Модели последовательных и параллельных вычислений

В.А. Захаров

Лекция 8.

1. Алгебраические модели систем взаимодействующих процессов
2. Алгебра процессов: синтаксис
3. Алгебра процессов: операционная семантика
4. Процессные графы и темпоральные логики
5. Отношение бисимуляции
6. Рекурсия и абстракция
7. Ветвящаяся бисимуляция
8. Применение алгебры процессов в задачах верификации распределенных систем

Алгебраические модели систем взаимодействующих процессов

Модели вычислений, которые рассматривались ранее, ведут свое происхождение или являются близкими родственниками конечных автоматов.

Их устройство и поведение описывается при помощи таких понятий как «состояние», «конфигурация», «переход».

Однако известно, что поведение конечных автоматов, которое проявляется в распознавании определенных языков, может быть описано и алгебраическими средствами при помощи операций конкатенации, сложения, итерации, из которых строятся регулярные выражения.

Если автоматы примутся «общаться» друг с другом, обмениваясь сообщениями, то мы получим систему параллельных взаимодействующих процессов.

Есть ли подходящие алгебраические средства для описания вычислений таких систем?

Алгебраические модели систем

Вычисление системы взаимодействующих процессов проявляется в выполнении действий .

Некоторые действия затрагивают только тот процесс, в котором они выполняются. Такие действия называются внутренними . Пример внутреннего действия — оператор присваивания.

Другие действия могут выполняться только синхронно в паре с другими действиями. С помощью таких согласованных пар действий процессы вступают во взаимодействие. Пример согласованной пары действий — операции отправления–приема сообщений.

Выполнение некоторые действия может быть зарегистрировано сторонним наблюдателем, тогда как другие действия остаются недоступными для наблюдения. Поэтому действия системы можно разделить также на видимые и скрытные .

Алгебраические модели систем

Для обозначения атомарных действий процессов введем множество имен

$$Names = \mathcal{E} \cup \mathcal{A} \cup \{\tau\},$$

которое включает в себя

- помножество $\mathcal{E}$ имен видимых внутренних действий процессов,
- помножество $\mathcal{A}$ имен взаимодействий процессов,
- имя τ для скрытных (невидимых) внутренних действий процессов.

Каждое взаимодействие с именем a реализуется парой комплиментарных действий $\overline{a}$ и a .

Алгебраические модели систем

Процессы формируются из действий при помощи операций композиции, таких как

- последовательная композиция $.$:
  выражение $p_1 . p_2$ обозначает процесс, всякое вычисление которого начинается вычислением процесса p_1 , и по завершении его продолжается вычислением процесса p_2 ;
- альтернативная композиция $+$:
  выражение $p_1 + p_2$ обозначает процесс, всякое вычисление которого либо является вычислением процесса p_1 , либо вычислением процесса p_2 ;
- параллельная композиция $\|$:
  выражение $p_1 \| p_2$ обозначает процесс, всякое вычисление которого представляет собой чередование вычислений процессов p_1 и p_2 , сопровождаемое выполнением взаимодействия.

Алгебра процессов: синтаксис

Процессом над множеством имен $Names$ называется всякое выражение (терм), которое может быть построено по следующим правилам:

1. **nil** — это процесс;
2. если p — это процесс, а n — это имя, то $(n.p)$ — это процесс;
3. если p_1, p_2 — это процессы, то $(p_1.p_2)$, $(p_1 + p_2)$, $(p_1 \parallel p_2)$ — это процессы;
4. и никаких других процессов не бывает.

Множество всех процессов обозначим записью $Proc$.
Процессами являются следующие термы:

- $\mathbf{nil} \in Proc$,
- $a.\mathbf{nil} \in Proc$, $\bar{a}.\mathbf{nil} \in Proc$, $\tau.\mathbf{nil} \in Proc$,
- $(a.\mathbf{nil}) \parallel (\bar{a}.\mathbf{nil}) \in Proc$,
- $(\tau.\mathbf{nil}) + ((a.\mathbf{nil}) \parallel (\bar{a}.\mathbf{nil})) \in Proc$.

Алгебра процессов: синтаксис

Попробуем построить систему взаимодействующих процессов, состоящую из

Кофейного аппарата

Человека-пользователя

Стакана

Алгебра процессов: синтаксис

Сценарий работы этой системы таков:

Алгебра процессов: синтаксис

Сценарий работы этой системы таков:

1) Пользователь опускает монету в автомат,
2) выбирает желаемый вид кофе,
3) нажимает на кнопку,
4) забирает стакан с напитком.

1) Автомат проверяет монету,
2) принимает заказ на кофе,
3) готовит кофе,
4) наливает кофе в стакан.

1) Стакан принимает в себя кофе,
2) отдается в руки пользователя.

Алгебра процессов: синтаксис

System: $\text{User} \parallel \text{Machine}$

$$
\begin{aligned}
\text{User} &= \overline{coin}.\text{Choice}.\overline{button}.wait.bring.\mathbf{nil} \\
\text{Choice} &= (\overline{espresso}.\mathbf{nil}) + (\overline{americano}.\mathbf{nil})
\end{aligned}
$$

$$
\begin{aligned}
\text{Machine} &= coin.\text{Select}.button.\text{Proceed}.\mathbf{nil} \\
\text{Select} &= (espresso.\mathbf{nil}) + (americano.\mathbf{nil}) \\
\text{Proceed} &= (make.\overline{fill}.\mathbf{nil}) \parallel \text{Glass} \\
\text{Glass} &= fill.\overline{bring}.\mathbf{nil}
\end{aligned}
$$

Алгебра процессов: синтаксис

Программа на языке программирования Occam

```
PROCEDURE Machine
  SEQ
    coin ? x
    select(y)
    button ? z
    proceed(x,y)
PROCEDURE select(x)
  ALT
    espresso ? u
      y:=1
    espresso ? u
      y:=2
```

```
PROCEDURE Proceed(x,y)
  PAR
    SEQ
      make_coffee(x,y,w)
      fill ! w
    Glass
PROCEDURE Glass
  SEQ
    fill ? v
    bring ! v
```

Алгебра процессов: операционная семантика

А как определить вычисление процессов?

В алгебре разные выражения могут иметь одно и то же значение, т.е. быть тождественно равными.
Тождественно равные процессы имеют одинаковое поведение.

Поэтому семантику выражений в алгебре процессов определим при помощи двух отношений:

- отношения тождественного равенства процессов, при помощи которого выделяются классы процессов с неотличимым поведением, и
- отношения переходов процессов, при помощи которого собственно определяются вычисления процессов.

Алгебра процессов: операционная семантика

Отношение тождественного равенства процессов $\equiv$ — это наименьшее отношение конгруентности (отношение эквивалентности, сохраняющееся при подстановке) на множестве $Proc$, которое удовлетворяет следующим соотношениям:

$$\text{Com}+ : \quad p + q \equiv q + p,$$

$$\text{Assoc}+ : \quad p + (q + r) \equiv (p + q) + r,$$

$$\text{Com} \parallel : \quad p \parallel q \equiv q \parallel p,$$

$$\text{Assoc} \parallel : \quad p \parallel (q \parallel r) \equiv (p \parallel q) \parallel r,$$

$$\text{Zero} : \quad p \equiv \mathbf{nil}.p \equiv \mathbf{nil} + p \equiv \mathbf{nil} \parallel p.$$

Алгебра процессов: операционная семантика

Таким образом,

$$
\begin{aligned}
& (\mathbf{nil}.\mathbf{nil}) \parallel (p \parallel (\mathbf{nil} + \mathbf{nil})) \\
\equiv\ & \mathbf{nil} \parallel (p \parallel (\mathbf{nil} + \mathbf{nil})) \\
\equiv\ & \mathbf{nil} \parallel (p \parallel \mathbf{nil}) \\
\equiv\ & \mathbf{nil} \parallel p \\
\equiv\ & p
\end{aligned}
$$

Далее мы не будем проводить различие между эквивалентными процессами.

