

# Дипломная работа

30 мая 2013 г.

## 1 Введение

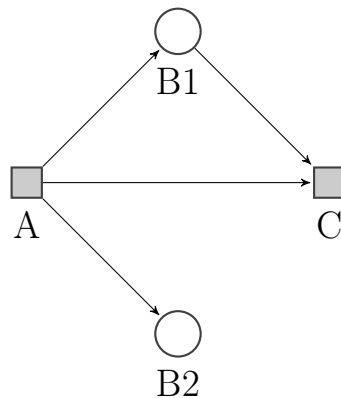


Рис. 1: This is a figure.

Today's scientific experiments typically involve running and refining a series of intertwined computational analysis and visualization tasks on large amounts of data. The complexity of these so-called analysis pipelines resulting in high costs for development and maintenance, the need for sharing knowledge encoded in these pipelines as well as hardware to execute them, and the need for re

Рис. 1

За последних два десятилетия в научном сообществе компьютерное моделирование, названное eScience, стало незаменимой частью исследовательского процесса наравне с традиционными инструментами, такими как эмпирические, основанное на экспериментальных наблюдениях, теоретическое моделирование. Компьютерное моделирование включает в себя

## 1.1 Мотивация

Мотивация

## 2 Базовые понятия и определения

## 3 Формальное описание модели актор-ориентированных исследовательских workflow

В данном разделе будут введены базовые понятия, используемые в данной работе, в том числе, понятия атомарного блока, составного блока, потока данных, автомата Мили соответствующего блоку и так далее.

### 3.1 Графы актор-ориентированные иерархических workflow (Actor-Oriented Hierarchical Workflow Graphs)

**Граф связей workflow** *Граф связей workflow  $W = (A, D)$  состоит из набора акторов  $A$ , представляющих задачи и набора связей  $D$ , соединяющих акторы через порты. Акторы по своей сути являются пассивными, т.е. на набор входных данных выдают. Акторы передают друг другу токены данных через сообщающиеся порты.*

**Порты** *Для каждого актора  $a \in A$  существует собственный набор портов  $port(a)$ , таких что каждый порт  $p \in port(a)$  является либо входящим, либо исходящим для этого актора, т.е.  $port(a) = in(a) \cup out(a)$*

**Связи** *Пусть для workflow  $in(W) = \bigcup_{a \in A} in(a)$  будет набор всех входных портов,  $out(W) = \bigcup_{a \in A} out(a)$  набором выходных портов.*

портов соответственно. Тогда связь  $d \in D$  будет направленным ребром  $d = \langle o, i \rangle$

**Обобщение и дробление workflow** Обобщение и дробление workflow являются ключевыми примитивами в составлении workflow. Для обобщения workflow  $W$  следовало бы представить его в виде составного актора  $a_W$ , иными словами "свернуть". И наоборот, при дроблении, "раскрывая" актор, получить

## 3.2 Описание актора

Базовой логической единицей потока данных является атомарный актор. Т.е. тот, который задаётся только Недетерминированным конечным автоматом(FSM).

Состояние любой системы или подсистемы в любой определённый момент времени характеризует, как система(подсистема) будет реагировать на входные данные. Формально, под состоянием мы будем подразумевать всю накопленную системой информацию, которая влияет на поведение системы в текущий момент и в последующие.

Для примера рассмотрим простой актор типа Counter, который считает количество собственных запусков. У него разумеется есть состояние, и его реакция на входные данные из порта trigger зависит от того, сколько раз он уже сработал. Для подсчёта и знания числа запусков он использует локальную переменную. И эту локальную переменную будем называть **переменной состояния**.

В этом случае количество возможных состояний определяется типом счётчика. Если, к примеру, это `int`, то возможно только  $2^{32}$  состояний, если `double`, то  $2^{64}$ . Если же тип счётчика `String`, то число состояний бесконечно. Но не смотря на то что число состояний может быть очень большим, логика переходов между ними достаточно проста.

**Finite State Machines** Автоматом называется система, выходы которой зависят не только от поступивших входов, но и от текущего состояния системы. Состояние системы - обобщение всей информации о предшествующих входах, необходимое системе для выполнения шага. Состояние системы может быть обозначено переменной состояния  $s \in \Sigma$ , где  $\Sigma$  - это набор всех возможных состояний системы. Конечным автоматом называется автомат,

для которого число состояний  $\Sigma$  конечно.

