Расширения сетей Петри

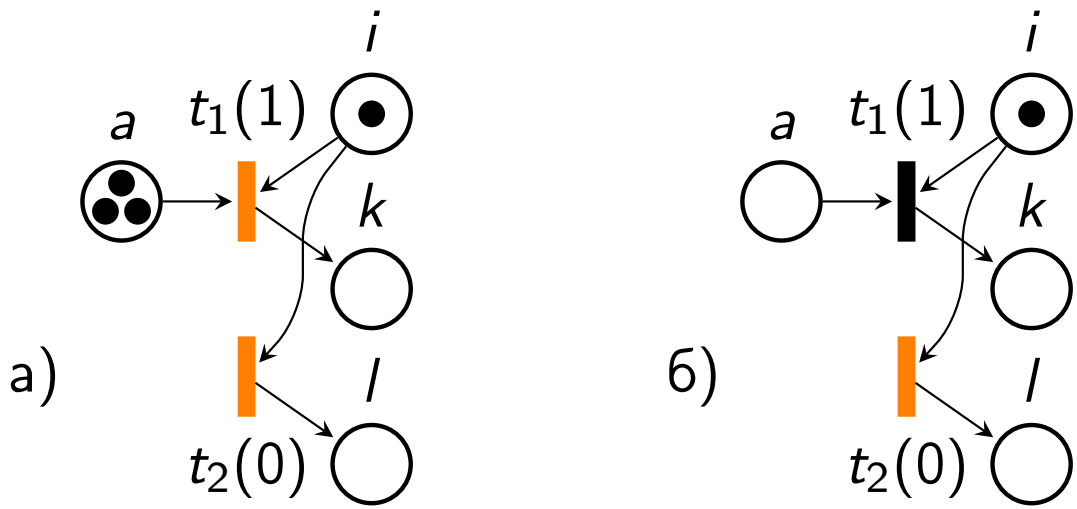
- В настоящее время имеется большое количество различных вариаций сетей Петри, тем или иным образом расширяющих (а иногда и сужающих) функционал этой модели.
- Мы рассмотрим несколько наиболее стандартных расширений:
  - сети с приоритетами;
  - ингибиторные сети;
  - цветные сети;
  - временные сети Петри.

Сети с приоритетами

- Сеть Петри с приоритетами — это стандартная сеть Петри, каждому переходу t которой поставлено в соответствие некоторое число $\Pr t$, называемое приоритетом этого перехода.
- Приоритеты используются для более полного управления разрешением конфликтных ситуаций.
- Если два перехода t_1 и t_2 конфликтуют из-за некоторого общего ресурса, то преимущество получит тот, который имеет больший приоритет.
- В частности, если все переходы данной сети имеют разные приоритеты, то конфликтов в такой сети вообще не будет.
- На практике, однако, достаточно определить только несколько приоритетов для потенциально конфликтных переходов.

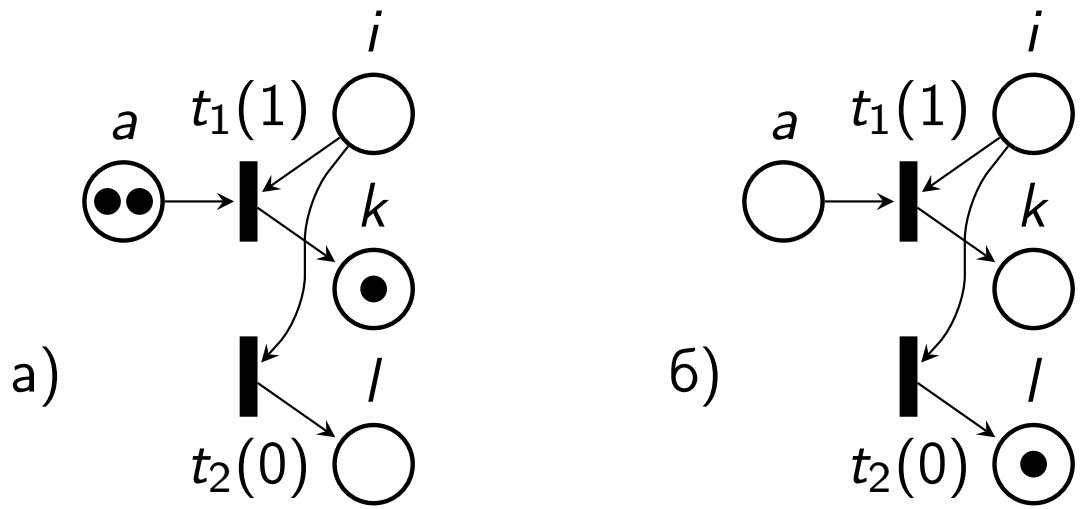
Реализация условного декремента

- В качестве примера рассмотрим сеть, реализующую операцию условного вычитания единицы.
- Если регистр a содержит хотя бы одну фишку (а), то активными являются оба перехода, которые конфликтуют из-за метки в i-ом месте. Т. к. приоритет перехода t_1 больше, то он и сработает.
- Если же регистр a пуст (б), то активным будет только переход t_2, срабатывание которого приведет к активации l-ой команды.



Реализация условного декремента

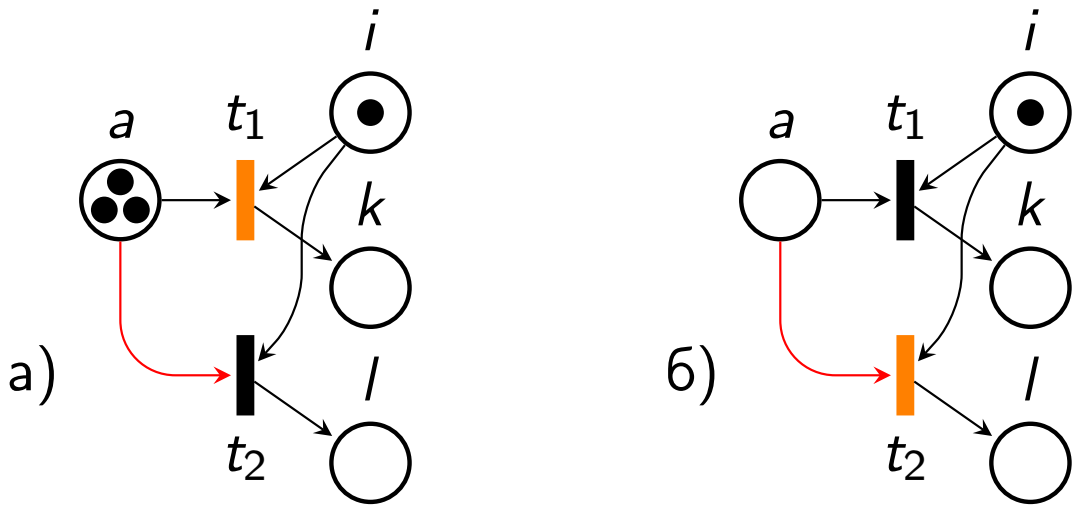
- В качестве примера рассмотрим сеть, реализующую операцию условного вычитания единицы.
- Если регистр a содержит хотя бы одну фишку (а), то активными являются оба перехода, которые конфликтуют из-за метки в i-ом месте. Т. к. приоритет перехода t_1 больше, то он и сработает.
- Если же регистр a пуст (б), то активным будет только переход t_2, срабатывание которого приведет к активации l-ой команды.