Алгебра процессов: операционная семантика

Введем функцию взаимодействия $\varphi : \mathcal{A} \to \mathcal{E}$, которая сопоставляет каждому типу взаимодействия a некоторое внутреннее атомарное действие $e = \varphi(a)$.

Взаимодействие двух параллельно работающих процессов осуществляется за счет синхронного выполнения пары взаимодействий $\bar{a}$ и a .

Но для стороннего наблюдателя и для той системы, которая включает в себя эти процессы, их взаимодействие будет выглядеть как выполнение системой внутренного действия $\varphi(a)$.

Алгебра процессов: операционная семантика

Отношение переходов $\rightarrow \subseteq Proc \times Names \times Proc$ (его также называют **отношением срабатывания** процессов) определяется как наименьшее отношение указанного типа, которое удовлетворяет следующим требованиям:

RA : $\dfrac{-}{x.p \xrightarrow{x} p}$ для любых $x \in \mathcal{E} \cup \{\tau\},\ p \in Proc$

R $\equiv$: $\dfrac{p' \equiv p,\ p \xrightarrow{x} q,\ q \equiv q'}{p' \xrightarrow{x} q'}$

R. : $\dfrac{p \xrightarrow{x} q}{p.r \xrightarrow{x} q.r}$ для любого $x \in Names$;

R+ : $\dfrac{p \xrightarrow{x} q}{(r + p) \xrightarrow{x} q}$

Алгебра процессов: операционная семантика

R1 $\|$: $\dfrac{-}{a.p \parallel \bar{a}.q \xrightarrow{\varphi(a)} p \parallel q}$ для любого взаимодействия $a \in \mathcal{A}$;

R2 $\|$: $\dfrac{p \xrightarrow{x} q}{(r \parallel p) \xrightarrow{x} (r \parallel q)}$ для любого внутреннего действия $x \in \mathcal{E} \cup \{\tau\}$

Алгебра процессов: операционная семантика

Пример

Предположим, что $\varphi(a) = c$. Тогда

$$(\bar{b}.\mathbf{nil} + \mathbf{nil}.a.\mathbf{nil}) \parallel \tau.\mathbf{nil} \parallel (b.\mathbf{nil} + \bar{a}.\mathbf{nil}) \xrightarrow{c} \tau.\mathbf{nil},$$

поскольку

$$\begin{array}{rcll}
a.\mathbf{nil} \parallel \bar{a}.\mathbf{nil} & \xrightarrow{c} & \mathbf{nil} \parallel \mathbf{nil} & \text{R1} \parallel \\
\mathbf{nil}.a.\mathbf{nil} & \equiv & a.\mathbf{nil} & 0 \\
\mathbf{nil}.a.\mathbf{nil} \parallel \bar{a}.\mathbf{nil} & \xrightarrow{c} & \mathbf{nil} & \text{R} \equiv \\
(\bar{b}.\mathbf{nil} + \mathbf{nil}.a.\mathbf{nil}) \parallel \bar{a}.\mathbf{nil} & \xrightarrow{c} & \mathbf{nil} & \text{R1}+ \\
(\bar{b}.\mathbf{nil} + \mathbf{nil}.a.\mathbf{nil} \parallel (b.\mathbf{nil} + \bar{a}.\mathbf{nil}) & \xrightarrow{c} & \mathbf{nil} & \text{R2}+ \\
(\bar{b}.\mathbf{nil} + \mathbf{nil}.a.\mathbf{nil}) \parallel (b.\mathbf{nil} + \bar{a}.\mathbf{nil}) \parallel \tau.\mathbf{nil} & \xrightarrow{c} & \mathbf{nil} \parallel \tau.\mathbf{nil} & \text{R2} \parallel \\
\mathbf{nil} \parallel \tau.\mathbf{nil} & \equiv & \tau.\mathbf{nil} & 0, \mathit{Com} \parallel \\
(\bar{b}.\mathbf{nil} + \mathbf{nil}.a.\mathbf{nil}) \parallel (b.\mathbf{nil} + \bar{a}.\mathbf{nil}) \parallel \tau.\mathbf{nil} & \xrightarrow{c} & \tau.\mathbf{nil} & \text{R} \equiv
\end{array}$$

Алгебра процессов: операционная семантика

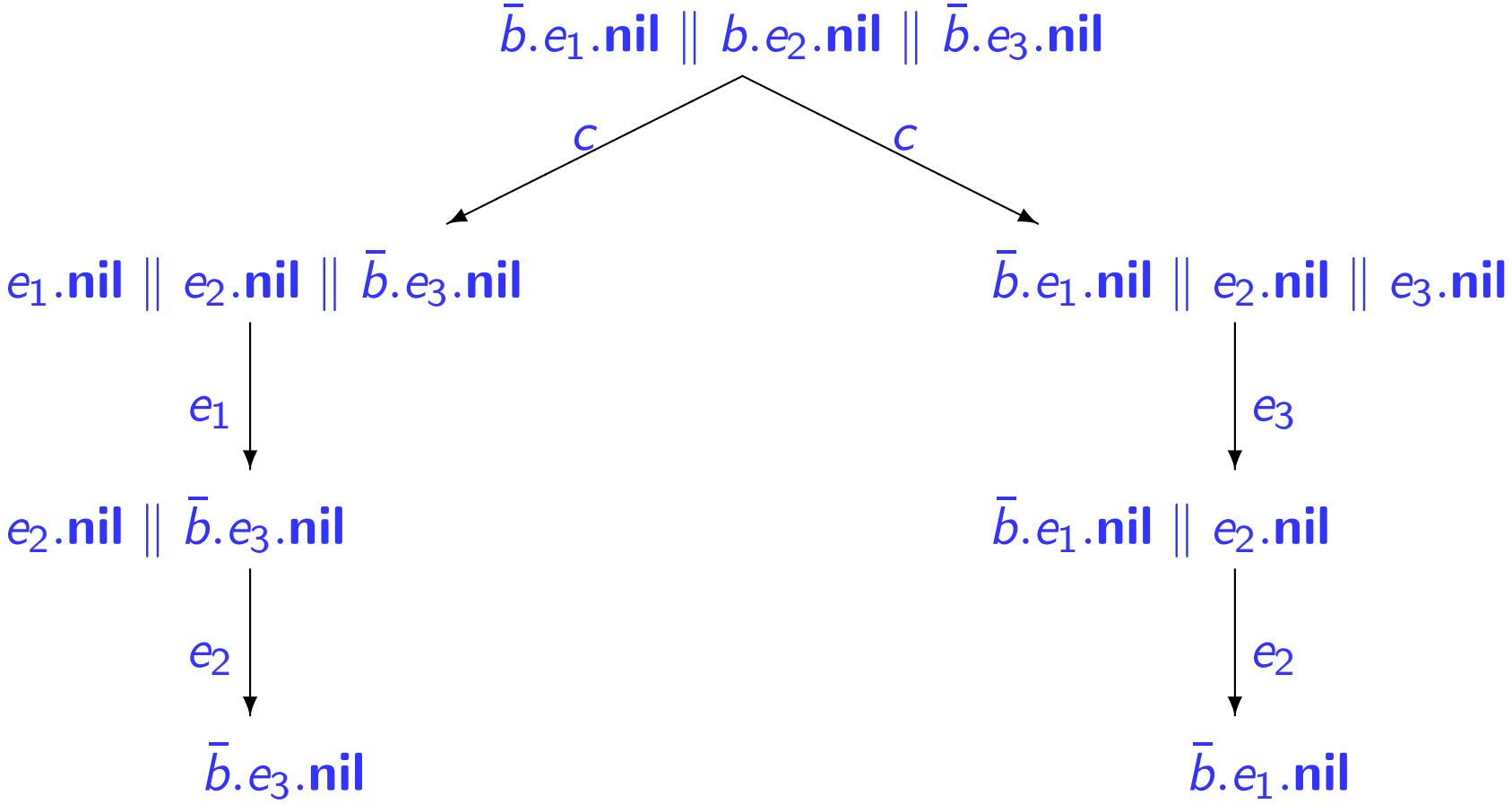
Вычисление системы взаимодействующих процессов, заданной процессным выражегием t_0 — это максимальная последовательность процессов

$$t_0 \xrightarrow{x_1} t_1 \xrightarrow{x_2} t_2 \xrightarrow{x_3} \cdot \xrightarrow{x_i} x_i \xrightarrow{x_{i+1}} \cdots,$$

связанных между собой отношением переходов.

