Трансляция вложенных сетей Петри в классические сети Петри для верификации разверток

В.О. Ермакова <ermakovavo@gmail.com>
И.А. Ломазова <ilomazova@hse.ru>
National Research University Higher School of Economics,
20 Myasnitskaya St., Moscow, 101000, Russia

Аннотация. Вложенные сети Петри являются одним из удобных формализмов для моделирования и анализа поведения распределенных мультиагентных систем. Они естественным образом представляют структуру мультиагентных систем, так как фишки в системной сети сами являются классическими сетями Петри и могут иметь автономное поведение. Мультиагентные системы являются системами с высоким уровнем параллелизма. При верификации таких систем методами проверки модели (model checking) возникают серьезные трудности, связанные с взрывным ростом числа промежуточных состояний системы (state-space explosion problem). Для решения этой проблемы в литературе был предложен подход, основанный на построении развертки поведения системы. Ранее была изучена применимость разверток для верификации вложенных сетей Петри и предложен метод построения разверток для безопасных консервативных вложенных сетей Петри. В этой работе предлагается другой метод построения разверток для безопасных консервативных вложенных сетей Петри, основанный на трансляции таких сетей в классические сети Петри. Для классических сетей Петри затем применяются стандартные методы построения разверток. Также в работе обсуждаются сравнительные достоинства двух подходов.

Ключевые слова: мультиагентные системы; верификация; сети Петри; вложенные сети Петри; развертки.

DOI: 10.15514/ISPRAS-2016-28(4)-7

Для цитирования: Ермакова В.О., Ломазова И.А. Трансляция вложенных сетей Петри в классические сети Петри для верификации разверток. Труды ИСП РАН, том 28, вып, 4, 2016, стр. 115-136. DOI: 10.15514/ISPRAS-2016-28(4)-7

1. Введение

Мультиагентные системы активно изучаются уже в течение нескольких десятилетий. Они используются в различных практических областях, таких как искусственный интеллект, облачные сервисы, грид системы, системы

встроенной реальности с интерактивными агентами среды, сборе информации, кооперации мобильных агентов и коммуникации.

Мультиагентные системы сложны из-за их распределенной структуры. Они состоят из взаимодействующих агентов, имеющих общую среду и автономное поведение. Когда разрабатывается такая система, важно проверить, будет ли она отвечать необходимым требованиям. Для сложных распределенных систем обычно сначала строят и анализируют (верифицируют) модель системы и только после проверки корректности модели переходят к этапу реализации.

Одним из формализмов, успешно представляющих поведение распределенных систем, являются сети Петри. Однако из-за плоской структуры классических сетей Петри они оказываются неудобны для моделирования сложных мультиагентных систем. Для таких систем используется специальные расширения сетей Петри, в частности, вложенные сети Пети [1]. Вложенные сети Петри естественно представляют структуру и поведение мультиагентных систем, так как фишки в системной сети сами являются сетями Петри и имеют собственное поведение.

Для проверки свойств сетей Петри часто используются методы верификации, основанные на проверке моделей (model checking). Основная идея этого подхода заключается в построении графа достижимости и проверке свойств на полученном графе. Однако при использовании этого метода для верификации высоко параллельных систем возникает серьезная проблема, связанная с размером графа достижимости системы. Эта проблема известна как проблема взрывного роста числа состояний (state-space explosion problem) – число промежуточных состояний системы растет экспоненциально от числа независимых параллельных агентов.

Одним из решений этой проблемы является проверка свойств системы не на графе достижимости, а на так называемой развертке (unfolding) ее поведения [2,3]. Ранее в работе [4] было показано, как теория разверток может быть применена для верификации вложенных сетей Петри, а именно, для безопасных консервативных вложенных сетей Петри было дано определение развертки и описан алгоритм ее построения. Было доказано, что для вложенных сетей Петри выполняется фундаментальное свойство разверток и, следовательно, развертки вложенных сетей могут быть использованы для верификации консервативных вложенных сетей Петри так же, как классические развертки используются для верификации классических сетей Петри.

В этой работе описывается другой способ построения разверток для безопасных консервативных вложенных сетей Петри. Консервативность означает, что сетевые фишки, представляющие агентов, не могут быть уничтожены или созданы, но могут изменять свое положение в системной сети, а также менять своё внутреннее состояние, т.е. количество агентов в системе не меняется. Безопасность означает, что в каждой позиции системной сети может одновременно находиться не более одной сетевой фишки (агента).

Мы показываем, что для любой безопасной консервативной вложенной сети Петри можно построить эквивалентную классическую сеть Петри, а затем применить метод построения разверток для классических сетей. Полученная в результате развертка будет изоморфна развертке вложенной сети Петри, построенной методом, описанным в [4].

1.1 Сравнение с другими исследованиями

Вложенные сети Петри широко используются в моделировании распределенных систем [5,6,7], последовательных и реконфигурируемых системах [8,9,10], верификации протоколов [11], координации сенсорных сетей с мобильными агентами [12], инновационных архитектурах космических систем [13], распределенных вычислениях [14].

В литературе было предложено несколько методов для поведенческого анализа вложенных сетей Петри, среди них композициональные методы для проверки ограниченности и живости вложенных сетей Петри [15], трансляция вложенных сетей Петри в раскрашенные сети Петри и верификация их с помощью CPNtools [16], верификация подкласса рекурсивных вложенных сетей Петри с помощью SPIN [17].

Подход, основанный на построении развертки, и проблема взрывного роста числа состояний подробно описаны в литературе. Начало разработкам в области построения разверток для классических сетей Петри было положено в [18]. К. МакМилан [2] был первым, кто использовал развертки для верификации. Он представил концепцию конечных префиксов разверток и показал применимость этого подхода для верификации асинхронных цепей.

Исходный алгоритм МакМиллана был использован для решения проблемы выполнимости перехода – проверить, может данный переход сработать или нет. Этот алгоритм применим также для проверки наличия дедлоков и для решения некоторых других проблем. Позже улучшения алгоритма были представлены в [19,20,21]. Имеются также работы по применению разверток для верификации высокоуровневых сетей Петри [22], алгебр процессов [23] и М-сетей [22].

Общий подход для отсечения разверток с сохранением информации в конечном префиксе развертки предложен в [24,25]. Этот метод основан на понятии усеченного контекста. Мы используем этот подход для определения ветвящегося процесса и развертки консервативной вложенной сети Петри.

1.2 Структура работы

В разделе 2 даны основные определения сетей Петри и вложенных сетей Петри. В разделе 3 представлен алгоритм для трансляции безопасных консервативных вложенных сетей Петри в классические. Раздел 4 посвящен сравнению метода построения разверток на основе трансляции вложенной сети Петри в классическую и метода непосредственного построения развертки вложенной сети. Последний раздел содержит выводы и обсуждения результатов работы.

2. Предварительные сведения

Пусть S – конечное множество. Мультимножеством m над множеством S называется функция $m: \mathrm{S} \to Nat$, где Nat – множество неотрицательных целых чисел. Другими словами, мультимножество может содержать несколько копий одного и того же элемента.

