Верификация коммуникационных протоколов с использованием сетей Петри*

Анисимов Н.А., Поступальский П.А.

Аннотация

Настоящая работа посвящена проблеме полной верификации протоколов, которая заключается в доказательстве соответствия протокола предоставляемому сервису. В качестве формального аппарата используются сети Петри. Показывается, что данная задача сводится к проверки двух помеченных сетей Петри на шаговую бисимуляционную эквивалентность. Для решения этой задачи используются алгоритмы построения сокращенного дерева достижимости и алгоритм проверки двух систем переходов на бисимуляционную эквивалентность. Вводится более слабая версия шаговой бисимуляционной эквивалентности, названная распределенной эквивалентностью, которая, как показано, достаточна для верификации протоколов. Описывается реализация введенного подхода.

1 Введение

Одной из наиболее сложных и актуальных задач при разработке протоколов информационно-вычислительных сетей является задача их верификации. В наиболее полной постановке эта задача заключается в демонстрации того, что распределенные в пространстве объекты, взаимодействуя в соответствии с протоколом, предоставляют пользователям необходимый сервис [28]. Для решения этой задачи необходимо, в первую очередь, иметь возможность представлять протокол и сервис в единых математических терминах. Во то же время необходимо что бы этот математический аппарат обладал подходящим понятием соответствия или, другими словами, эквивалентности. Следует отметить, что в данном случае необходима поведенческая эквивалентность объектов, т.е.

*Работа поддержана Российским фондом фундаментальных исследований (Грант No. 96-01-00177).

такая эквивалентность, которая абстрагируясь от внутренней структуры объектов учитывает только их проявления во внешнем мире.

Все существующие формализмы, которые принципиально применимы для решения задачи верификации протоколов, можно разбить на две группы. К одной из них относятся модели алгебраических теорий параллельных процессов, представителями которых являются CCS [23], TCSP [4], ACP [7]. Основным недостатком этих теорий является то, что они ограниченно трактуют параллелизм событий как недетерминированный выбор возможных последовательностей. В частности, в такой трактовке два параллельных события не могут выполниться одновременно. С другой стороны этот подход обладает хорошо разработанными понятиями поведенческой эквивалентности [11, 13], среди которых фундаментальную роль играет бисимуляционная эквивалентность [25]. Известен ряд алгоритмов проверки процессов на бисимуляционную эквивалентность [9, 20, 15, 22, 24] и их реализаций [15, 22, 12, 14].

К другой группе формализмов относятся модели теории сетей Петри [26, 3, 10], основным преимуществом которых является более адекватная трактовка параллелизма событий. В частности, этот подход допускает и одновременное выполнение двух параллельных событий. Для сетей Петри были адаптировано и обобщено большинство поведенческих эквивалентностей теорий параллельных процессов [27], учитывающих бóльшую степень параллелизма. Однако необходимая алгоритмическая поддержка этих эквивалентностей для сетей Петри в настоящее время практически отсутствует, что существенно препятствует использованию сетей Петри на практике.

В данной работе мы рассматриваем задачу верификации сетевых протоколов, используя подход сетей Петри, которая, как показывается, сводится к проверке двух помеченных сетей Петри на бисимуляционную эквивалентности. Предлагается двухэтапная схема такой проверки, заключающаяся в построении графов достижимости обеих сетей и сравнении их известными алгоритмами. Определенные обстоятельства, обусловленные спецификой задачи, позволяют сократить размеры графов достижимости и, в конечном счете, уменьшить сложность проверки. Данный подход реализован в рамках автоматизированной системы разработки протоколов названной PN^3-Tool [6].

2 Задача верификации протоколов и бисимуляционная эквивалентность

Рассмотрим задачу верификации протоколов, используя общепринятую стандартную терминологию Эталонной модели архитектуры взаимодействующих систем [16]. Согласно этой модели архитектура разбивается на протокольные уровни, где структура каждого уровня представляется в терминах сервиса, точек доступа к сервису, протокольных объектов, протокола. Принадлежность этих понятий уровню с номером N будет обозначаться соответствующим префиксом, например: N-протокол, N-объект, $(N-1)$-сервис.

N-уровень может быть представлен двумя способами (Рис.1). Во-первых, назначением N-уровня является предоставление объектам более верхнего $(N+1)$-уровня N-сервиса. Поэтому N-уровень может быть представлен в виде эталонного N-сервиса $\mathbf{SE_N^R}$, т.е. сервиса, который **должен** предоставлять N-уровень, см. Рис.1.(i)[1]. Во-вторых, для обеспечения эталонного сервиса N-уровень содержит множество N-объектов, которые взаимодействуют друг с другом в соответствии с N-протоколом. Для обмена протокольными командами N-объекты используют сервис нижнего уровня – $(N-1)$-сервис. На Рис.1.(ii) представлена структура N-уровня, содержащая два N-объекта $\mathbf{PE_N}$, взаимодействующих в соответствии с протоколом $\mathbf{P_N}$, и $(N-1)$-сервис $\mathbf{SE_{(N-1)}}$. Такая модель N-уровня соответствует реальному N-сервису и обозначается $\mathbf{SE_N^I}$.

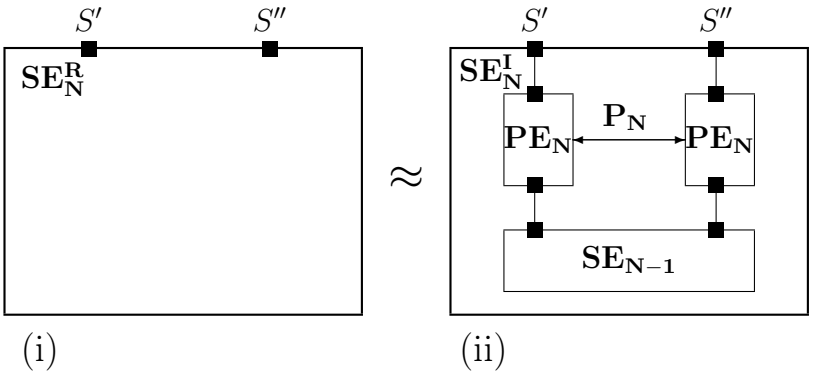

Рис. 1: Модель верификации протокола

Теперь можно сказать, что задача верификации протокола N–уровня

[1]Без ограничения общности будем считать, что сервис имеет две точки доступа S' и S''.

заключается в доказательстве эквивалентности эталонного сервиса реальному. Иными словами, необходимо показать, что сервисы $\mathbf{SE^R_N}$ и $\mathbf{SE^I_N}$ эквивалентны в точках доступа S' и S''. Протокол $\mathbf{P_N}$ в этом случае называется корректным. Заметим, что корректность протокола определяется по отношению к предоставляемому и используемому сервисам.

Рассмотрим каким требованиям должна удовлетворять данная эквивалентность. Согласно фундаментальному принципу эталонной модели уровни независимы друг от друга. Это означает, что изменения во внутренней структуре, скажем, N-уровня нисколько не должно сказаться на работу других уровней. В частности, возвращаясь к структуре на Рис.1, функционирование объектов $(N+1)$-уровня не должно измениться при замене эталонного сервиса на реальный. Иными словами данная эквивалентность должна быть конгруэнтной по отношению к операции замены одной подсистемы на другую.

Для спецификации сервисов и протокольных объектов в работе [1] была введена модель, названная объектом сети Петри. Объект сети Петри представляет собой сеть Петри, снабженную множеством пометочных функций, где каждая пометочная функция интерпретируется как точка доступа к объекту, через которую можно взаимодействовать с объектом. Также введено правило композиции объектов через точки доступа, которое формализует правила композиции, используемые на Рис.1. Кроме того было введено понятие эквивалентности объектов в точках доступа, основанное на бисимуляционно эквивалентности сетей Петри. В работе [5] было показано, что для того, что бы бисимуляционная эквивалентность была конгруэнтна по отношению к операции композиции, она должна быть шаговая, т.е. должна учитывать срабатывание не только одиночных переходов, но и одновременное срабатывание мультимножеств переходов – шагов.