Алгебра процессов: операционная семантика

Процесс может иметь несколько различных вычислений



Все вычисления системы разумно объединить в одной структуре — процессном графе .

Процессные графы

Пусть имеется процесс $p, p \in Proc$.

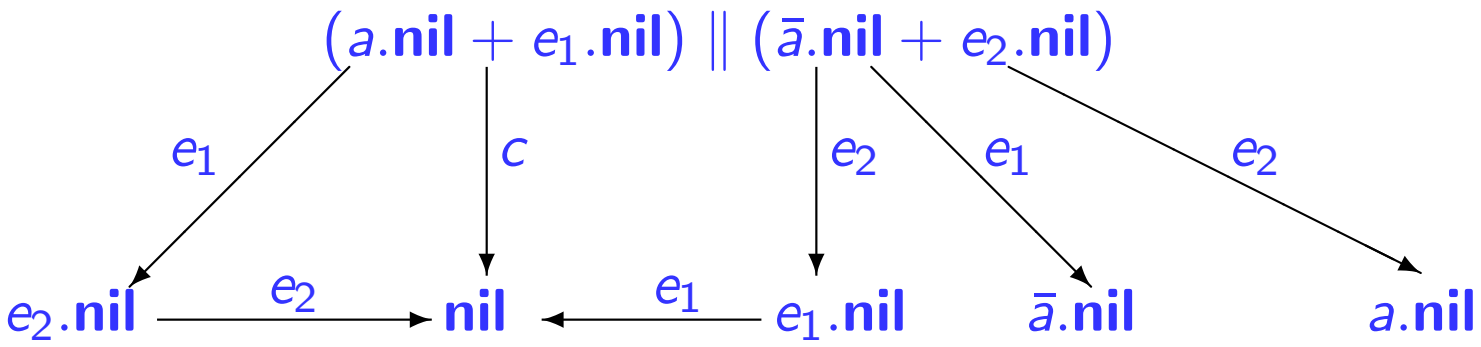
Обозначим записью $\longrightarrow_*$ рефлексивно-транзитивное замыкание отношения переходов, и выделим множество процессов $p \downarrow = \{p' \ : \ p \longrightarrow_* p'\}$ достижимых из процесса p по отношению переходов.

Процессным графом , порожденным процессом p , называется ориентированный размеченный граф $\Gamma(p) = ([p \downarrow]_\equiv, \longrightarrow)$, в котором

- множество вершин $[p \downarrow]_\equiv$ — это фактор-множество процессов, достижимых из процесса p , по отношению тождественности процессов $\equiv$;
- из вершины $[q]_\equiv$ в вершину $[r]_\equiv$ ведет дуга, помеченная именем x в том и только том случае, если допустим переход $q \xrightarrow{x} r$.

Процессные графы

Пример процессного графа



Процессные графы

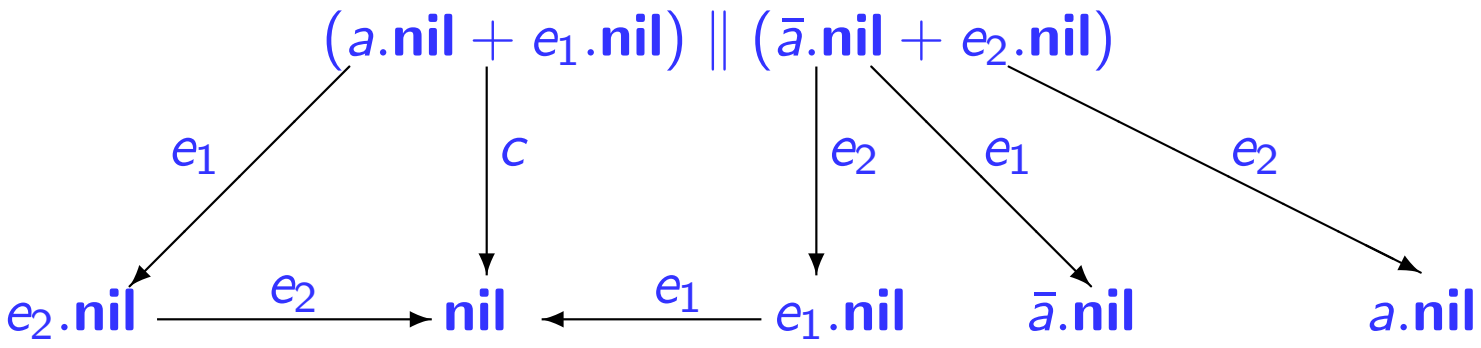
Процессный граф $\Gamma(p) = ([p \downarrow]_{\equiv}, \longrightarrow)$ — это система переходов, дуги которой размечены именами базовых действий.

Имена базовых действий можно рассматривать как элементарные (атомарные) логический условия — индикаторы выполнения этих действий.

Процессные графы

Процессный граф $\Gamma(p) = ([p\downarrow]_{\equiv}, \longrightarrow)$ — это система переходов, дуги которой размечены именами базовых действий.

Имена базовых действий можно рассматривать как элементарные (атомарные) логический условия — индикаторы выполнения этих действий.

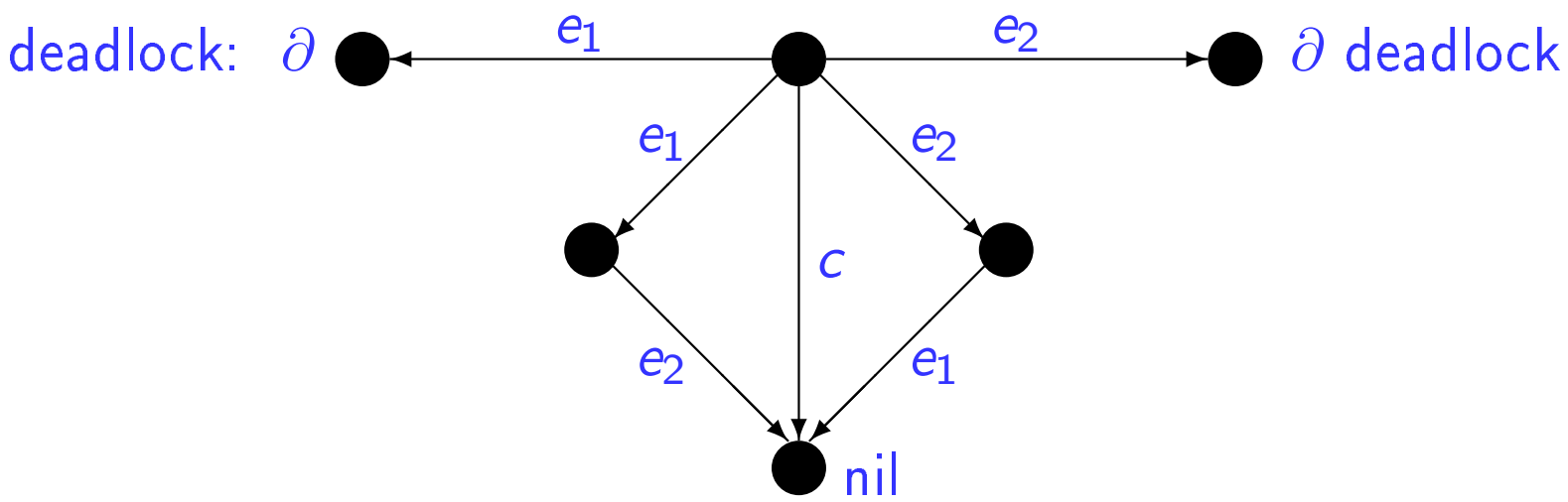


И тогда для описания свойств вычислений взаимодействующих процессов можно использовать темпоральные логики CTL, LTL, CTL*, семантика которых задается при помощи размеченных систем переходов (моделей Крипке).