Для двух мультимножеств m и m' полагаем $\mathrm{m} \subseteq m'$, если $\forall s \in S : m(s) \leq m'(s)$ (отношение включения). Сумма и объединение двух мультимножеств m и m' также определяются стандартно: $\forall s \in S : (m + m')(s) = m(s) + m'(s), (m \cup m')(s) = max\big(m(s), m'(s)\big)$.

2.1 Классические сети Петри

Пусть P и T – два конечных непересекающихся множества позиций и переходов и $F \subseteq (P \times T) \cup (T \times P)$ – функция инцидентности. Тогда $N = (P, T, F)$ является сетью Петри. Разметкой сети $N = (P, T, F)$ называется мультимножество над множеством позиций P. Через $\mathcal{M}(N)$ будем обозначать множество всех разметок сети N. Размеченная (маркированная) сеть Петри (N, M_0) – это сеть Петри вместе с её начальной разметкой M_0.

Графически сеть Петри представляется в виде ориентированного графа, в котором вершины-позиции изображаются кругами, а вершины-переходы – прямоугольниками. Позиции могут содержать фишки, представленные закрашенными кружками. Текущая разметка m определяется помещением $m(p)$ фишек в каждую позицию $p \in P$.

Для перехода $t \in T$ дуга (x, t) называется входящей дугой, а (t, x) – исходящей. Для каждой вершины $x \in P \cup T$ мы определяем пред-множество элементов для вершины x как $\bullet x = \{y \mid (y, x) \in F\}$.

Мы говорим, что переход t в сети Петри $N = (P, T, F)$ активен в разметке M если $\bullet t \subseteq M$. Активный переход может сработать и произвести новую разметку $M' = M - \bullet t + t \bullet$ (обозначается как $M \xrightarrow{t} M'$). Разметка M называется достижимой, если существует (возможно, пустая) последовательность срабатываний $M_0 \xrightarrow{t1} M_1 \xrightarrow{t2} M_2 \to \cdots \to M$ из начальной разметки в M. Через $\mathcal{RM}(N)$ обозначим множество всех достижимых разметок в N.

Разметка M называется безопасной, если для всех позиций $p \in P$ имеем $M(p) \leq 1$. Маркированная сеть Петри называется безопасной, если каждая достижимая разметка $M \in \mathcal{RM}(N)$ безопасна. Граф достижимости сети Петри (N, M_0) представляет детальную информацию о поведении сети. Это помеченный ориентированный граф, в котором вершины являются достижимыми разметками сети (N, M_0), а дуги соответствуют срабатываниям переходов. В графе достижимости дуга t между разметками M и M'существует тогда и только тогда, когда $M \xrightarrow{t} M'$.

2.2. Развертки классических сетей Петри

Развертки используются для представления семантики истинного параллелизма (true concurrency) сетей Петри. Для верификации используются конечные префиксы развёрток. Здесь мы приводим необходимые основные понятия и определения, связанные с развертками. Более детальное описание можно найти в [26,27].

Пусть, $N = (P, T, F)$ – сеть Петри. Следующие отношения определены на множестве $P \cup T$ вершин в N:

- Отношение (каузальной) зависимости (обозначается $<$) – это транзитивное замыкание F, соответственно, $\leq$ – рефлексивное замыкание $<$; мы говорим, что y зависит от x, если $x < y$.
- Отношение конфликта (обозначается #): для вершин $x, y \in P \cup T, x\#y \coloneqq \exists t, t' \in T. t \neq t' \wedge {}^\bullet t \cap {}^\bullet t' \neq \emptyset \wedge t \leq x \wedge t' \leq y$;
- Отношение параллельности (обозначается co): две вершины сети параллельны, если они не находятся в конфликте и ни одна из них не зависит от другой.

Для множества вершин B мы пишем $co(B)$ если все вершины в B являются попарно параллельными.

Сетью событий называется безопасная сеть Петри $ON = (B, E, G)$ такая, что:

- ON – ациклична;
- $\forall p \in B: |{}^\bullet p| \leq 1$;
- $\forall x \in B \cup E$ множество $\{y \mid y < x\}$ конечно, то есть каждая вершина в ON имеет конечное число предшествующих вершин;
- $\forall x \in B \cup E: \neg(x\#x)$, то есть, ни одна вершина не находится в конфликте с собой.

В сетях событий элементы из B обычно называются условиями, а элементы принадлежащие E – событиями.

Сети событий представляют поведение системы в семантике «истинного параллелизма» (true concurrency semantics). Семантика истинного параллелизма отличается от последовательной (интерливинговой) семантики тем, что в последовательной семантике в каждый конкретный момент времени может происходить не более одного события. В семантике истинного параллелизма это не так, и несколько событий могут происходить одновременно. Внешний наблюдатель не различает эти две семантики. Также, последовательная семантика проще и более удобна для анализа поведенческих свойств, поэтому она часто используется. Тем не менее, когда модель должна учитывать время, разница между этими семантиками становится заметной.

Перейдем к определению ветвящихся процессов и разверток. Конфигурацией $\mathcal{C}$ в сети событий $ON = (B, E, G)$ называют бесконфликтное подмножество вершин, которое замкнуто относительно отношения $<$, то есть $\forall x, y \in \mathcal{C}: \neg(x\#y)$ и $(x < y) \wedge y \in C$, где $x \in \mathcal{C}$. Для каждого $x \in B \cup E$ мы определяем

локальную конфигурацию x такую что $[x] = \{y | y \in B \cup E, y < x\}$. Определение локальной конфигурации может быть обобщено на любое бесконфликтное множество вершин $X \subseteq B \cup E$, а именно $[X] = \{y | y \in B \cup E, x \in X, y < x\}$.

Определим множество ветвящихся процессов для данной маркированной сети Петри $N = (P, T, F, M_0)$, используя, так называемое, каноническое представление.

Множество C канонических имен N определено рекурсивно, как минимальное множество такое, что если $x \in P \cup T$ и A – конечное подмножество $\mathcal{C}$, то $(A, x) \in \mathcal{C}$.

Сеть событий (B, E, G) называется -сетью, если выполняются следующие условия:

- $B \cup E \subseteq \mathcal{C}$;
- $\forall (A, x) \in B \cup E, \ {}^\bullet(A, x) = A$.

Начальная разметка -сети Петри – это подмножество вершин $\{(\emptyset, x) | (\emptyset, x \in B)\}$. Для каждой $\mathcal{C}$-сети CN определяется функция (морфизм) h, отображающая вершины CN на вершины сети N: $h\big((A, x)\big) = x$.

