Модел. и анализ информ. систем. Т. **19**, № **5** (2012) 115–130

УДК 517.51+514.17

Имитационное моделирование и верификация вложенных сетей Петри с использованием CPNTools

Дворянский Л. В., Ломазова И. А.[1]

Национальный исследовательский университет "Высшая школа экономики"
Институт программных систем РАН

e-mail: leo@mathtech.ru, ilomazova@hse.ru

получена 22 октября 2012

Ключевые слова: вложенные сети Петри, раскрашенные сети Петри, имитационное моделирование, верификация, граф достижимости

Вложенные сети Петри (NP-сети) — это расширение сетей Петри в рамках подхода "nets-within-nets", когда фишки в разметке сети сами являются сетями Петри и обладают автономным поведением, при этом имеются средства синхронизации сетевых фишек и системной сети. Формализм NP-сетей позволяет естественным образом моделировать многоуровневые мультиагентные системы с динамической структурой. В настоящее время не существует инструментальной поддержки имитационного моделирования и анализа вложенных сетей Петри. В работе предлагается проводить имитационное моделирование и построение графа достижимости для NP-сетей путем перевода NP-сетей в раскрашенные сети Петри и использования инструментария CPN Tools в качестве виртуальной машины для исполнения и средства автоматического анализа исходных NP-сетей.

1. Введение

В настоящее время наблюдается смещение парадигмы вычислений от традиционной, в которой вычислительная мощность рабочей станции, сервера или группы серверов является основным ресурсом производимых вычислений, к новым способам организации вычислений — облачные, повсеместные, коммунальные, краевые вычисления. В таких вычислениях все атрибуты вычислений, касаются ли они программного или аппаратного обеспечения, логического управления или данных, используемых вычислительных или физических ресурсов, становятся максимально децентрализованными и распределенными.

[1]Работа выполнена в рамках программы Научного фонда НИУ ВШЭ в 2012/2013 гг. (проект 11-01-0032) и при поддержке Российского фонда фундаментальных исследований (гранты 11-01-00737, 11-07-005549) .

Новые способы организации и структурирования вычислений в программных и аппаратных системах привели к распространению мультиагентных систем (МАС). В МАС вычисление распределено между множеством узлов-агентов взаимодействующих друг с другом для достижения локальных целей, а их совместная скоординированная работа приводит к достижению глобальных целей системы. В качестве наглядных примеров МАС могут выступать одноранговые сети, грид-системы, облачные сервисы, системы дополненной реальности, беспроводные самоорганизующиеся сети и другие. Подобные системы встречаются в самых разных прикладных областях, и их неправильное поведение может привести к материальным убыткам, ущербу здоровью и качеству жизни людей. Поэтому проблемы обеспечения надежности и корректности работы МАС являются весьма актуальными, и для построения корректных МАС необходимо применение строгих математических методов моделирования и анализа.

Одним из успешных и популярных формализмов для моделирования распределенных и параллельных систем являются сети Петри. Однако классические сети Петри обладают статической плоской структурой, что является существенным недостатком при моделировании мультиагентных систем. В качестве решения этой проблемы Р. Фальк [20] предложил подход "сети-внутри-сетей" (*nets-within-nets*), который активно развивается в теории сетей Петри как способ моделирования мобильных объектов в распределенных системах [3, 4, 6, 9, 13, 14, 15, 16, 18, 19, 21].

Вложенные сети Петри (NP-сети) [1] — это расширение сетей Петри, в котором фишки, задающие маркировку сети, сами являются сетями Петри. В отличие от других расширений сетей Петри с сетевыми фишками, вложенные сети обладают автономным поведением и взаимодействуют друг с другом и с основной сетью. Координация активных объектов на разных уровнях NP-сети обеспечивается переходами синхронизации (одновременное срабатывание одинаково помеченных переходов в смежных уровнях). Модели NP-сетей имеют слабосвязанную динамическую многоуровневую структуру, что позволяет естественным образом моделировать распределенные и многоуровневые мультиагентные системы с мобильными агентами. В работах [1, 8, 17] было показано, как с помощью NP-сетей моделировать такие мультиагентные системы, как, например, одноранговые сетевые протоколы, производственные процессы, интеллектуальные системы. Поэтому разработка методов анализа поведенческих свойств вложенных систем Петри является важной проблемой.

Отметим, что NP-сети не являются Тьюринг полными, но строго более выразительны, чем обыкновенные сети Петри. Так, достижимость, живость и ограниченность неразрешимы для двухуровневых вложенных сетей Петри [1]. В связи с этим большой интерес представляет разработка методов и подходов для имитационного моделирования и верификации вложенных сетей Петри. Этой теме посвящена настоящая работа.

Имитационное моделирование является важным инструментом валидации построенной модели. Имитационное моделирование позволяет лучше понять поведение модели, обнаружить явные дефекты и узкие места в поведении изучаемой или разрабатываемой системы на ранних стадиях проектирования. Исполнение нескольких типичных сценариев повышает нашу уверенность в корректности модели.

Другим важнейшим инструментом анализа поведенческих свойств модели является граф достижимости. Имея граф достижимости модели, можно с помощью автоматического анализа проверять важные поведенческие свойства — живость, справедливость, безопасность, и обнаруживать такие дефекты поведения, как тупики, активные блокировки, дивергенции, нарушения условий взаимного исключения и другие. Для анализа всех возможных исполнений модели необходимо построить полный граф достижимости, что возможно для моделей с конечным пространством состояний. Для систем с бесконечным или очень большим числом состояний возможно построение фрагментов пространства состояний. Такие фрагменты могут быть ограничены максимальной глубиной исполнения, максимальным временем построения фрагмента, числом дуг, выбираемых для обхода в узлах ветвления, или просто объемом памяти, доступной системе анализа. Несмотря на неполноту результатов, получаемых в случае моделей с большим или бесконечным пространством состояний, анализ таких фрагментов способен выявить большое количество дефектов модели.

Для практического применения любого поведенческого формализма необходима компьютерная поддержка первичных инструментов анализа моделей — имитационного моделирования и построения графа достижимости. Несмотря на достоинства NP-сетей, отсутствие инструментальной поддержки для имитационного моделирования и анализа свойств NP-сетей сильно ограничивает возможности их применения и препятствует дальнейшему развитию и распространению этого формализма.