Таким образом задача верификации протоколов может быть сведена к задаче проверки двух помеченных сетей Петри на шаговую бисимуляционную эквивалентность. Последующие разделы работы посвящены решению этой задачи.

3 Сети Петри

Приведем некоторые понятия и определения из теории сетей Петри необходимые для дальнейшего изложения. Определения из теории множеств используемые в данной работе можно найти в приложении А.

Определение 1 *Сетью называется набор* $N = \langle S, T, F \rangle$*, где*

1. $S = \{s_1, ..., s_n\}$ – *множество мест;*
2. $T = \{t_1, ..., t_m\}$ – *множество переходов таких, что* $S \cap T = \emptyset$*;*
3. $F \subseteq \mu S \times T \times \mu S$ – *отношение инцидентности такое, что*
   (a) $\forall \langle Q'_1, t_1, Q''_1\rangle, \langle Q'_2, t_2, Q''_2\rangle \in F : \langle Q'_1, t_1, Q''_1\rangle \neq \langle Q'_2, t_2, Q''_2\rangle \Rightarrow t_1 \neq t_2$*;*
   (b) $\{t \mid \langle Q', t, Q''\rangle \in F\} = T$

Нотация 2 *Пусть задана сеть* $N = \langle S, T, F\rangle$, $s \in S$, $t \in T$, $S' \subseteq S$, $T' \subseteq T$.

1. *Если для некоторого перехода* t *имеем* $\langle Q', t, Q''\rangle \in F$, *то будем обозначать* ${}^\bullet t = Q'$, $t^\bullet = Q''$*;*
2. ${}^\bullet s = \{(t, n) \mid (s, n) \in t^\bullet\}$ *и* $s^\bullet = \{(t, n) \mid (s, n) \in {}^\bullet t\}$*;*
3. ${}^\bullet(S') = \sum_{s\in S'} {}^\bullet s$, $(S')^\bullet = \sum_{s\in S'} s^\bullet$, ${}^\bullet(S')^\bullet = {}^\bullet(S') + (S')^\bullet$*;*
4. ${}^\bullet(T') = \sum_{t\in T'} {}^\bullet t$, $(T')^\bullet = \sum_{t\in T'} t^\bullet$, ${}^\bullet(T')^\bullet = {}^\bullet(T') + (T')^\bullet$*;*

Расширим функции ${}^\bullet(\cdot)$ и $(\cdot)^\bullet$ на мультимножества переходов. Пусть $\Theta \in \mu T$ есть мультимножество переходов такое, что $\Theta = n_1 t_1 + n_2 t_2 + ... + n_k t_k$. Тогда положим ${}^\bullet\Theta = n_1 {}^\bullet t_1 + n_2 {}^\bullet t_2 + ... + n_k {}^\bullet t_k$ $\Theta^\bullet = n_1 t_1^\bullet + n_2 t_2^\bullet + ... + n_k t_k^\bullet$. Из определения легко заметить, что приведенные функции обладают свойством аддитивности: ${}^\bullet(\Theta_1 + \Theta_2) = {}^\bullet\Theta_1 + {}^\bullet\Theta_2$, $(\Theta_1 + \Theta_2)^\bullet = \Theta_1^\bullet + \Theta_2^\bullet$

Само по себе понятие сети имеет статическую природу. Для задания динамических характеристик используется понятие *маркировки* сети $M \in \mu S$, т.е. функции $M : S \to \mathcal{N}_0$, сопоставляющей каждому месту целое число.

Определение 3 *Маркированной сетью (или сетью Петри) называется набор* $\Sigma = \langle S, T, F, M_0\rangle$, *где* $\langle S, T, F\rangle$ – *сеть, а* $M_0 \in \mu S$ – *ее начальная маркировка.*

Определим два способа функционирования маркированных сетей, основанные на срабатывании отдельных переходов и мультимножеств переходов.

Определение 4 *Пусть* $\Sigma = \langle S, T, F, M_0\rangle$ *маркированная сеть.*

1. *Переход $t \in T$ считается возбужденным при маркировке $M \in \mu S$, если $M \geq {}^\bullet t$;*

2. *Переход t, возбужденный при маркировке M, может сработать, приведя к новой маркировке M', которая вычисляется по правилу: $M' = M - {}^\bullet t + t^\bullet$. Срабатывание перехода обозначается как $M[t\rangle M'$.*

Иными словами переход считается возбужденным при некоторой маркировке, если в каждом его входном месте имеется количество меток не менее кратности соответствующих дуг. Возбужденный переход может сработать, причем при срабатывании из каждого его входных места изымается, а в каждое входное добавляется некоторое количество меток, равное кратности соответствующих дуг. Срабатывание перехода считается неделимой мгновенной операцией. Если одновременно возбуждено несколько переходов, сработать может любой из них.

Это определение является классическим [26] и не позволяет рассматривать одновременное срабатывание параллельных переходов. Более общее правило базируется на срабатывании мультимножеств переходов.

Определение 5 *Пусть $\Sigma = \langle S, T, F, M_0\rangle$ маркированная сеть.*

1. *Мультимножество переходов, называемое шагом, $\Theta \in \mu T$ считается возбужденным при маркировке $M \in \mu S$, если $M \geq {}^\bullet\Theta$;*

2. *Шаг Θ, возбужденный при маркировке M, может сработать, приведя к новой маркировке $M' = M - {}^\bullet\Theta + \Theta^\bullet$. Это срабатывание записывается как $M[\Theta\rangle M'$.*

Срабатывание шага является также мгновенным неделимым действием. Если возбуждено несколько шагов одновременно, то сработать может произвольный. Ясно, что предыдущее правило является частным случаем настоящего, где шаг состоит из единственного перехода. Поэтому следующие обозначения будут даны для второго правила, полагая, что их перепись в терминах первого не составит труда.

Если $\Phi = \Theta_1\Theta_2...\Theta_k \in (\mu T)^*$, тогда $M[\Phi\rangle M'$ означает, что существуют маркировки $M_1, M_1, ..., M_{k-1} \in \mu S$ такие, что $M[\Theta_1\rangle M_1[\Theta_2\rangle ...M_{k-1}[\Theta_k\rangle M'$. Запись $M[\rangle M'$ означает, что $\exists\Phi \in (\mu T)^* \ : \ M[\Phi\rangle M'$, а запись $M[\Phi\rangle$ означает, что $\exists M' \in \mu S \ : \ M[\Phi\rangle M'$. *Множество достижимых маркировок* из маркировки M обозначается как $[M\rangle \stackrel{def}{=} \{M' \mid M[\rangle M'\}$.

Далее будем говорить, что правило 1 задает *"интерливинговую"* семантику, а правило 2 — *"шаговую"* семантику сетей Петри.

Пусть Vis есть некоторое множество видимых символов и $\tau \notin Vis$ есть выделенный невидимый символ. Обозначим $Act = Vis \cup \{\tau\}$

Определение 6 *Пометкой сети Петри* $\Sigma = \langle S, T, F, M_0 \rangle$ *называется набор* $\lambda = \langle Vis, \sigma \rangle$, *где* $\sigma : T \to Act$ *есть пометочная функция.*

Нотация 7 *Пусть дана маркированная сеть* $\Sigma = \langle S, T, F, M_0 \rangle$, $\lambda = \langle Vis, \sigma \rangle$ *– её пометка,* $V \in Act^*$, $W \in (Act \backslash \{\tau\})^*$. *Запись* $M(V\rangle^s_\lambda M'$ *означает, что* $\exists \Phi \in (\mu^+(T))^* \ : \ M[\Phi\rangle M'$ *и* $\sigma_\lambda(\Phi) = V$; *Запись* $M(W\rangle\rangle^s_\lambda M'$ *будет означать, что* $\exists \Phi \in (\mu^+(T))^* \ : \ M[\Phi\rangle M'$ *и* $\sigma^+_\lambda(\Phi) = W$; W *будем называть видимостью последовательности* Φ.

Предыдущие определения сделаны для общего случая функционирования сети с помощью шаговой семантики. Верхний индекс в выражениях $M(V\rangle^s_\lambda M'$ и $M(W\rangle\rangle^s_\lambda M'$ подчеркивает этот факт. Случай интерливинговой семантики является частным случаем, где каждый шаг состоит из одного перехода. В этом случае будем писать: $M(V\rangle^i_\lambda M'$ и $M(W\rangle\rangle^i_\lambda M'$.