Пусть S – конечное или бесконечное множество -сетей. Тогда объединение сетей из S определяется покомпонентно, то есть:

$$\cup S = (\cup_{(P,T,F,M) \in S} P, \cup_{(P,T,F,M) \in S} T, \cup_{(P,T,F,M) \in S} F, \cup_{(P,T,F,M) \in S} M).$$

Множество ветвящихся процессов маркированной сети Петри $N = (P, T, F, M_0)$ определяется как наименьшее множество -сетей, удовлетворяющее следующим условиям:

- $\mathcal{C}$-сеть $(I, \emptyset, \emptyset)$, где $I = \{(\emptyset, p) | p \in M_0\}$ (состоящая из условий I и не содержащая событий) – это ветвящийся процесс.
- Пусть $\mathcal{B}_1$ – ветвящийся процесс, M – достижимая разметка для $\mathcal{B}_1$, и $M' \subseteq M$, так что $h(M') = {}^\bullet t$ для некоторого t в T. Пусть $\mathcal{B}_2$ – сеть, полученная с помощью добавления события (M', t) и условий $\{(\{(M', t)\}, p) | p \in t^\bullet\}$ к $\mathcal{B}_1$. Тогда $\mathcal{B}_2$ – ветвящийся процесс.
- Пусть $\mathcal{BB}$ – конечное или бесконечное множество ветвящихся процессов. Объединение $\cup\mathcal{BB}$ также является ветвящимся процессом.

На Рис.2 показан пример ветвящегося процесса для сети Петри $PN1$ с начальной разметкой $\{p1\}$, изображенной на Рис.1.

Ветвящийся процесс $\mathcal{B}_1 = ((P_1, E_1, F_1), h_1)$ называется префиксом ветвящегося процесса $\mathcal{B}_2 = ((P_2, E_2, F_2), h_2)$, (обозначается $\mathcal{B}_1 \sqsubseteq \mathcal{B}_2$), если $P_1 \subseteq P_2$ и $E_1 \subseteq E_2$.

Максимальный относительно частичного порядка $\sqsubseteq$ ветвящийся процесс сети N называется разверткой сети N и обозначается как $U(N)$.

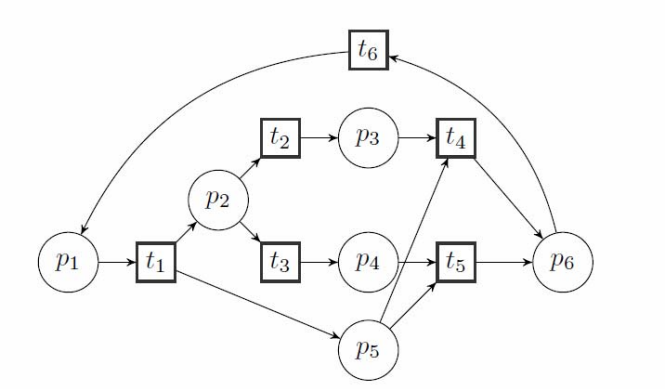

Рис. 1 Сеть Петри $PN1$
Fig. 1 Petri Net PN1

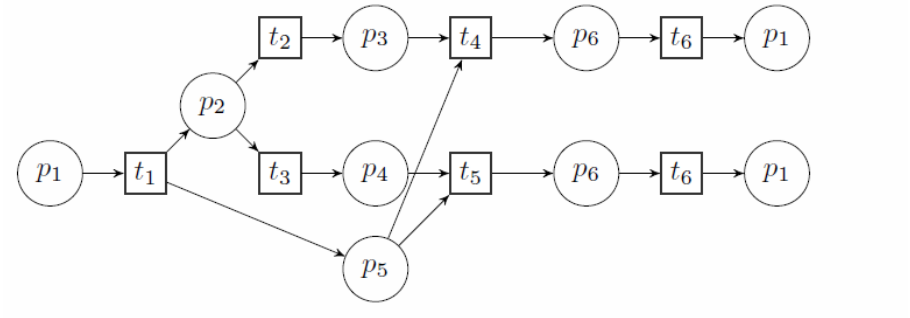

Рис. 2 Ветвящийся процесс для $PN1$
Fig 2 Branching process for PN1

2.3. Фундаментальное свойство разверток

Выполнение фундаментального свойства разверток означает, что для любой сети Петри поведение развертки эквивалентно поведению исходной сети. Фундаментальное свойство формулируется следующим образом.

Пусть M – достижимая маркировка сети Петри N, и пусть M_U является достижимой разметкой в $U(N)$ такой, что $h(M_U) = M$. Тогда:

- если существует шаг $M_U \xrightarrow{t_U} M_U'$ в $U(N)$, то существует шаг $M \xrightarrow{t} M'$ в N такой, что $h(t_U) = t \wedge h(M'_U) = M'$;
- если существует шаг $M \xrightarrow{t} M'$в N, то существует шаг $M_U \xrightarrow{t_U} M_U'$ в $U(N)$ такой, что $h(t_U) = t \wedge h(M'_U) = M'$.

Другими словами, фундаментальное свойство разверток гласит, что граф достижимости развертки изоморфен графу достижимости исходной сети Петри. Это свойство очень важно для использования разверток при анализе семантических свойств и верификации. Развертки определены и изучены для различных классов сетей Петри: для сетей Петри высокого уровня [22], контекстных сетей [28], временных сетей Петри [29], гиперсетей [30]. Все эти конструкции имеют сходные свойства, которые служат обоснованием применимости определенных в этих работах разверток для верификации. Далее будет приведено определение развертки для вложенных сетей Петри, для которого также выполняется фундаментальное свойство разверток.

2.4 Вложенные сети Петри

В этой работе мы рассматриваем вложенные сети Петри, точнее, их специальный подкласс, называемый строго консервативными вложенными сетями Петри. Более подробную информацию о вложенных сетях Петри можно найти в [1,7]. Здесь мы приводим сокращенное определение, подходящее для рассматриваемого случая.

Во вложенных сетях Петри фишки сами могут быть сетями Петри. Вложенная сеть Петри состоит из системной и элементных сетей. Мы называем их компонентами вложенной сети Петри. Маркированные элементные сети называются сетевыми фишками. Сетевые фишки, также как и обычные черные фишки, могут находиться в позициях системной сети. Некоторые переходы во вложенных сетях Петри могут быть помечены метками синхронизации. Непомеченные переходы во вложенных сетях Петри могут срабатывать автономно, согласно правилам срабатывания переходов в классических сетях Петри. Помеченные переходы в системной сети должны синхронизироваться с переходами (имеющими такую же пометку) в сетевых фишках, задействованных в это срабатывание перехода.

В этой работе мы рассматриваем безопасные и типизированные вложенные сети Петри, то есть каждая позиция системной сети может содержать не более одной фишки: черной, либо сетевой фишки определенного типа.

Пусть $Type$ – множество типов, Var – множество типизированных (типами из $Type$) переменных, а Lab – множество меток. Типизированной вложенной сетью Петри NP называется кортеж $(SN, (EN_1, \dots EN_k), v, \lambda, W)$, где:

- $SN = (P_{SN}, T_{SN}, F_{SN})$ – сеть Петри, называемая системной сетью;
- Для каждого $i = \overline{1,k}$, $EN_i = P_{EN_i}, T_{EN_i}, F_{EN_i})$ есть сеть Петри, называемая элементной сетью; множества переходов и позиций в системной и элементных сетях попарно не пересекаются; каждой элементной сети приписан тип из *Type*;
- $v: P_{SN} \to Type \cup \{\bullet\}$ – функция типизации позиций системной сети;
- $\lambda: T_{NP} \to Lab$ – частичная функция пометки переходов, где $T_{NP} = T_{SN} \cup T_{EN_1} \cup \dots \cup T_{EN_k}$. Будем писать $\lambda(t) = \perp$, если λ не определена для t.

