Распределенные алгоритмы и системы

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

Блок 37

Задача обнаружения завершения вычислений

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

ВМК МГУ, 2022/2023, весенний семестр

Основные понятия

Иногда узел распределённого алгоритма завершает своё выполнение явно: переходит в особое заключительное состояние, в котором ни при каких обстоятельствах нельзя выполнить ни один переход

Например, так устроены многие обсуждавшиеся алгоритмы — те, в которых «по задумке» узел выполняет свой код до самого конца (низа записи)

Но узел может завершить выполнение и неявно: перейти в состояние, в котором возможен только приём сообщения, и только такого, которое никогда не появится в канале

Такие алгоритмы тоже обсуждались ранее

Например, в алгоритме Ле-Ланна избрания лидера в кольце последователь в бесконечном цикле выполняет команды

$$\begin{array}{l} \mathit{receive}(\mathbf{tok}, v) \\ \mathit{send}(\mathbf{tok}, v) \\ \mathit{lost} \end{array}$$

и никогда не переходит в заключительное состояние, то есть завершается неявно после получения последней отправленной ему фишки

Основные понятия

Неявное завершение можно расценивать как «зависание» узла, а явное — как сигнал к тому, что узел может переходить к выполнению следующей задачи

Хотелось бы и при использовании алгоритма с неявным завершением уметь получать такой сигнал, то есть уметь преобразовывать этот алгоритм в завершающийся явно

Задача обнаружения завершения вычислений (для краткости — задача о.з.в.) состоит (гарантированном) обнаружении неявного завершения вычисления узла (каждого узла заданного распределённого алгоритма), с тем чтобы можно было в этом случае выполнить команду принудительного явного завершения без ущерба для этого алгоритма

Основные понятия

Решить задачу о.з.в. можно, **например**, так:

- Время от времени собирать в узле снимок сети при помощи
  - алгоритма сохранения снимка и
  - централизованного волнового алгоритма, в котором решение принимает инициатор

- Проверять по собранному снимку, достигнута ли алгоритмом заключительная конфигурация
  - (это устойчивое свойство, а значит, можно убедиться в его наличии по значимому снимку)

Но хотелось бы иметь и другие (*более эффективные*) средства обнаружения завершения вычислений

Основные понятия

Для решения задачи о.з.в. потребуется, как и для задачи сохранения снимка, надстроить имеющийся распределённый алгоритм (базовый) алгоритмом, добавляющим возможность о.з.в. (контрольным)

Если в состоянии узла можно выполнить хотя бы одно базовое внутреннее действие или базовое действие отправки, то будем называть это состояние активным, а иначе — пассивным

Узел назовём активным, если он находится в активном состоянии, и пассивным иначе

Активность узла означает, что узел может выполнить ещё хотя бы одно базовое действие, то есть базовый алгоритм ещё не завершился

Пассивность узла означает, что базовый алгоритм в этом узле

- либо неявно завершился,
- либо рано или поздно выполнит действие приёма сообщения и продолжит выполняться, а до тех пор не выполнит ни одного действия

Основные понятия

Конфигурацию (надстроенного алгоритма) назовём целевой, если в ней не допустимо ни одно базовое действие

$\mathcal{F}$ — так обозначим множество всех целевых конфигураций

Контрольный алгоритм в задаче о.з.в. обычно разбивается на два подалгоритма:

1. Собственно алгоритм о.з.в., то есть алгоритм обнаружения свойства $\mathcal{F}$ в каком-либо узле сети
2. Алгоритм оповещения всех узлов о том, что свойство $\mathcal{F}$ достигнуто (и можно явно завершить вычисление)

Способы оповещения всех узлов уже обсуждались ранее (например, волновые алгоритмы и PIF)

Поэтому подробно обсу́дим только алгоритм о.з.в.