В данной работе мы показываем, что имитационное моделирование и верификацию NP-сетей можно проводить путем их трансляции в раскрашенные сети Йенсена. Раскрашенные сети Петри (CP-сети) [11] — это класс сетей Петри высокого уровня, примечательный широким спектром применений, а также системой CPN Tools [12], обеспечивающей мощную инструментальную поддержку имитационного моделирования и анализа поведенческих свойств.

Классические высокоуровневые сети Петри выразительно эквивалентны обыкновенным сетям Петри, в то время как раскрашенные сети Петри, допускающие счетное число различных цветов (натуральные числа), полны по Тьюрингу. Кортежи натуральных чисел могут быть использованы для представления разметок сетевых фишек вложенных сетей Петри в раскрашенных сетях Петри. Важно отметить, что такое представление хорошо сохраняет структуру исходной модели, облегчая интерпретацию результатов.

В результате трансляции по исходной NP-сети строится поведенчески эквивалентная ей целевая CP-сеть. После этого имитационное моделирование или анализ проводятся на целевой CP-сети. Результаты такого моделирования или анализа интерпретируются на исходной NP-сети.

Работа имеет следующую структуру. Раздел 2 содержит основные понятия и определения, связанные с раскрашенными и вложенными сетями Петри. Раздел 3 посвящен трансляции вложенных сетей Петри в раскрашенные. Обосновываются возможности и полезность трансляции NP-сетей в CP-сети и использования инструментария CPN Tools для имитационного моделирования и анализа NP-сетей. Явно выделены требования к трансляции, необходимые для проведения имитационного моделирования и построения графа достижимости в CPN Tools. Описан ал-

горитм трансляции, удовлетворяющей указанным требованиям. Раздел 4 посвящен реализации транслятора NPN2CPN. Кратко описан пример использования этого транслятора для имитационного моделирования и анализа в CPN Tools простого одноранговового протокола. Раздел 5 содержит заключение и описание направления дальнейших исследований и развития инструментальной поддержки NP-сетей.

Отметим также, что другой подход к разработке практически применимых методов верификации вложенных сетей Петри может быть связан с разработкой эффективных алгоритмов для отдельных подклассов NP-сетей. В работах [7, 8] было показано, что при некоторых условиях живость и ограниченность NP-сети может быть установлена композиционально, то есть как следствие соответственно живости и ограниченности ее компонентов.

2. Основные определения

Приведенное далее определение раскрашенных сетей Петри почти не отличается от классического [11]. Вложенные сети Петри являются расширением раскрашенных сетей Петри над специальным универсумом. Поэтому определение раскрашенных сетей Петри параметризовано универсумом значений U. Мы также добавляем метки на переходах, необходимые для реализации синхронизации переходов во вложенных сетях Петри.

Сначала приведем определения, которые будут использованы в оставшейся части работы. Обозначаем с помощью $\mathbb{N}$ множество натуральных чисел, включая ноль. Для множества S *мультимножество* m над S есть отображение $m : S \to \mathbb{N}$. Множество всех мультимножеств над S обозначается $\mathbb{N}^S$. Мы используем $+$ и $-$ для операций сложения и вычитания двух мультимножеств, и $=, <, >, \leq, \geq$ для сравнения мультимножеств, которые определены естественным образом. Пустое мультимножество обозначаем через $\emptyset$, а принадлежность элемента мультимножеству — через $\in$.

2.1. Раскрашенные сети Петри

Определим раскрашенные сети Петри над универсумом U.

Каждая позиция раскрашенной сети типизирована подмножеством U. Дуги помечены выражениями языка *Expr* над множеством переменных *Var* и множеством констант *Con*. Имеется фиксированная интерпретацией $\mathcal{I}$, такая что для любой согласованной по типам оценки $\nu : Var \to U$ значение $\mathcal{I}(e, \nu) \in \mathbb{N}^U$ выражения $e \in Expr$ определено. Множество меток переходов *Lab* не содержит τ. Меткой τ помечаются скрытые переходы, тогда как метками из *Lab* помечаются наблюдаемые переходы сети.

Определение 1. Раскрашенная сеть *над универсумом* U *– это кортеж* $(P, T, F, \upsilon, \rho, \Lambda)$, *где*

- P *и* T *– непересекающиеся конечные множества* позиций *и* переходов*;*
- $F \subseteq (P \times T) \cup (T \times P)$ *– множество* дуг*;*

- $\upsilon : P \to 2^U$ – *функция* типов позиций, *отображающая* P *в подмножества* U*;*
- $\rho : F \to Expr$ – *функция* меток дуг*;*
- $\Lambda : T \to Lab \cup \{\tau\}$ – *функция* меток переходов.

Для раскрашенной сети $N = (P, T, F, \upsilon, \rho, \Lambda)$ над универсумом U *разметка* N есть функция $m : P \to \mathbb{N}^U$, такая, что $m(p) \in \mathbb{N}^{\upsilon(p)}$ для $p \in P$. Пара (N, m) называется размеченной сетью Петри.

Пусть $N = (P, T, F, \upsilon, \rho, \Lambda)$ – раскрашенная сеть Петри. Переход $t \in T$ *активен* в разметке m тогда и только тогда, когда $\exists \nu \forall p \in P : (p, t) \in F \Rightarrow m(p) \geq \mathcal{I}(\rho(p, t), \nu)$. Активный переход t может *сработать*, порождая новую разметку $m'(p) = m(p) - \mathcal{I}(\rho(p, t), \nu) + \mathcal{I}(\rho(t, p), \nu)$ для каждого $p \in P$ (обозначается $m \xrightarrow{t} m'$).

Размеченная раскрашенная сеть определяет систему переходов, представляющую наблюдаемое поведение сети.

2.2. Вложенные сети Петри

Определим *вложенные сети Петри* (*NP-сети*) как расширенные раскрашенные сети Петри над специальным универсумом. Этот универсум состоит из элементов некоторого конечного множества S (называемых атомарными фишками) и размеченными сетями Петри (называемыми сетевыми фишками). Для простоты мы рассматриваем только двухуровневые NP-сети, в которых сетевые фишки являются раскрашенными сетями Петри над конечным универсумом.

Пусть S будет конечное множество атомарных объектов. Для раскрашенной сети Петри N с помощью $\mathfrak{M}(N, S)$ мы обозначим множество всех размеченных сетей, получаемых из N добавлением разметок над универсумом S.