Определение 8 *Две маркированные сети* $\Sigma_1 = \langle S_1, T_1, F_1, M_{01} \rangle$ *и* $\Sigma_2 = \langle S_2, T_2, F_2, M_{02} \rangle$ *с пометками соответственно* α *и* β *называются интерливинго бисимуляционно (би–) эквивалентными, записывается как* $\Sigma_1 \ {}_\alpha{\approx}^i_\beta \ \Sigma_2$, *если и только если существует отношение бисимуляции* $\Re \subseteq [M_{01}\rangle \times [M_{02}\rangle$ *такое, что*

1. $(M_{01}, M_{02}) \in \Re$;
2. *если* $(M_1, M_2) \in \Re$, *то*
   - *(a)* $M_1(W\rangle^i_\alpha M'_1 \Longrightarrow \exists M'_2 \ : \ M_2(W\rangle^i_\beta M'_2$ *и* $(M'_1, M'_2) \in \Re$;
   - *(b)* $M_2(W\rangle^i_\beta M'_2 \Longrightarrow \exists M'_1 \ : \ M_1(W\rangle^i_\alpha M'_1$ *и* $(M'_1, M'_2) \in \Re$;

Шаговая би–эквивалентность сетей Σ_1 *и* Σ_1 *получается заменой индекса* i *на* s *в вышестоящем определении и обозначается как* $\Sigma_1 \ {}_\alpha{\approx}^s_\beta \ \Sigma_2$.

Две маркировки M_1 и M_2 такие, что $(M_1, M_2) \in \Re$ будут называться эквивалентными. Неформально говоря, две сети би–эквивалентны, если, во–первых, их начальные маркировки эквивалентны. Во–вторых, если из одной из двух эквивалентных маркировок, скажем M_1, существует путь с видимостью W в маркировку M'_1, то из M_2 существует путь с видимостью W в некоторую маркировку M'_2, которая эквивалентна M'_1. И наоборот. Следует отметить, что в этом определении учитываются только видимые действия. В литературе такое определение называется слабой би–эквивалентностью, в отличии от сильной би–эквивалентности, учитывающей все переходы.

4 Верификация протоколов

Как было показано ранее, для обеспечения верификации необходимо наличие алгоритма проверки двух сетей Петри на эквивалентность. Теория сетей Петри и алгебраические теории параллельных процессов содержат достаточное количество результатов, позволяющих обеспечить решение этой задачи. Действительно, известен ряд алгоритмов проверки систем переходов[2] на бисимуляционную эквивалентность (см., например, [9, 14, 20, 24]) и имеющих полиномиальную сложность от размеров системы (количество состояний и переходов). С другой стороны, системы переходов для сетей Петри представляют не что иное как хорошо известный граф достижимых маркировок, для построение которого так же существует ряд алгоритмов (см., например, [2]). Заметим, однако, что проверяемые сети должны быть ограниченными, что обеспечивает конечность графа достижимых маркировок. В общем же случае проблема проверки двух сетей Петри на бисимуляционную эквивалентность является не разрешимой [17]. Таким образом, проверка двух процедур на эквивалентность будет заключаться в построении графов достижимых маркировок для соответствующих сетей Петри и проверки их на бисимуляционную эквивалентность.

Отметим ряд особенностей, которые заставляют вносить некоторые модификации в упомянутые методы. Известные методы построения графа достижимости и проверки их на бисимуляционную эквивалентность разработаны в интерливинговой семантике. Настоящий же подход требует использования шаговой семантики.

4.1 Проверка помеченных сетей Петри на бисимуляционную эквивалентность

Представленные в литературе существующие алгоритмы проверки параллельных процессов на бисимуляционную эквивалентность приводятся в терминах т.н. помеченных систем переходов. Для того, что бы ими прямо воспользоваться для сравнения сетей Петри, необходимо установить соответствие между двумя этими системами понятий. Приведем далее некоторые определения и понятия из области систем переходов, следуя терминологии, используемой в [21].

Графом переходов называется набор $TG = \langle Q, q_0, Act, \rightarrow \rangle$, где Q — множество состояний (узлов графа) , $q_0 \in Q$ — начальное состояние , Act — множество действий , $\rightarrow \subseteq Q \times Act \times Q$ — множество переходов

[2]То есть систем, состоящих из состояний и помеченных переходов между ними.

(помеченных дуг графа).

Вместо записи $(q_1, a, q_2) \in \rightarrow$ часто удобнее писать $q_1 \stackrel{a}{\rightarrow} q_2$, что соответствует переходу из состояния q_1 в состояние q_2 с выполнением действия a. Состояние q' называется достижимым из q, если либо $q = q'$, либо существует последовательность $q \stackrel{a_1}{\rightarrow} q_1 \stackrel{a_2}{\rightarrow} q_2 \stackrel{a_3}{\rightarrow} ... \stackrel{a_n}{\rightarrow} q_n \stackrel{a_{n+1}}{\longrightarrow} q'$. Далее будет предполагаться, что все состояния из Q достижимы из q_0.

Граф переходов, в котором начальное состояние q_0 не определено называется *системой переходов* $TS = \langle Q, Act, \rightarrow \rangle$. Отношение $\Re \subseteq Q \times Q$ называется отношением строгой бисимуляции для системы переходов $TS = \langle Q, Act, \rightarrow \rangle$ если оно симметрично и если для любых $(q_1, q_2) \in \Re$ выполняется следующее условие: если $q_1 \stackrel{a}{\rightarrow} q_1'$, то существует q_2' такое, что $q_2 \stackrel{a}{\rightarrow} q_2'$ и $(q_1', q_2') \in \Re$.

Два графа переходов $TG_1 = \langle Q_1, q_{01}, Act, \rightarrow_1 \rangle$ и $TG_2 = \langle Q_2, q_{02}, Act, \rightarrow_2 \rangle$ называются строго бисимуляционно эквивалентными, записывается как $TG_1 \sim TG_2$, если существует отношение бисимуляции $\Re$ для системы переходов $TS = \langle Q_1 \uplus Q_2, Act, \rightarrow_1 \uplus \rightarrow_2 \rangle$ такое, что $(q_{01}, q_{02}) \in \Re$. Иными словами, для того, чтобы сравнить два графа переходов необходимо объединить их в одну систему переходов[3], построить для последней отношение строгой бисимуляции $\Re$ и проверить — принадлежит ли к нему пара начальных состояний (q_{01}, q_{02}).

Сам алгоритм построения строгого отношения бисимуляции [20] заключается в пошаговом разбиении множества состояний Q на непересекающиеся подмножества (блоки) определенным образом так, что в конце каждое такое подмножество образует класс эквивалентности состояний. В абстрактной форме этот алгоритм приведен в Приложении В, а в наиболее разработанной форме, пригодной для непосредственной реализации, его можно найти в [12].

Для верификации протоколов, однако, необходима более слабая эквивалентность, не учитывающая невидимые τ-события, которая обычно называется *слабой бисимуляцией* или *наблюдательной эквивалентностью* [23]. Пусть отношение $\Rightarrow$ есть транзитивное и рефлексивное замыкание отношения $\stackrel{\tau}{\rightarrow}$, $\tau \in Act$. Отношение $\stackrel{a}{\Rightarrow}$ определяется как композиция отношений $\Rightarrow$, $\stackrel{a}{\rightarrow}$ и $\Rightarrow$. Тогда отношение $\Re \subseteq Q \times Q$ называется слабым отношением бисимуляции для системы переходов $TS = \langle Q_1 \uplus Q_2, Act, \rightarrow_1 \uplus \rightarrow_2 \rangle$, если оно симметрично и если для любых $(q_1, q_2) \in \Re$ выполняется следующее условие: если $q_1 \stackrel{a}{\Longrightarrow} q_1'$, то существует q_2' такое, что $q_1 \stackrel{a}{\Longrightarrow} q_2'$ и $(q_1', q_2') \in \Re$.