Процессные графы

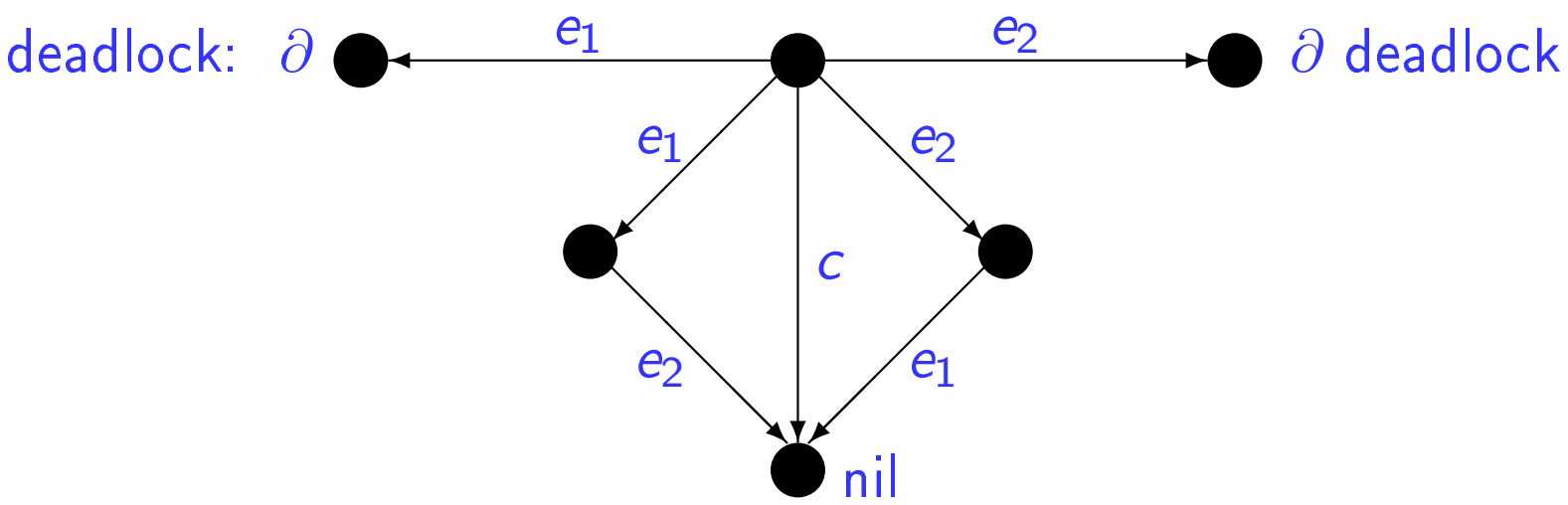
Процессный граф $\Gamma(p) = ([p \downarrow]_{\equiv}, \longrightarrow)$ — это система переходов, дуги которой размечены именами базовых действий.

Имена базовых действий можно рассматривать как элементарные (атомарные) логический условия — индикаторы выполнения этих действий.



И тогда для описания свойств вычислений взаимодействующих процессов можно использовать темпоральные логики CTL, LTL, CTL*, семантика которых задается при помощи размеченных систем переходов (моделей Крипке).

Процессные графы



Например, интерес представляют такие свойства вычислений:

- **AF nil**

  Правда ли, что каждое вычисление завершается успешно ? (Ответ: НЕТ)
- **EG EF nil**

  Правда ли, что некоторое вычисление можно всегда направить к успешному завершению ? (Ответ: ДА)
- **A** $(c$ **R** $\neg e_1)$

  Правда ли, что действие e_1 выполняется не раньше, чем действие c ? (Ответ: НЕТ)

Отношение бисимуляции

Процессные графы можно сравнивать между собой по отношению взаимного моделирования (бисимуляции).

Отношение бисимуляции на множестве процессов $Proc$ — это такое симметричное рефлексивное бинарное отношение R, $R \subseteq Proc \times Proc$, которое удовлетворяет следующим требованиям:

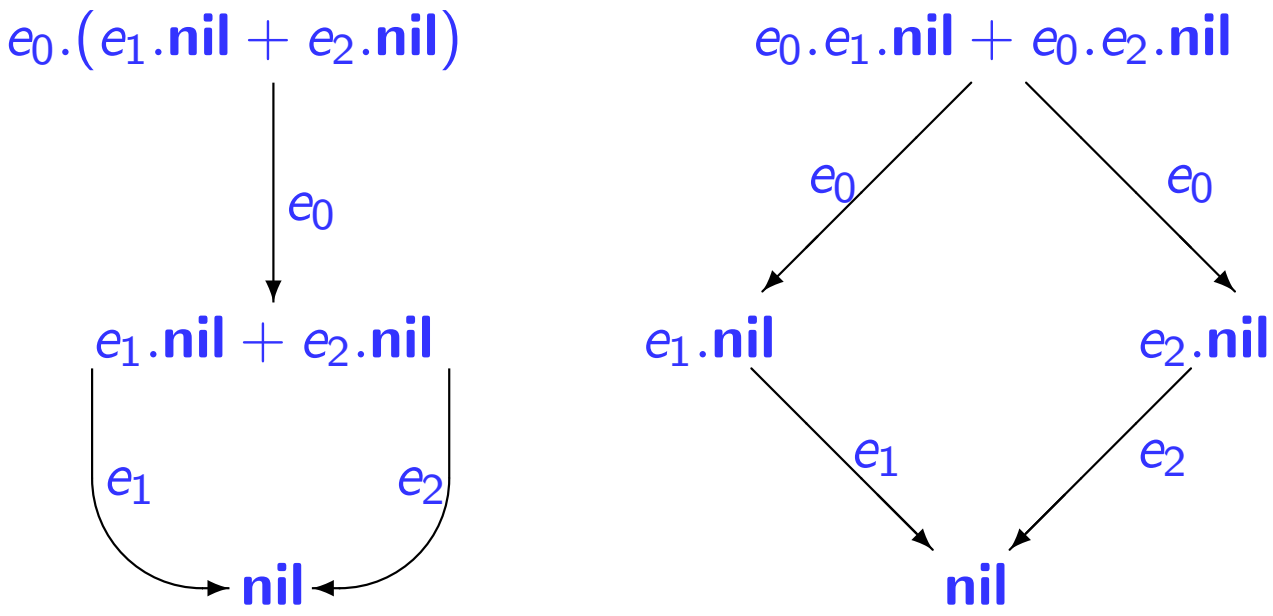
1. $\mathbf{nil}\, R\, p \;\Rightarrow\; p \equiv \mathbf{nil}$,
2. $p\, R\, p' \;\wedge\; p \xrightarrow{x} q \;\Rightarrow\; \exists q' :\; p' \xrightarrow{x} q' \;\wedge\; q\, R\, q'$.

Два процесса p и q считаются бисимуляционно эквивалентными ($p \sim q$), если существует такое отношение бисимуляции R , для которого выполняется pRq .

Отношение бисимуляции

Процессы $e_1.\mathbf{nil} \parallel e_2.\mathbf{nil}$ и $e_1.e_2.\mathbf{nil} + e_2.e_1.\mathbf{nil}$ бисимуляционно эквивалентны — их процессные графы изоморфны.

Процессы $e_0.(e_1.\mathbf{nil} + e_2.\mathbf{nil})$ и $e_0.e_1.\mathbf{nil} + e_0.e_2.\mathbf{nil}$ **не являются** бисимуляционно эквивалентными.



