

МАТЕМАТИЧЕСКОЕ МОДЕЛИРОВАНИЕ

УДК 519.21

А.С.Сорокин

ПРЕДСТАВЛЕНИЕ ДИНАМИЧНЫХ ПОТОКОВ МОДЕЛЕЙ МАССОВОГО ОБСЛУЖИВАНИЯ

Введение. За прошлое десятилетие было разработано большое количество моделей, с помощью которых появилась возможность определять количественно продолжительности пребывания систем в различных состояниях [1-7].

Предполагается, что случайная величина, распределенная по экспоненте, описывает пребывание в основной математической модели. Эта величина является непрерывным временем пребывания в марковском процессе.

Такие модели могут быть реализованы для любого динамического состояния системы, а переходные распределения вероятностей есть независимые неотрицательные случайные величины с невырожденными функциями распределения.

Отметим, что в более общем случае, когда математическая модель представляет собой обобщенный полумарковский процесс, моделирование проводится обычными средствами анализа.

Этот анализ показывает, что существует несколько способов задания полумарковского процесса. В любом случае появляется задача описания пространства состояний. Дискретный, основанный на описании процесса состояний с помощью алгебры, формализм моделирования, является неудачной попыткой, так как это приводит к чрезвычайно большим пространствам состояний. Создание численной реализации на основе линейной, очень громоздкой алгебры и отнимающее много времени моделирование потенциально приводит к заведомо неточным результатам.

Занятие задачей описания пространства состояний для непрерывного марковского процесса во времени (например [7-10]) потребовало большой изобретательности исследования.

Размерность этого процесса, который может быть описан, действительно продолжает увеличиваться, хотя и достаточно медленно.

В этой работе предлагается радикально отличающийся подход к изучению этой задачи, моделируя с алгеброй процесса такой как PEPA. Подход основан на двух дополнениях обычной перспективы.

- Во-первых, не стремимся вычислять распределение вероятности по всему пространству со-

стояний модели. Выберем более абстрактное представление в терминах переменных состояний, определяя количество типов очевидного поведения их в модели.

- Во-вторых, предполагаем, что эти переменные состояний подчинены непрерывному, а не дискретному изменению. Как только эти корректировки сделаны, то система подготовлена к эффективному решению в виде ряда обыкновенных дифференциальных уравнений (ОДУ). Результат решения этого ряда уравнений приводит к оценке переходного процесса, как предела меры по установленвшемуся состоянию.

В п. 1 содержится некоторая основная информация на языке PEPA и предлагается новое представление в форме числового вектора. В п. 3 изложены изменения к непрерывному анализу и иллюстрированы примером в п. 6. Подход к другим методам дан в п. 9.

1 Описание языка PEPA. Язык PEPA использовался, чтобы изучить характеристику широкого разнообразия систем [11-15]. Как и во всех алгебрах процесса, системы представлены в PEPA как композиция, состоящая из компонентов, которые предпринимают действия. В PEPA действия, как предполагается, имеют продолжительность, или задержку.

Таким образом, выражение $(\alpha, r)_P$ обозначает компонент, который может предпринять α действие, по норме r , чтобы выделить компонент P . Здесь $\alpha \in A$, где A множество типов действия и $P \in C$, где C - множество составляющих типов.

У языка PEPA есть небольшое множество комбинаторов, которые будут созданы как параллельное выполнение и взаимодействие простых последовательных компонентов.

В табл. 1 представлена структурная оперативная семантика языка PEPA.

Префикс: основной механизм для того, чтобы описать поведение системы PEPA. Модель должна выдавать компонент, определяющий первое действие, используя префиксный комбинатор, обозначенный при точке. Как было отмечено, что $(\alpha, r)_P$ выполняет действие α с нормой r , и это действие впоследствии ведет себя как P .

Выбор: компонент $P+Q$ представляет систему, которая ведет себя или как P или как Q .

Действия и P и Q допустимы. Первая ситуация, которая произойдет, выбирает один из них, а другому отказано. Система будет вести себя как производная, выделяющая выбранный компонент.

Таблица 1

Предфикс	$(\alpha, r).E \xrightarrow{(\alpha, r)} E$
Выбор	$E \xrightarrow{(\alpha, r)} E'$ $E + F \xrightarrow{(\alpha, r)} E'$ $F \xrightarrow{(\alpha, r)} F'$ $E + F \xrightarrow{(\alpha, r)} F'$
Кооперация	$E \xrightarrow{(\alpha, r)} E' (\alpha \notin L)$ $E \triangleright \triangleleft F \xrightarrow{L} E \triangleright \triangleleft F (\alpha \notin L)$ $F \xrightarrow{(\alpha, r)} F' (\alpha \notin L)$ $E \triangleright \triangleleft F \xrightarrow{L} E \triangleright \triangleleft F' (\alpha \notin L)$ $E \xrightarrow{(\alpha, r_1)} E' (\alpha \in L)$ $F \xrightarrow{(\alpha, r_2)} F' (\alpha \in L)$ $E \triangleright \triangleleft F \xrightarrow{L} E \triangleright \triangleleft F' (\alpha \in L)$ $R = \frac{r_1}{r_\alpha(E)} \frac{r_2}{r_\alpha(F)} \times \min(r_\alpha(E), r_\alpha(F))$
Сокрытие	$E \xrightarrow{(\alpha, r)} E' (\alpha \notin L)$ $E / L \xrightarrow{(\alpha, r)} E' / L (\alpha \notin L)$ $E \xrightarrow{(\alpha, r)} E' (\alpha \in L)$ $E / L \xrightarrow{(\alpha, r)} E' / L (\alpha \in L)$
Constant	$E \xrightarrow{(\alpha, r)} E' \left(\begin{array}{l} def \\ A = E \end{array} \right)$ $A \xrightarrow{(\alpha, r)} E' \left(\begin{array}{l} def \\ A = E \end{array} \right)$