- W: $F_{SN} \to Var \cup \{\bullet\}$ – функция пометки дуг такая, что для дуги r тип выражения $W(r)$ совпадает с типом позиции, инцидентной r.

Маркированная элементная сеть называется сетевой фишкой. Далее для данной вложенной сети Петри через $A_{net} = \{(EN, m) | \exists i = 1, \dots k: EN = EN_i, m \in \mathcal{M}(EN_i)\}$ обозначим множество всех возможных фишек сети вместе с черными фишками (black dot token).

Элементно-автономный шаг. Пусть t – непомеченный переход в одной из сетевых фишек. Тогда срабатывание перехода t определяется стандартными правилами срабатывания перехода в сетях Петри. Сама фишка остается в той же самой позиции системной сети.

Системно-автономный шаг – это срабатывание непомеченного перехода $t \in T_{SN}$ в системной сети в соответствии с правилами срабатывания для высокоуровневых сетей Петри (например, раскрашенных сетей Петри [31]).

Синхронный шаг. Пусть t – переход, помеченный λ в системной сети SN, t активный в разметке M при означивании b переменных в выражениях на дугах. Пусть далее $\alpha_1, \dots \alpha_n \in A_{net}$ – сетевые фишки, задействованные в срабатывании t. Тогда t может сработать при условии, что в каждой сетевой фишке α_i $(1 \leq i \leq n)$, задействованной в срабатывании t, имеется активный переход, помеченный той же синхронизационной меткой λ. Синхронный шаг выполняется в два этапа: сначала срабатывает по одному переходу, помеченному λ, в каждой из сетевых фишек, задействованных в срабатывании t, и затем выполняется срабатывание t в системной сети.

Вложенная сеть NP называется безопасной, если в каждой ее достижимой разметке имеется не более одной фишки в каждой позиции в системной сети и в каждой позиции сетевых фишек.

Далее мы будем рассматривать только безопасные сети.

2.5 Консервативные вложенные сети Петри

Безопасная вложенная сеть Петри $N = (SN, (EN_1, \dots EN_k), v, \lambda, W)$ называется строго консервативной если

- Для каждого $t \in T_{SN}$ и для каждого $p \in \bullet t$, $\exists! p' \in t \bullet$. $W(p,t) = W(t,p')$ или $W(p,t) = \bullet$;
- Для каждого $t \in T_{SN}$ и для каждого $p \in t \bullet$, $\exists! p' \in \bullet t$. $W(p',t) = W(t,p)$ или $W(p,t) = \bullet$.

Строгая консервативность означает, что сетевые фишки не могут появляться или исчезать после срабатывания перехода в системной сети.

Заметим, что в [15] вложенные сети Петри называются консервативными, если фишки не могут исчезать после срабатывания перехода, но могут копироваться, таким образом, количество сетевых фишек в таких консервативных вложенных сетях Петри может быть не ограничено. Здесь мы рассматриваем более узкий подкласс вложенных сетей Петри с постоянным количеством сетевых фишек

(фишки не могут копироваться). Следует отметить, что хотя это ограничение довольно строгое, с помощью консервативных вложенных сетей Петри можно моделировать много важных и интересных мультиагентных систем.

3. Трансляция безопасных консервативных вложенных сетей Петри в классические сети Петри

Поскольку безопасные консервативные вложенные сети Петри ограничены, т.е. множество достижимых состояний любой такой сети конечно, для каждой такой сети существует эквивалентная по поведению классическая ограниченная сеть Петри. В этом разделе будет представлен алгоритм трансляции безопасной консервативной вложенной сети Петри в классическую сеть Петри с эквивалентным поведением и структурой, которая соответствует структуре исходной вложенной сети. Далее будет показано, что развертка полученной классической сети Петри изоморфна развертке исходной вложенной сети Петри. Таким образом, трансляция в классические сети Петри может быть использована для построения развертки и верификации ограниченных консервативных вложенных сетей Петри.

Алгоритм трансляции безопасных консервативных вложенных сетей Петри будет проиллюстрирован на примере вложенной сети Петри $NP2$, показанной на Рис. 3 (системная сеть) и Рис. 4 (элементная сеть). Эта сеть будет транслирована в безопасную классическую сеть Петри PN.

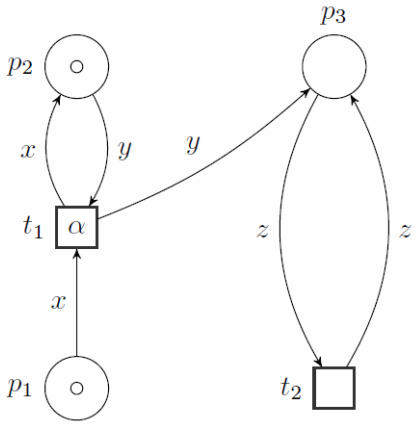

Рис. 3 Вложенная сеть Петри $NP2$
Fig. 3 Nested Perti Net NP2

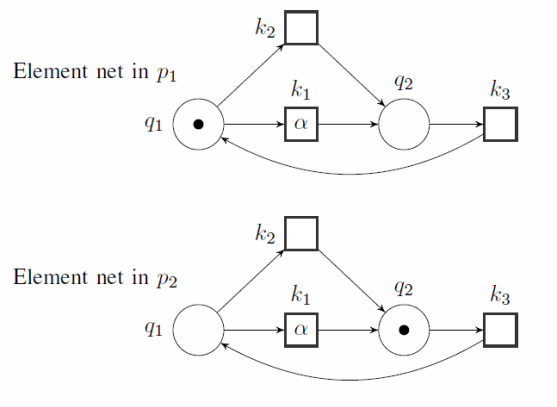

Рис. 4 Вложенная сеть Петри $NP2$

Fig. 4 Nested Perti Net NP2

3.1 Алгоритм трансляции

Пусть $NP = (SN, (EN_1, \dots EN_k), v, \lambda, W)$ – вложенная сеть Петри с множеством $NTok$ идентифицированных сетевых фишек в начальной разметке. Через I мы обозначим множество всех идентификаторов, использованных в $NTok$, а через $I_E \subseteq E$ обозначим подкласс идентификаторов сетевых фишек типа E. Сеть NP будет транслирована в сеть Петри $PN = (P_{PN}, T_PN, F_{PN})$ с начальной разметкой m_0.

- Сначала определяем множество P_{PN} позиций целевой сети PN. Для каждого типа E некоторой позиции в системной сети SN мы создаем множество $\mathfrak{S}_E$ позиций для P_{PN}. Множество $\mathfrak{S}_E$будет содержать копию каждой позиции типа E в системной сети для каждой сетевой фишки типа E (помеченную идентификатором сетевой фишки) и копию каждой позиции в PE для каждой сетевой фишки типа E, то есть, $\mathfrak{S}_E = \{(p, id) | p \in P_{SN}, v(p) = E, id \in I_E\} \cup \{(q, id) | q \in P_E, id \in I_E\}$. Для каждой позиции в SN типа черной фишкой создаем только одну копию p без идентификатора. Затем множество P_{PN} позиций для целевой сети PN определяется как объединение этих множеств. Результат выполнения первого шага алгоритма для вложенной сети $NP2$ изображен на Рис. 5.