Ингибиторные сети Петри

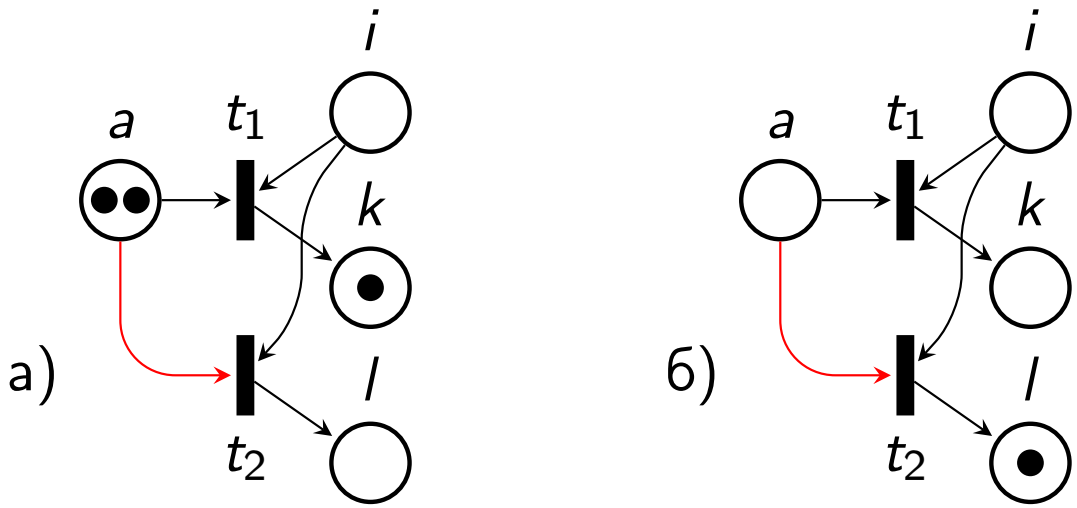
- В ингибиторных сетях Петри к стандартным дугам, ведущим из мест в переходы, добавляется специальный вид *ингибиторных* (тормозящих) дуг.
- Такая дуга, при наличии в соответствующем месте хотя бы одной метки, *препятствует* активации соответствующего перехода.
- Поэтому, такой переход будет активизирован только тогда, когда в данном месте закончатся все метки.
- Таким образом, тормозящие дуги позволяют явным образом выполнять проверку *на отсутствие меток* в заданном месте.

Реализация условного декремента



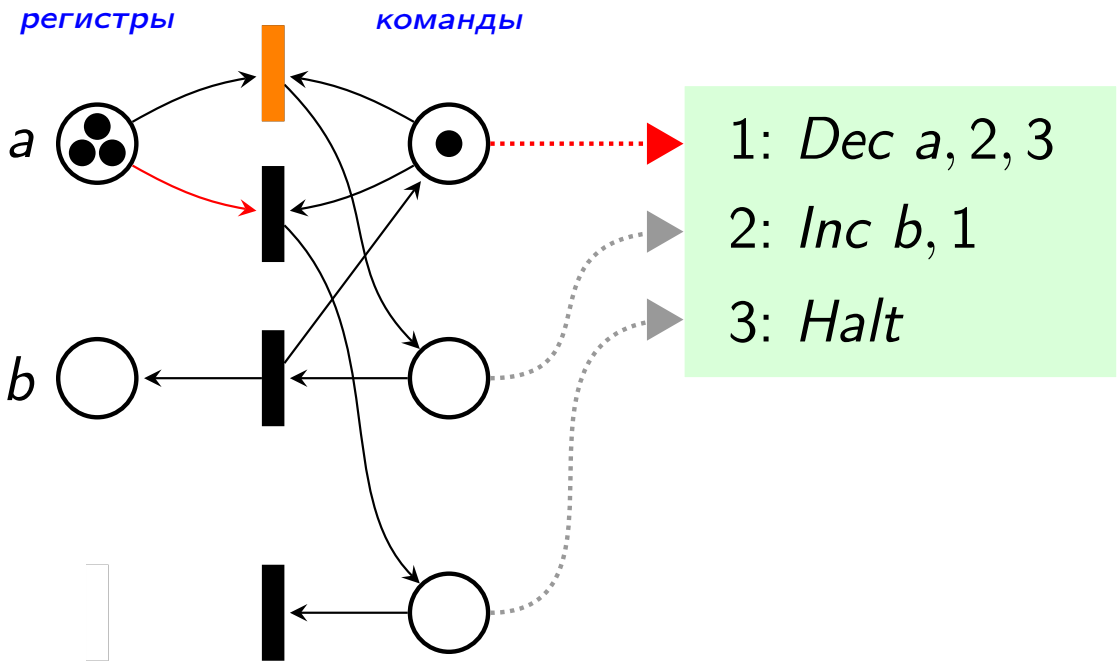
Ингибиторные дуги показаны линиями красного цвета.

Реализация условного декремента



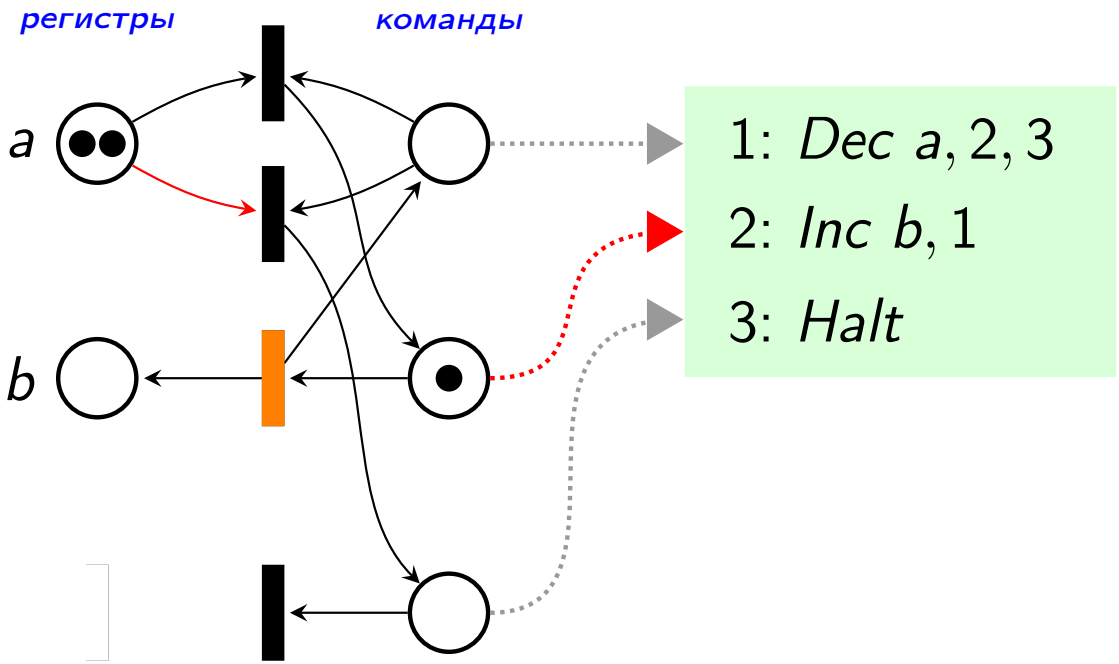
Ингибиторные дуги показаны линиями красного цвета.

Моделирование программы для машины Минского



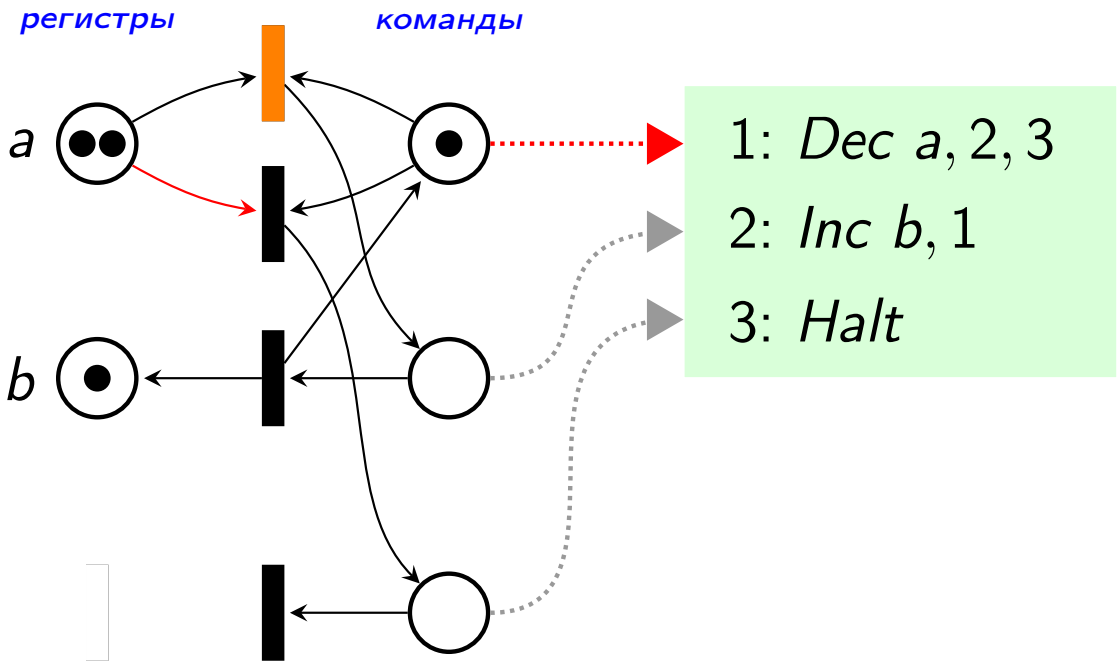
Программа копирования содержимого регистра a в регистр b.