Пусть $N_1, \ldots, N_k$ – раскрашенные сети Петри над универсумом S. Определим универсум $\mathcal{U}(N_1, \ldots, N_k) = S \cup \mathfrak{M}(N_1, S) \cup \cdots \cup \mathfrak{M}(N_k, S)$ с множеством типов $\Omega(N_1, \ldots, N_k) = \{S, \mathfrak{M}(N_1, S), \ldots, \mathfrak{M}(N_k, S)\}$. Такие множества $\mathfrak{M}(N, S)$ мы будем называть типами.

Определение 2. *Пусть* Lab *— множество меток переходов, и пусть* $N_1, \ldots, N_k$ *— раскрашенные сети Петри над универсумом* S*, в которых все переходы помечены метками из* $Lab \cup \{\tau\}$*. NP-сеть определим как кортеж* $NP = (N_1, \ldots, N_k, SN)$*, где* $N_1, \ldots, N_k$ *называются элементными сетями, и* SN *называется системной сетью. Системная сеть* $SN = (P_{SN}, T_{SN}, F_{SN}, \upsilon, \rho, \Lambda)$ *— это раскрашенная сеть Петри над универсумом* $\mathcal{U} = \mathcal{U}(N_1, \ldots, N_k)$*, где позиции типизированы типами из* $\Omega = \Omega(N_1, \ldots, N_k)$*, переходы помечены метками из* $Lab \cup \{\tau\}$*, а язык меток дуг* $Expr$ *определен, как показано ниже. Для краткости говорим, что позиция типа* $\mathfrak{M}(N, S)$ *имеет тип* N*.*

Пусть Con *– множество констант, интерпретированных над* $\mathcal{U}$ *и* Var *— множество переменных типизированных* Ω*-типами. Тогда выражения языка* $Expr$ *суть мультимножества элементов одного типа над* $Con \cup Var$ *с двумя дополнительными ограничениями для каждого перехода* $t \in T_{SN}$*:*

1. *константы и множественное вхождение переменной не допускаются в выражениях на входных дугах переходов;*

2. *каждая переменная из выражений на выходных дугах перехода t должна встречаться в одном из выражений на входных дугах этого перехода.*

В силу первого ограничения нельзя проверить равенство внутренних разметок сетевых фишек. Без такого ограничения можно реализовать проверку на нулевую разметку сетевых фишек и формализм NP-сетей становится Тьюринг полным. Второе ограничение исключает бесконечное ветвление в системе переходов, представляющей поведение NP-сети.

Размеченная элементная сеть называется сетевой фишкой, а элемент множества S — атомарной фишкой.

Разметка NP-сети полностью определяется разметкой ее системной сети. Разметка в NP-сети отображает каждую позицию системной сети в мультимножество атомарных или сетевых фишек соответствующего типа.

Поведение NP-сети состоит из трех видов шагов. *Элементно-автономный шаг* – это срабатывание перехода, помеченного τ, в одной из сетевых фишек текущей разметки, в соответствии с обычными правилами срабатывания для раскрашенных сетей Петри. Такие срабатывания изменяют только внутреннюю разметку одной из сетевых фишек.

Системно-автономный шаг – это срабатывание перехода, помеченного τ, в системной сети, в соответствии с правилами срабатывания для раскрашенных сетей Петри, как если бы сетевые фишки были просто раскрашенными фишками без внутренней разметки. Автономный шаг в системной сети может перемещать, копировать, генерировать или уничтожать фишки, задействованные в этом шаге, но не может изменить их внутренние разметки.

Шаг (вертикальной) синхронизации – это одновременное срабатывание перехода, помеченного меткой $\lambda \in Lab$, в системной сети вместе со срабатыванием переходов, помеченных λ, во всех сетевых фишках задействованных (поглощаемых) срабатыванием этого перехода системной сети.

Таким образом, шаг – это множество переходов (одноэлементное множество в случае автономного шага). Мы обозначаем $M \xrightarrow{s} M'$ то, что срабатывание шага s преобразует разметку M в разметку M'. Описанные шаги естественным образом определяют поведение NP-сети в виде системы переходов.

Детальное описание NP-сетей можно найти в [15, 1]. В данной статье мы рассматриваем типизированный вариант NP-сетей, в котором для каждой позиции определен её тип.

3. Трансляция вложенных сетей Петри в раскрашенные

В зависимости от того, какой вид анализа требуется провести для исходной модели, возникают различные требования, накладываемые на трансляцию. Первое требование связано с необходимостью проводить имитационное моделирование. Для имита-

ционного моделирования необходимо, чтобы целевая модель на каждом шаге вела себя так же, как исходная модель. Это означает, что модель должна предлагать такой же выбор действий и срабатывание выбранного действия должно переводить целевую модель в состояние, подобное состоянию исходной модели после срабатывания такого же действия. При этом допускается, что целевой модели необходимо совершить несколько промежуточных шагов, соответствующих одному шагу в исходной модели. Описанное требование является по существу неформальным описанием отношения слабой бисимуляции. Заметим, что описанная далее трансляция обеспечивает даже более сильное отношение сильной бисимуляции.

Второе требование возникает из необходимости строить граф достижимости для исходной модели по целевой модели. Для того чтобы по графу достижимости целевой модели можно было анализировать исходную модель, необходимо установить, каким образом узлы построенного графа достижимости (состояния полученной модели) соотносятся с состояниями исходной модели. Далее мы установим взаимооднозначное соответствие между состояниями исходной и полученной модели.

Наконец, третье требование обусловлено алгоритмом построения графа достижимости в инструменте (State Space Tool) из инструментария CPN Tools. Для корректного построения графа достижимости необходимо, чтобы срабатывание перехода при любой заданной оценке в CP-сети однозначно определяло следующее состояние CP-сети [22], т.е. получаемая CP-сеть должна быть детерминированной относительно срабатывания перехода. В частности, это накладывает ограничения на реализацию переходов вертикальной синхронизации NP-сети в полученной CP-сети, что будет рассмотрено при построении трансляции.

Таким образом определены следующие требования к трансляции NP-сетей в CP-сети: подобие поведений исходной и целевой модели; соответствие состояний исходной и целевой модели; совместимость целевых моделей с алгоритмом построения графа достижимости в CPN Tools.

Алгоритм трансляции *NP*-сетей в *CP*-сети

Опишем трансляцию NP-сетей в CP-сети. Для заданной NP-сети $NPN = (N_1, \ldots, N_k, SN)$ с начальной разметкой M_0 результатом трансляции будет CP-сеть $CPN = (P, T, F, \upsilon, \rho, \Lambda, m_0)$, которая удовлетворяет критериям, перечисленным выше.