Отношение бисимуляции

Оказывается, что отношение бисимуляции процессов сохраняет выполнимость формул темпоральных логик на процессных графах.

Теорема 1.

Два процесса p и q бисимуляционно эквивалентны в том и только том случае, если для любой формулы φ темпоральной логики CTL* справедливо соотношение

$$\Gamma(p) \models \varphi \;\Leftrightarrow\; \Gamma(q) \models \varphi.$$

Отношение бисимуляции

1. Наряду с введенным отношением бисимуляции существуют и другие отношения между процессами, отражающие большую или меньшую степень схожести их поведения. Некоторые из этих отношений мы рассмотрим далее.
2. Все процессы, которые можно определить при помощи введенных выражений алгебры процессов, имеют только конечные вычисления. Поэтому их процессные графы конечны, и их поведение легко анализировать.
   А как можно определить процессы, имеющие бесконечные вычисления?

Рекурсия

Есть несколько способов определения процессов, имеющих неограниченно длительные вычисления.

Наиболее часто используются два из них: **оператор репликации** и **рекурсивные уравнения**.

Оператор репликации $!$ в алгебре процессов является аналогом оператора итерации Клини в алгебре регулярных множеств.

Синтаксическое определение, вводящее оператор репликации таково:

- если $p \in Proc$, то $!p \in Proc$.

Семантика оператора репликации определяется тождеством

$$\text{R!} \quad : \quad !p \equiv p \parallel !p \quad \text{для любых} \quad p \in Proc.$$

Рекурсия

Пример использования оператора репликации — описание системы передачи данных по каналу связи с подтверждением:

$$!(send.\overline{ack}.\mathbf{nil}) \parallel !(\overline{send}.ack.\mathbf{nil})$$

$$\equiv \quad !(send.\overline{ack}.\mathbf{nil}) \parallel send.\overline{ack}.\mathbf{nil} \parallel \overline{send}.ack.\mathbf{nil} \parallel !(\overline{send}.ack.\mathbf{nil})$$

$$\xrightarrow{mes} \quad !(send.\overline{ack}.\mathbf{nil}) \parallel \overline{ack}.\mathbf{nil} \parallel ack.\mathbf{nil} \parallel !(\overline{send}.ack.\mathbf{nil})$$

$$\xrightarrow{repl} \quad !(send.\overline{ack}.\mathbf{nil}) \parallel !(\overline{send}.ack.\mathbf{nil})$$

Рекурсия

Другой, более общий способ описания бесконечных процессов — определять их, как решения систем уравнений, содержащих процессные переменные.

Добавим к множеству имен $Names$ множество процессных переменных X и будем рассматривать процессные выражения над множеством базовых символов $Names \cup X$.

Тогда рекурсивной спецификацией процессов называется система уравнений вида

$$\begin{array}{lcl} X_1 & = & \Phi_1(X_1, \ldots, X_k) \\ & dots & \\ X_k & = & \Phi_k(X_1, \ldots, X_k) \end{array}$$

где $X_1, \ldots, X_k$ — процессные переменные, а $\Phi_1, \ldots, \Phi_k$ — процессные термы, зависящие только от указанных переменных.

Рекурсия

Например, систему передачи данных по каналу свзи с подтверждением можно описать вот такой системой уравнений:

$$\begin{array}{lcl} S & = & X_1 \parallel X_2 \\ X_1 & = & send.\overline{ack}.\mathbf{nil} \parallel X_1 \\ X_2 & = & \overline{send}.ack.\mathbf{nil} \parallel X_2 \end{array}$$

Задача.

Каково решение следующей системы уравнений

$$\begin{array}{lcl} X & = & Y \parallel Z \\ Y & = & a + Z \\ Z & = & \bar{a}.Z + \mathbf{nil} \end{array}$$

Рекурсия

Однако не всякая система процессных уравнений имеет единственное решение.

Это можно гарантировать лишь для предохраняемых систем уравнений (guarded recursive specifications), состоящих из уравнений вида

$$X_i = a_1.\Psi_1(X_1, \ldots, X_k) + \cdots + a_m.\Psi_m(X_1, \ldots, X_k) + b_1.\mathbf{nil} + \cdots + b_\ell.\mathbf{nil}.$$

Задача.

Найдите метод решения линейных систем уравнений, состоящих из уравнений вида

$$X_i = a_1.X_{i_1} + \cdots + a_m.X_{i_m} + b_1.\mathbf{nil} + \cdots + b_\ell.\mathbf{nil}.$$

Абстракция

Когда проводится анализ поведения какой-нибудь сложной распределенной системы (например, микроэлектронной схемы или операционной системы), многие действия, выполняемые системой, бывают недоступны для наблюдения или попросту неинтересны.

Такие действия можно считать лишенными индивидуальных отличительных черт, неотличимыми друг от друга, скрытыми от наблюдения. Для их обозначения вводится специальное скрытное действие τ .

Для работы со скрытным действием используются операция **абстракции** и отношение **ветвящейся бисимуляции**.

Абстракция позволяет сделать «невидимыми» некоторые действия системы, а ветвящаяся бисимуляция позволяет оценивать схожесть поведения двух систем, игнорируя эти «невидимыми» действия.

Абстракция

Операция абстракции.

Пусть задано процессное выражение p и выделено некоторое подмножество внутренних действий $I, I \subseteq \mathcal{E}$.
Тогда абстракций процесса p относительно множества I называется процесс $\tau_I(p)$, который получается заменой в процессе p и во всех вычислениях этого процесса всех действий из множества I скрытным действием τ

Например, если $I = \{c, e\}$ и $\varphi(a) = c$, то

$$\tau_I\Big(a.d.\mathbf{nil} \parallel \bar{a}.e.\mathbf{nil}\Big) = a.d.\mathbf{nil} \parallel \bar{a}.\tau.\mathbf{nil}$$

и вычисление процесса $\tau_I(p)$ имеет вид

$$a.d.\mathbf{nil} \parallel \bar{a}.\tau.\mathbf{nil} \xrightarrow{\tau} d.\mathbf{nil} \parallel \tau.\mathbf{nil} \xrightarrow{\tau} d.\mathbf{nil} \xrightarrow{d} \mathbf{nil}$$

Алгебра процессов: синтаксис

С точки зрения пользователя работа кофейной машины — это абстракция ее процесса от невидимых для человека действий $make, fill$:

$$\tau_{\{make,fill\}}(\text{System}) : \qquad \text{User} \parallel \tau_{\{make,fill\}}(\text{Machine})$$

$$\begin{aligned} \text{User} &= \overline{coin}.\text{Choice}.\overline{button}.wait.bring.\mathbf{nil} \\ \text{Choice} &= (\overline{espresso}.\mathbf{nil}) + (\overline{americano}.\mathbf{nil}) \end{aligned}$$

$$\begin{aligned} \tau_{\{make,fill\}}(\text{Machine}) \quad = \quad & coin.(espresso.\mathbf{nil}) + (americano.\mathbf{nil}). \\ & button.(\tau.\overline{fill}.\mathbf{nil}) \parallel fill.\overline{bring}.\mathbf{nil}).\mathbf{nil}, \end{aligned}$$

где $\varphi(fill) = \tau$

Ветвящаяся бисимуляция

Отношение ветвящейся бисимуляции.

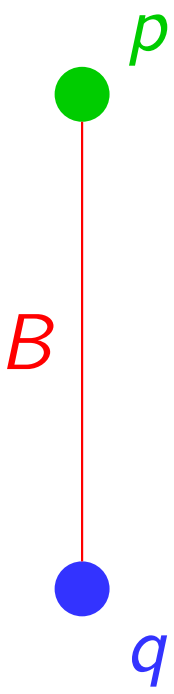
Симметричное и рефлексивное отношение B, $B \subseteq Proc \times Proc$, называется отношением ветвящейся бисимуляции , если оно удовлетворяет следующему требованию:
если pBq и $p \xrightarrow{x} p'$, то верно одно из двух

1. либо $x = \tau$ и $p'Bq$,
2. либо существует такая последовательность переходов
$$q \xrightarrow{\tau} q_1 \xrightarrow{\tau} \cdots \xrightarrow{\tau} q_n \xrightarrow{x} q' ,$$
для которой выполняются отношение $p'Bq'$, а также отношения pBq_i для всех $i, 1 \leq i \leq n$.