Моделирование программы для машины Минского



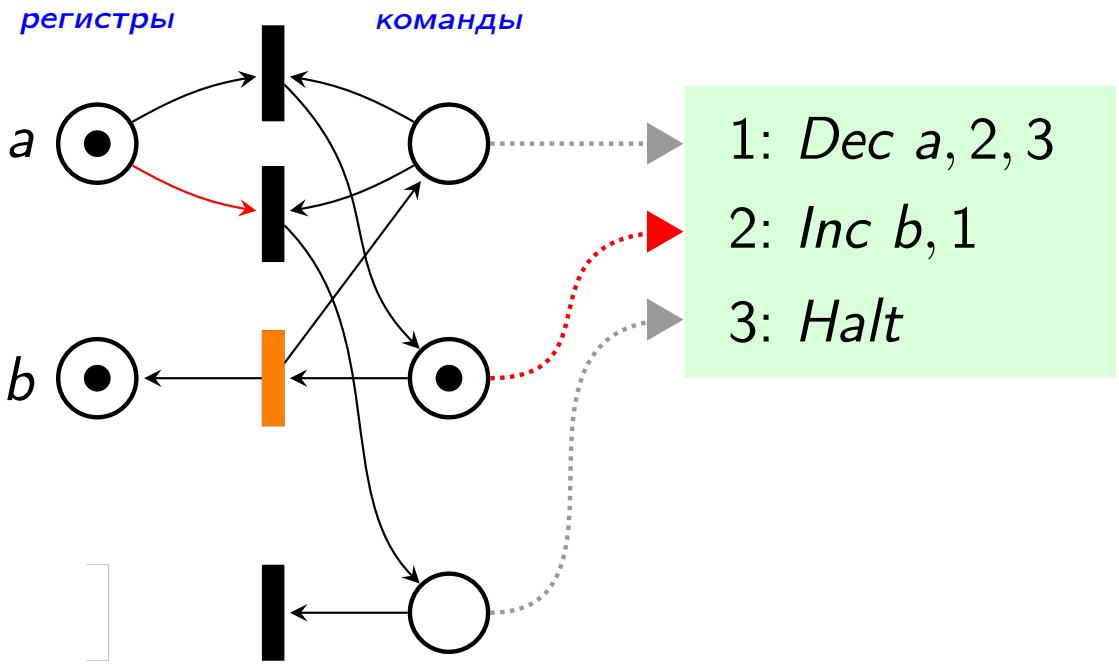
Программа копирования содержимого регистра a в регистр b.

Моделирование программы для машины Минского



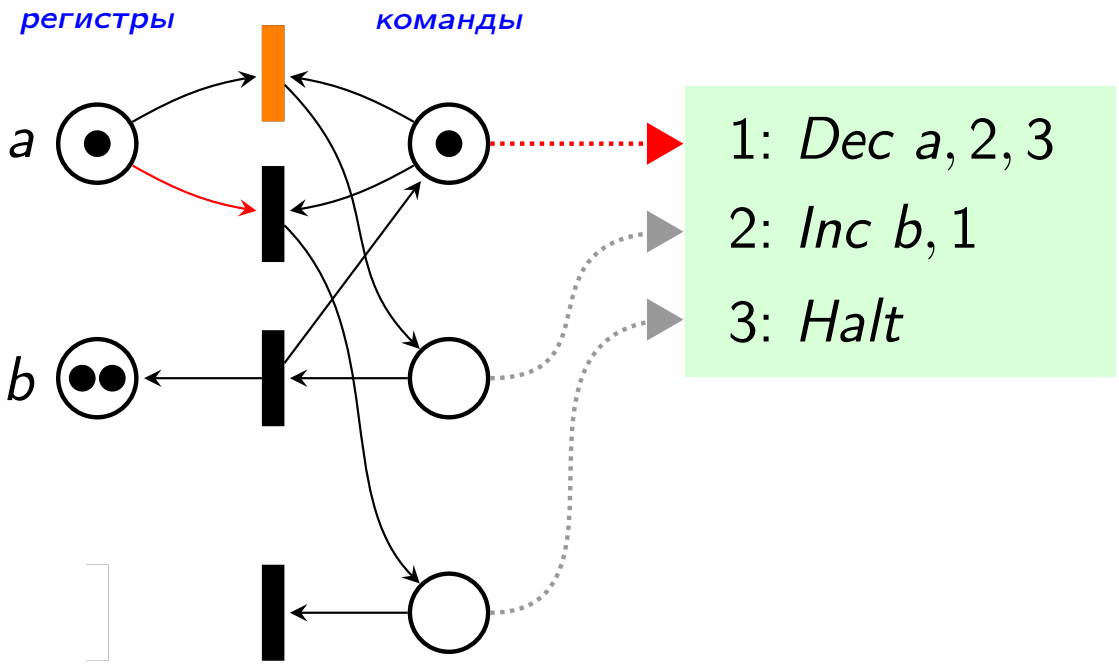
Программа копирования содержимого регистра a в регистр b.

Моделирование программы для машины Минского



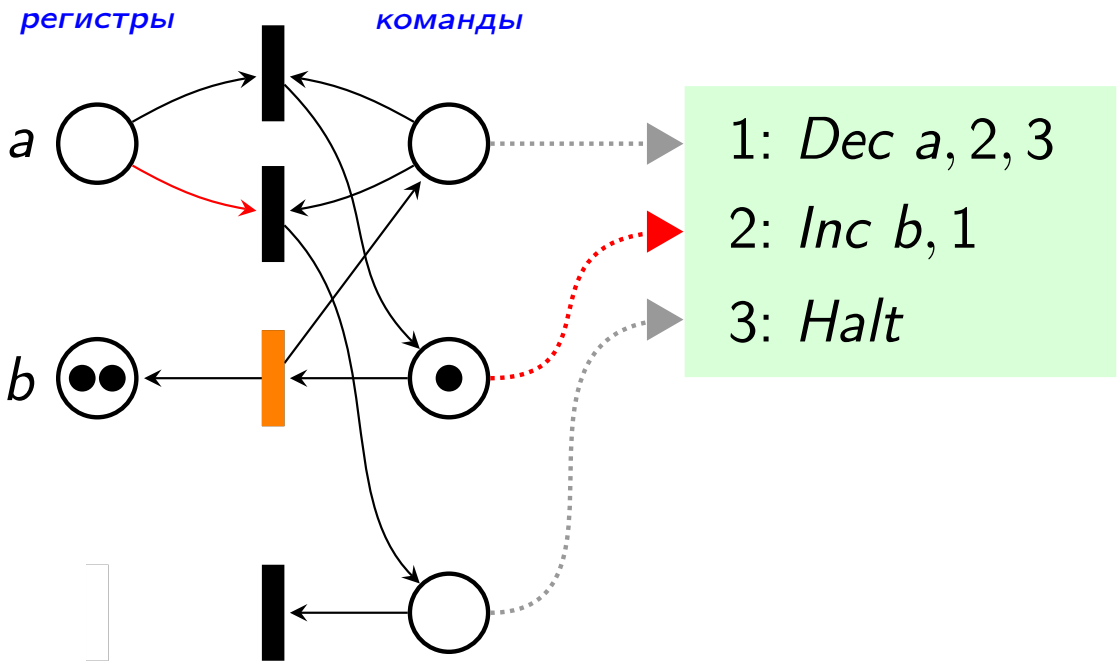
Программа копирования содержимого регистра a в регистр b.