[3]Как обычно, здесь используется раздельное объединение, обозначаемое как $\uplus$, которое предполагает непересечение множеств Q_1 и Q_2. Если же все же каким то образом они имеют общие элементы, то необходимо просто перекодировать элементы одного из множеств, заменив один из графов другим изоморфным экземпляром.

Два графа переходов $TG_1 = \langle Q_1, q_{01}, Act, \rightarrow_1 \rangle$ и $TG_2 = \langle Q_2, q_{02}, Act, \rightarrow_2 \rangle$ называются слабо бисимуляционно эквивалентными, записывается как $TG_1 \approx TG_2$, если существует отношение бисимуляции $\Re$ для системы переходов $TS = \langle Q_1 \uplus Q_2, Act, \rightarrow_1 \uplus \rightarrow_2 \rangle$ такое, что $(q_1, q_2) \in \Re$. Далее будет использоваться в основном слабое отношение бисимуляционной эквивалентности, которое для краткости будет называться просто эквивалентностью.

Для вычисления отношения слабой бисимуляции система переходов модифицируется процедурой τ-насыщения, которая заключается в следующих действиях:

1. Добавление каждому состоянию системы τ-петли, т.е. перехода $q \xrightarrow{\tau} q$ для каждого $q \in Q$;
2. Соединение начала и конца каждого τ-последовательности τ-переходом;
3. Добавление a-перехода, соединяющего начало и конец каждой последовательности из τ-переходов, где встречается один a-переход.

После τ-насыщения к полученному графу состояний применяется алгоритм построения строгого отношения эквивалентности, который дает слабое отношение эквивалентности для исходной системы.

Рассмотрим как приведенные результаты могут быть использованы для анализа помеченных сетей Петри на бисимуляционную эквивалентность. Пусть $\Sigma = \langle S, T, F, M_0 \rangle$ есть ограниченная сеть Петри с пометкой $\alpha = \langle Alph, \sigma \rangle$. Построим для нее граф переходов $TG(\Sigma, \alpha) = \langle Q, q_0, Act, \rightarrow \rangle$ следующим образом:

1. $Q = [M_0\rangle$;
2. $q_0 = M_0$;
3. $Act = \mu(Alph_\alpha) \cup \{\tau\}$;
4. $(M_1, A, M_2) \in \rightarrow \iff M_1(A\rangle_\alpha M_2$, где $A \in Act$.

Иными словами, множество состояний построенного графа есть все множество достижимых маркировок сети, причем начальное состояние есть начальная маркировка. Каждое действие из Act представляет собой либо непустое мультимножество, определенное на алфавите имен $Alph$ пометки, либо невидимый символ τ. Между двумя состояниями - маркировками M_1 и M_2 в графе строится переход с именем A, если в сети Петри существует переход $M_1[\Theta\rangle M_2$ с шагом $\Theta \in \mu T$ таким, что $\sigma_\alpha(\Theta) = A$. Ограниченность сети Петри гарантирует конечность множества состояний и всего графа переходов в целом.

Теорема 9 *Две сети Петри Σ_1 с пометкой α и Σ_2 с пометкой β шагово бисимуляционно эквивалентны тогда и только тогда, когда их графы переходов слабо бисимуляционно эквивалентны, т.е.*

$$\Sigma_1 \ {}_{\alpha}{\approx}^{s}_{\beta} \ \Sigma_2 \iff TG(\Sigma_1, \alpha) \approx TG(\Sigma_2, \beta)$$

Док-во: Пусть – $TG_1 = TG(\Sigma_1, \alpha) = \langle Q_1, q_{01}, Act, \rightarrow_1 \rangle$ и $TG_2 = TG(\Sigma_2, \beta) = \langle Q_2, q_{02}, Act, \rightarrow_2 \rangle$ – графы переходов сетей, а $TS = \langle Q, Act, \rightarrow \rangle$ – их система переходов.

($\Rightarrow$) Из эквивалентности сетей следует существование отношения $\Re_\Sigma \subseteq [M_{01}\rangle \times [M_{02}\rangle$, удовлетворяющего условиям (1) и (2) определения 8. Покажем, что $\Re_{TS} = (\Re_\Sigma)^{TR}$ есть отношение слабой бисимуляции для системы переходов TS. Здесь запись $(\cdot)^{TR}$ обозначает транзитивное и рефлексивное замыкание. Пусть $(q_1, q_2) \in \Re_{TS}$. Рассмотрим тори возможных случая.

Случай 1: $q_1 \in [M_{01}\rangle, q_2 \in [M_{02}\rangle$, т.е. $(q_1, q_2) \in \Re_\Sigma$. Пусть $q_1 \stackrel{A}{\Longrightarrow} q_1'$, где $A \in Act$. Тогда, согласно процедурам построения графов переходов и систем переходов, в одной из сетей, скажем для определенности в сети Σ_1, существует последовательность переходов $M_1(A\rangle_\alpha M_1'$, где $M_1 = q_1$, $M_1' = q_1'$. Тогда согласно условию 2(а) определения 8 и в сети Σ_2 существует последовательность $M_2(A\rangle_\beta M_2'$, $(M_1', M_2') \in \Re_\Sigma$, где $M_1 = q_2$. Положим $q_2' = M_2'$, тогда эту последовательность можно переписать как $q_2 \stackrel{A}{\Longrightarrow} q_2'$ и $(q_1', q_2') \in \Re_\Sigma$. Из последнего следует, что $(q_1', q_2') \in \Re_{TS}$, т.к. очевидно, что $\Re_\Sigma \subseteq \Re_{TS}$.

Случай 2: $q_1.q_2 \in [M_{01}\rangle$. Пусть $q_1 \stackrel{A}{\Longrightarrow} q_1'$ и $q_1 = M_1$, $q_2 = M_2$. Положим для определенности, что эта последовательность соответствует последовательности переходов $M_1(A\rangle_\alpha M_1'$ в сети Σ_1. Тогда в сети Σ_2 существует хотя бы одна маркировка M_3 такая, что $(M_1, M_3) \in \Re_\Sigma$. Кроме того существует и последовательность $M_3(A\rangle_\beta M_3'$ с $(M_1', M_3') \in \Re_\Sigma$. Обозначим $M_3 = q_3$ и $M_3' = q_3'$. Далее, т.к. $(q_1, q_2) \in \Re_{TS}$ и $(q_1, q_3) \in \Re_{TS}$, то $(q_2, q_3) \in \Re_{TS}$. А из этого следует, что и в сети Σ_1 существует последовательность $M_2(A\rangle_\alpha M_2'$ такая, что $(M_2', M_3') \in \Re_\Sigma \subseteq \Re_{TS}$. То есть из $q_1 \stackrel{A}{\Longrightarrow} q_1'$ следует, что $\exists q_2' : q_2 \stackrel{A}{\Longrightarrow} q_2'$, $(q_1', q_2') \in \Re_{TS}$.

Случай 3: $q_1.q_2 \in [M_{02}\rangle$. Этот случай симметричен случаю 2 и доказывается совершенно аналогично.