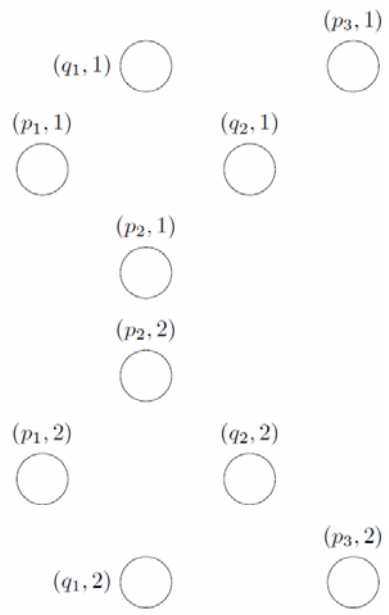

Рис. 5 Создание позиций PN

Fig. 5 Position creation for PN

- Чтобы определить начальную разметку для сети PN разметку позиций P_{NP} во вложенной сети Петри кодируем разметками на построенных позициях сети P_{PN}. Если сетевая фишка $\eta = (id, E, m)$ находится в позиции p в разметке M системной сети, то в целевой сети черные фишки помещаются в позицию (p, id) и во все позиции (q, id) для всех q таких, что $m(q) = 1$. Если позиция типа черной точки в системной сети SN содержит черную фишку, то единственная соответствующая позиция в сети PN также будет содержать черную фишку. Легко заметить, что эта кодировка определяет взаимно-однозначное соответствие между разметками консервативной безопасной вложенной сети Петри и безопасными разметками в PN. В нашем примере первая элементная сеть находится в позиции $p1$, вторая – в позиции $p2$. Таким образом, черные фишки помещаются в позиции $(p_1, 1)$ и $(p_2, 2)$; а также в $(q_1, 1)$ и $(q_1, 2)$.

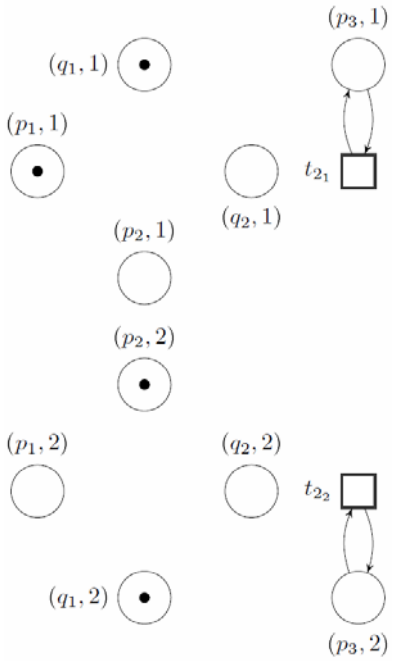

Рис. 6 Системно-автономный шаг

Fig. 6 System-autonomous step

- Для каждого автономного перехода t в системной сети SN мы строим множество T_t переходов следующим образом. Поскольку каждая переменная на входной дуге перехода t может быть означена, вообще говоря, любой сетевой фишкой подходящего типа, то для каждого такого означивания строится отдельный переход в PN с соответствующими входными и выходными дугами. В нашем примере для перехода t_2 мы строим два перехода t_{21}и t_{22}.
- Для каждого автономного перехода в сетевой фишке из $NTok$ строится соответствующий переход, инцидентный позициям, помеченным id. Так в нашем примере мы получим четыре перехода: k_{21}, k_{22}, k_{31} и k_{32}.

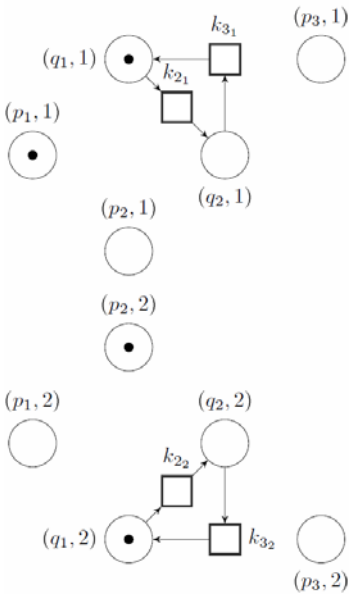

Рис. 7. Элементно-автономный шаг

Fig. 7. Element-autonomous step

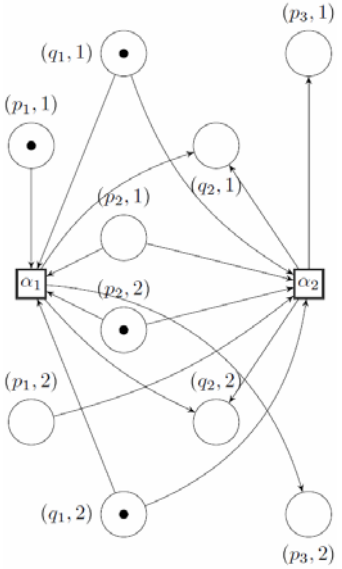

Рис. 8 Синхронный шаг

Fig 8. Synchronous step

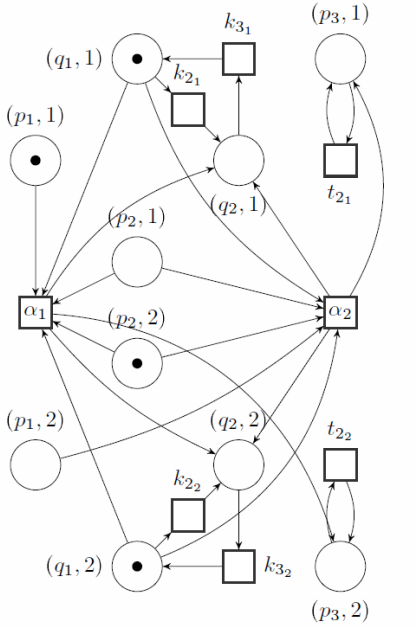

Рис. 9 Результат трансляции $NP2$ в классическую сеть Петри
Fig. 9. The result of NP2 translation into classical Petri Net

- Срабатывание синхронного перехода означает одновременное срабатывание перехода в системной сети и срабатывание переходов, помеченных такой же меткой в каждой задействованной в этом срабатывании сетевой фишке. Таким образом, синхронный шаг является комбинацией шага 3 и шага 4. В нашем примере есть две сетевые фишки, и переходы α_1 и α_2 строятся для каждой из них. Таким образом мы можем моделировать синхронный шаг для каждой из возможных начальных разметок системной сети.

Корректность определенной выше трансляции обеспечивает следующая

Теорема 1.
Пусть NP – вложенная сеть Петри и PN – сеть Петри, полученная из сети NP с помощью трансляции, описанной выше. Тогда графы достижимости сетей NP и PN изоморфны.

Доказательство. Шаг 2 алгоритма определяет взаимное соответствие между достижимыми разметками сетей NP и PN. Легко заметить, что в соответствии с определением трансляции соответствующие шаги срабатывания в обеих сетях не нарушают это соответствие.