Для данной NP-сети $NPN = (N_1, \ldots, N_k, SN)$, структура графа соответствующей CP-сети $CPN = (P_C, T_C, F_C, \upsilon, \rho, \Lambda, m_0)$ подобна структуре системной сети SN. Одновременно с построением CPN определяем биекцию $tr_P : P_{SN} \to P_C$ и сюръекцию $tr_T : T_{SN} \cup T_{N1} \cup \cdots \cup T_{Nk} \to T_C$. Первая функция отображает позиции системной сети на позиции целевой CP-сети CPN, а вторая отображает переходы системной и элементных сетей на переходы CPN.

Трансляция состоит из следующих пяти шагов.

Шаг 1 определяет позиции CPN. Для каждой позиции $p \in P_{SN}$ в системной сети SN NP-сети NPN строится новая позиция $p' \in P_C$, и задаем $tr_P(p) = p'$.

Если позиция p в *SN* типизирована множеством S атомарных объектов, тогда $p' = tr_P(p)$ также типизирована S в *CPN*.

Пусть теперь p типизирована элементной сетью E в *SN*. Тогда позиция $p' = tr_P(p)$ в *CPN* будет типизирована множеством векторов натуральных чисел (кортежи натуральных чисел), где размерность вектора равна размерности разметок элементной сети E, т.е. мощности множества позиций в E.

Тогда графовая структура сетевой фишки определяется типом позиции *CPN*, в которой располагается фишка. Разметки сетевой фишки будут представляться как векторы натуральных чисел в соответствии со стандартным векторным представлением разметок сетей Петри. С помощью $tr_M(M)$ обозначаем кортеж натуральных чисел, представляющих разметку M.

CPN Tools поддерживает кортежи, но имеет ограничения на размерность векторов. Поэтому для представления векторов натуральных чисел используются списки языка CPN ML. Операции над векторами *сложение(ms_ add)*, *лексикографическое(ms_ comp)* и *поэлементное сравнение(ms_ comp_ el)* реализованы как функции на языке CPN ML. Эти функции используются в выражениях на дугах для эмуляции срабатываний переходов NP-сети.

Шаг 2 задает элементно-автономные переходы в *CPN*. Для каждой позиции $p \in P_{SN}$, типизированной элементной сетью $E = \upsilon(p)$, и для каждого перехода $t \in T_E$, помеченного τ, строим новую переход-петлю t' в *CPN* и определяем $tr_T(t) = t'$. Переход t' имеет одну входную дугу $(tr_P(p), t')$ подписанную переменной x соответствующего типа целочисленных векторов. Единственная выходная дуга $(t', tr_P(p))$ подписана *"ms_ add x te"*, где *te* – это разность результирующей и исходной разметки при срабатывании перехода t' в векторном представлении.

Чтобы избежать неправильных срабатываний, накладываем условие срабатывания на t' для проверки активности перехода — *"ms_ comp_ el x prete"*, где *prete* – это необходимое предусловие в векторном представлении. Результат применения данного шага проиллюстрирован на Рис. 1.

Шаг 3 задает эмуляцию системно-автономных переходов. Для каждого перехода $t \in T_{SN}$, помеченного τ, строим новый переход t' в *CPN* и задаем $tr_T(t) = t'$. Для каждой дуги $f = (p, t)$ или $f = (t, p)$ в F_{SN} добавляем новую дугу $f' = (tr_P(p), tr_T(t))$ или, соответственно, $f' = (tr_T(t), tr_P(p))$, помеченную $\rho_{SN}(f)$. Этот шаг проиллюстрирован на Рис. 2.

Шаг 4 задает эмуляцию переходов синхронизации. Для каждого перехода $t \in T_{SN}$ с меткой синхронизации строим новый переход t', который эмулирует шаги вертикальной синхронизации в *NPN*. Срабатывание перехода синхронизации состоит из одновременного срабатывания системного перехода и срабатываний переходов с такой же меткой синхронизации в каждой вовлеченной сетевой фишке. Таким образом, Шаг 4 – это комбинация Шагов 2 и 3. Выражения на дугах, описывающие срабатывания переходов в сетевых фишках, приписываем к выходящим дугам t' в *CPN*. Добавляем условия срабатывания аналогично