($\Leftarrow$) Из эквивалентности TG_1 и TG_2 следует, что существует отношение слабой бисимуляции $\Re_{TS} \subseteq \Big([M_{01}\rangle \cup [M_{02}\rangle\Big) \times \Big([M_{01}\rangle \cup [M_{02}\rangle\Big)$. Покажем, что отношение

$$\Re_\Sigma = \Re_{TS} \cap [M_{01}\rangle \times [M_{02}\rangle$$

есть отношение бисимуляции для сетей Σ_1 и Σ_2 с пометками α и β. Очевидно, что $(M_{01}, M_{02}) \in \Re_\Sigma$. Пусть $(M_1, M_2) \in \Re_\Sigma$ и $M_1(A\rangle_\alpha M_1'$. Эта последовательность в графе переходов соответствует последовательности $q_1 \stackrel{A}{\Longrightarrow} q_1'$, где $q_1 = M_1$, $q_1' = M_1'$. Тогда существует и последовательность $q_2 \stackrel{A}{\Longrightarrow} q_2$ с $(q_1', q_2') \in \Re_{TS}$. Так как $q_2 = M_2 \in [M_{02}\rangle$, то, очевидно, и $q_2' = M_2' \in [M_{02}\rangle$. То есть (M_1', M_2') содержится так же и в $\Re_\Sigma$, т.е. $(M_1', M_2') \in \Re_\Sigma$. ■

Этот результат позволяет использовать существующие алгоритмы анализа параллельных процессов на бисимуляционную эквивалентность, выраженную в терминах графов переходов, для помеченных сетей Петри. Заметим, что здесь используется шаговая семантика сетей Петри. Это обстоятельство может привести к построению из сетей Петри графов с большим количеством переходов. Действительно, если в сети Петри возбужден шаг $\Theta \in \mu^+(T)$, то так же возбуждены и могут сработать и все его подшаги $\Theta' < \Theta$, и для каждого такого подшага приходится строить переход в графе. С другой стороны, заметим, что если возбужденный шаг Θ имеет 'невидимую' часть, т.е. $\Theta = \Theta_1 + \Theta_2$ с $\sigma(\Theta_2) = \tau$, то такой переход в каком то смысле будет избыточным. То есть, если не порождать переход Θ, а породить только его 'видимую' часть Θ_2, то это не должно привести к изменению внешнего проявления сети, т.е. эта сеть будет продолжать быть эквивалентной / неэквивалентной другим сетям. Интуитивно, этот факт следует из того, что если шаг $\Theta = \Theta_1 + \Theta_2$ возбужден, то так же возбужден и шаг Θ_1 и, более того, после срабатывания Θ_1 оставшийся подшаг Θ_2 продолжает быть возбужденным.

Предложение 10 *Если в сети Σ может сработать шаг $M_1[\Theta\rangle M_2$ и $\Theta = \Theta_1 + \Theta_2$, то $\exists M' : M_1[\Theta_1\rangle M'$ и $M'[\Theta_2\rangle M_2$.*

Док-во: $M_1[\Theta\rangle M_2$ означает, что ${}^\bullet\Theta \le M_1$ и $M_2 = M_1 - {}^\bullet\Theta + \Theta^\bullet$. Так как $\Theta = \Theta_1 + \Theta_2$, то очевидно, что ${}^\bullet\Theta_1 \le M_1$ и, следовательно, шаг Θ_1 возбужден. При его срабатывании сеть переходит в некоторую маркировку M' так, что $M_1[\Theta_1\rangle M'$ и $M' = M_1 - {}^\bullet\Theta_1 + \Theta_1^\bullet$. Из аддитивности функции ${}^\bullet(\cdot)$ следует, что ${}^\bullet\Theta = {}^\bullet\Theta_1 + {}^\bullet\Theta_2$. Далее, $M_1 \ge {}^\bullet\Theta \Rightarrow M_1 \ge {}^\bullet\Theta_1 + {}^\bullet\Theta_2 \Rightarrow M_1 - {}^\bullet\Theta_1 \ge {}^\bullet\Theta_2 \Rightarrow M_1 - {}^\bullet\Theta_1 + \Theta_2^\bullet \ge {}^\bullet\Theta_2 \Rightarrow M' \ge {}^\bullet\Theta_2$. То есть Θ_2 возбужден при новой маркировке M' и может сработать. То есть существует $M'' : M'[\Theta_2\rangle M''$. Тогда $M'' = M' - {}^\bullet\Theta_2 + \Theta_2^\bullet = M_1 - {}^\bullet\Theta_1 + \Theta_1^\bullet - {}^\bullet\Theta_2 + \Theta_2^\bullet = M_1 - {}^\bullet(\Theta_1 + \Theta_2) + (\Theta_1 + \Theta_2)^\bullet = M_1 - {}^\bullet\Theta + \Theta^\bullet = M_2$. ■

Таким образом можно 'дробить' шаги вплоть до одиночных переходов, о чем свидетельствует следующий факт:

Предложение 11 *Если в сети Σ может сработать шаг $M_1[\Theta\rangle M_2$, то тогда может сработать и последовательность переходов $v \in T^*$ такая, что $M_1[v\rangle M_2$, и количество вхождений каждого перехода $t \in T$ в строку v равно его кратности в шаге Θ, т.е. $\Theta(t)$.*

Док-во: Для шага Θ возьмем в качестве Θ_1 произвольный переход $t \in \Theta$, т.е. $\Theta = t + \Theta_2$. Далее то же самое сделаем с Θ_2 и т.д. ■

Приведем далее процедуру построения сокращенного графа переходов для сети Петри $\Sigma = \langle S, T, F, M_0\rangle$ с пометкой $\alpha = \langle Alph, \sigma\rangle$. Сокращенный граф переходов сети Σ есть граф $TG^*(\Sigma, \alpha) = \langle Q, Act, \rightarrow\rangle$ где

1. $Q = [M_0\rangle$;
2. $q_0 = M_0$;
3. $Act = \mu^+(Alph_\alpha) \cup \{\tau\}$;
4. $(M_1, A, M_2) \in \rightarrow \iff \Big(M_1[\Theta\rangle M_2, \sigma(\Theta) = A, \Theta \in \mu(T^{vis})\Big)$
   $\bigvee\Big(M_1[\Theta\rangle M_2, \Theta = \{t\}, \sigma(t) = \tau = A\Big)$, где $T^{vis} = \{t \in T \mid \sigma(t) \neq \tau\}$

Иными словами, в сокращенном графе переходов строятся только те переходы – шаги, которые составлены только из 'видимых' переходов. Из 'невидимых' переходов образуются только одиночные шаги. Множество состояний у сокращенного графа такое же как и у обыкновенного.

Теорема 12 *Две сети Петри Σ_1 с пометкой α и Σ_2 с пометкой β шагово бисимуляционно эквивалентны тогда и только тогда, когда их сокращенные графы переходов слабо бисимуляционно эквивалентны, т.е.*

$$\Sigma_1 \ {}_\alpha\!\approx^s_\beta \ \Sigma_2 \iff TG^*(\Sigma_1, \alpha) \approx TG^*(\Sigma_2, \beta)$$

Док-во: Для доказательства достаточно показать, что для любой сети Петри $\Sigma = \langle S, T, F, M_0\rangle$ с пометкой $\alpha = \langle Alph, \sigma\rangle$ справедливо следующее равенство

$$TG(\Sigma, \alpha) \approx TG^*(\Sigma, \alpha) \tag{1}$$

Обозначим $TG_1 = TG(\Sigma, \alpha)$, $TG_2 = TG^*(\Sigma, \alpha)$ и построим из них систему переходов TS. Пусть $\Re^a = B_1 \cup B_2 \cup ... \cup B_k$, $B_i \subseteq Q_1$ есть автобисимуляция[4] для графа переходов TG_1. Покажем, что отношение

$$\Re = \{B \times B \mid B \in \Re^a\}$$

[4]Здесь отношение автобисимуляции задано в виде объединения раздельных блоков – классов эквивалентности состояний .

есть отношение бисимуляции для равенства (1). Пусть $(q_1, q_2) \in \Re$ и $q_1 \stackrel{A}{\Longrightarrow} q_1'$. Последнее означает, что существует последовательность

$$q_1 \stackrel{\tau}{\to} q_1^1 \stackrel{\tau}{\to} q_1^2 ... \stackrel{\tau}{\to} q_1^k \stackrel{A}{\to} q_1^{1\prime} \stackrel{\tau}{\to} q_1^{2\prime} \stackrel{\tau}{\to} ... \stackrel{\tau}{\to} q_1^{l\prime} \stackrel{\tau}{\to} q_1{}', \qquad (2)$$

где $k, l \geq 0$. Переход $q_1^i \stackrel{\tau}{\to} q_1^{i+1}$ означает, что в сети сети существует соответствующий шаг $\Theta \in \mu T$, состоящий из невидимых переходов. Согласно следствию 11, в сети существует и последовательность, составленная из этих переходов, чему соответствует последовательность $q_1^i \stackrel{\tau}{\to}\stackrel{\tau}{\to} ... \stackrel{\tau}{\to} q_1^{i+1}$. Переход $q_1^k \stackrel{A}{\to} q_1^{1\prime}$ означает, что в сети существует шаг $\Theta \in \mu T$ такой, что $\sigma(\Theta) = A$. Представим $\Theta = \Theta_1 + \Theta_2$, где Θ_1 составлен только из видимых переходов, а Θ_2 – из невидимых. Тогда, согласно предложению 10 и следствию 11, существует и последовательность $q_1^k \stackrel{A}{\to} q_1^* \stackrel{\tau}{\to}\stackrel{\tau}{\to} ... \stackrel{\tau}{\to} q_1^{1\prime}$, где $q_1^k \stackrel{A}{\to} q_1^*$ есть переход в графе, соответствующий шагу Θ_1 в сети Петри. Таким образом для последовательности (2) в графе TG_1 существует и последовательность из одиночных τ-переходов и одного шага A:

$$q_1 \stackrel{\tau}{\to} ... \stackrel{\tau}{\to} q_1^k \stackrel{A}{\to} q_1^* \stackrel{\tau}{\to} ... \stackrel{\tau}{\to} q_1',$$

которая имеет ту же 'видимость'. Согласно процедуре построения сокращенного графа, в нем так же имеется эта последовательность.