Ветвящаяся бисимуляция

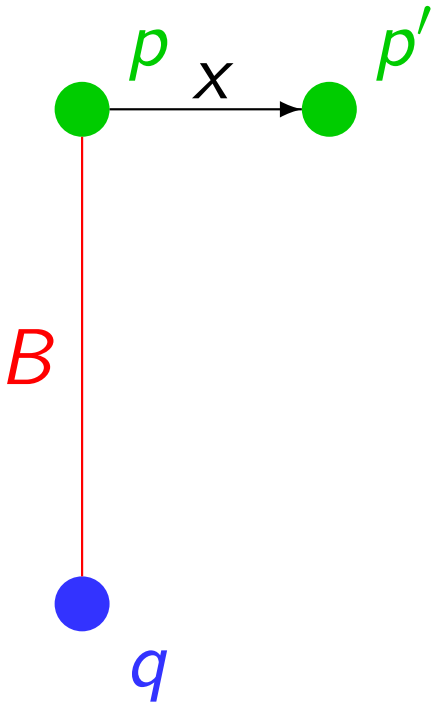
Для любой пары процессов p и q

Ветвящаяся бисимуляция



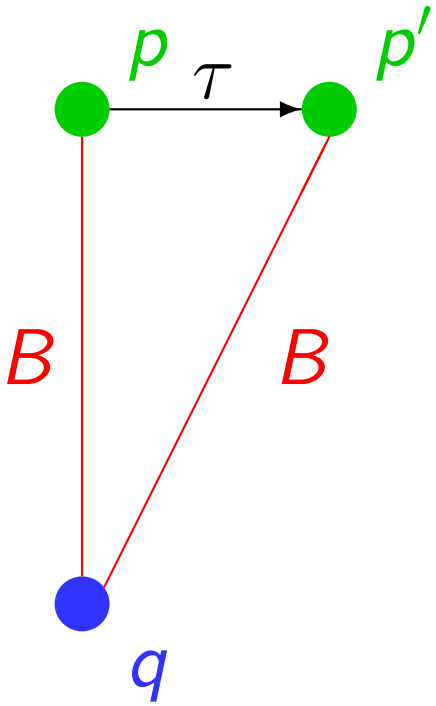
Для любой пары процессов p и q если pBq

Ветвящаяся бисимуляция



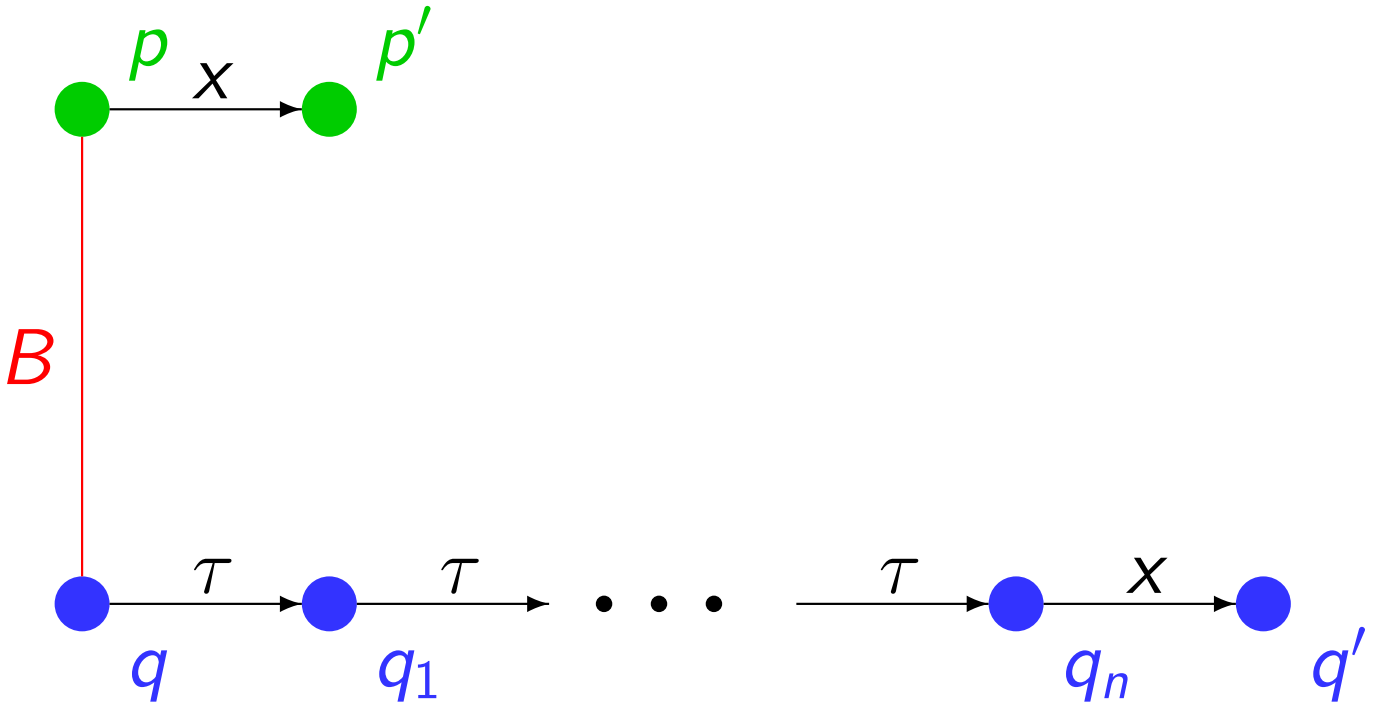
Для любой пары процессов p и q если pBq и $p \xrightarrow{x} p'$,

Ветвящаяся бисимуляция



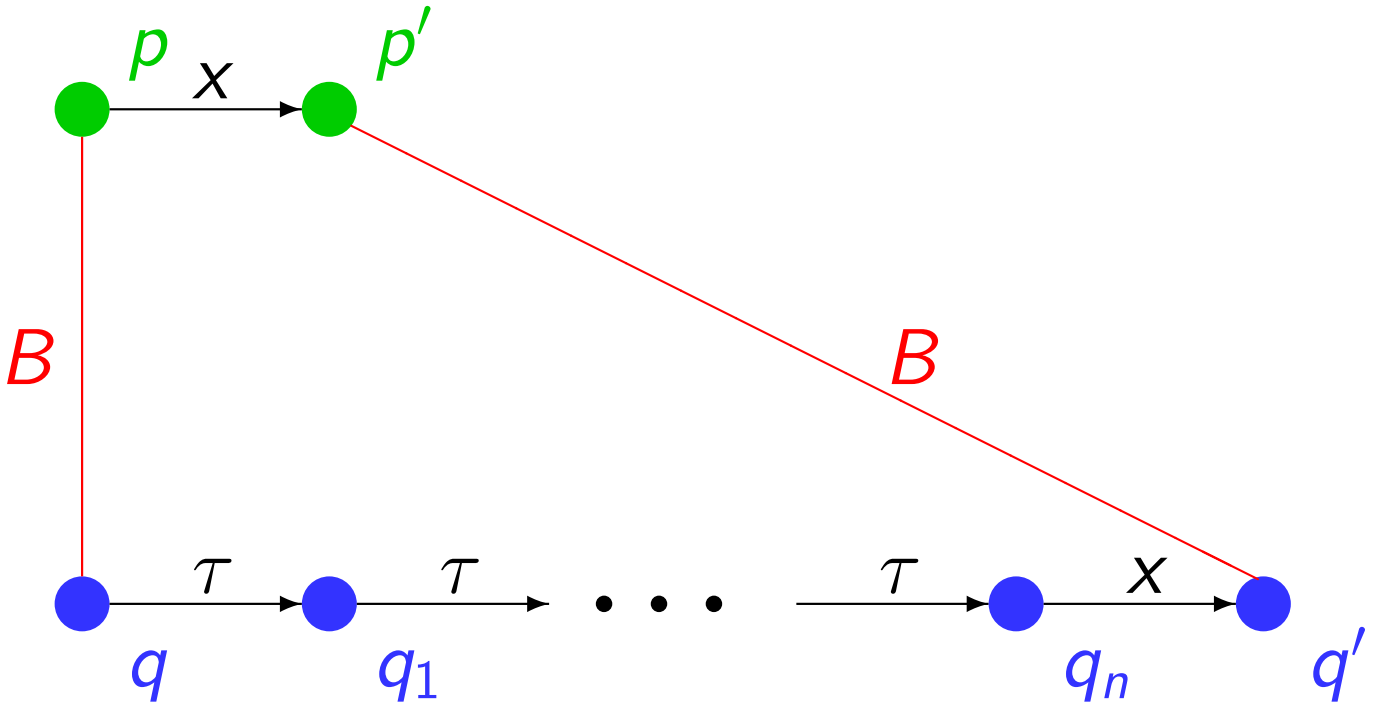
Для любой пары процессов p и q если pBq и $p \xrightarrow{x} p'$, либо $x = \tau$ и $p'Bq$