4 Построение развертки для ограниченной консервативной вложенной сети Петри

После трансляции вложенной сети Петри в классическую для полученной классической сети Петри можно построить ветвящиеся процессы и развертки, пользуясь известными методами. Так на Рис. 10 показан один из ветвящихся процессов для сети $NP2$. Развертка тогда путем отсечения конечного префикса ветвящегося процесса по сечению, определяющему состояние, которое уже входит в этот префикс.

Сравним метод построения развертки путем трансляции вложенной сети Петри в классическую покомпонентным методом, предложенным в [4].

Теорема 2.
Развертка безопасной консервативной вложенной сети Петри *NP*, полученная описанным в [4] методом, изоморфна развертке классической безопасной сети Петри, полученной в результате описанной выше трансляции сети *NP*.

Доказательство. В [4] было доказано выполнение фундаментального свойства развертки для описанных там покомпонентных разверток, а именно, что граф достижимости вложенной сети Петри изоморфен графу достижимости его покомпонентной развертки. По *Теореме 1* граф достижимости классической сети, полученной в результате трансляции, изоморфен графу достижимости исходной вложенной сети. Для классических сетей Петри выполняется фундаментальное свойство разверток, т.е. граф достижимости развертки изоморфен графу достижимости сети Петри. Из всего этого следует, что развертки, построенные методом из [4] и путем трансляции в классические сети Петри, изоморфны.

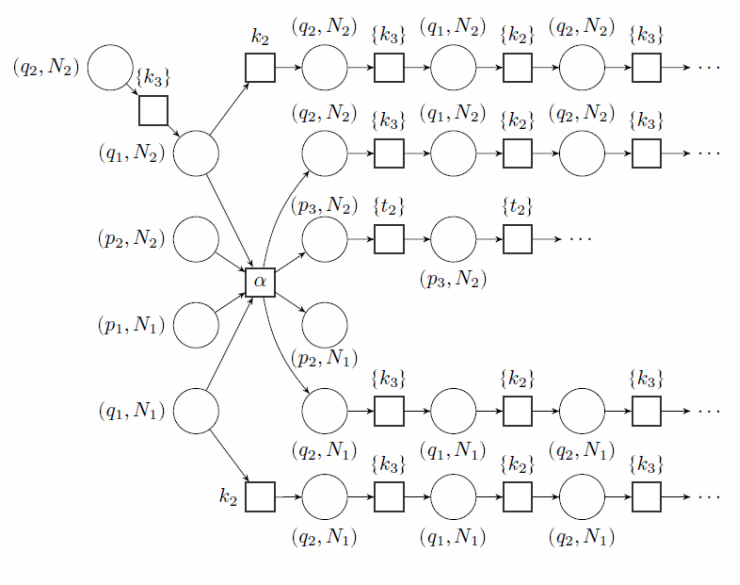

Рис. 10 Ветвящийся процесс для сети NP2

Fig. 10. Branching process for NP2 net

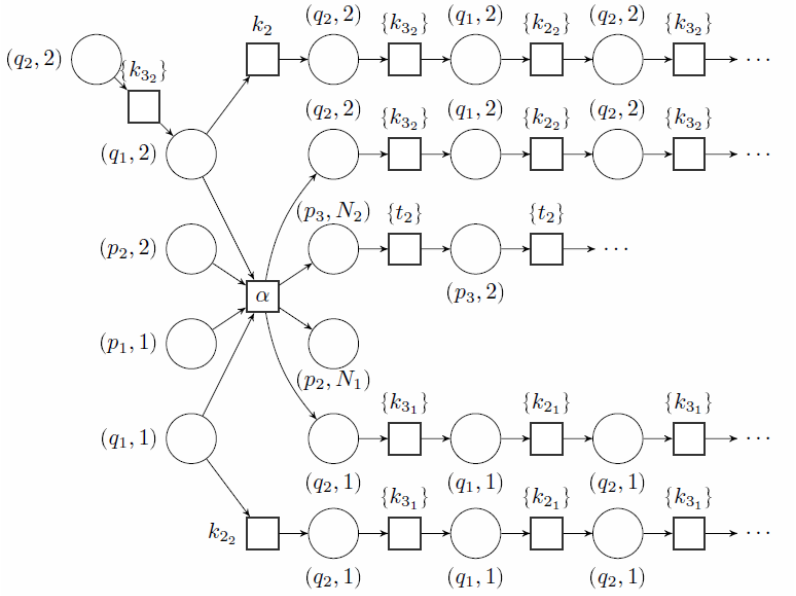

Рис. 11 Ветвящийся процесс, построенный путем трансляции сети NP2

Fig. 11. Branching process built after NP2 translation

5 Заключение

В этой работе мы показали, что каждая консервативная безопасная вложенная сеть Петри может быть транслирована в поведенчески эквивалентную классическую сеть Петри так, что их графы достижимости изоморфны. Таким образом, чтобы построить развертки для вложенной сети Петри достаточно транслировать её в классическую сеть Петри и потом применить метод развертки для классических сетей Петри.

Другой способ построения развертки безопасной консервативной вложенной сети Петри описан в [4]. В этой работе развертка сети строится непосредственно путем построения разверток отдельных компонентов вложенной сети Петри. Мы показали, что оба метода дают одинаковый с точностью до изоморфизма результат.

Каждый из этих методов имеет свои преимущества и недостатки. Построение разверток вложенных сетей Петри путем трансляции в классические сети Петри позволяет использовать готовые методы и инструментарий для построения разверток и верификации классических сетей Петри. Применение метода построения разверток непосредственно для вложенных сетей Петри, описанного в [4], требует разработки специального программного обеспечения. С другой стороны, построение промежуточной классической сети Петри может привести к значительному росту затрат времени и памяти при построении развертки вложенной сети.

Поэтому интересно сравнить сложность этих двух методов: метода, предложенного в [4], и подхода, основанного на трансляции вложенных сетей Петри в классические. Нетрудно заметить, что если в начальной разметке сети имеется несколько сетевых фишек одного и того же, то трансляция вложенной сети в классическую сеть Петри ведет к значительному росту сети. Так для перехода системной сети с n входными позициями одного типа и k фишек этого типа в начальной разметке строится k^n копий этого перехода в целевой сети Петри. Понятно, что этого нельзя избежать, так как необходимо различать разметки сетевых фишек, находящихся в разных позициях системной сети.

Чтобы сравнить временные затраты этих двух методов на практике, нами были реализованы компьютерные программы, позволяющие выполнять

- трансляцию консервативной безопасной сетей Петри в классическую сеть Петри и построение развертки для нее;
- построение развертки для вложенной сети Петри напрямую.

Наша гипотеза состояла в том, что увеличение количества сетевых фишек ведет к значительному росту классической сети во время трансляции. Чтобы получить репрезентативные результаты, мы провели эксперименты на сетях, имеющих сходную структуру, но разное количество элементных сетей разных типов.