Внутренняя логика блока представляется моделью недетерминированного конечного автомата (*Finite State Machine, FSM*) Определение: конечным автоматом называется набор  $M = (\Sigma, I, \Lambda, T, s_0)$ , где

- $S$  -набор конечных состояния,
- $I$  – набор входных портов актора,
- $O$  - набор возможных выходных портов актора
- $s_0 \in \Sigma$  - начальное состояние,
- $T : S \times 2^I \rightarrow 2^{S \times 2^O}$  отображение сопоставляющее каждому состоянию с набором входных портов набор состояний с соответствующим набором выходных портов

Как уже было определено выше, у актора имеется **начальное состояние**  $s$ , т.е. то в котором модель находится перед запуском. Так же могут быть и **конечные состояния**. Визуально состояния соединены переходами, рядом с которыми указано, что происходит при срабатывании работе.

Классическая теория конечных автоматов (Hopcroft and Ullman, 1979) различает два вида автоматов: **Автомат Мили** и **Автомат Мура**. В Автомате Мили, от входных значений, в отличии от Автомата Мура, выходное значение сигнала в котором зависит лишь от текущего состояния данного автомата.

## 4 The Need for Composing Models of Computation in E-science

## 5 Workflows and Hierarchy

Описание понятия актора(actor,atomic actor. ,composite actor (a.k.a. sub-workflow))

## 6 (Модели управления потоком) Models of Computation

### Process Networks (PN)

#### 6.1 Использование PN

- Тупики
- 

Более подробно модель описывается через Kahn networks. Каждый актор запускается в отдельно треде(thread)(или вычислительном узле), и все акторы запускаются в конкурирующем режиме.

### SDF

#### 6.2 Задачи

#### 6.3 Использование SDF

Deadlocks

- Consistency of data rates
- The value of the iterations parameter
- The granularity of execution Более подробно в секции 3.3.

#### 6.4 Тупики(Deadlocks)

#### 6.5 Consistency of data rates

Все акторы внутри *однородного(homogeneous)* SDF принимают только по одному токену из каждого входного порта и выписывают по 1 токену в каждый выходной порт. В каждом цикле должно быть "сдерживающий"актор, так что dataflow-граф будет ациклическим, если убрать все "сдерживающие"акторы. Схему запуска можно определить статически, например, через топологическая сортировка графа.

## 6.6 Свойства SDF

SDF не связан с временными событиями. Для всех акторов, входящих в SDF, поглощение токенов из входных портов, выполнение вычисленных операций и отправливание токенов в выходные порты является атомарной операцией. Запуск композитного актора соответствует одной итерации содержащейся в нём модели по предварительно вычисленной схеме(schedule). При этой схеме выполнения рассчитывается так, чтобы, и при бесконечном числе итераций в модели не возникали тупики и накопление токенов.

### 6.6.1 Вычисление схем запуска

Вложенные друг в друга SDF модели, можно уплотнять(flattern), Запуск последовательный, преимущественно на одном узле.

## FSM

Конечные автоматы (Finite State Machines)

## DDF

В этой модели управления вместо того, чтобы выдавать каждому актору отдельный тред, система управления запускает отдельный актор, когда необходимые удовлетворены его зависимости на входе. Dataflow делятся на два типа, dynamic dataflow (DDF) и synchronous dataflow (SDF). В случае DDF, система управления динамически определяет какой актор необходимо запустить в следующем, и следовательно составляет схему запуска(firing schedule) динамически во время работы. В случае SDF, система управления

- Сети Петри
- Анализ Сетей Петри
- Граф Карпа и Миллера
- Marked Graphs

- Однородные dataflow
- Обобщённые dataflow
- Модели Кана (Khan) для параллельных вычислений

9.0 Специальная лексика Dataflow - поток данных

10.0 Используемая литература

## 7 СЕТИ

### 7.1 Сети Петри

Сети Петри широко используются для моделирования и исследования динамических дискретных систем. И прежде чем рассмотреть частные случаи использования Сетей Петри, приведём описание их каноничной формы, согласно определению Петерсона [Pet81].

Сеть Петри представляет собой двудольный ориентированный граф, состоящий из вершин двух типов — *позиций* и *переходов*, соединённых между собой дугами. Вершины одного типа не могут быть соединены непосредственно. В позициях могут размещаться метки (маркеры), способные перемещаться по сети.