Ветвящаяся бисимуляция



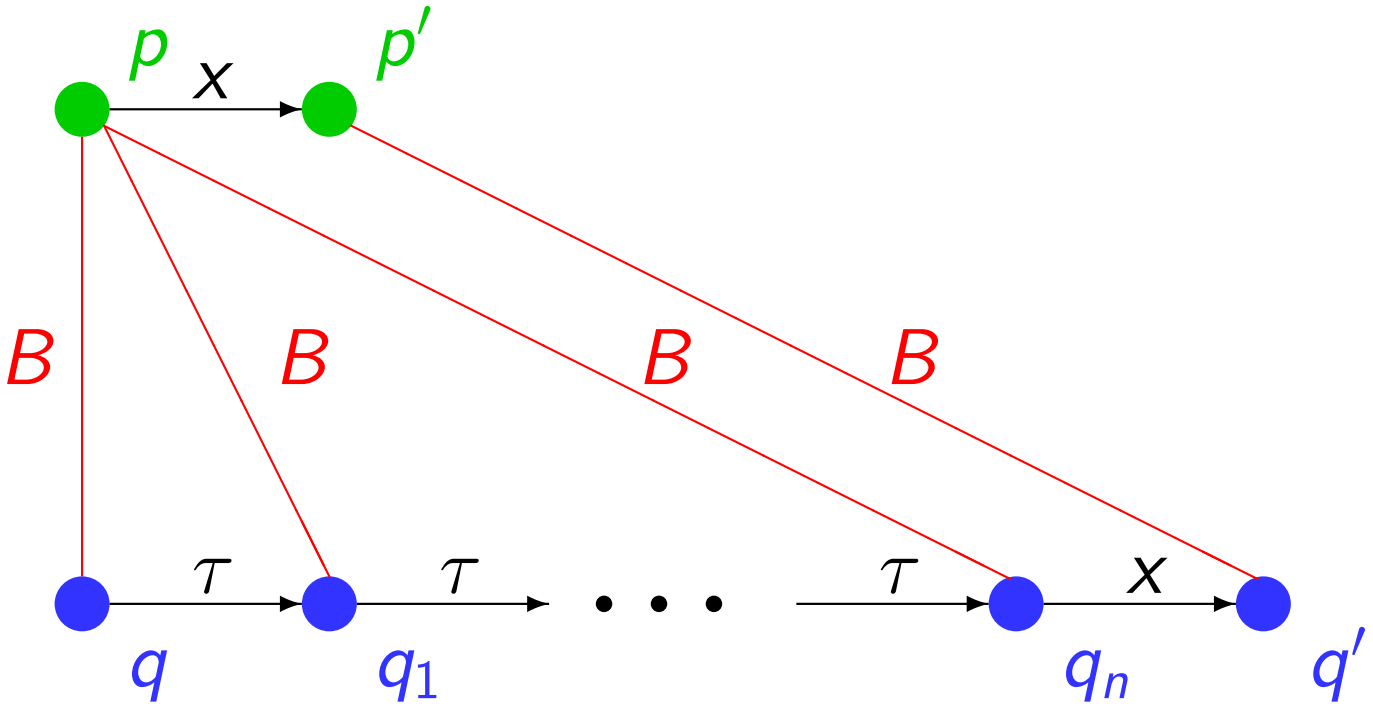
Для любой пары процессов p и q если pBq и $p \xrightarrow{x} p'$, либо существует такая последовательность переходов

Ветвящаяся бисимуляция



Для любой пары процессов p и q если pBq и $p \xrightarrow{x} p'$, либо существует такая последовательность переходов для которой $p'Bq'$,

Ветвящаяся бисимуляция



Для любой пары процессов p и q если pBq и $p \xrightarrow{x} p'$, либо существует такая последовательность переходов для которой $p'Bq'$, а также pBq_i для всех $i, 1 \leq i \leq n$

Ветвящаяся бисимуляция

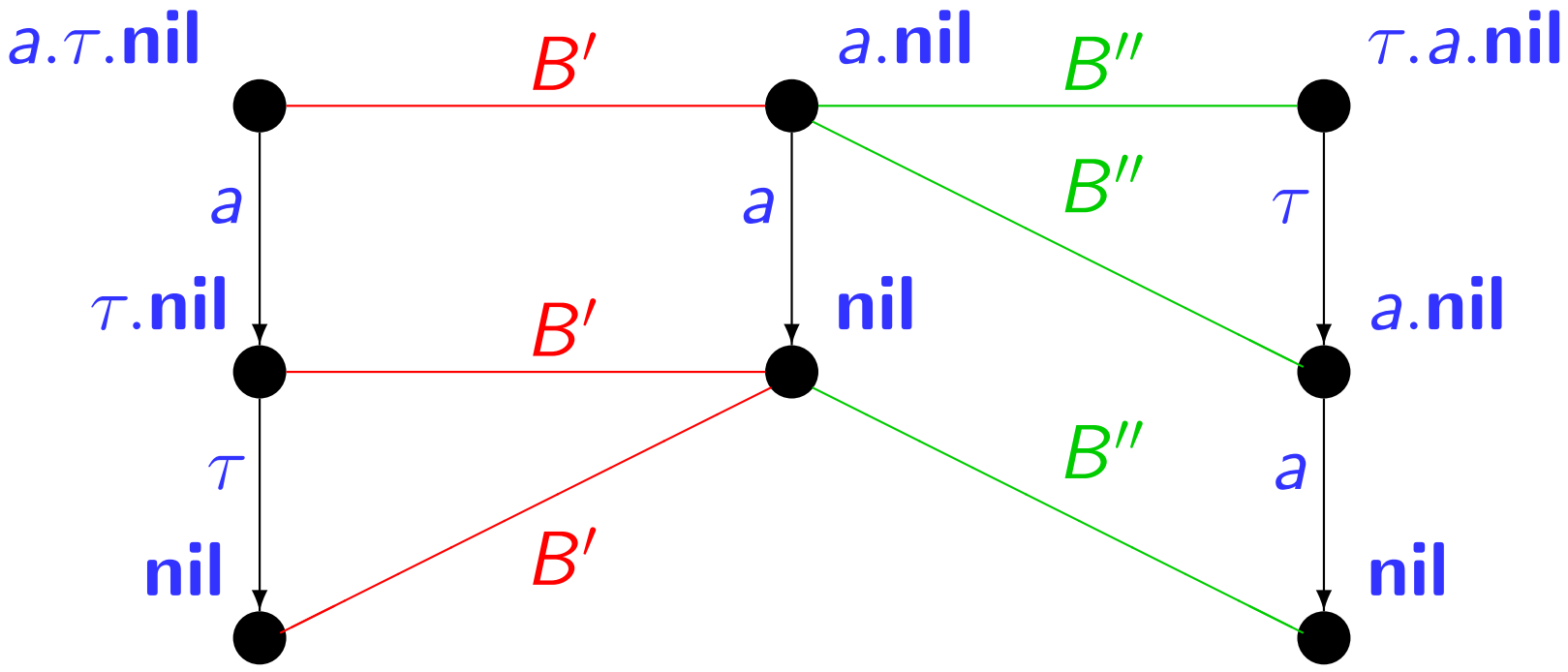
Примеры.

