$

Вопрос: зачем узел сбрасывает информацию о наилучшем граничном ребре, останется ли алгоритм корректным, если этого не делать, и если нет, то почему?

GHS: приём initiate узлом p

Процедура $TEST_p$:

1. Если $ch\_stat_p[q] = basic$ для какого-либо $q \in Neigh_p$:
   1.1 Выбрать указанный выше q с наименьшим весом ребра (p, q)
   1.2 $test\_ch_p := q$;
   1.3 $send(\mathbf{test}, (rank_p, name_p)) \to q$
2. Иначе:
   2.1 $test\_ch_p := \perp$;
   2.2 $REPORT_P$

Узел p в рамках поиска минимального граничного ребра

- выбирает инцидентное ребро типа *basic* (неопробованное) с наименьшим весом и отправляет в него сообщение **test** для проверки, является ли оно граничным,
- а если таких рёбер нет, то считает поиск завершённым и докладывает о результатах в сторону стержня

GHS: приём initiate узлом p

Процедура $REPORT_p$:

1. Если $N\_ch_p = |\{q \mid q \in Neigh_p,\ ch\_stat_p = branch,\ q \neq parent_p\}|$ и $test\_ch_p \neq \perp$:
   1.1 $state_p := found$;
   1.2 $send($**report**$, best\omega_p) \to parent_p$

Если узел получил от всех соседей из его фрагмента, кроме родителя, мнения о наилучших рёбрах и сам завершил поиск наилучшего ребра, то он отправляет своё мнение о наилучшем ребле в сторону стержня

GHS: приём test узлом p

Предусловие: в канале хотя бы от одного соседа q есть сообщение $(\textbf{test}, (rank, name))$, где $rank \leq rank_p$

1. $receive(\textbf{test}, (rank, name)) \leftarrow q$ для сообщения, указанного в предусловии
2. Если $name \neq name_p$ (*это граничное ребро*): $send(\textbf{accept}) \rightarrow q$
3. Иначе (*это внутреннее ребро*):
   3.1 Если $ch\_stat_p[q] = basic$: $ch\_stat_p[q] := reject$;
   3.2 Если $q \neq test\_ch_p$: $send(\textbf{reject}) \rightarrow q$
   3.3 Иначе: $TEST_P$

Если узел q в фрагменте не большего ранга решил проверить ребро, то

1. если это другой фрагмент, то отправляется сообщение о том, что ребро граничное,
2. а иначе статус ребра изменяется на «внутреннее», и в зависимости от того, проверяет ли сейчас p это ребро, этот узел либо продолжает перебирать инцидентные рёбра (запускает $TEST_p$), либо сообщает q о том, что ребро внутреннее

GHS: приём test узлом p

Предусловие: в канале хотя бы от одного соседа q есть сообщение $(\textbf{test}, (rank, name))$, где $rank \leq rank_p$

1. $receive(\textbf{test}, (rank, name)) \leftarrow q$ для сообщения, указанного в предусловии
2. Если $name \neq name_p$ (*это граничное ребро*): $send(\textbf{accept}) \rightarrow q$
3. Иначе (*это внутреннее ребро*):
   3.1 Если $ch\_stat_p[q] = basic$: $ch\_stat_p[q] := reject$;
   3.2 Если $q \neq test\_ch_p$: $send(\textbf{reject}) \rightarrow q$
   3.3 Иначе: $TEST_P$

$\boxed{\text{Вопрос:}}$ зачем блокировать приём сообщения от старшего фрагмента?

$\boxed{\text{Вопрос:}}$ почему считается правильным в некоторых случаях (3.3) не отправлять «ожидаемый» ответ **reject**, и зачем вместо этого запускать процедуру $TEST_p$?

GHS: приём accept узлом p

Предусловие: в канале хотя бы от одного соседа q есть сообщение **accept**

1. $receive(\textbf{accept}) \leftarrow q$ для любого $q \in Neigh_p$
2. $test\_ch_p := \perp$;
3. Если $\omega_{(p,q)} < best\omega_p$:
   3.1 $best\omega_p := \omega_{(p,q)}$;
   3.2 $best\_ch_p := q$;
4. $REPORT_p$

Если на сообщение **test** в $TEST_p$ получен ответ, что проверяемое ребро действительно является граничным, то это ребро признаётся наилучшим инцидентным, поиск такого ребра завершается, и результат отправляется в сторону стержня

Вопрос: зачем «сбрасывать» значение $test\_ch_p$ — останется ли алгоритм корректным, если удалить этот «сброс», и если нет, то почему?

GHS: приём reject узлом p

Предусловие: в канале хотя бы от одного соседа q есть сообщение **reject**

1. $receive(\mathbf{reject}) \leftarrow q$ для любого $q \in Neigh_p$
2. Если $ch\_stat_p[q] = basic$: $ch\_stat_p[q] := reject$;
3. $TEST_p$

Если на сообщение **test** получен ответ, что проверяемое ребро является внутренним, то статус ребра обновляется и поиск наилучшего инцидентного граничного ребра продолжается

GHS: приём report узлом p

Предусловие: в канале хотя бы от одного соседа q есть сообщение типа **report**, такое что $q \neq parent_p$ или $state_p \neq find$

1. $receive(\mathbf{report}, \omega) \leftarrow q$ для любого $q \in Neigh_p$
2. Если $q \neq parent_p$:
   - 2.1 Если $\omega < best\omega_p$: $(bst\omega_p, best\_ch_p) := (\omega, q)$;
   - 2.2 $N\_ch_p := N\_ch_p + 1$;
   - 2.3 $REPORT_p$
3. Иначе:
   - 3.1 Если $\omega > best\omega_p$: $CHANGEROOT_p$
   - 3.2 Иначе: если $\omega = best\omega_p = \infty$, то завершить выполнение узла