**Определение: Сеть Петри** *Простой сетью Петри называется набор  $PN = (S, T, F)$ , где*

1.  $S = \{s_1, \dots, s_n\}$  - множество позиций
2.  $T = \{t_1, \dots, t_r\}$  - множество переходов таких, что  $S \cap T = \emptyset$ .
3.  $F \subseteq \mu S \times T \times \mu S$  - отношение инцидентности такое, что
  - $\forall \langle Q'_1, t_1, Q''_1 \rangle, \langle Q'_2, t_1, Q''_2 \rangle \in F : \langle Q'_1, t_1, Q''_1 \rangle \neq \langle Q'_2, t_1, Q''_2 \rangle \Rightarrow t_1 \neq t_2;$
  - $\{t | \langle Q'_1, t, Q''_1 \rangle \in F\} = T$

Условия в пункте 3 говорят, что для каждого перехода  $t \in T$  существует единственный элемент  $\langle Q', t, Q'' \rangle$ , задающий для него входное множество  $Q'$  и выходное множество  $Q''$ . Дадим определение входному и выходному множеству.

**Определение: Входное и выходное множества мест и переходов**

Пусть задана сеть  $N = (S, T, F)$ .

1. Если для некоторого перехода  $t$  имеем  $\langle Q', t, Q'' \rangle \in F$ , то будем обозначать

$$\bullet t = Q' = \langle s | t \in s \bullet \rangle, t \bullet = Q'' = \langle s | t \in \bullet s \rangle$$

2. И соответственно

$$\bullet s = \langle t | s \in t \bullet \rangle, s \bullet = \langle t | s \in \bullet t \rangle$$

Будем говорить, что  $\bullet t$  - входные, а  $t \bullet$  - выходные позиции для перехода  $t$ . Таким образом, согласно определению, справедливо:

$$\forall t \in T : \langle \bullet t, t, t \bullet \rangle \in F.$$

Позиция  $s$  называется инцидентной переходу  $t$ , если  $s \in \bullet t$  или  $s \in t \bullet$ .

Сети Петри имеют удобную графическую форму представления в виде графа, в котором места изображаются кружками, а переходы прямоугольниками. Места и переходы, причем место  $s$  соединяется с переходом  $t$  если  $s \in \bullet t$  и  $t$  соединяется с  $s$  если  $s \in t \bullet$ .

Само по себе понятие сети имеет статическую природу. Для задания динамических характеристик используется понятие маркировки сети  $M \in \mu S$ , т.е. функции  $M : S \rightarrow N_0$ , сопоставляющей каждому месту целое число. Графически маркировка изображается в виде точек, называемых метками (tokens), и располагающихся в кружках, соответствующих местам сети. Отсутствие меток в некотором месте говорит о нулевой маркировке этого места.

**Определение: Маркированная сеть Петри** Маркированной сетью Петри называется набо  $(PN, M_0)$ , где

1.  $PN = (S, T, F)$  - сеть;

2.  $M_0 \in \mu S$  - начальная маркировка.

Сети Петри были разработаны и используются для моделирования параллельных и асинхронных систем. При моделировании в сетях Петри места символизируют какое-либо состояние системы, а переход символизируют какие-то действия, происходящие в системе. Система, находясь в каком-то состоянии, может порождать определенные действия, и наоборот, выполнение какого-то действия переводит систему из одного состояния в другое.



Текущее состояние системы определяет маркировка сети Петри, т.е. расположение меток (токенов) в местах сети, т.е.  $M \in S \rightarrow N$ . Под маркировку, представленной в виде:  $1s_1 + 2s_2 + 1s_3 + 0s_4$ , понимаем, что в позиции  $s_1$  находится 1 метка, 2 метки в  $s_2$ , 1 метка в  $s_3$  и ни одной метки в  $s_4$ . Так же представление этой маркировки можно сократить до  $1s_1 + 2s_2 + 1s_3$ . Для сравнения двух маркировок, мы определяем частичный порядок.

$$\forall_{M_1, M_2} M_1 \leq M_2, \forall_{s \in S} : M_1(s) \leq M_2(s).$$

Выполнение действия в системе, в сетях Петри определяется как срабатывание переходов. Срабатывание переходов порождает новую маркировку, т.е. порождает новое размещение меток (токенов) в сети.