Моделирование программы для машины Минского



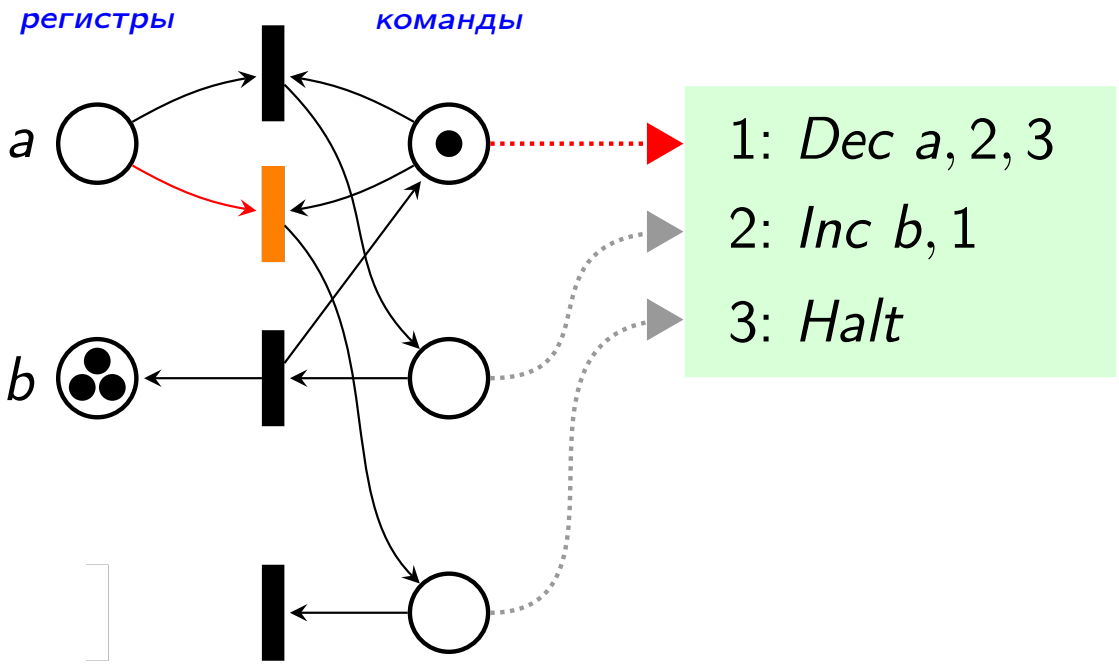
Программа копирования содержимого регистра a в регистр b.

Моделирование программы для машины Минского



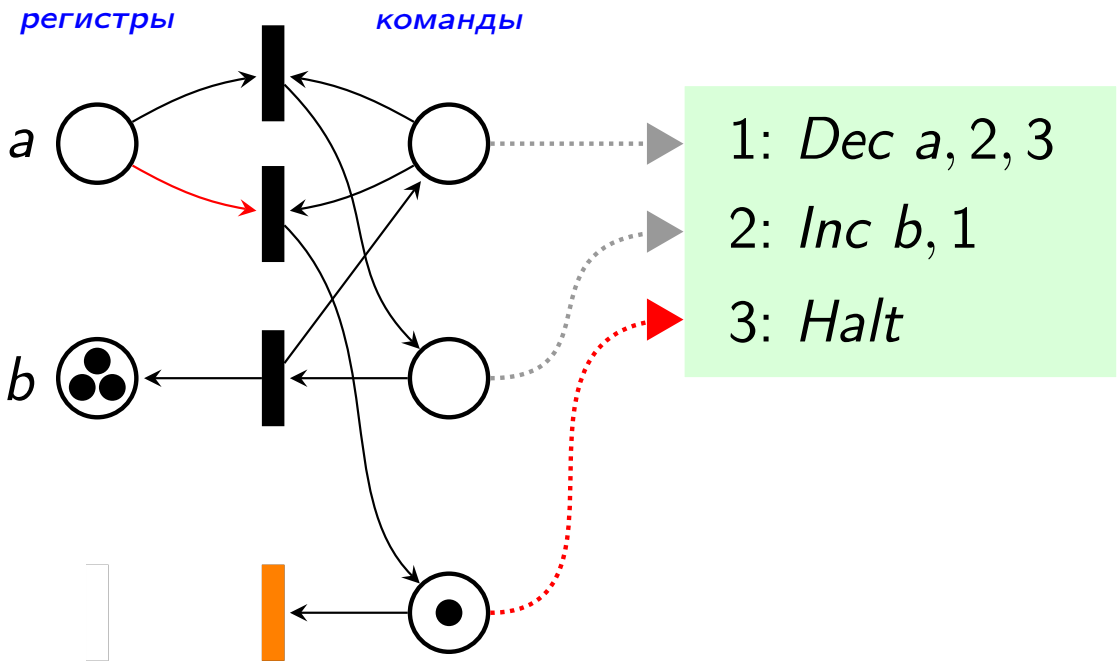
Программа копирования содержимого регистра a в регистр b.

Моделирование программы для машины Минского



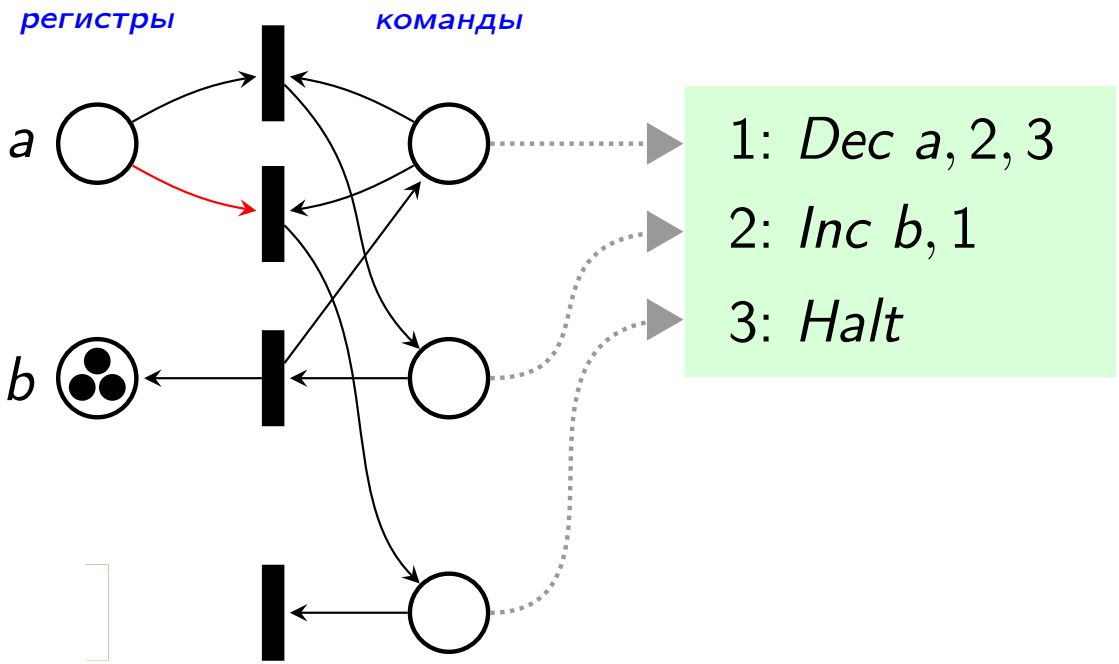
Программа копирования содержимого регистра a в регистр b.

Моделирование программы для машины Минского



Программа копирования содержимого регистра a в регистр b.

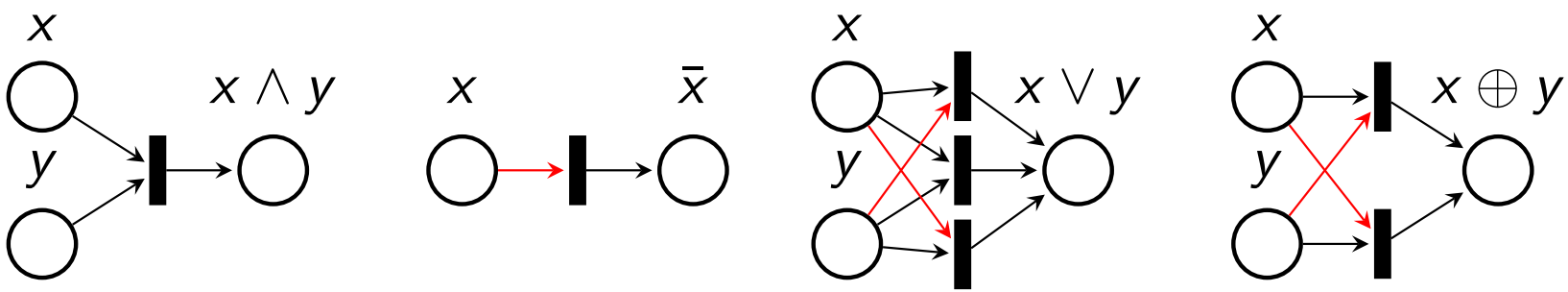
Моделирование программы для машины Минского



Программа копирования содержимого регистра a в регистр b.