Последовательность же $q_2 \stackrel{A}{\Longrightarrow} q_2'$ в TG_2 имеется и в графе TG_1, т.к. множество переходов графа TG_2 есть подмножество переходов графа TG_1. ■

4.2 Построение сокращенного графа достижимости

В предыдущем разделе было показано как помеченные сети Петри могут быть проверены на бисимуляционную эквивалентность, причем для этого использовался граф достижимых маркировок сети в шаговой семантике. В данном разделе будет показано как такой граф может быть построен. Основной сложностью здесь является то, что необходимо одновременно проверять – является ли сеть Петри ограниченной и, следовательно, конечный ли у нее граф достижимых маркировок.

Как известно (см., например, [26]), проблема проверки сети Петри на ограниченность разрешима, и соответствующий алгоритм заключается в построении дерева покрытия для сети Петри. Этот алгоритм может быть использован и для построения графа достижимых маркировок с одновременной проверкой его конечности. Далее приводится описание процедуры построения графа достижимых маркировок, которая состоит

из построения дерева достижимых маркировок и далее, в случае его конечности, трансформации его в граф.

Пусть $\Sigma = \langle S, T, F, M_0 \rangle$, $\alpha = \langle Alph_\alpha, \sigma_\alpha \rangle$ есть сеть Петри с пометкой. Дерево достижимых маркировок определим как помеченный граф $RT = \langle V, D, \nu_s, \nu_t \rangle$, где

1. $V = \{v_1, v_2, ...\}$ – конечное множество вершин графа;
2. $D \subseteq V \times V$ – конечное множество его ориентированных дуг;
3. $\nu_s : V \to [M_0\rangle$ – пометка вершин, сопоставляющая каждой вершине дерева достижимую маркировку сети;
4. $\nu_t : D \to \mu T$ – пометка дуг, сопоставляющая каждой дуге дерева мультимножество переходов – шаг[5].

На Рис.2 приводится этот алгоритм. Множество вершин дерева $V = \{v_1, v_2, ...\}$ строится с помощью множества целых неотрицательных чисел $\mathcal{N}_0 = \{0, 1, 2, ...\}$, для чего используется переменная nid, начальное значение которой устанавливается в ноль, что соответствует корневой вершине дерева. Далее при образовании новой вершины значение nid увеличивается на единицу и результат становится идентификатором этой вершины. В процессе построения дерева строится и функция ν_s, сопоставляющая каждой вершине соответствующую маркировку. В начальном состоянии строится корень дерева $V = \{0\}$, связанный с начальной маркировкой $\nu_s(0) = M_0$. Множество $ToProcess$ содержит граничные вершины, которые необходимо будет далее обработать. В начальном состоянии $ToProcess$ содержит одну вершину – корень дерева.

Далее алгоритм выбирает граничную вершину v из множества $ToProcess$, для которой строится множество всех выводов – пар (Θ, M), где Θ есть возбужденный шаг в маркировке $\nu_s(v)$, а M – соответствующая достижимая маркировка. Множество выводов строится с помощью функции $Next$. Далее для каждой пары (Θ, M) проверяется наличие в дереве другой вершины v' с такой же маркировкой. Если она есть, то образуется новая вершина с идентификатором $v_n = nid + 1$ и с соответствующей маркировкой $\nu_s(v_n) = M$. Также образуется новая дуга (v, v_n), соответствующая шагу $\nu_t(v, v_n) = \Theta$[6].

[5]В качестве множества вершин дерева V нельзя прямо использовать множество достижимых маркировок $[M_0\rangle$, т.к. в дереве могут быть различные вершины, соответствующие одной и той же маркировке (т.н. дублирующие маркировки).

[6]Заметим, что в этом случае новая вершина v в множество $ToProcess$ не заносится, т.к. она является дублирующей и, следовательно, в дальнейшей обработки не нуждается.

```
function Next(v) = {(Θ, M) ∈ μT × μS | ν_s(v)[Θ⟩M}

algorithm RTree(Σ, α)
begin
    nid := 0; ν_s = {(nid, M_0)};
    D := ∅; ν_t = ∅;
    ToProcess := {nid};
    while ToProcess ≠ ∅ do
        begin for each v ∈ ToProcess do
            begin
            for each (Θ, M) ∈ Next(v) do
                begin
                if ∃v' ∈ V : ν_s(v') = M
                then
                    nid := nid + 1;
                    V := V ∪ {nid}; ν_s := nu_s ∪ {nid};
                    D := D ∪ {(v, nid)}; ν_t := nu_t ∪ {((v, nid), Θ)};
                    go to end_proc;
                if на пути от v к корню дерева ∃v' : ν_s(v') < M
                then print "Сеть Петри не ограниченна";
                    go to exit;
                else
                    nid := nid + 1;
                    ToProcess := ToProcess ∪ {nid};
                    V := V ∪ {nid}; ν_s := nu_s ∪ {nid};
                    D := D ∪ {(v, nid)}; ν_t := nu_t ∪ {((v, nid), Θ)};
                end
        end_proc:end
        end
end
```

Рис. 2: Алгоритм построения дерева достижимых маркировок

Если в дереве нет дублирующих вершин, то тогда проверяется – существует ли на пути от вершины v до корня дерева другая вершина v' с маркировкой строго меньшей чем M. Если да, то такая сеть является неограниченной и алгоритм заканчивает свою работу, выдав соответствующее сообщение.

Если же нет ни дублирующих ни меньших маркировок, то для пары (Θ, M) строятся новая вершина $v_n = nid+1$ с маркировкой $\nu_s(v_n) = M$ и дуга (v, v_n) с шагом Θ. Эта вершина добавляется в множество $ToProcess$, как вершина, нуждающаяся в дальнейшей обработке.

После обработки вершины v она удаляется из $ToProcess$.

Заметим, что если для некоторой вершины v множество выводов пустое: $Next(v) = \emptyset$, то соответствующая маркировка тупиковая, и эта вершина просто удаляется из $ToProcess$.

Согласно известным результатам о конечности дерева покрытия сетей Петри (см., например, [26]), этот алгоритм конечен и строит конечное дерево достижимых состояний в случае ограниченности сети Петри. В противном случае от прекращает свою работу с соответствующим сообщением.

Заметим, что здесь этот алгоритм приведен в довольно абстрактной форме и для его конкретной реализации необходимо дополнительно определить стратегию перебора вершин, например “в глубину” или “в ширину” [2]. Конкретная реализация алгоритма приводится далее.

Из дерева достижимости $RT = \langle V, D, \nu_s, \nu_t \rangle$ нетрудно получить граф достижимых маркировок сети (граф переходов): $TG(\Sigma, \alpha) = \langle Q, q_0, Act, \rightarrow \rangle$, где

1. $Q = \{\nu_s(v) \mid v \in V\}$;
2. $q_0 = \nu_s(0)$;
3. $Act = \mu Alph \ \cup \ \{\tau\}$;
4. $\rightarrow = \Big\{\langle \nu_s(v_1), \sigma_\alpha(\nu_t(v_1, v_2)), \nu_s(v_2)\rangle \mid (v_1, v_2) \in D\Big\}$,

который далее может быть непосредственно использован для проверки сети на бисимуляционную эквивалентность.