Работа маркировочной сети Петри управляется наличием или отсутствием маркировочных токенов. В сети Петри срабатывает переход  $t$ , в процессе которого с каждого входа  $s \in \bullet t$  снимается метка и каждому выходу  $s \in t \bullet$  прибавляется метка.

**Определение: Правило срабатывания переходов** Пусть  $\Sigma = (S, T, F, M_0)$  маркировочная сеть.

1. Переход  $t \in T$  считается возбуждённым при маркировке  $M \in \mu S$ , если каждое положение  $s \in \bullet t$  имеет хотя бы одна метка;
2. Переход  $t$ , возбуждённый при маркировке  $M$ , может сработать, приведя к новой маркировке  $M'$ , которая получается путём поглощения метки на каждом позиции  $s \in \bullet t$  и появления новой метки на каждой позиции  $s' \in t \bullet$ .

Для сети Петри  $(S, T, F)$  и её текущей маркировки  $M_1 \in \mu S$  опишем следующие обозначения:

- $M_1 \xrightarrow{t} M_2$ : переход  $t$  при маркировке  $M_1$  возбуждён и при его срабатывании приводит к маркировке  $M_2$
- $M_1 \rightarrow M_2$ : существует переход  $t$ , такой что  $M_1 \xrightarrow{t} M_2$
- $M_1 \xrightarrow{\sigma} M_n$ : последовательность переходов  $\sigma = t_1 t_2 t_3 \dots t_{n-1} \in T^*$  переводящая маркировку  $M_1$  в  $M_n$  через набор (возможно пустой) промежуточных маркировок  $M_2, \dots, M_{n-1}$ , т.е.:  $M_1 \xrightarrow{t_1} M_2 \xrightarrow{t_2} \dots \xrightarrow{t_{n-1}} M_n$

Маркировка  $M_n$  называется *достижимой* из  $M_1$  (и имеет обозначение  $M_1 \xrightarrow{*} M_n$ ) тогда и только тогда, когда существует последовательность такая переходов  $\sigma$ , что  $M_1 \xrightarrow{\sigma} M_n$ . Отметим, что пустая последовательность переходов тоже допустима,  $M_1 \xrightarrow{*} M_1$ .

Когда же мы имеем сеть Петри с начальной маркировкой  $(S, T, F, M_0)$ , то маркировка  $M'$  будет допустимой, если  $M_0 \xrightarrow{*} M'$ .

**Определение: Живость(Live)** Маркированная сеть Петри  $(PN, M)$  называется *живой*, если для любой достижимой маркировки  $M'$  и любого перехода  $t$  существует маркировка  $M''$  достижимая из маркировки  $M'$  и задеиствующая переход  $t$ .

**Определение: Ограниченность(Bounded), безопасность(Safe)** Маркированная сеть Петри  $(PN, M)$  называется *ограниченной*, если для каждого положения  $s$  существует такое натуральное число  $n$ , что для каждой достижимой маркировки меток в этом положении меньше  $n$ . Сеть называется *безопасной*, если максимальное число меток не превышает 1.

**Определение: Правильность(Well-formed)** Сеть Петри  $PN$  называется *правильной*, если существует такая начальная маркировка  $M_0$ , что сеть  $(PN, M_0)$  будет живой и ограниченной.

**Определение: Путь** Пусть дана сеть Петри  $PN$ . Путь  $C$  из вершины  $n_1$  в вершину  $n_k$  представляет собой последовательность вершин  $\langle n_1, n_2, \dots, n_k \rangle$  такую, что  $\langle n_i, n_{i+1}, n_{i+2} \rangle \in F$  для  $1 \leq i \leq k - 2$ .

Путь  $C$  называется *простым*, если для любых двух вершин  $n_i, n_j \in C$  выполняется:  $i \neq j \Rightarrow n_i \neq n_j$ .

Путь  $C$  называется *безконфликтным*, ели для любой положения  $n_j$  и любого перехода  $n_i$ ,  $n_i, n_j \in C$ , выполняется  $j \neq i - 1 \Rightarrow n_j \notin \bullet n_i$ . Для удобства введём оператор  $\alpha$  над путями. Тогда для пути  $C = \langle n_1, n_2, \dots, n_k \rangle$   $\alpha(C) = \{n_1, n_2, \dots, n_k\}$

**Определение: Сильная связность(Strongly Connected)** Сеть Петри называется *сильно звязной*, если для любой пары вершин  $x$  и  $y$  существует путь из  $x$  в  $y$ .