В отношении ветвящейся бисимуляции находятся процессы

1. $a.\mathbf{nil}$, $\tau.a.\mathbf{nil}$ и $a.\tau.\mathbf{nil}$
2. $a.\mathbf{nil} + \tau.(a.\mathbf{nil} + b.\mathbf{nil}).\mathbf{nil}$ и $\tau.(a.\mathbf{nil} + b.\mathbf{nil}).\mathbf{nil} + b.\mathbf{nil}$
3. $a.\mathbf{nil} + b.\mathbf{nil}.$ и $\tau.a.\mathbf{nil} + b.\mathbf{nil}$

Почему?

Ветвящаяся бисимуляция



Ветвящаяся бисимуляция

Примеры.

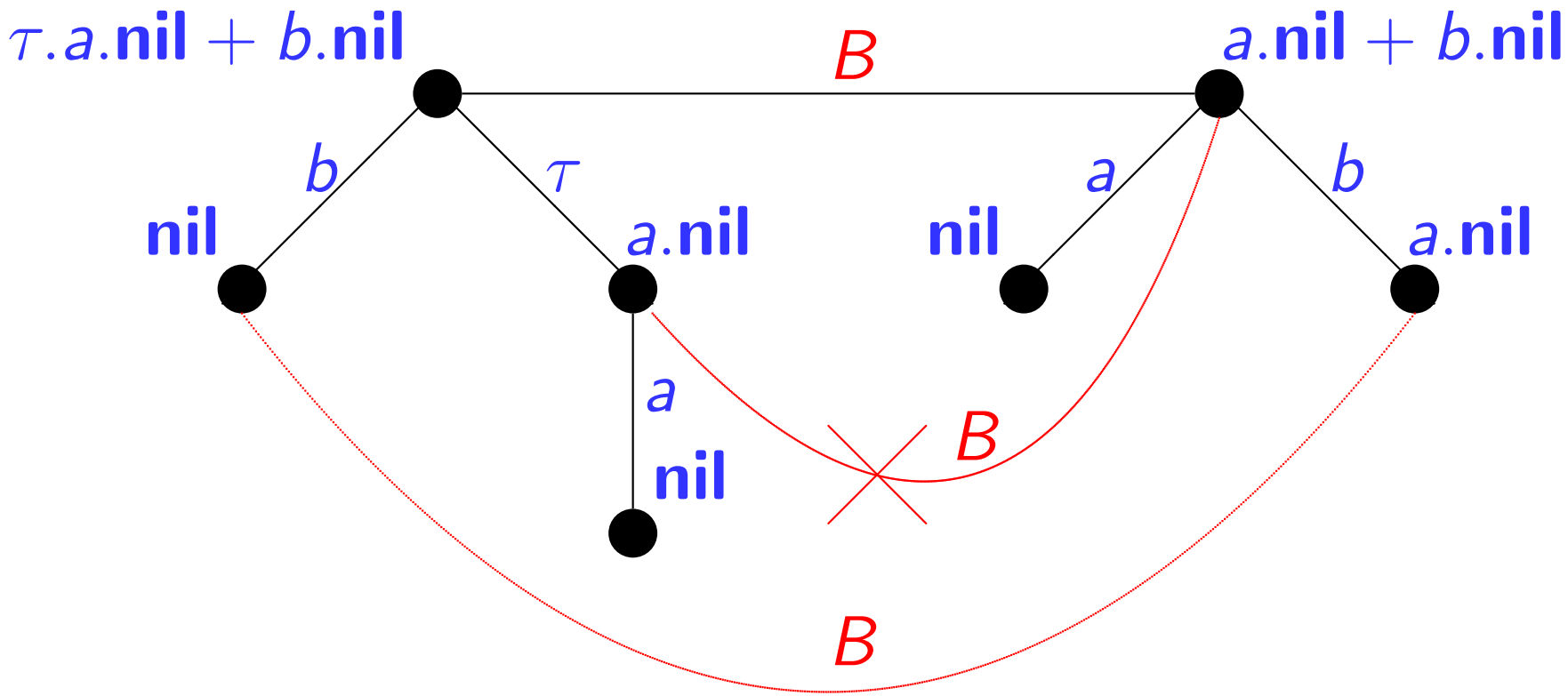
Следующие пары процессов **не находятся** в отношении ветвящейся бисимуляции:

4. $a.\mathbf{nil} + b.\mathbf{nil}$ и $\tau.a.\mathbf{nil} + b.\mathbf{nil}$
(иначе τ уже не было бы скрытным действием — оно приводит к выполнению действия a , но не приводит к выполнению действия b)

5. $a.\mathbf{nil} + b.\mathbf{nil}$ и $\tau.a.\mathbf{nil} + \tau.b.\mathbf{nil}$

Почему?

Ветвящаяся бисимуляция



Применение алгебры процессов

Алгебры процессов применяются для спецификации и верификации распределенных информационных систем, состоящих из взаимодействующих процессов.

Технология применения алгебры процессов такова.

Применение алгебры процессов

1. Требования, предъявляемые к проектируемой распределенной системе, — техническое задание — записываются на одном из языков формальных спецификаций (например, на языке одной из темпоральных логик). Полученная таким образом, **спецификация** $Spec$ исследуется на непротиворечивость при помощи одной из систем автоматического доказательства теорем.

2. **Модель проектируемой распределенной** системы описывается явно или неявно (при помощи процессных уравнений Eq) в алгебре процессов.
Основное внимание уделяется описанию протокола взаимодействия процессов проектируемой системы.

3. Вычисляется решение $p_1, \ldots, p_N$ системы процессных уравнений Eq , описывающих поведение проектируемой системы, и строится параллельная композиция $Model = p_1 \parallel \cdots \parallel p_N$, которая представляет модель проектируемой системы.

Применение алгебры процессов

4. Проводится проверка соответствия построенной модели $Model$ заданной спецификации $Spec$: $\Gamma(Model) \models Spec$. Для решения этой задачи применяются инструменты верификации моделей программ (model checking). Обнаруженные ошибки устраняются за счет уточнения спецификации проекта или изменения модели проекта.

5. На основе построенной модели $Model$ создается программная реализация проектируемой распределенной системы $Prog$. На каждом этапа создания распределенной системы

- ее программная реализация $Prog$ транслируется в выражение алгебры процессов $Syst$,
- к этому выражению применяется операция абстракции $Abstr = \tau_I(Syst)$, скрывающая все несущественные действия процесса-реализации,
- проводится проверка отношения ветвящейся бисимуляции $Abstr \sim_{br} Model$.

Применение алгебры процессов

Математическая теория гарантирует

$$\Gamma(Model) \models Spec \Leftrightarrow \Gamma(Syst) \models Spec.$$

Но чтобы воспользоваться этой теорией надо справиться с задачами

1. решения систем процессных уравнений
   $Eq \Rightarrow Model$,
2. верификации моделей программ
   $\Gamma(Model) \models Spec$,
3. абстракции программ
   $Prog \Rightarrow Syst \Rightarrow Abstr = \tau_I(Syst)$,
4. проверки бисимуляционной эквивалентности
   $Abstr \sim_{br} Model$.

КОНЕЦ ЛЕКЦИИ 8