Заметим, что в приведенном алгоритме строится полный граф достижимости, т.е. граф со всеми возможными шагами. Для построения сокращенного графа этот алгоритм также можно использовать, модифицировав функцию $Next$ следующим образом:

$$\begin{array}{lll} \texttt{function}\ Next_1(v) = & & \{(\Theta, M) \in \mu T \times \mu S \mid \nu_s(v)[\Theta\rangle M, \Theta \in \mu T^{vis}\} \\ & \cup & \{(\{t\}, M) \in \mu T \times \mu S \mid \nu_s(v)[t\rangle M, \sigma_\alpha(t) = \tau\} \end{array}$$

Действительно, функция $Next_1$ строит выводы с возбужденными шагами только для видимых переходов из T^{vis}. Для невидимых переходов строятся выводы только из одиночных переходов.

4.3 Распределенная бисимуляционная эквивалентность

При более подробном рассмотрении приведенных алгоритмов для верификации распределенных систем оказывается, что ранее введенная шаговая бисимуляционная эквивалентность даже с сокращенным графом достижимости является слишком сильной. В некоторых ситуациях можно расширить множество переходов для которых возможность одновременного срабатывания не повлияет на эквивалентность.

С точки зрения эквивалентности спецификации сервиса и его реализации необходимо эквивалентное поведение реализации при ее композиции с вышележащем уровнем. В этом случае важно учитывать одновременное срабатывания лишь тех переходов которые при последующих композициях могут быть объединены в один. В случае протоколов можно выделить *распределенные* переходы соответствующие логической распределенности задачи. Эта распределенность означает, что никогда при последующих композициях эти переходы не сольются в один и, следовательно, возможность выполнения шага состоящего из распределенных переходов непринципиальна. Иными словами, необходима такая шаговая бисимуляция, которая бы учитывала шаги только тогда, когда они имеют локальный характер и могут в перспективе быть объединены в другие переходы синхронизации. Далее приводится формальное определение понятия распределенной шаговой эквивалентности и соответствующий алгоритм анализа.

Пусть алфавит имен действий пометки $\alpha = \langle Alph, \sigma \rangle$ сети Σ разбивается на непересекающиеся подмножества имен, где каждое такое подмножество соответствует действиям, выполняемым в одном локальном месте. Иными словами задано разбиение:

$$\mathcal{P} = \{Alph^1, Alph^2, ..., Alph^n\}, \bigcup_{i=1}^{n} Alph^i = Alph, Alph^i \cap Alph^j = \emptyset, i \neq j.$$

Пометочная функция $\sigma : t \to \mu(Alph) \cup \{\tau\}$ будет называться $\mathcal{P}$-*состоятельной,* если

$$\forall t \in T : \sigma(t) = \tau \vee (\quad \sigma(t) \lceil Alph^i = \sigma(t) \text{ для некоторого } i \in \{1, .., n\}$$
$$\wedge \quad \sigma(t) \lceil Alph^j = \emptyset \text{ для остальных } j \neq i)$$

Иными словами, имя перехода в $\mathcal{P}$-состоятельной пометочной функции должно быть либо τ-символом, либо состоять из имен только одного локального подмножества имен $Alph^i$.

$\mathcal{P}$–состоятельная пометочная функция индицирует и разбиение множества переходов: $T = T^1 \cup T^2 \cup ... \cup T^n$, где

$$T^i = \{t \in T \mid \sigma(t)\lceil Alph^i = \sigma(t)\} \text{ и } T^\tau = \{t \in T \mid \sigma(t) = \tau\}.$$

Сокращенный граф переходов с распределенными именами сети Петри Σ с пометкой α и разбиением $\mathcal{P}$ есть граф переходов $TG^{**}(\Sigma, \alpha) = \langle Q, q_0, Act, \rightarrow \rangle$

1. $Q = [M_0\rangle$;
2. $q_0 = M_0$;
3. $Act = \cup_i Alph^i \cup \{\tau\}$;
4. $(M_1, A, M_2) \in \rightarrow \Longleftrightarrow \bigvee_{i=1}^n (M_1[\Theta\rangle M_2, \sigma(\Theta) = A, \Theta \in \mu T^i)$
   $\bigvee (M_1[\{t\}\rangle M_2, t \in T^\tau)$.

Таким образом в сокращенном графе с распределенными именами строятся только те дуги, которым соответствуют шаги, составленные из видимых переходов, причем все эти переходы должны принадлежать одному локальному подмножеству разбиения. Две сети Петри Σ_1, Σ_2 с пометками α, β и разбиениями будут называться *распределенно бисимуляционно эквивалентными,* если их графы $TG^{**}(\Sigma_1, \alpha)$ и $TG^{**}(\Sigma_2, \beta)$ бисимуляционно эквивалентны.

Очевидно, что распределенная бисимуляционная эквивалентность слабее шаговой эквивалентности и сильнее интерливинговой эквивалентности сетей Петри.

Алгоритм анализа сетей Петри на распределенную бисимуляционную эквивалентность может быть получен из классического алгоритма (см.Рис.2) дальнейшей модификацией функции $Next$:

$$\begin{array}{lll} \texttt{function}\ Next_2(v) = & \bigcup_{i=1}^n & \{(\Theta, M) \in \mu T \times \mu S \mid \nu_s(v)[\Theta\rangle M, \Theta \in \mu T^i\} \\ & \bigcup & \{(\{t\}, M) \in \mu T \times \mu S \mid \nu_s(v)[t\rangle M, t \in T^\tau\} \end{array}$$

4.4 Реализация

Верификация сетей реализована в качестве подсистемы в рамках автоматизированной системы разработки протоколов PN^3-Tool [6]. Модуль

верификации как и вся система написана на языке C/C++ с использованием инструментальных средств MS Visual C++ 1.5 и предназначена на работу под управлением графической оболочки Microsoft Windows 3.1. Верификация происходит в два этапа:

- Построение дерева покрытия верифицируемых сетей используя шаговую семантику срабатывания переходов. Одновременно с генерацией дерева создаются ассоциаторы между дублирующими маркировками, что позволяет получить замыкание дерева в граф при незначительных дополнительных затратах памяти, сохранив при этом структуру дерева покрытия необходимую для других алгоритмов. На этапе построения происходит определение таких свойств сети как ограниченность и наличие тупиков. Ввиду того, что алгоритмы проверки сети на различные свойства, как правило, анализируют дерево покрытия, то предусмотрена возможность его сохранения для использования другими модулями программы.

- Проверка на бисимуляционной эквивалентности. Графы покрытия полученные на предыдущем этапе объединяютя и проверяются на эквивалентность. В программе реализован алгоритм бисимуляционной эквивалентности описанный в [14] позволяющий обойтись без замыкания графа τ - переходами.

В качестве примера предлагаемого подхода можно привести верификацию альтернативно-битового протокола.

Для создания сети представляющий альтернативно-битовый протокол был использован композициональный подход. Для каждого протокольного объекта – приемника, передатчика и среды передачи данных в базовом редакторе PN^3-Tool были созданы сети с соответствующими точками доступа. Далее, эти сети были объединены с помощью операции параллельной композиции с синхронизацией по соответствующим точкам доступа. Получившаяся сеть имела 20 переходов и 16 мест.

Время построения дерева достижимости около – 5 секунд, число вершин дерева покрытия – 129, из которых дублирующих вершин – 75, число достижимых маркировок – 54, глубина дерева достижимости – 15, время проверки на бисимуляционную эквивалентность около трех секунд. Для верификации использовался компьютер Pentium 120 Мгц, под управлением Windows NT 4.0.