Эксперименты с сетями небольшого размера подтвердили нашу гипотезу. Так, для сети $NP2$ из нашего примера время построения развертки составило 0.38

мс. при применении метода покомпонентной развертки и 0.54 мс. при построении развертки путем трансляции в классическую сеть Петри. Таким образом, даже в случае двух сетевых фишек, разница времени исполнения весьма заметна. В будущем мы планируем провести эксперименты по сравнительной оценке сложности двух методов построения разверток на больших примерах. Также планируется выполнить эксперименты по верификации конкретных свойств вложенных сетей Петри с помощью построения разверток.

Благодарность

Работа выполнена при поддержке Программы фундаментальных исследований НИУ ВШЭ и Российского фонда фундаментальных исследований (проект 16-01-00546).

Список литературы

[1]. Lomazova I.A. Nested Petri nets—a formalism for specification and verification of multi-agent distributed systems. Fundamenta Informaticae 43(1), 2000, pp. 195–214.
[2]. McMillan K.L. Using unfoldings to avoid the state explosion problem in the verification of asynchronous circuits. Computer Aided Verification, Springer, 1992, pp. 164–177.
[3]. Nielsen M., Plotkin G., Winskel G. Petri nets, event structures and domains, part I. Theoretical Computer Science 13(1), 1981, pp. 85–108.
[4]. Frumin D., Lomazova I.A. Branching processes of conservative nested Petri nets. VPT 2014. Second International Workshop on Verification and Program Transformation Vol. 28: EPiC Series. EasyChair, 2014. P. 19-35.
[5]. Lomazova I.A., van Hee K.M., Oanea O., Serebrenik A., Sidorova N., Voorhoeve M. Nested nets for adaptive systems. Application and Theory of Petri Nets and Other Models of Concurrency, LNCS, 2006, pp. 241–260.
[6]. Lomazova I.A. Modeling dynamic objects in distributed systems with nested petri nets. Fundamenta Informaticae 51(1-2), 2002, pp. 121–133.
[7]. Lomazova I.A. Nested petri nets for adaptive process modeling. Pillars of computer science. Springer, 2008, pp. 460–474
[8]. L´opez-Mellado E., Villanueva-Paredes N., Almeyda-Canepa H. Modelling of batch production systems using Petri nets with dynamic tokens. Mathematics and Computers in Simulation 67(6), 2005, pp. 541–558.
[9]. Kahloul L., Djouani K., Chaoui A. Formal study of reconfigurable manufacturing systems: A high level Petri nets based approach. Industrial Applications of Holonic and Multi-Agent Systems. Springer, 2013, pp. 106–117.
[10]. Zhang, L., Rodrigues, B. Nested coloured timed Petri nets for production configuration of product families. International journal of production research 48(6), 2010, pp. 1805–1833.
[11]. Venero M.L.F., da Silva F.S.C. Modeling and simulating interaction protocols using nested Petri nets. Software Engineering and Formal Methods. Springer, 2013, pp. 135–150.
[12]. Chang L., He X., Lian J., Shatz S. Applying a nested Petri net modeling paradigm to coordination of sensor networks with mobile agents. Proc. of Workshop on Petri Nets and Distributed Systems, Xian, China, 2008, pp. 132–145.
[13]. Cristini F., Tessier C. Nets-within-nets to model innovative space system architectures. Application and Theory of Petri Nets. Springer, 2012, pp. 348–367.

[14]. Mascheroni M., Farina F. Nets-within-nets paradigm and grid computing. Transactions on Petri Nets and Other Models of Concurrency V. Springer, 2012, pp. 201–220.
[15]. Dworza´nski L.W., Lomazova I.A. On compositionality of boundedness and liveness for nested Petri nets. Fundamenta Informaticae 120(3-4), 2012, pp. 275–293.
[16]. Dworza´nski L., Lomazova I.A. CPN tools-assisted simulation and verification of nested Petri nets. Automatic Control and Computer Sciences 47(7), 2013, pp. 393–402.
[17]. Venero M.L.F. Verifying cross-organizational workflows over multiagent based environments. Enterprise and Organizational Modeling and Simulation. Springer, 2014, pp. 38–58.
[18]. Winskel G. Event structures. Springer, 1986.
[19]. Bonet B., Haslum P., Hickmott S., Thi´ebaux S. Directed unfolding of petri nets. Transactions on Petri Nets and Other Models of Concurrency I. Springer, 2008, pp. 172–198.
[20]. McMillan K.L. A technique of state space search based on unfolding. Form. Methods Syst. Des. 6(1), 1995, pp. 45–65.
[21]. Heljanko K. Using logic programs with stable model semantics to solve deadlock and reachability problems for 1-safe petri nets. Fundamenta Informaticae 37(3), 1999, pp. 247–268.
[22]. Khomenko V., Koutny M. Branching processes of high-level Petri nets. In Garavel, H., Hatcliff, J., eds.: Tools and Algorithms for the Construction and Analysis of Systems. Volume 2619 of Lecture Notes in Computer Science. Springer, 2003, pp. 458–472.
[23]. Langerak R., Brinksma E. A complete finite prefix for process algebra. Computer Aided Verification, Springer, 1999, pp. 184–195.
[24]. Khomenko V., Koutny M., Vogler W. Canonical prefixes of Petri net unfoldings. Acta Informatica 40(2), 2003, pp. 95–118.
[25]. Khomenko V. Model Checking Based on Prefixes of Petri Net Unfoldings. Ph.D. Thesis, School of Computing Science, Newcastle University, 2003.
[26]. Esparza J., Heljanko K. Unfoldings: a partial-order approach to model checking. Springer, 2008.
[27]. Engelfriet J. Branching processes of Petri nets. Acta Informatica 28(6), 1991, pp.575–591.
[28]. Baldan P., Corradini A., Knig B., Schwoon S. Mcmillans complete prefix for contextual nets. Jensen, K., Aalst, W.M., Billington, J., eds.: Transactions on Petri Nets and Other Models of Concurrency I. Volume 5100 of Lecture Notes in Computer Science. Springer, 2008, pp. 199–220.
[29]. Fleischhack H., Stehno C. Computing a finite prefix of a time Petri net. Esparza J., Lakos C., eds.: Application and Theory of Petri Nets 2002. Volume 2360 of Lecture Notes in Computer Science. Springer, 2002, pp. 163–181.
[30]. Mascheroni, M. Hypernets: a Class of Hierarchical Petri Nets. Ph.D. Thesis, Facolt di Scienze Naturali Fisiche e Naturali, Dipartimento di Informatica Sistemistica e Comunicazione, Universit`a Degli Studi Di Milano Bicocca, 2010.
[31]. Jensen K., Kristensen L.M. Coloured Petri nets: modelling and validation of concurrent systems. Springer, 2009.

Translation of Nested Petri Nets into Classical Petri Nets for Unfoldings Verification

V.O. Ermakova <ermakovavo@gmail.com>
I.A. Lomazova <ilomazova@hse.ru>
National Research University Higher School of Economics,
20 Myasnitskaya St., Moscow, 101000, Russia

Аннотация. Nested Petri nets (NP-nets) have proved to be one of the convenient formalisms for distributed multi-agent systems modeling and analysis. It allows representing multi-agent systems structure in a natural way, since tokens in the system net are Petri nets themselves, and have their own behavior. Multi-agent systems are highly concurrent. Verification of such

systems with model checking method causes serious difficulties arising from the huge growth of the number of system intermediate states (state-space explosion problem). To solve this problem an approach based on unfolding system behavior was proposed in the literature. Earlier in [4] the applicability of unfolding for nested Petri nets verification was studied, and the method for constructing unfolding for safe conservative nested Petri nets was proposed. In this work we propose another method for constructing safe conservative nested Petri nets unfoldings, which is based on translation of such nets into classical Petri nets and applying standard method for unfolding construction to them. We discuss also the comparative merits of the two approaches.