Реализация логических операций

- С помощью ингибиторных сетей Петри оказывается возможным реализовать и работу логических элементов.
- На рисунке приведены примеры моделирования четырех базовых логических операций с использованием тормозящих дуг.
- Предполагается, что пустое место соответствует логическому нулю, а место с одной меткой — логической единице.

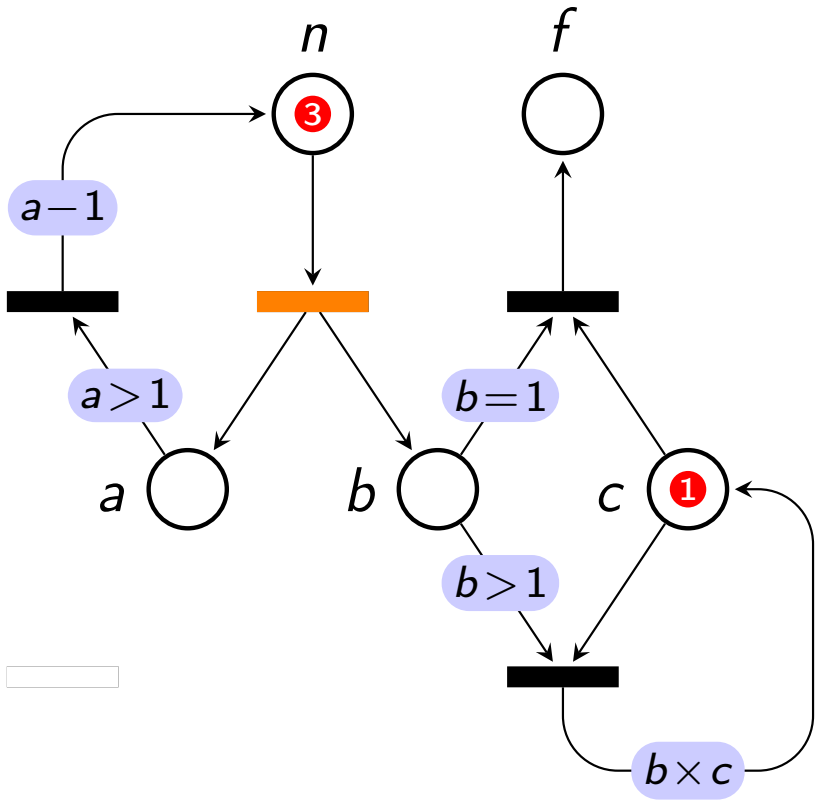


Цветные сети Петри

- В традиционных сетях Петри все метки по определению являются одинаковыми, следовательно, неразличимыми.
- Однако в тех системах, которые моделируются сетями Петри, метки часто представляют собой *различные* объекты.
- В *цветных сетях Петри* каждая метка имеет свое значение (цвет), что дает возможность отличать одни метки от других.
- Значение метки может быть простым, или любого сколь угодно сложного типа.
- Переходы в цветных сетях Петри определяют отношения между значениями входных меток и выходных меток: для этого входным дугам каждого перехода приписываются предусловия, которые определяют, метки с какими значениями поглощаются данным переходом; выходные дуги перехода определяют значения меток, которые будут помещены в соответствующие места.

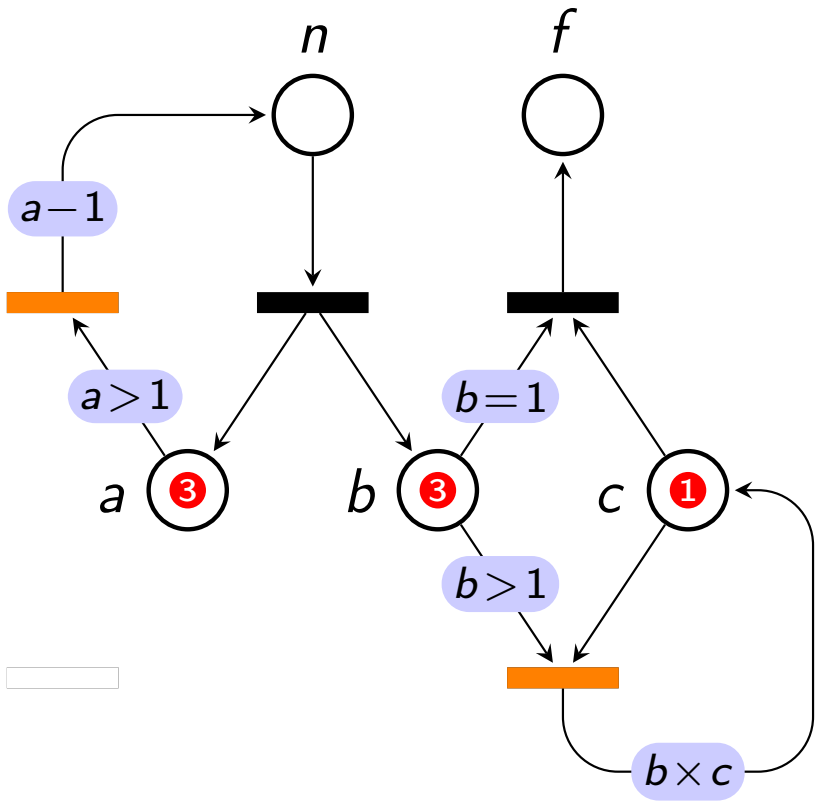
Пример цветной сети Петри

- По своей выразительности цветные сети Петри уже сопоставимы с блок–схемами. В качестве примера на рисунке показана цветная сеть Петри, вычисляющая факториал числа n.



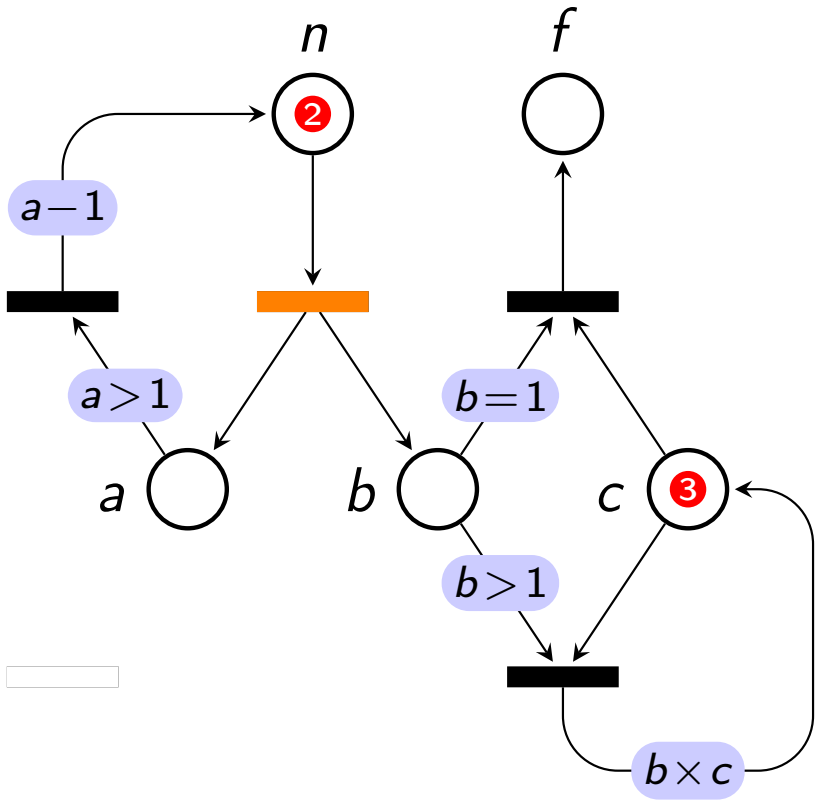
Пример цветной сети Петри

- По своей выразительности цветные сети Петри уже сопоставимы с блок–схемами. В качестве примера на рисунке показана цветная сеть Петри, вычисляющая факториал числа n.