Основные понятия

Алгоритм обнаружения завершения вычисления — это распределённый алгоритм, надстраивающийся над произвольным базовым алгоритмом и обладающий следующими свойствами:

1. **Невмешательство**: алгоритм о.з.в. не влияет на последовательность выполняемых базовых действий

2. **Живость**: если достигнута конфигурация из $\mathcal{F}$, то рано или поздно (спустя конечное время) должен быть запущен алгоритм оповещения
   - Запуск алгоритма оповещения будем обозначать командой *announce*

3. **Безопасность**: алгоритм оповещения запускается только в конфигурациях множества $\mathcal{F}$

Технические упрощения

Далее будем использовать три технических упрощения (допущения):

1. Выполнение любого базового внутреннего действия делает узел пассивным
   - Если это не так, то можно «укрупнить» действия, присоединив внутренние действия к выполняющимся перед ними и после них
   - То есть алгоритм о.з.в. не интересуется тем, сколько именно и каких именно внутренних действий выполняется между действиями отправки и приёма
2. Активный узел может стать пассивным только после выполнения внутреннего действия
   - Если это не так, то можно добавить «фиктивное» внутреннее действие перед каждым пассивным состоянием
   - В частности, узел всегда считается активным после приёма сообщения
3. Приём сообщения, содержащегося в канале, всегда допустим
   - Если это не так, то можно исправить базовый алгоритм так, чтобы все сообщения принимались при поступлении и сохранялись внутри узла с дальнейшей обработкой (также внутри узла) согласно *receive*

Технические упрощения

$active_p$ — будем использовать эту запись в качестве булевого выражения, обозначающего активность узла p:

$$\text{узел } p \text{ активен} \quad \Leftrightarrow \quad active_p = \mathbb{t}$$

Как и в алгоритме Лаи-Янга, будем полагать, что в контрольный алгоритм «перехватывает» базовые сообщения, и в связи с этим содержит следующие действия:

1. $S_p(m, q)$ — выполняется вместо отправки базового сообщения m узлу q
2. $R_p(m, q)$ — выполняется вместо приёма базового сообщения m от узла q
3. $I_p(\alpha)$ — выполняется вместо базового внутреннего действия α

Основные свойства

Теорема (о целевых конфигурациях). Конфигурация γ является целевой $\Leftrightarrow$ в γ все узлы пассивны и в коммуникационной подсистеме ($\mathcal{M}$) не содержится ни одного базового сообщения

Доказательство.

($\Rightarrow$) Пусть $\gamma \in \mathcal{F}$

Если в γ хотя бы один узел p активен, то это (по определению) означает, что в p допустимо некоторое базовое действие — чего быть не может по определению $\mathcal{F}$

Если в $\mathcal{M}$ содержится хотя бы одно сообщение m, отправленное некоторому узлу p, то по техническим допущениям в p допустимо действие приёма m — чего быть не может по определению $\mathcal{F}$

($\Leftarrow$) Пусть $\gamma \notin \mathcal{F}$

Тогда в γ допустимо хотя бы одно базовое действие α некоторого узла p

Если α — внутреннее действие или действие отправки, то узел p активен

Если α — действие приёма, то в $\mathcal{M}$ содержится хотя бы одно сообщение, отправленное узлу p ▼

Основные свойства

Теорема 1 о сложности о.з.в.. **В любом алгоритме о.з.в. количество отправляемых контрольных сообщений в худшем случае не меньше коммуникационной сложности наилучшего волнового алгоритма, в котором все узлы являются инициаторами**

Доказательство.

Рассмотрим такой базовый алгоритм: каждый узел выполняет одно внутреннее действие и завершается

Согласно живости алгоритма о.з.в., после выполнения действия в каждом узле следует рано или поздно выполнить *announce*

Согласно невмешательству алгоритма о.з.в., никакое контрольное действие не может причинно-следственно предшествовать никакому базовому действию

Из этого и теоремы о перестановке действий следует, что если базовое действие α хотя бы одного узла не предшествует выполнению *announce*, то существует вычисление, в котором *announce* выполняется раньше α — чего быть не может по безопасности алгоритма о.з.в.

Основные свойства

Доказательство (продолжение).

Значит, каждое базовое действие предшествует выполнению *announce*

Следовательно, алгоритм о.з.в. — это волновой алгоритм, в котором *announce* = *decide*

Согласно невмешательству алгоритма о.з.в., в начальном состоянии каждого узла допустимо внутреннее действие — а значит, каждый узел является инициатором контрольного волнового алгоритма ▼

Теорема 2 о сложности о.з.в. (задача 1, трудная). В любом алгоритме о.з.в. количество отправляемых контрольных сообщений в худшем случае не меньше количества отправляемых базовых сообщений