Key words: multi-agent systems; verification; Petri nets; nested Petri nets; unfoldings.

DOI: 10.15514/ISPRAS-2016-28(4)-7

For citation: Ermakova V.O., Lomazova I.A. Translation of Nested Petri Nets into Classical Petri Nets for Unfoldings Verification. Trudy ISP RAN/Proc. ISP RAS, vol. 28, issue 4, 2016, pp. 115-136 (in Russian). DOI: 10.15514/ISPRAS-2016-28(4)-7

References

[1]. Lomazova I.A. Nested Petri nets—a formalism for specification and verification of multi-agent distributed systems. Fundamenta Informaticae 43(1), 2000, pp. 195–214.
[2]. McMillan K.L. Using unfoldings to avoid the state explosion problem in the verification of asynchronous circuits. Computer Aided Verification, Springer, 1992, pp. 164–177.
[3]. Nielsen M., Plotkin G., Winskel G. Petri nets, event structures and domains, part I. Theoretical Computer Science 13(1), 1981, pp. 85–108.
[4]. Frumin D., Lomazova I.A. Branching processes of conservative nested Petri nets. VPT 2014. Second International Workshop on Verification and Program Transformation Vol. 28: EPiC Series. EasyChair, 2014. P. 19-35.
[5]. Lomazova I.A., van Hee K.M., Oanea O., Serebrenik A., Sidorova N., Voorhoeve M. Nested nets for adaptive systems. Application and Theory of Petri Nets and Other Models of Concurrency, Lecture Notes in Computer Science, vol. 4024, 2006, pp. 241–260.
[6]. Lomazova I.A. Modeling dynamic objects in distributed systems with nested petri nets. Fundamenta Informaticae 51(1-2), 2002, pp. 121–133.
[7]. Lomazova I.A. Nested petri nets for adaptive process modeling. Pillars of computer science. Lecture Notes in Computer Science, vol. 4800, Springer, 2008, pp. 460–474
[8]. L´opez-Mellado E., Villanueva-Paredes N., Almeyda-Canepa H. Modelling of batch production systems using Petri nets with dynamic tokens. Mathematics and Computers in Simulation 67(6), 2005, pp. 541–558.
[9]. Kahloul L., Djouani K., Chaoui A. Formal study of reconfigurable manufacturing systems: A high level Petri nets based approach. Industrial Applications of Holonic and Multi-Agent Systems. Springer, 2013, pp. 106–117.
[10]. Zhang, L., Rodrigues, B. Nested coloured timed Petri nets for production configuration of product families. International journal of production research 48(6), 2010, pp. 1805–1833.
[11]. Venero M.L.F., da Silva F.S.C. Modeling and simulating interaction protocols using nested Petri nets. Software Engineering and Formal Methods. Springer, 2013, pp. 135–150.
[12]. Chang L., He X., Lian J., Shatz S. Applying a nested Petri net modeling paradigm to coordination of sensor networks with mobile agents. Proc. of Workshop on Petri Nets and Distributed Systems, Xian, China, 2008, pp. 132–145.

[13]. Cristini F., Tessier C. Nets-within-nets to model innovative space system architectures. Application and Theory of Petri Nets. Springer, 2012, pp. 348–367.
[14]. Mascheroni M., Farina F. Nets-within-nets paradigm and grid computing. Transactions on Petri Nets and Other Models of Concurrency V. Springer, 2012, pp. 201–220.
[15]. Dworza´nski L.W., Lomazova I.A. On compositionality of boundedness and liveness for nested Petri nets. Fundamenta Informaticae 120(3-4), 2012, pp. 275–293.
[16]. Dworza´nski L., Lomazova I.A. CPN tools-assisted simulation and verification of nested Petri nets. Automatic Control and Computer Sciences 47(7), 2013, pp. 393–402.
[17]. Venero M.L.F. Verifying cross-organizational workflows over multiagent based environments. Enterprise and Organizational Modeling and Simulation. Springer, 2014, pp. 38–58.
[18]. Winskel G. Event structures. Springer, 1986.
[19]. Bonet B., Haslum P., Hickmott S., Thi´ebaux S. Directed unfolding of petri nets. Transactions on Petri Nets and Other Models of Concurrency I. Springer, 2008, pp. 172–198.
[20]. McMillan K.L. A technique of state space search based on unfolding. Form. Methods Syst. Des. 6(1), 1995, pp. 45–65.
[21]. Heljanko K. Using logic programs with stable model semantics to solve deadlock and reachability problems for 1-safe petri nets. Fundamenta Informaticae 37(3), 1999, pp. 247–268.
[22]. Khomenko V., Koutny M. Branching processes of high-level Petri nets. In Garavel, H., Hatcliff, J., eds.: Tools and Algorithms for the Construction and Analysis of Systems. Volume 2619 of Lecture Notes in Computer Science. Springer, 2003, pp. 458–472.
[23]. Langerak R., Brinksma E. A complete finite prefix for process algebra. Computer Aided Verification, Springer, 1999, pp. 184–195.
[24]. Khomenko V., Koutny M., Vogler W. Canonical prefixes of Petri net unfoldings. Acta Informatica 40(2), 2003, pp. 95–118.
[25]. Khomenko V. Model Checking Based on Prefixes of Petri Net Unfoldings. Ph.D. Thesis, School of Computing Science, Newcastle University, 2003.
[26]. Esparza J., Heljanko K. Unfoldings: a partial-order approach to model checking. Springer, 2008.
[27]. Engelfriet J. Branching processes of Petri nets. Acta Informatica 28(6), 1991, pp.575–591.
[28]. Baldan P., Corradini A., Knig B., Schwoon S. Mcmillans complete prefix for contextual nets. Jensen, K., Aalst, W.M., Billington, J., eds.: Transactions on Petri Nets and Other Models of Concurrency I. Volume 5100 of Lecture Notes in Computer Science. Springer, 2008, pp. 199–220.
[29]. Fleischhack H., Stehno C. Computing a finite prefix of a time Petri net. Esparza J., Lakos C., eds.: Application and Theory of Petri Nets 2002. Volume 2360 of Lecture Notes in Computer Science. Springer, 2002, pp. 163–181.
[30]. Mascheroni, M. Hypernets: a Class of Hierarchical Petri Nets. Ph.D. Thesis, Facolt di Scienze Naturali Fisiche e Naturali, Dipartimento di Informatica Sistemistica e Comunicazione, Universit`a Degli Studi Di Milano Bicocca, 2010.
[31]. Jensen K., Kristensen L.M. Coloured Petri nets: modelling and validation of concurrent systems. Springer, 2009.