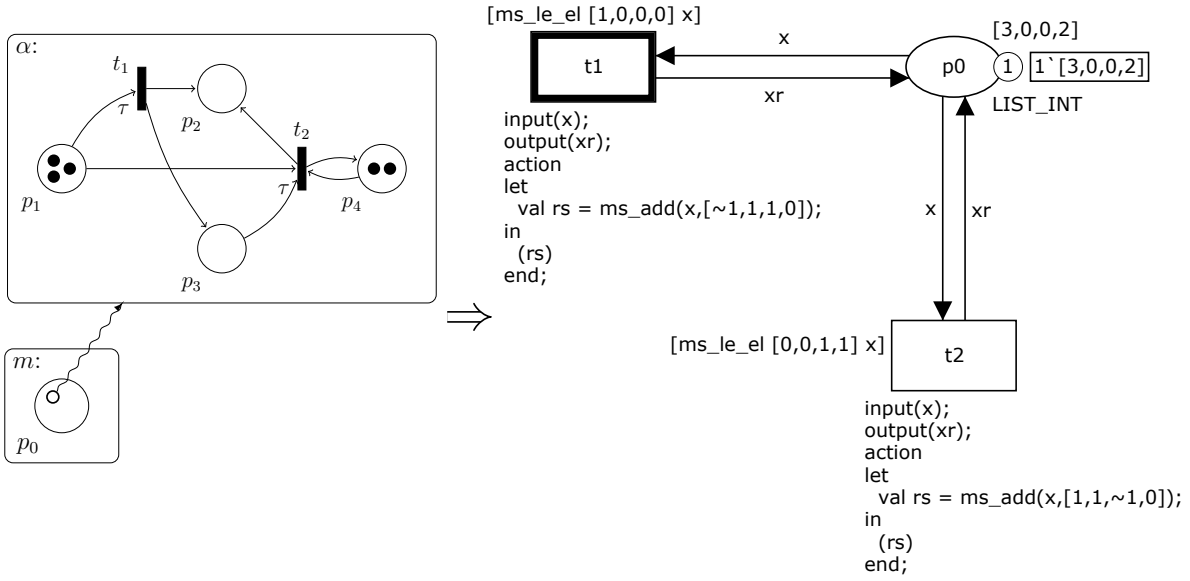

Рис. 1. Шаг трансляции для элементно-автономного перехода

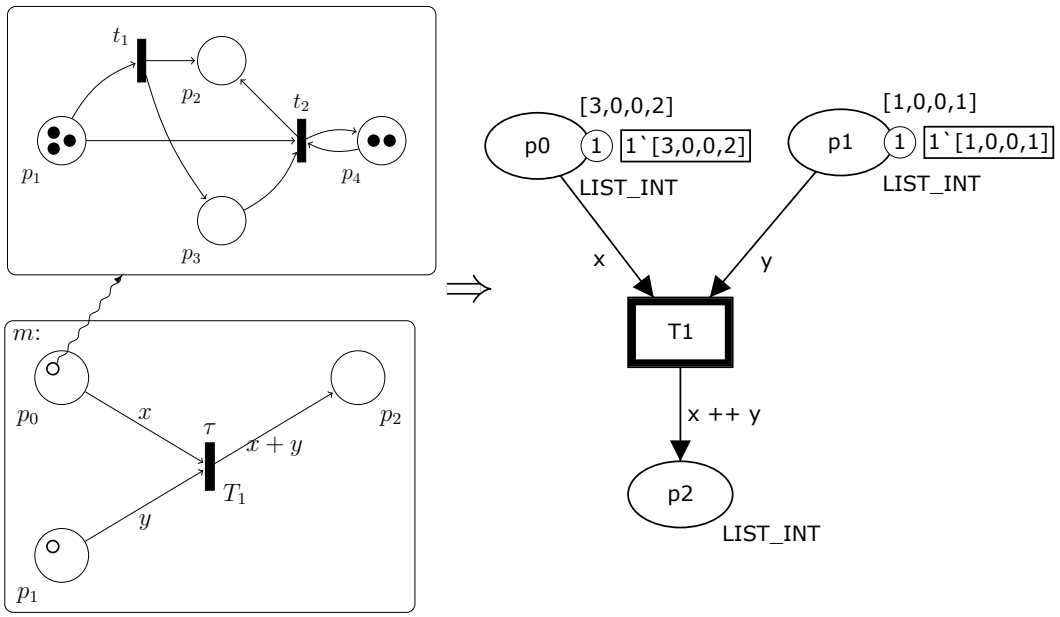

Рис. 2. Шаг трансляция для системно-автономного перехода

Шагу 2. Результат применения Шага 4 проиллюстрирован на Рис. 3. Мы не рассматриваем все технические решения для реализации трансляции. Однако заметим, что в случае, когда более одного перехода помечено одинаковой меткой синхронизации в элементой сети N, то срабатывание перехода синхронизации с оценкой ν, задействующей сетевую фишку типа N, может из данного состояния перевести систему в различные состояния в зависимости от того, какой из одинаково помеченных переходов сетевой фишки был задействован

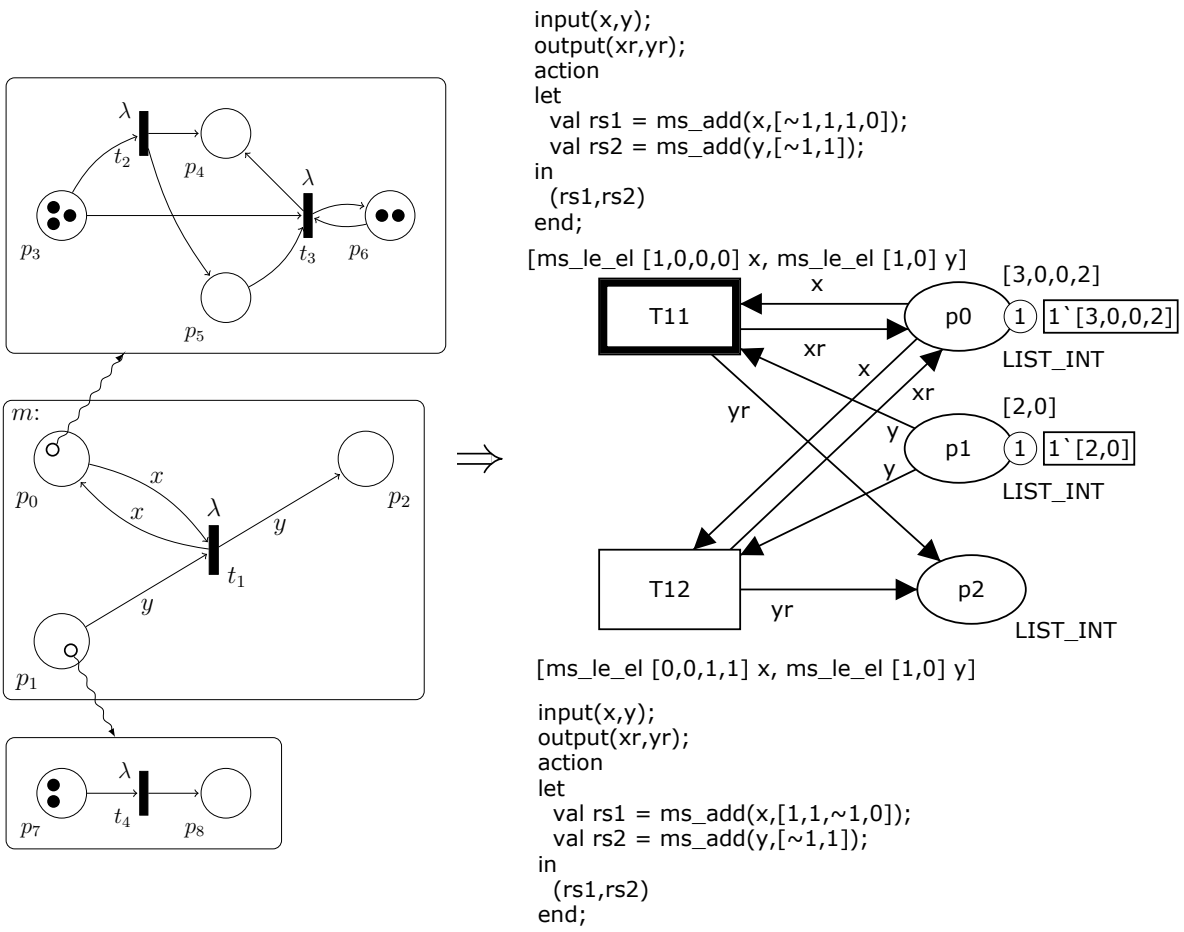

Рис. 3. Шаг трансляции для перехода синхронизации

в шаге синхронизации. В силу ограничения 3, такой переход не может быть реализован с помощью одного перехода в конструируемой CP-сети. Поэтому для каждой возможной комбинации переходов системной и элементных сетей, вовлеченных в срабатывание, строится отдельный переход.