Constant : Служит для присвоения названия компонентам. Значения констант компонент задаются равенствами. Так в системе обозначений $X = E$ имя X определяет область действия в выражении, находящемся справа, например, $X = (\alpha, r).X$ выполняет α по норме r .

Сокрытие: Представлен скрытый оператор, обозначенный P/L . Здесь, множество L идентифицирует те действия, которые считаются внутренними или частными для компонента. Он объявлен как неизвестный тип τ .

Кооперация: Пишем $P \triangleright \triangleleft Q_L$ для того, чтобы обозначить коопeração между P и Q по L . Множество, которое используется как нижний индекс в символе кооперации, кооперацию устанавливает L , определяет те действия, в которых кооперанды синхронизируют. Для типов действия не установленных L , компоненты независимы и их действия одновременно допустимы.

Будем писать $P \parallel Q$ как сокращение для $P \triangleright \triangleleft Q_L$, когда множество L пусто. Если компо-

нент допускает деятельность, тип действия которой в кооперации не будет в состоянии продолжать совместно деятельность с другим компонентом, то также допускает деятельность того же типа. Эти два компонента тогда продолжают вместе разделенную деятельность. Норма разделенной деятельности может быть изменена, чтобы отразить работу, выполненную двумя компонентами для завершения деятельности. Суммарную мощность компонент C для выполнения действий типа α называют очевидной нормой α в P и обозначают через $r_\alpha(P)$. В отличие от других алгебр стохастического процесса, язык РЕРА имеет ограниченную способность, а именно, при кооперации компонент не может выполнить деятельность быстрее, таким образом, норма разделенной деятельности является минимумом норм деятельности в кооперации компонент.

В некоторых случаях, когда деятельность выполняют в кооперации с другим компонентом, компонент может быть пассивным относительно этой деятельности. Это означает то, что норма деятельности не определена (обозначена \perp) и определена в кооперации, при норме деятельности в другом компоненте. Все пассивные действия должны быть синхронизированы в конечной модели.

Синтаксис может быть формально представлен посредством следующей грамматики:

$$\begin{aligned} S &::= (\alpha, r).S \mid S + S \mid C_s \\ P &::= P \triangleright \triangleleft P \mid P / L \mid C, \end{aligned}$$

где S обозначает последовательный компонент, и P обозначает эталонный компонент, который выполняется параллельно. C - постоянная, которая обозначает или последовательный компонент или эталонный компонент. C_s константы, которые обозначают последовательные компоненты.

Эффект этого синтаксического разделения между типами констант должен ограничить

компоненты языка PEPA так, чтобы кооперация последовательных процессов была необходимым условием для выполнения эргодического основного марковского процесса.

2 Представление состояний. В стохастическом процессе алгебра моделирует обычно представление состояний в терминах синтаксических форм эталонного выражения.

Структурная оперативная семантика определяет развитие модели и формируется маркированная система перехода (обычно её называют образованием диаграммы в системе PEPA) представления пространства состояний модели.

Это - диаграмма, в которой каждый узел - синтаксическое формирование или производная (или класс эквивалентности синтаксических выражений для сильной эквивалентности), а каждая дуга представляет возможную деятельность, вызывающую изменение состояния. Отметим, что в системе PEPA представление состояний - фактически маркированная система мультиперехода, потому что фиксирование кратности дуг крайне важно, особенно когда повторенные компоненты возведены в степень.

Когда марковская интерпретация реализована в системе PEPA, продолжительность каждой деятельности, как и предполагается, является случайной переменной управляемой отрицательным экспоненциальным распределением.

В этом случае диаграмма образования будет диаграммой переходов непрерывного марковского времени. Таким образом, каждое марковское состояние соединено с каждым синтаксическим членом.

Анализ характеристики выполнен в терминах распределения вероятности установившегося состояния, или переходное распределение вероятности, которое является чрезвычайно громоздким, когда пространство состояний является достаточно большим.

Вместо законченной синтаксической формы, начиная со статической, комбинаторы кооперации остаются неизменяемыми во всех состояниях, при этом часто удобно представлять состояния модели в векторной форме. Вектор состояний делает запись одного входа для каждого последовательного компонента модели PEPA. Эти компоненты будут присутствовать в каждой производной модели, хотя они изменяют свое локальное состояние или производную.