**Определение: Свободный выбор(Free-Choice)** Сеть Петри обладает свойством *свободного выбора*, если для любых двух переходов  $t_1, t_2 \in T$  выполняется  $\bullet t_1 \cap \bullet t_2 \neq \emptyset \Rightarrow \bullet t_1 = \bullet t_2$

## ПРИМЕРЫ

**Определение: WF-сеть(Workflow-net)** Пусть дана сеть Петри  $PN = (S, T, F)$ .

- Если  $PN$  - WF-сеть с входной позицией  $i$ , тогда для любой позиции  $s \in S : \bullet s \neq \{s = i\}$ , т.е.  $i$  -единственная входная позиция.
- Если  $PN$  - WF-сеть с выходной позицией  $o$ , тогда для любой позиции  $s \in S : s\bullet \neq \{s = o\}$ , т.е.  $o$  -единственная выходная позиция.
- Если  $PN$  - WF-сеть и мы добавляем в  $PN$  переход  $t^*$ , соединяющий выходную позицию  $o$  и входную позицию  $i$  (т.е.  $\bullet t^* = o, t^*\bullet = i$ ), тогда получившаяся сеть будет обладать сильной связностью.

Для WF-сети начальную маркировку с единственной меткой только во входной позиции  $i$  будем обозначать  $i$ , а маркировку с единственной меткой только в выходной позиции  $o$  соответственно  $o$ .

**Определение: Корректность(Soundness)** WF-сеть  $PN = (S, T, F)$  корректна тогда и только тогда, когда:

- (i) Для каждой маркировки  $M$ , достижимой из  $i$ , маркировка  $o$  так же достижима из  $M$ .

$$\forall M (i \xrightarrow{*} M) \Rightarrow (M \xrightarrow{*} o)$$

- (ii) Маркировка  $o$  - единственная достижимая из  $i$  маркировка, имеющая хотя бы одну метку в позиции  $o$ .

$$\forall M (i \xrightarrow{*} M \wedge M \geq o) \Rightarrow (M = o)$$

- (iii) В  $(PN, i)$  нет неживых переходов.

$$\forall t \in T \exists_{M, M'} i \xrightarrow{*} M \xrightarrow{t} M'$$

Рассмотрим примеры нарушения условий корректности WF-сетей.

- Для WF-сети, изображённой на Рис. 2(а) нарушено первое условие корректности. Срабатывание любого из переходов  $t_1, t_2$  или  $t_3$  приводит к маркировке, из которой маркировка  $o$  недостижима. При маркировке  $s_1$  мы имеем бесконечный цикл, а в случае маркировки  $s_2$  или  $s_3$  переход  $t_5$  никогда не сработает, т.к. требует по метке на каждой из входных позиций.

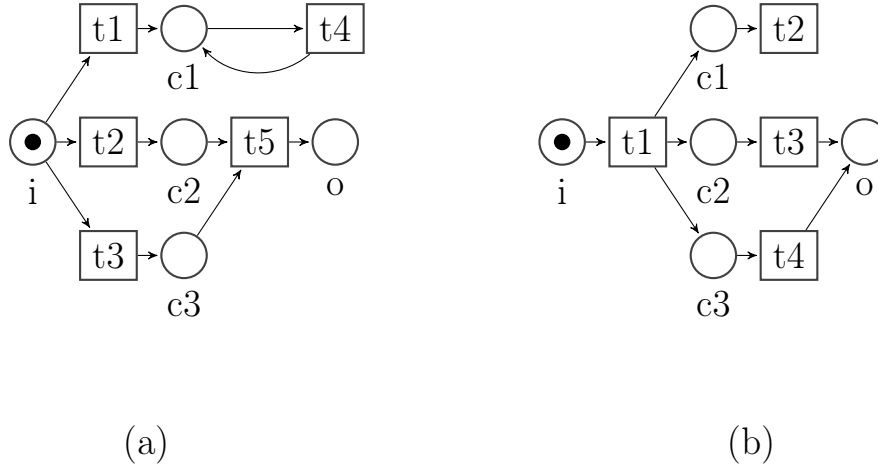


Рис. 2: This is a figure.