Шаг 5. Пусть M_0 – начальная разметка в *NPN*. Отметим, что разметка NP-сети *NPN* определена как разметка ее системной сети *SN*. Начальную разметку m_0 целевой CP-сети *CPN* получим следующим образом. Для каждой позиции $p \in SN$, типизированной атомарным типом, пусть $m_0(tr_P(p)) = M_0(p)$, а для позиции p, типизированной элементной сетью, $m_0(tr_P(p)) = tr_M(M_0(p))$.

Теорема 1. *Пусть NPN — размеченная NP-сеть, и пусть CPN — размеченная CP-сеть, полученная как результат описанной выше трансляции. Тогда NPN и CPN бисимулярны.*

Пусть *SN* — системная сеть в *NPN*. Пусть $\mathfrak{M}(NPN)$ и $\mathfrak{M}(CPN)$ – множества всех достижимых разметок для *NPN* и *CPN* соответственно. Определим отношение $\mathcal{R} \subset \mathfrak{M}(NPN) \times \mathfrak{M}(CPN)$, такое что $(M, m) \in \mathcal{R}$ тогда и только тогда, когда для каждой

$p \in SN$, типизированной атомарным типом, $m_0(tr_P(p)) = M_0(p)$, и для каждой $p \in SN$, типизированной элементной сетью, $m_0(tr_P(p)) = tr_M(M_0(p))$. То, что отношение $\mathcal{R}$ является бисимуляцией, напрямую следует из определения трансляции.

4. Реализация и апробация

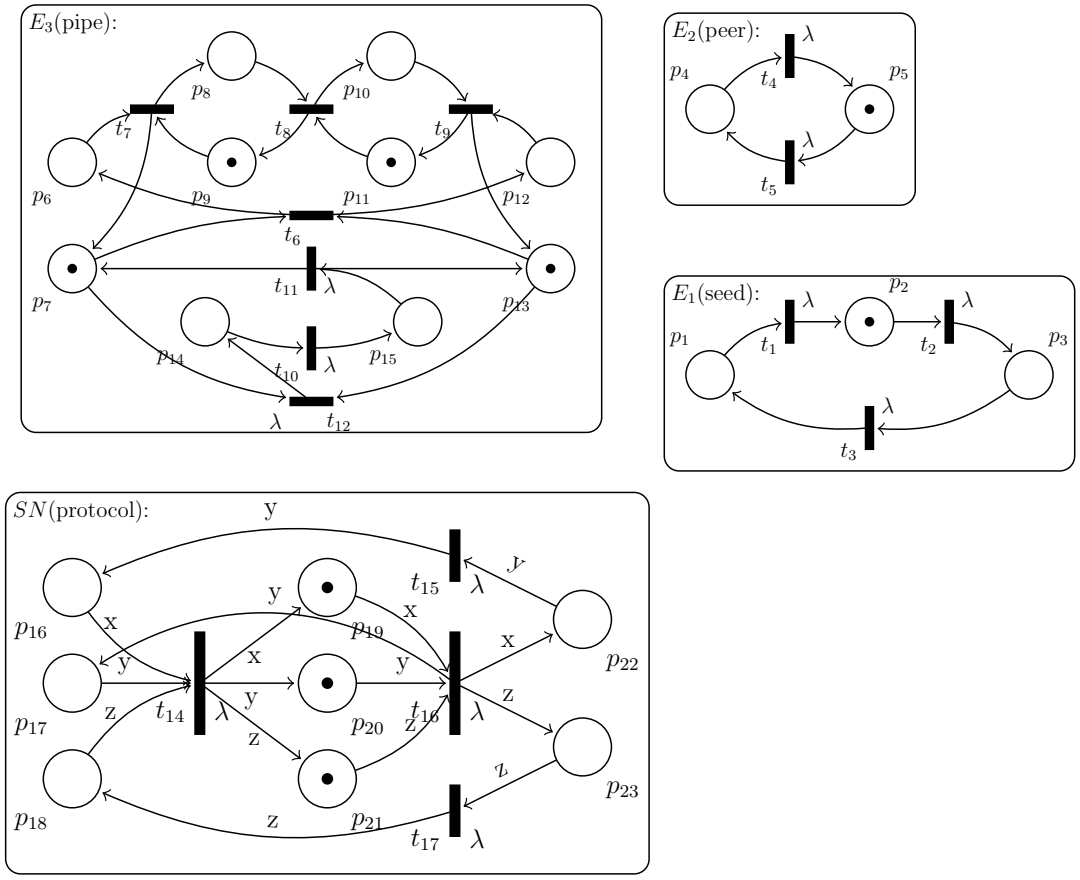

Рис. 4. NP-сеть, моделирующая простой одноранговый протокол из [8]

Описанная выше трансляция была реализована в виде транслятора NPN2CPN на языке Java. В качестве входного файла транслятор получает описание NP-сети в формате XML. Результатом работы транслятора является файл в формате CPN Tools, содержащий целевую CP-сеть. Так как полученная CP-сеть является исполнимой, то становится возможным провести имитационное моделирование с помощью инструмента Simulation tools или анализ производительности с помощью механизма Monitors и функции CPN‘Replications, позволяющей автоматически производить несколько исполнений целевой CP-сети. Полный граф достижимости или его начальный фрагмент CP-сети может быть построен с помощью инструмента State space tools. Если пространство состояний CP-сети небольшое и удается полностью построить граф достижимости, то становится возможным применить все инструменты анализа State space tools – проверку достижимости разметки, живо-