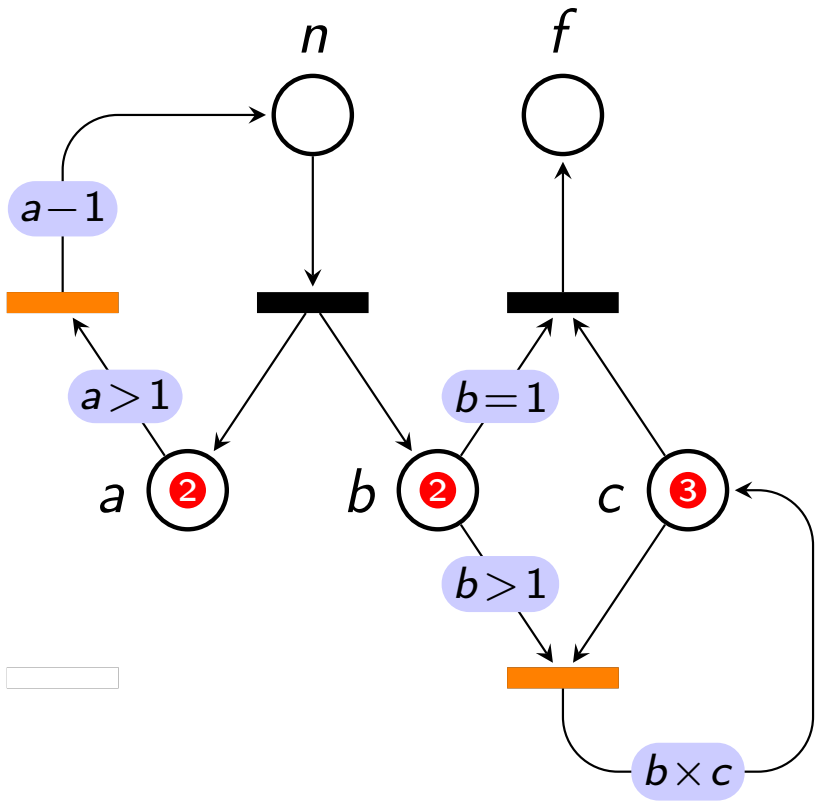
Пример цветной сети Петри

- По своей выразительности цветные сети Петри уже сопоставимы с блок–схемами. В качестве примера на рисунке показана цветная сеть Петри, вычисляющая факториал числа n.



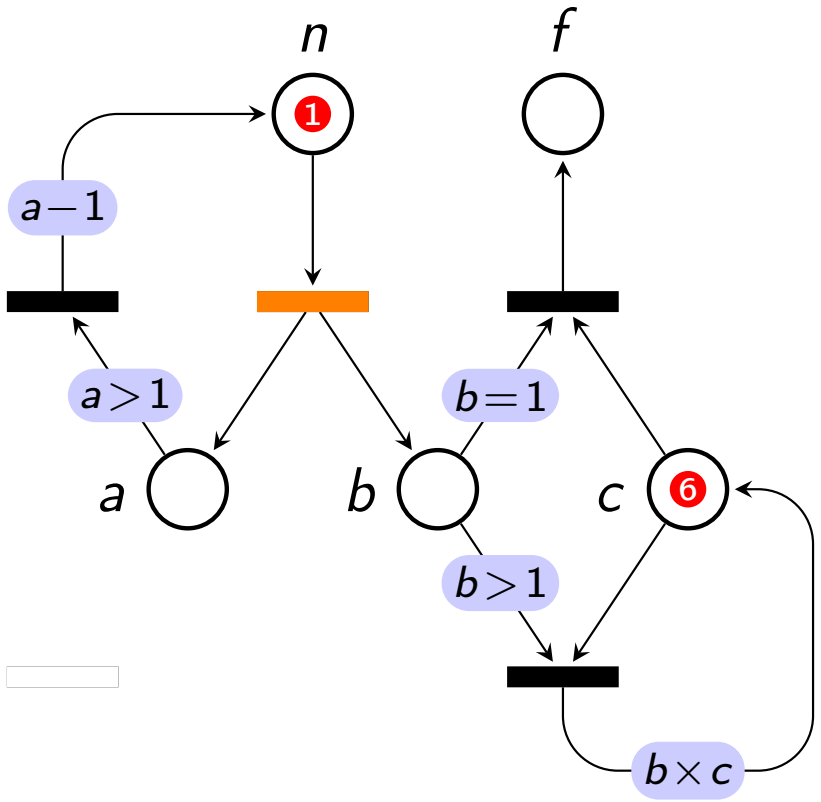
Пример цветной сети Петри

- По своей выразительности цветные сети Петри уже сопоставимы с блок–схемами. В качестве примера на рисунке показана цветная сеть Петри, вычисляющая факториал числа n.



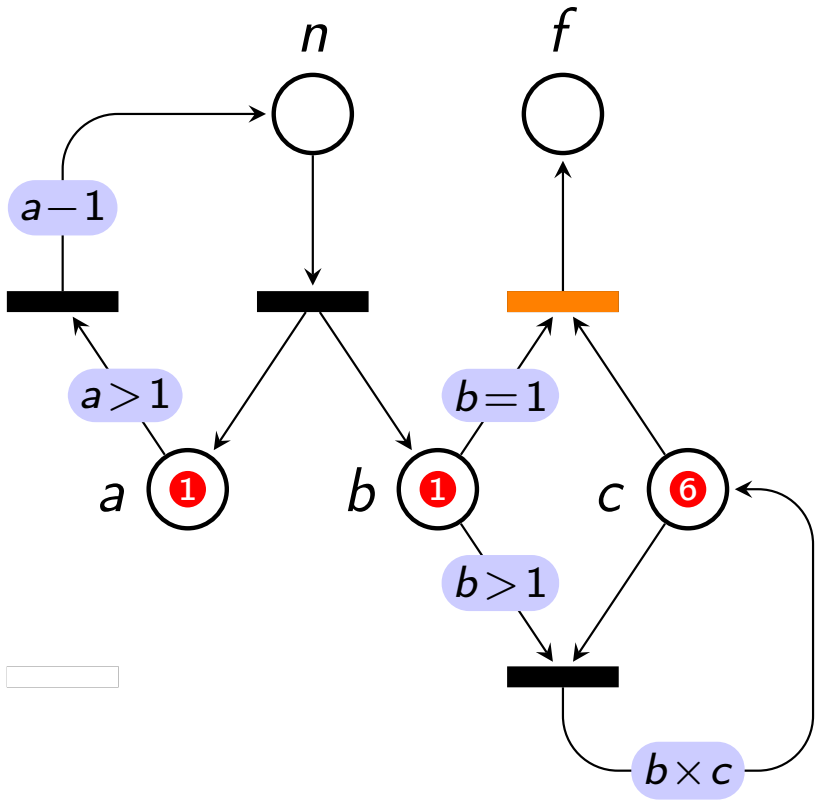
Пример цветной сети Петри

- По своей выразительности цветные сети Петри уже сопоставимы с блок–схемами. В качестве примера на рисунке показана цветная сеть Петри, вычисляющая факториал числа n.



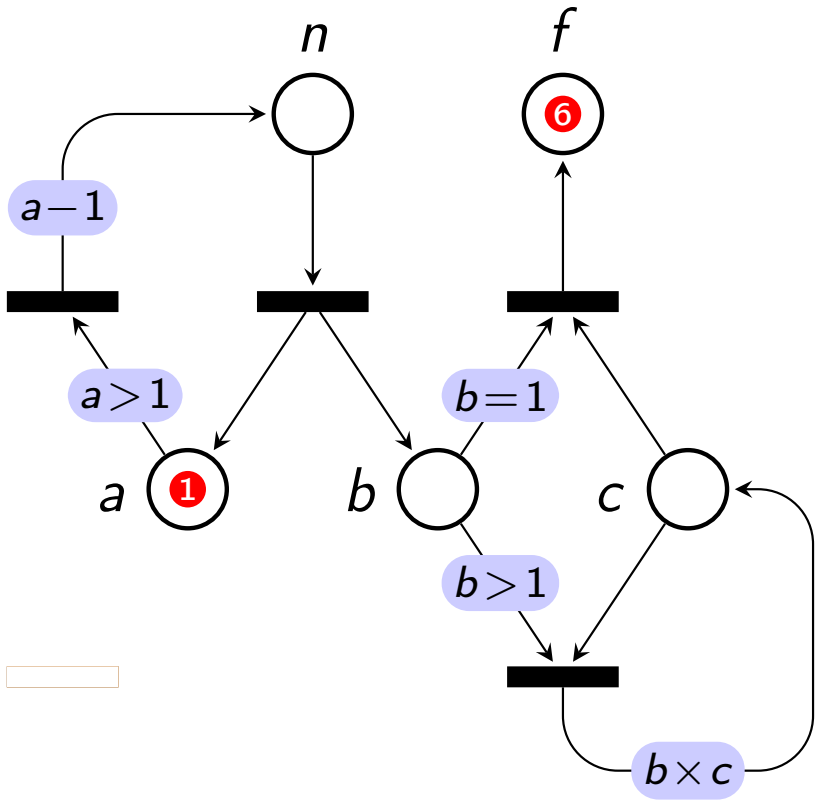
Пример цветной сети Петри

- По своей выразительности цветные сети Петри уже сопоставимы с блок–схемами. В качестве примера на рисунке показана цветная сеть Петри, вычисляющая факториал числа n.