- В результате выполнения последовательности переходов  $\langle t1, t3, t4 \rangle$  WF-сети Рис. 2(b) нарушается второе условие корректности,
- В сети , изображённой на Рис. 2(b), так же нарушено третье условие корректности, т.к. переход t2 - неживой.

Для данной WF-сети  $PN = (S, T, F)$  мы хотим определить, является ли она корректной. В [2] показано, как свойство корректности соотносится с живостью и ограниченностью сети. Для того, чтобы связать эти понятия, дадим определение расширенной сети  $\overline{PN} = (\overline{S}, \overline{T}, \overline{F})$ , где  $\overline{S} = S$ ,  $\overline{T} = T \cup \{t^*\}$ ,  $\overline{F} = F \cup \langle o, t^* \rangle, \langle t^*, i \rangle$ .

**Теорема 1.** *WF-сеть  $PN$  корректна тогда и только тогда, когда  $(\overline{PN}, i)$  - живая и ограниченная маркировочная сеть.*

## 7.2 Структурные характеристики корректности

Корректности WF-сети является её динамической характеристикой и из это вытекают некоторые проблемы:

- Для сложных WF-сетей задача определения корректности может быть весьма затратной (Для произвольных WF-сетей вычисление ограниченности и живости имеет экспоненциальную сложность[8]).
- Теорема 1 не определяет в какие именно компоненты WF-сети нарушают свойство корректности.

Поэтому полезно было бы знать какими структурными характеристиками обладают корректные WF-сети. Для этого будут рассмотрены некоторые классы WF-сетей: WF-сети со свободным выбором, хорошо структурированные WF-сети/

### 7.2.1 WF-сети со свободным выбором

Напомним определение свободного выбора в сети. Если для любых двух переходов  $t_1, t_2 \in T$ , для которых выполняется  $\bullet t_1 \cap \bullet t_2 \neq \emptyset$ , то необходимо чтобы  $\bullet t_1 = \bullet t_2$ , т.е. эти переходы являются частью одного OR-split.

Соблюдение правила свободного выбора не мешает моделировать параллельное, последовательное и циклическое выполнение задач.

Если же мы будем допускать сети не обладающие свойством свободного выбора, то тогда на выбор между конкурирующими задачами влияет очерёдность в которой выполнялись предшествующие задачи, что является недопустимым при грамотном проектировании. Нарушение свободы выбора, чаще всего возникает при сочетании распараллеливания задач и маршрутизации с выбором.

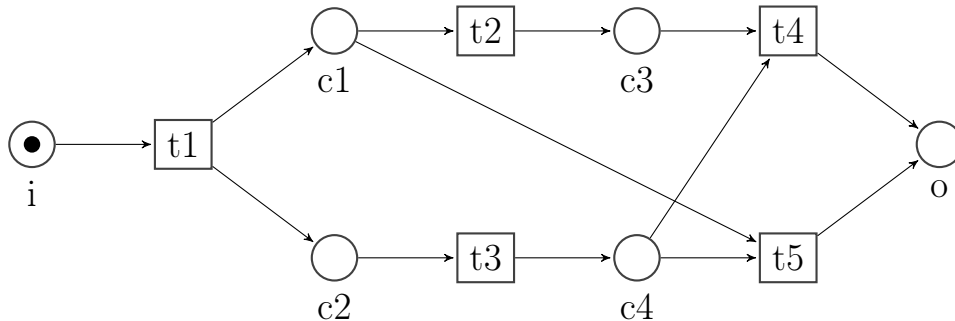


Рис. 3: This is a figure.

На Рис. 3 изображена такая ситуация. Сработавший переход  $t_1$  вводит параллелизм. Но тем не менее, в выбор между  $t_2$  и  $t_5$ , т.к. переход  $t_5$  не возмуждён. Параллельное выполнение  $t_2$  и  $t_3$  приводит к ситуации, когда выполнить переход  $t_5$  невозможно. Тем не менее, если выполнение перехода  $t_2$  отложено до окончания выполнения  $t_3$ , то тогда будет существовать выбор между  $t_2$  и  $t_5$ . Но хотелось бы, чтобы параллелизм был отделён от выбора между альтернативами. Поэтому мы считаем конструкцию, приведённую на Рис. 3, неверной.