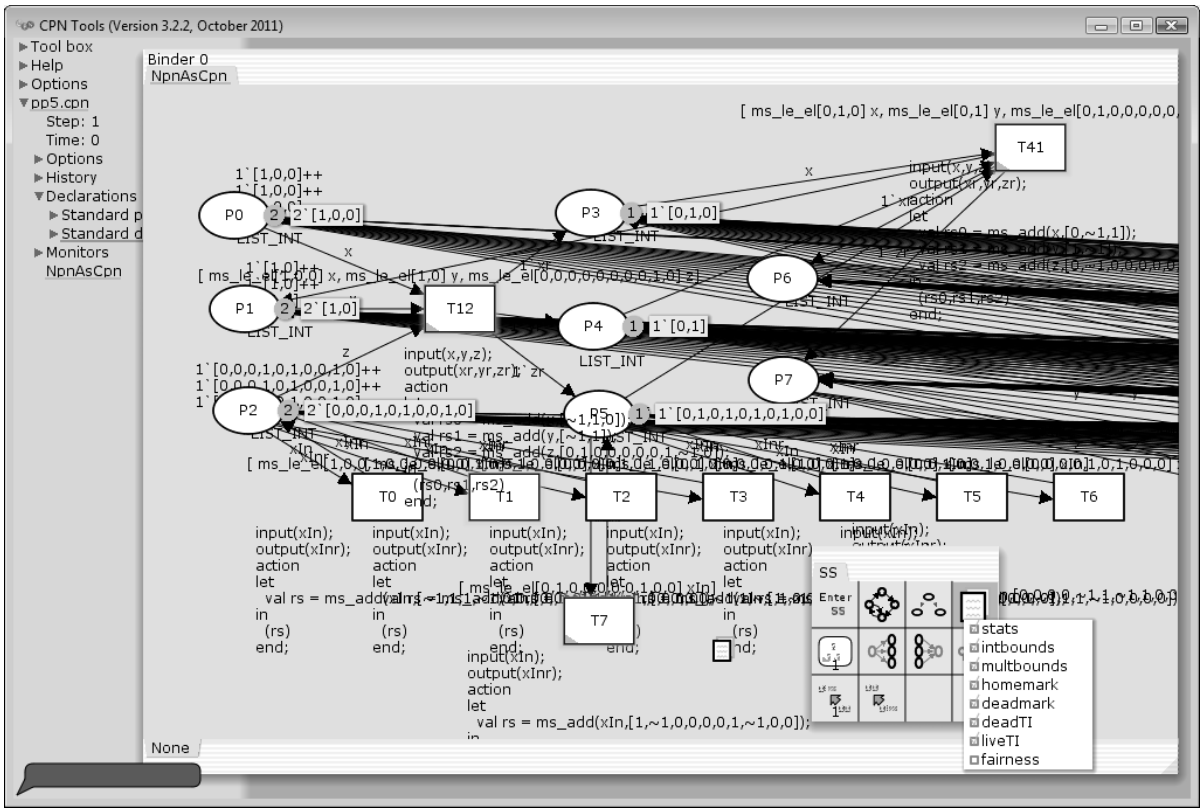

Рис. 5. CPN Tools с загруженным результатом трансляции

сти, справедливости и других свойств, а также инструмент ASK-CTL для проверки выполнимости темпоральных свойств, выраженных на расширении CTL логики, называемой ASK-CTL. Если удается построить только начальный фрагмент графа достижимости, то нельзя проверить глобальные свойства системы, но сохраняется возможность обнаружить дефекты системы, проявляющиеся уже в начале исполнения системы. При обнаружении такого дефекта можно получить как достижимую разметку целевой CP-сети, так и путь из начальной разметки в данную разметку. Однако для интерпретации полученной разметки или пути на исходной NP-сети необходимо осуществлять этот перевод вручную, используя метод, описанный в Шаге 5. Несмотря на простоту такого перевода, это затрудняет интерпретацию длинных путей и является недостатком текущей реализации.

Транслятор был апробирован на нескольких небольших модельных примерах. Рассмотрим, в частности, анализ NP-сети, изображенной на Рис. 4, моделирующей простой одноранговый протокол из статьи [8]. С помощью трансляции была получена поведенчески эквивалентная CP-сеть. На Рис. 5 показано окно CPN Tools, в котором открыт фрагмент полученной CP-сети. На рисунке видно большое количество переходов, которые получаются в результате перебора всех возможных шагов вертикальной синхронизации для каждого перехода синхронизации системной сети, а также построения переходов-петель в каждой позиции для всех возможных элементно-автономных шагов. Однако соответствие между позициями и разметками целевой и исходной сети устанавливается легко.

Был проведен анализ модели, соответствующей начальной конфигурации протокола с 10 клиентами, 10 поставщиками данных и 10 доступными каналами. Анализ был проведен на компьютере с процессором AMD Brisbone 2.2 GHz, операционной системой Windows Vista и инструментом CPN Tools 3.4.0. Замеры выполнялись с помощью стандартного инструмента Windows Reliability and Performance Monitor. Граф достижимости содержит 13013 узлов и был построен за 76 секунд. Комбинированная проверка живости, поиск опорных разметок (home markings) и границ разметок позиций заняли 106 секунд. Проверка справедливости выполнялась 154 минуты и показала наличие несправедливых переходов. Также было проверено несколько поведенческих свойств системы, выраженных на языке ASK-CTL. Например, проверка существования путей, в которых начальная разметка не встречается (`NOT(POS(NOT(EV(NF(“”,IsInitial))))`), заняла 12 секунд. В результате анализа были обнаружены недостатки построенной модели и достигнуто лучшее понимание работы моделируемой системы.

5. Заключение

В этой работе разработана трансляция вложенных сетей Петри в раскрашенные сети и обосновано применение ее для имитационного моделирования и анализа вложенных сетей средствами инструментария CPN Tools.

Идея трансляции моделей одного формализма в другой с целью использовать существующий инструментарий в качестве виртуальной машины для имитационного моделирования и анализа исходных моделей не нова. Например, тот же CPN Tools используется в качестве виртуальной машины для исполнения EPC моделей в инструменте Prom и в инструменте YAWL. YAWL в свою очередь используется в качестве целевого языка в инструменте BPMN2YAWL для BPMN моделей [10].