Вопрос: объясните, почему сообщение типа **report** может быть получено только от узла того же фрагмента

Если доклад получен от ребёнка, то следует обновить мнение о лучшем граничном ребре, и если сбор докладов от детей и проверка рёбер завершены, то переслать доклад родителю

GHS: приём report узлом p

Предусловие: в канале хотя бы от одного соседа q есть сообщение типа **report**, такое что $q \neq parent_p$ или $state_p \neq find$

1. $receive(\textbf{report}, \omega) \leftarrow q$ для любого $q \in Neigh_p$
2. Если $q \neq parent_p$:
   2.1 Если $\omega < best\omega_p$: $(bst\omega_p, best\_ch_p) := (\omega, q)$;
   2.2 $N\_ch_p := N\_ch_p + 1$;
   2.3 $REPORT_p$
3. Иначе:
   3.1 Если $\omega > best\omega_p$: $CHANGEROOT_p$
   3.2 Иначе: если $\omega = best\omega_p = \infty$, то завершить выполнение узла

Если же доклад получен от родителя, то это доклад от одного стержневого узла другому (они и только они являются взаимными родителями), и стержневой узел, нашедший ребро лучшего веса, запускает процедуру $CHANGEROOT_p$ отправки сообщения **connect** в ребро с этим весом

GHS: приём report узлом p

Предусловие: в канале хотя бы от одного соседа q есть сообщение типа **report**, такое что $q \neq parent_p$ или $state_p \neq find$

1. $receive(\textbf{report}, \omega) \leftarrow q$ для любого $q \in Neigh_p$
2. Если $q \neq parent_p$:
   2.1 Если $\omega < best\omega_p$: $(bst\omega_p, best\_ch_p) := (\omega, q)$;
   2.2 $N\_ch_p := N\_ch_p + 1$;
   2.3 $REPORT_p$
3. Иначе:
   3.1 Если $\omega > best\omega_p$: $CHANGEROOT_p$
   3.2 Иначе: если $\omega = best\omega_p = \infty$, то завершить выполнение узла

Если оба стержневых узла не нашли ни одного граничного ребра, то процедура построения MST в узле завершается

GHS: приём report узлом p

Процедура $CHANGEROOT_p$:

1. Если $ch\_stat_p[best\_ch_p] = branch$: $send(\textbf{changeroot}) \to best\_ch_p$
2. Иначе:
   2.1 $send(\textbf{connect}, rank_p) \to best\_ch_p$
   2.2 $ch\_stat_p[best\_ch_p] := branch$;

Сообщение **changeroot** доставляется узлу, которому инцидентно найденное минимальное граничное ребро

В это ребро отправляется сообщение **connect** о намерении объединиться с фрагментом на другом конце ребра

GHS: приём changeroot в узле p

Предусловие: в канале хотя бы от одного соседа q есть сообщение **changeroot**

1. $\mathit{receive}(\textbf{changeroot}) \leftarrow q$ для любого $q \in \mathit{Neigh}_p$
2. $\mathit{CHANGEROOT}_p$

Задача 2. Ответьте хотя бы на три поставленных ранее вопроса

GHS: свойства

Теорема. Вычисление GHS обязательно конечно, и в заключительной конфигурации в каждом узле статусом *branch* отмечены рёбра MST и только они

Доказательство.

Из описания и подробных пояснений должно быть ясно, что сообщение **connect** отправляется только в минимальное граничное ребро фрагмента

По утверждению 2, если каждый фрагмент, имеющий хотя бы одно граничное ребро, действительно отправляет сообщение **connect**, то алгоритмом действительно строится MST согласно условию теоремы

GHS: свойства

Доказательство.

Покажем, что в любом фрагменте, отличном от MST, рано или поздно какой-либо узел отправляет сообщение **connect**

Предположим от противного, что это не так

Согласно описанию алгоритма, это означает, что в каждом фрагменте на очередном этапе выполнения действий этого фрагмента происходит блокировка: бесконечное безрезультатное ожидание того, что предусловие очередного действия станет истинным

Согласно схеме пересылки сообщений внутри фрагмента и между фрагментами, блокировка фрагмента F_1 (с рангом $rank_1$ и именем $name_1$) возможна только в тех случаях, когда сообщение из F_1 отправляется узлу другого фрагмента (F_2 с рангом $rank_2$ и именем $name_2$) и никогда не принимается в F_2 из-за невыполненного предусловия, и это возможно только для двух сообщений:

1. (**connect**, $rank_1$)
2. (**test**, $(rank_1, name_1)$)

GHS: свойства

Доказательство.

Для каждого из этих видов сообщений невозможность принять его в F_2 означает, что верно одно из следующих условий:

1. $rank_1 > rank_2$
2. $rank_1 = rank_2$ и $\omega_{e_{F_1}} > \omega_{e_{F_2}}$
3. $rank_1 = rank_2$, $\omega_{e_{F_1}} = \omega_{e_{F_2}}$, и в F_2 выполняется поиск минимального граничного ребра

Задача 3. Покажите, что невозможно выполнение хотя бы одного из трёх последних условий одновременно для каждого фрагмента

Это означает, что если существует текущий фрагмент, отличный от MST, то существует и незаблокированный фрагмент (*противоречие*) ▼

Задача 4.

- Оцените, сколько сообщений каждого типа отправляется при выполнении алгоритма
- Основываясь на этой оценке, покажите, что коммуникационная сложность GHS не превосходит $2|V|\log_2|V| + 5|E|$