**Вывод 1.** Для WF-сети со свободным выбором, задача проверки на корректность решается за полиномиальное время.

*Док-во.* Пусть PN - WF-сеть со свободным выбором, тогда  $\overline{PN}$  тоже будет обладать свободным выбором. В работе [10] доказано, что определение живости и ограниченности для  $(\overline{PN})$ , можно за полиномиальное время. По Теорема 1 это эквивалентно корректности WF-сети.

**Лемма 1.** Корректная WF-сеть со свободным выбором - безопасная.

*Док-во.* Пусть PN - корректная WF-сеть со свободным выбором. Тогда  $\overline{PN}$  тоже будет со свободным выбором и правильностью (well-formed). Следовательно,  $\overline{PN}$  S-покрываема [10], т.е. каждая позиция в  $\overline{PN}$ . И т.к. в начальной маркировке есть всего одна метка, то  $(\overline{PN}, i)$  будет безопасной, а следовательно и  $(PN, i)$ .

Безопасность является предпочтительным свойством, т.к. нелогично было бы допускать несколько меток в позиции, представляющей состояние. Состояние может быть активным (1 метка) или неактивным (ни одной метки). Non-free-choice constructs such as the construct shown in Figure 4 are a potential source of anomalous behavior (e.g., deadlock) which is difficult to trace.

## 7.2.2 Хорошо структурированные WF-сети

Другой подход для определения хорошей структурной характеристики workflow - это сбалансированность компонент типов AND/OR-split и AND/OR-join. Очевидно, что два параллельных потока иницированных AND-split, не должны объединяться через OR-join. Также как и два альтернативных потока, иницированных компонентом OR-split не должны синхронизироваться через AND-join.

Для того чтобы формализовать конструкции на рисунке Рис. 4 дадим следующее определение.

**Хорошая организованность (Well-handled)** Сеть Петри PN называется хорошо организованной, если для любых пар вершин  $x$  и  $y$  таких, что одна из вершин является позицией, а другая переходом и для любой пары простых путей  $C_1$  и  $C_2$  из  $x$  в  $y$ ,  $\alpha(C_1 \cup \alpha(C_2)) = \{x, y\} \Rightarrow C_1 = C_2$ .

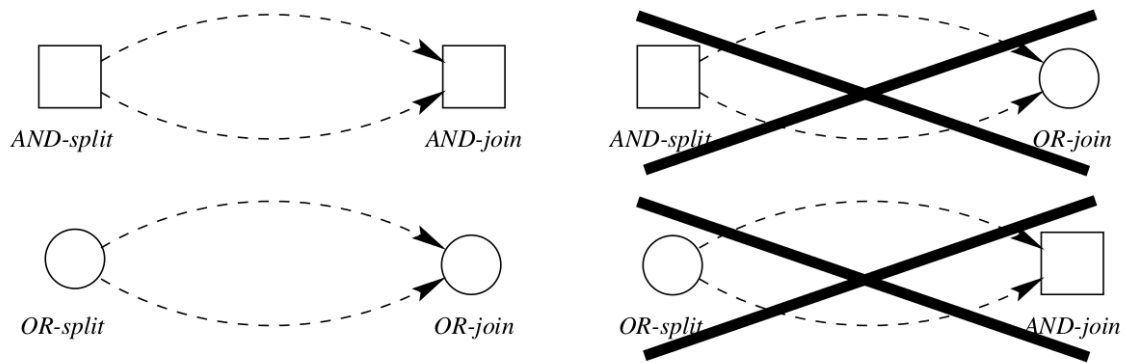


Рис. 4: Хорошие и плохие конструкции

Хорошая организованность может быть определена за полиномиальное время применяя алгоритмы максимального потока и минимального среза, описанные в [5].

**Лемма 2.** *Хорошая организованность сеть петри с сильной связностью будет хорошо организованной.*

*Доказательство.* Пусть  $PN$  - хорошая организованность сеть петри с сильной связностью. Очевидно, что не будет существовать ни одно пути

**Хорошая структурированность (Well-structured)  $WF$ -сеть** является хорошей структурированной, если  $\overline{PN}$  хорошая организованная.

## Список литературы

- [1] [GG] Gray L., Griffeath D. The ergodic theory of traffic jams // J. Stat.