Предложенная трансляция реализована в виде инструментария NPN2CPN, который позволяет проводить имитационное моделирование и анализ производительности NP-сетей с помощью CPN Tools. В случае NP-сетей с конечным и не очень большим пространством состояний, когда удается полностью построить граф достижимости, использование NPN2CPN позволяет дополнительно проверить исходную систему на ограниченность, а также использовать весь функционал CPN Tools для анализа поведенческих свойств и проверки выполнимости темпоральных формул. В случае систем с большим или бесконечным пространством состояний подобный анализ поведенческих свойств и проверка темпоральных формул возможна только на начальном фрагменте графа достижимости. Однако даже анализ начального фрагмента имеет практическую ценность, т.к. многие дефекты проявляются на исполнениях небольшой глубины. Разработанный подход успешно апробирован на небольших примерах.

Отметим, что в настоящее время результаты анализа целевой модели приходится интерпретировать на исходной модели “вручную”. В случае найденного дефекта пользователь сам должен осуществить перевод состояния или истории исполнения обратно, на язык NP-сетей. Хотя такой перевод достаточно прост, указанный недостаток затрудняет применение технологии для больших систем. В дальнейшем

предполагается реализовать компьютерную поддержку интерпретации результатов анализа на исходной модели. Также большой интерес представляет использование новых возможностей CPN Tools [2].

Описанная в работе трансляция может быть расширена на многоуровневые вложенные сети Петри. Для этого необходимо реализовать внутренние переходы фишек из более высоких уровней как переходы-петли на позициях CP-сети, в которой могут оказаться эти фишки, возможно, внутри других фишек. Применение трансляции в таком случае осложняется комбинаторным ростом числа переходов-петель.

Список литературы

1. *Ломазова И. А.* Вложенные сети Петри: моделирование и анализ распределенных систем с объектной структурой. М.: Научный Мир, 2004. 208 с.

2. *van der Aalst, W.M.P., Stahl C., Westergaard M.* Strategies for Modeling Complex Processes Using Colored Petri Nets // ToPNoC V. 2012. LNCS 6900. P. 265–291.

3. *Bashkin V.A., Lomazova I.A.* Resource Driven Automata Nets // Fundamenta Informaticae. 2011. Vol. 109(3). P. 223–236.

4. *Bednarczyk M.A., Bernardinello L., Pawlowski W., Pomello L.* Modelling mobility with Petri Hypernets // Recent Trends in Algebraic Development Techniques. 2005. LNCS 3423. P. 28–44.

5. *Bocăneală C.* Modeling inteligent systems with level Petri nets // Annals of Dunarea de Jos Year. 2008. Vol. 31(2). P. 31–36.

6. *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. P. 132–145.

7. *Dvoryansky L.V., Lomazova I.A.* Compositionality of some behavioral properties for free-choice nested Petri nets. // Proc. Second Workshop "Program Semantics, Specification and Verification: Theory and Application". Yaroslavl, 2011. P. 27–36.

8. *Dworzański L. W., Lomazova I.A.* On Compositionality of Boundedness and Liveness for Nested Petri Nets // Fundamenta Informaticae. 2012. Vol. 120(3–4).P. 277–295.

9. *Hoffmann K., Ehrig H., Mossakowski T.* High-level nets with nets and rules as tokens // Proc. ICATPN. 2005. LNCS 3536. P. 268–288.

10. *ter Hofstede A.H.M., van der Aalst W.M.P., Adams M., Russell N.* Modern Business Process Automation: YAWL and its support environment. Springer, 2010. 676 p.

11. *Jensen K., Kristensen L. M.* Coloured Petri Nets: Modelling and Validation of Concurrent Systems. Springer, 2009. 396 p.

12. *Jensen K., Kristensen L.M., Wells L.* Coloured Petri Nets and CPN Tools for Modelling and Validation of Concurrent Systems // International Journal on Software Tools for Technology Transfer. 2007. Vol. 9(3–4). P. 213–254.

13. *Köhler M., Farwer M.* Object nets for mobility // Proc. ICATPN. 2007. LNCS 4546. P. 244–262.

14. *Köhler-Bussmeier M.* Hornets: Nets within nets combined with net algebra // Proc. Applications and Theory of Petri Nets. 2009. LNCS 5606. P. 243–262.

15. *Lomazova I. A.* Nested Petri nets — a Formalism for Specification and Verification of Multi-Agent Distributed Systems // Fundamenta Informaticae. 2000. Vol. 43(1–4). P. 195–214.

16. *Lomazova I. A.* Nested Petri Nets: Multi-level and Recursive Systems // Fundamenta Informaticae. 2001. Vol. 47(3–4). P.283–293.

17. *Lomazova I. A.* Nested Petri Nets for Adaptive Process Modeling // Pillars of Computer Science: Essays Dedicated to Boris (Boaz) Trakhtenbrot on the Occasion of His 85th Birthday. 2008. LNCS 4800. P. 460–474.

18. *Mascheroni M., Farina F.* Nets-Within-Nets paradigm and grid computing // Transactions on Petri Nets and Other Models of Concurrency V. 2012. LNCS 7347. P. 201–220.

19. *Pawlowski W.* Petri Hypernets with Constraints // Proc. "Concurrency, Specification and Programming (CS&P 2009)". 2009. P. 467–479.

20. *Valk R.* Petri Nets as Token Objects: An Introduction to Elementary Object Nets // Proc. of ICATPN'98. LNCS1420. 1998. P. 1–25.

21. *Valk R.* Object Petri nets: Using the nets-within-nets paradigm // Lectures on Concurrency and Petri Nets. 2004. LNCS 3098. P. 819–848.

22. CPN Tools Manual: Nondeterministic nets. `http://cpntools.org/documentation/tasks/verification/nondeterministic_nets`

CPN Tools-Assisted Simulation and Verification of Nested Petri Nets

Dworzański L. W., Lomazova I. A.

Keywords: nested Petri nets, colored Petri nets, simulation, verification, reachability graph

Nested Petri nets (NP-nets) are an extension of Petri net formalism within the "nets-within-nets" approach, when tokens in a marking are Petri nets, which have an autonomous behavior and are synchronized with the system net. The formalism of NP-nets allows modeling multi-level multi-agent systems with dynamic structure in a natural way. Currently, there is no tool for supporting NP-nets simulation and analysis. The paper proposes the translation of NP-nets into Colored Petri nets and the use of CPN Tools as a virtual machine for NP-nets modeling, simulation and automatic verification.

Сведения об авторах:
Дворянский Леонид Владимирович,
Национальный исследовательский университет "Высшая школа экономики",
преподаватель;
Ломазова Ирина Александровна,
Национальный исследовательский университет "Высшая школа экономики",
профессор.