5 Заключение

Итак в данной работе предложен подход к верификации протоколов, основанный на сравнении двух помеченных сетей Петри на шаговую бисимуляционную эквивалентность. Этот подход позволяет использовать существующие алгоритмы построения графа достижимых маркировок сетей Петри и проверки двух систем переходов на бисимуляционную эквивалентность с некоторыми модификациями. Подход программно реализован на в рамках автоматизированной системы. Однако, как показывает опыт применения к верификации протокола альтернативного бита, имеются определенные недостатки, устранение которых входит в ближайшие планы авторов. Во-первых, предполагается расширение подхода на сети Петри высокого уровня [19], что позволит охватить более сложные протоколы. Во-вторых, подлежит исследованию применение современных подходов к построению более компактных графов достижимости [18], что должно уменьшить сложность задачи верификации.

Список литературы

[1] *Анисимов Н.А.* Иерархическая композиция протоколов // Автоматика и вычислительная техника, 1990, N 1, с.3–10.

[2] *Ачасова С.М., Бандман О.Л.* Корректность параллельных вычислительных процессов. – Новосибирск: Наука, 1990.

[3] *Котов В.Е.* Сети Петри. – М.: Наука, 1984. – 160 с.

[4] *Ч.Хоар,* Взаимодействующие последовательные процессы, – М.: Мир, 1989.

[5] *N.A.Anisimov,* A Notion of Petri Net Entity: A Modular Approach to Design of Distributed Sysytems, Internal Report CN-92-03, Institute of Automation and Control Processes, Vladivostok, 1992. Submitted.

[6] *N.A. Anisimov, A.A. Kovalenko, P.A. Postupalski,* PN^3-Editor: Compositional Petri Net Editor for Protocol Specification, Proc. of the Third Int. Workshop on Modelling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS'95), 1995, Durham, NC, USA, pp. 325–328.

[7] *J.C.M. Baeten, W.P. Weijland,* Process Algebra. Cambridge Tracts in Theoretical Computer Science, 18, Cambridge University Press, 1990.

[8] *E. Best, R. Devillers, A. Kiehn, and L. Pomello,* Concurrent Bisimulations in Petri Nets. Acta Informatica, Vol. 28, 1991 pp.231–261.

[9] *T. Bolognesi, S. A. Smolka,* Fundamental Results for the Verification of Observational Equivalence: a Survey, in Protocol Specification, Testing, and Verification, VII: Proc.of the IFIP WG 6.1 7th International Symposium, eds. H. Rudin, C. H. West, North–Holland, 1987, pp.165–179.

[10] *W. Brauer, W. Reisig, G. Rozenberg (eds).* Petri Nets. Part I and II. Proc. of an Advanced Course, Bad Honnef. Lecture Notes in Computer Science, 254/255, Springer–Verlag, 1987

[11] *R. De Nicola,* Extensional Equivalences for Transition Systems, Acta Informatica, Vol. 24, 1987, pp.211–237.

[12] *J.-C.Fernandez,* An Implementation of an Efficient Algorithm for Bisimulation Equivalence, Science of Computer Programming, Vol.13, 1989/90.

[13] *R. J. Van Glabbeek,* The Linear Time — Branching Time Spectrum, in: CONCUR'90, Theories of Concurrency: Unification and Extension, ed. J.C.M.Baeten, J.W.Klop, Lecture Notes in Computer Science, 458, Springer–Verlag, 1990.

[14] *J.F.Groote, F.W. Vaandrager,* An Efficient Algorithm foe Branching Bisimulation and Stuttering Equivalence, In: Proc. 17th ICALP, ed. M.S.Paterson. Lecture Notes in Computer Science, 443, Springer–Verlag, 1990, pp.626–638.

[15] *M. Hillerström,* Verification of CCS-processes, M. Sc. Thesisis, Computer Science Department, Aalborg University, Denmark, 1987.

[16] ISO, IS 7498. Information Processing Systems – Open Systems Interconnection – Basic Reference model, 1983.

[17] *P.Jančar,* Decidability Questions for Bisimulation of Petri Nets and Some Related Problems, Report ECS-LFCS-93-261, University of Edinburg, 1993, 12 p.

[18] *R. Janicki, M. Koutny,* On Some Implementation of Optimal Simulations. Proceedings of Computer-Aided Verification'90, Lecture Notes in Computer Sciences, 254, Springer Verlag, 1991, pp.166–175.

[19] *K. Jensen, G. Rozenberg (Eds.)* Hogh–level Petri Nets. Theory and Application, Springer–Verlag, 1991, 724 p.

[20] *P.C.Kanellakis, S.C.Smolka,* CCS Expressions, Finite State Processes and Three Problems of Equivalence. In: Proc. of the Second ACM Symposium on Principles of Distributed Computing, 1983.

[21] *H.P.Korver,* The Current State og Bisimulation Tools, Report P9101, Programming Research Group, University of Amsterdam, 1991, 28 p.

[22] *K.G. Larsen,* Context-Dependent Bisimulation Between Processes, Ph.D Thesisis, University of Edinburg, 1986.

[23] *R.Milner,* A Calculus for Communication Systems. Lecture Notes in Computer Science, 92, Springer–Verlag, 1980, p.170.

[24] *R.Paige, R.E.Tarjan,* Three Partitioning Refinement Algorithms. SIAM Journal of Computing, Vol.16, No.6, 1987, pp.973–989.

[25] *D. M. R. Park,* Concurrency and Automata on Infinite Sequences, in: Procedeengs 5th GI Conference, ed.P. Deussen, Lecture Notes in Computer Science, 104, Springer–Verlag, 1981, pp.167–183.

[26] *J.L.Peterson.* Petri Net Theory and the Modelling of Systems, Prentice–Hall Inc., 1981

[27] *L. Pomello, G. Rosenberg, and C. Simone,* A Survey of Equivalence Notions for Net Based Systems. in: Advances in Petri Nets 1992, ed.G. Rozenberg, Lecture Notes in Computer Science, 609, Springer–Verlag, 1992.

[28] *C.Sunshine.* A Formal Techniques for Protocol Specification and Verification. Computer, Vol.12, 1979, pp.20–27.

A Определения

Приведем некоторые определения из теории множеств используемые в данной работе.

Пусть f некоторая функция, $f : A \to B$. Тогда определим два типа *проекций функции* f на $A' \subseteq A$ как следующие функции:
$f\lceil A' = \{(a,b) \in f \mid a \in A'\}$ и $f\lfloor A' = \{(a,b) \in f \mid a \notin A'\}$.

Мультимножества. Пусть $A = \{a_1, a_2, ..., a_n\}$ некоторое множество. Мультимножеством на множестве A называют функцию $\mu : A \to \mathcal{N}_0$, которая каждому элементу из A сопоставляет неотрицательное целое число. Мультимножество как обычную функцию можно задавать в виде множества пар $\{(a_1, n_1), (a_2, n_2), ..., (a_k, n_k)\}$. Однако, часто их удобнее записывать в виде формальной суммы: $\mu = n_1 a_1 + n_2 a_2 + ... + n_k a_k$ или $\mu = \sum_i n_i a_i$, где $n_i = \mu(a_i)$ – число экземпляров элемента a_i в мультимножестве (кратность). Далее, в зависимости от контекста, будут использоваться обе эти формы записи мультимножеств. Если $n_i = 0$, то этот член в формальной сумме будет опускаться.

Будем обозначать множество всех мультимножеств, определённых на множестве A как $\mu(A)$ или, когда это не приведет к недоразумениям, как μA. Определим также множество всех непустых мультимножеств как $\mu^+(A) \stackrel{def}{=} \mu(A) \setminus \{\mathbf{0}\}$.

B Алгоритм проверки систем на бисимуляционную эквивалентность

В данном приложении приводится алгоритм построения отношения строгой бисимуляции для систем переходов, впервые введенный в работе [20]. Здесь используется форма, представленная в [21].

```
function split(B, a, B') =
    {{s ∈ B | ∃s' ∈ B' : s -a-> s'}, {s ∈ B | ¬∃s' ∈ B' : s -a-> s'}} \ {∅};

algorithm bisim(Q, Act, →);
    begin
        P1 := {Q};
        P2 := ∅;
        while find(P1, a, B') do begin
            P2 := P1;
            P1 := ∅;
            foreach B ∈ P2 do P1 := P1 ∪ split(B, a, B');
        end
    end
```

Рис. 3: Алгоритм разбиения для построения отношения бисимуляции