Пример цветной сети Петри

- По своей выразительности цветные сети Петри уже сопоставимы с блок–схемами. В качестве примера на рисунке показана цветная сеть Петри, вычисляющая факториал числа n.

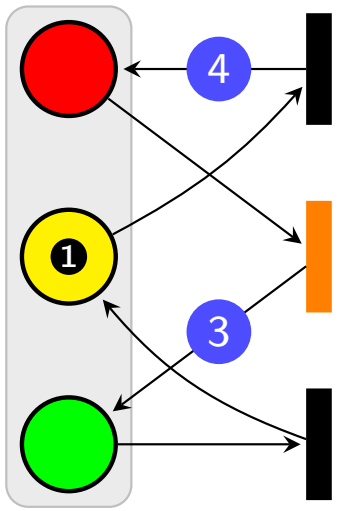


Временные сети Петри

- Для моделирования систем, в которых отдельные подпроцессы имеют различную продолжительность, можно использовать временные сети Петри.
- В таких сетях каждая метка получает дополнительный атрибут — время задержки. Если метка появилась в каком-то месте в момент времени t и имеет время задержки τ, то она будет доступна для всех переходов (связанных с данным местом) начиная с момента времени $t + \tau$.
- Задержки, которые назначаются меткам, приписываются дугам, ведущим от переходов. Соответственно, все метки, поставляемые по какой-либо дуге, будут получать одну и ту же задержку, которая приписана этой дуге.
- По умолчанию считается, что задержка каждой дуги является единичной.

Пример временной сети Петри

- На рисунке ниже приведен пример моделирования с помощью временной сети Петри светофора, у которого продолжительность красного сигнала равна четырем тактам, желтого — одному такту (значение по умолчанию), зеленого — трем тактам.



Сети Петри и параллельные вычисления

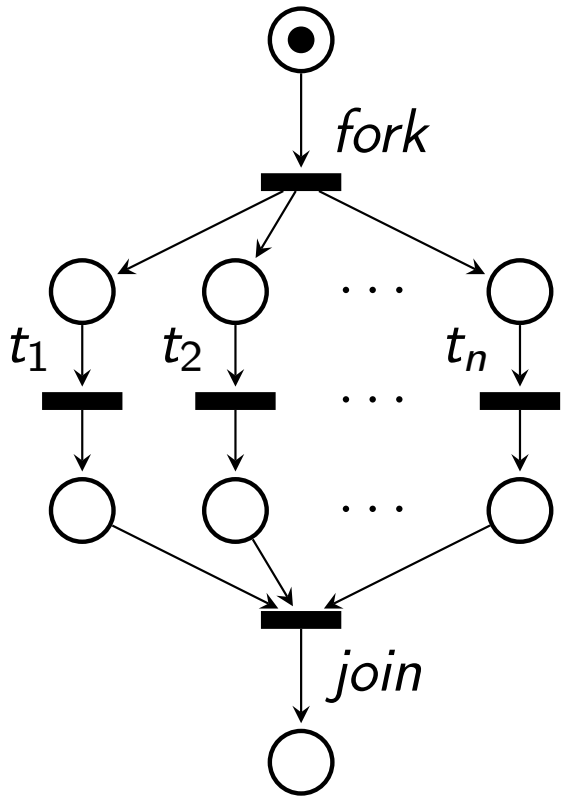
- Сети Петри по своему определению обладают встроенным параллелизмом, поэтому они традиционно используются для моделирования разннообразных параллельных процессов, как искусственных (например, вычислительные сети и системы), так и естественных (например, сети химических реакций).
- Это делает сети Петри с одной стороны адекватным инструментом анализа параллельных процессов, с другой — удобным объектом для реализации на параллельных вычслительных системах.

Моделирование параллельных вычислений

- Стандартная интерпретация сетей Петри с точки зрения параллельных вычислительных процессов заключается в том, что метки представляют собой данные, места — память, в которых хранятся данные, а переходы — подпрограммы, которые обрабатывают эти данные.
- Каждая подпрограмма ожидает, когда будут готовы все ее входные данные, после чего производит необходимые вычисления и помещает результаты на свои выходные места.
- Если какие-либо два перехода не конфликтуют из-за данных, то при некоторых условиях соответствующие им подпрограммы могут быть выполнены одновременно, т. е. параллельно.

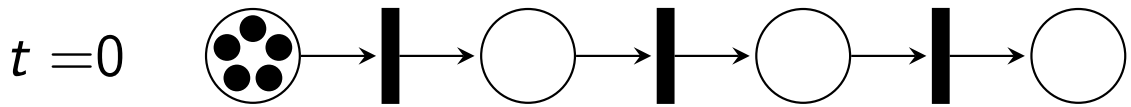
Моделирование операций fork и join

- На рисунке показана сеть Петри, моделирующая две стандартные операции **fork** и **join** для управления параллельными процессами.
- Переход *fork* служит для разветвления данного процесса на n независимых процессов $t_1, \ldots, t_n$, переход *join* — для их синхронизации и слияния обратно в один последовательный процесс.



Конвейерная обработка

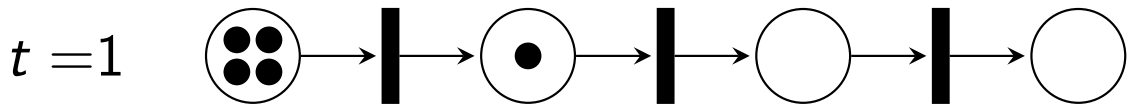
- Так же легко сети Петри позволяют моделировать и конвейерную обработку данных.
- Простейший пример такой модели показан на рисунке.



- Хотя в данном случае каждая метка должна быть обработана тремя переходами, за счет конвейеризации пять меток будут обработаны не за $3 \times 5 = 15$ тактов, а за 7 тактов, т. е. с некоторым ускорением (чем больше данных и длиннее конвейер, тем больше будет ускорение).

Конвейерная обработка

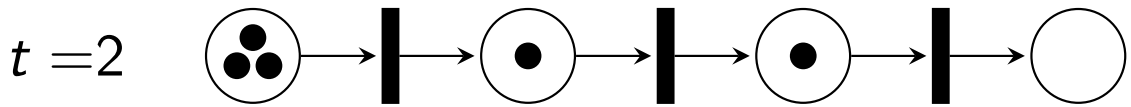
- Так же легко сети Петри позволяют моделировать и конвейерную обработку данных.
- Простейший пример такой модели показан на рисунке.



- Хотя в данном случае каждая метка должна быть обработана тремя переходами, за счет конвейеризации пять меток будут обработаны не за $3 \times 5 = 15$ тактов, а за 7 тактов, т. е. с некоторым ускорением (чем больше данных и длиннее конвейер, тем больше будет ускорение).

Конвейерная обработка

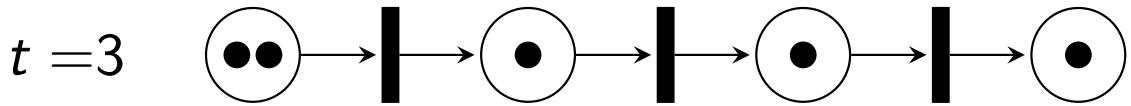
- Так же легко сети Петри позволяют моделировать и конвейерную обработку данных.
- Простейший пример такой модели показан на рисунке.



- Хотя в данном случае каждая метка должна быть обработана тремя переходами, за счет конвейеризации пять меток будут обработаны не за $3 \times 5 = 15$ тактов, а за 7 тактов, т. е. с некоторым ускорением (чем больше данных и длиннее конвейер, тем больше будет ускорение).

Конвейерная обработка

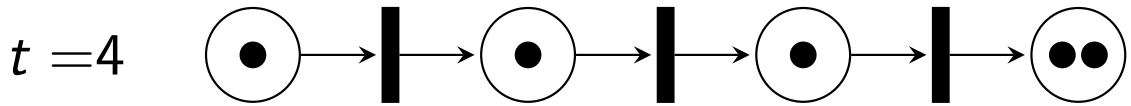
- Так же легко сети Петри позволяют моделировать и конвейерную обработку данных.
- Простейший пример такой модели показан на рисунке.



- Хотя в данном случае каждая метка должна быть обработана тремя переходами, за счет конвейеризации пять меток будут обработаны не за $3 \times 5 = 15$ тактов, а за 7 тактов, т. е. с некоторым ускорением (чем больше данных и длиннее конвейер, тем больше будет ускорение).

Конвейерная обработка

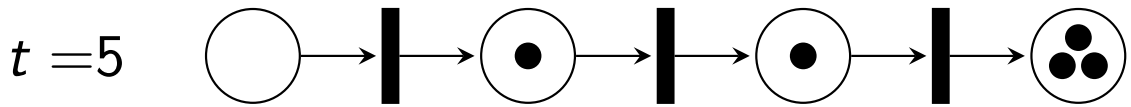
- Так же легко сети Петри позволяют моделировать и конвейерную обработку данных.
- Простейший пример такой модели показан на рисунке.



- Хотя в данном случае каждая метка должна быть обработана тремя переходами, за счет конвейеризации пять меток будут обработаны не за $3 \times 5 = 15$ тактов, а за 7 тактов, т. е. с некоторым ускорением (чем больше данных и длиннее конвейер, тем больше будет ускорение).

Конвейерная обработка

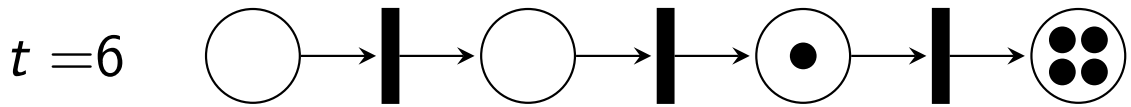
- Так же легко сети Петри позволяют моделировать и конвейерную обработку данных.
- Простейший пример такой модели показан на рисунке.



- Хотя в данном случае каждая метка должна быть обработана тремя переходами, за счет конвейеризации пять меток будут обработаны не за $3 \times 5 = 15$ тактов, а за 7 тактов, т. е. с некоторым ускорением (чем больше данных и длиннее конвейер, тем больше будет ускорение).

Конвейерная обработка

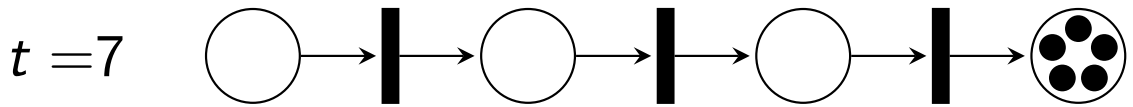
- Так же легко сети Петри позволяют моделировать и конвейерную обработку данных.
- Простейший пример такой модели показан на рисунке.



- Хотя в данном случае каждая метка должна быть обработана тремя переходами, за счет конвейеризации пять меток будут обработаны не за $3 \times 5 = 15$ тактов, а за 7 тактов, т. е. с некоторым ускорением (чем больше данных и длиннее конвейер, тем больше будет ускорение).

Конвейерная обработка

- Так же легко сети Петри позволяют моделировать и конвейерную обработку данных.
- Простейший пример такой модели показан на рисунке.

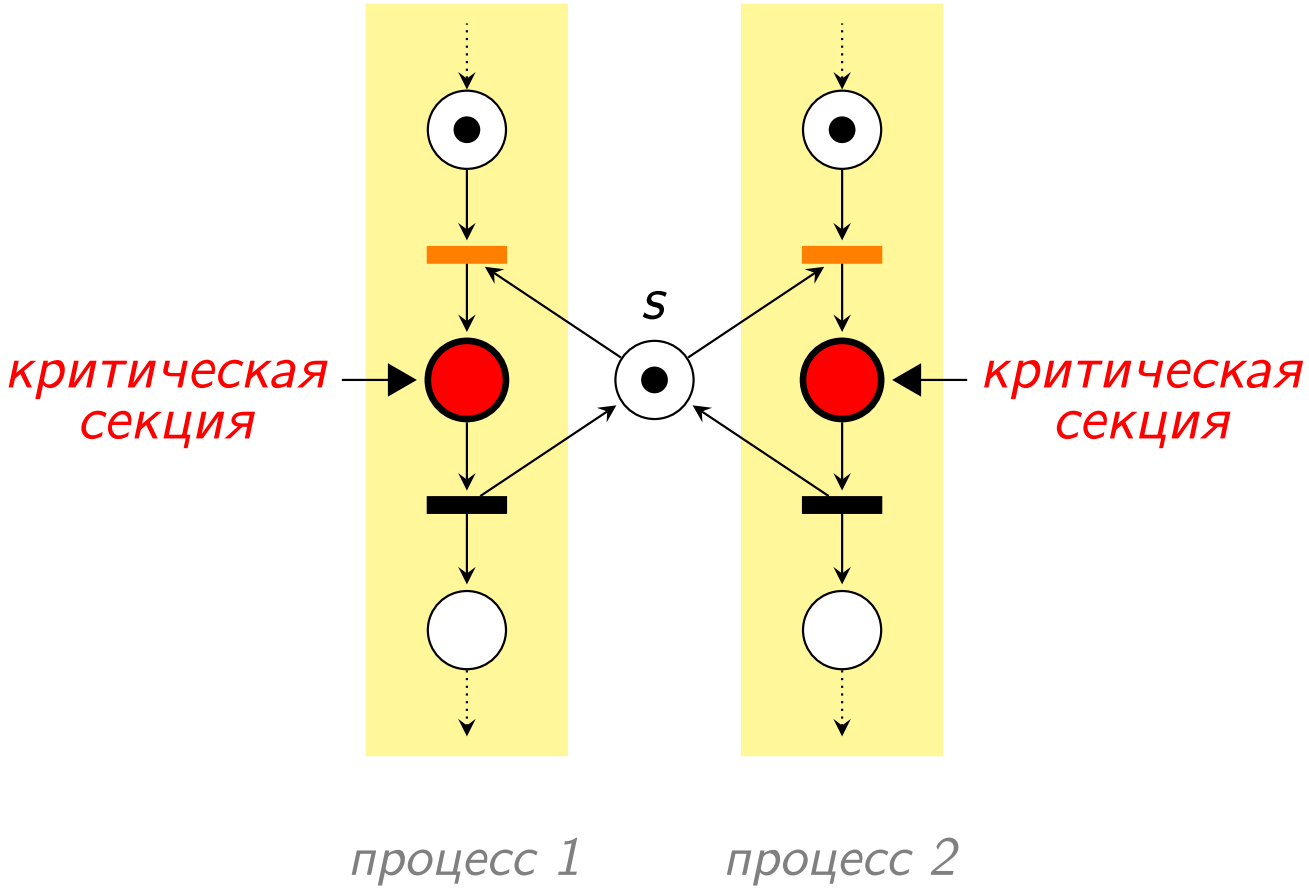


- Хотя в данном случае каждая метка должна быть обработана тремя переходами, за счет конвейеризации пять меток будут обработаны не за $3 \times 5 = 15$ тактов, а за 7 тактов, т. е. с некоторым ускорением (чем больше данных и длиннее конвейер, тем больше будет ускорение).

Синхронизация процессов

- Для организации взаимодействия параллельных процессов применяются различные схемы и механизмы синхронизации, критические секции, семафоры, операции обмена и т. д., которые относительно просто и наглядно моделируются с помощью сетей Петри.
- На рисунке (следующий слайд) показан фрагмент сети Петри, который реализует механизм взаимного исключения, применяемый для обеспечения корректного доступа нескольких процессов к одному разделяемому ресурсу (файлу, устройству).
- Часть процесса, которая используется для доступа и модификации разделяемого объекта, называется *критической секцией*. Выполнение процессом критической секции должно блокировать выполнение другими процессами своих критических секций, связанных с тем же разделяемым объектом.

Моделирование процесса взаимного исключения



Библиография

1. В. Котов, *Сети Петри* — М. : Наука, 1984.
2. Дж. Питерсон, *Теория сетей Петри и моделирование систем* — М. : Мир, 1984.
3. Hsu-Chun Yen, *Introduction to Petri Net Theory* — Recent Advances in Formal Languages and Applications, 2006, pp. 343-373.
4. C. A. Petri, W. Reisig, *Petri net* — Scholarpedia (http://www.scholarpedia.org/article/Petri_net) — *страница Карла Петри, посвященная сетям Петри*.

Спасибо за внимание!