Таким образом, глобальное состояние может быть представлено как вектор или последовательность локальных производных.

Если модель содержит эквивалентные компоненты, могут возникать множественные состояния в пределах модели, поведение которых подобно и поэтому можно объединить модель.

Тогда диаграмма создана в терминах классов эквивалентности синтаксических членов и это используют для построения СТМС [16]. В основ-

ве этого метода лежит использование канонического вектора состояний, для того чтобы фиксировать синтаксическую форму эталонного выражения. Если у двух состояний одинаковый канонический вектор, то они эквивалентны и не должны отличаться в объединенной диаграмме образования. Канонизация вводит в степень входы переупорядочения в пределах вектора. А способ с учетом сильной эквивалентности упорядочивает марковское бимоделирование в системе PEPA и размещает в лексикографическом порядке элементы в пределах подвекторов эквивалентных компонентов. Подробности в [16].

В этой работе предлагается альтернативная векторная форма для фиксации информации о состоянии моделей с повторными компонентами. В векторной форме состояний, когда используется каноническое представление, в векторе есть один вход для каждого последовательного компонента модели. Когда число повторных компонентов становится большим, то это может занять большой объём памяти.

В альтернативе в векторной форме состояний есть один вход для каждой локальной производной каждого типа компонента модели. Два компонента имеют тот же самый тип, если их диаграммы образования изоморфны. Входы в векторе не имеют синтаксического представления членов локальной производной последовательного компонента, но число компонентов в данное время показывает, что это локальная производная.

Увидеть различие между двумя векторными формами можно на следующем примере, демонстрирующем взаимодействие процессоров и ресурсов:

$$\begin{aligned}
 \text{Processor } &_0 = (\text{task } 1, r_1). \text{Processor } &_1 \\
 \text{Processor } &_1 = (\text{task } 2, r_2). \text{Processor } &_0 \\
 \text{Resource } &_0 = (\text{task } 1, r_1). \text{Resource } &_1 \\
 \text{Resource } &_1 = (\text{reset }, s). \text{Resource } &_0 \\
 &(\text{Resource }_0 \parallel \text{Resource }_0) \triangleright \triangleleft_{\text{task } 1} \\
 &(\text{Processor }_0 \parallel \text{Processor }_0).
 \end{aligned}$$

В этом примере каноническое векторное соответствие формы с данной конфигурацией показано на рис. 1. Здесь начальное состояние представлено явно в виде

$$((\text{Resource }_0, \text{Resource }_0), \\
 (\text{Processor }_0, \text{Processor }_0))_{(\text{task } 1)}.$$

Напротив, в векторной числовой форме, которая представлена на рис. 2., задано начальное состояние $(2, 0, 2, 0)$, где входами в вектор считаются числа $\text{Resource}_0, \text{Resource}_1, \text{Processor}_0, \text{Processor}_1$. Представление в канонической форме вектора состояний делается записью ряда элементов в каждом эквивалентном классе (показано в квадратных скобках на рис. 1).

Полная норма переходов между канонически-

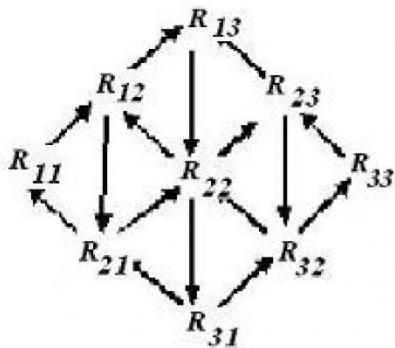


Рис. 1. Агрегированное пространство состояний в канонической форме

ми состояниями получается из числа экземпляров класса, где допущены действия над величинами и их относительными вероятностями.

В числовом векторном представлении каждый вектор является единственным для состояния и нормы переходов между состояниями получены непосредственно из вектора и степени активности. В текущей конфигурации модели, с двумя экземплярами класса из каждого составляющего типа ясно, что вектор состояний сформирован, а числовой вектор формируется из четырех элементов.

Если рассматривается конфигурация из десяти экземпляров класса, то из каждого составляющего типа становится ясно, что получаемая числовая форма намного более компактна. Кроме того, не происходит существенной потери информации.

На рис.1. вершинам графа, описывающего агрегированное пространство состояний в канонической форме, поставлены в соответствие:

$$\begin{aligned} R_{11} &:= ((Resource_0, Resource_0), \\ &\quad (Processor_1, Processor_1))_{(1)} . \\ R_{12} &:= ((Resource_0, Resource_0), \\ &\quad (Processor_0, Processor_1))_{(2)} . \\ R_{13} &:= ((Resource_0, Resource_0), \\ &\quad (Processor_0, Processor_0))_{(1)} . \\ R_{21} &:= ((Resource_0, Resource_1), \\ &\quad (Processor_1, Processor_1))_{(2)} . \\ R_{22} &:= ((Resource_0, Resource_1), \\ &\quad (Processor_0, Processor_1))_{(4)} . \\ R_{23} &:= ((Resource_0, Resource_1), \\ &\quad (Processor_0, Processor_0))_{(2)} . \\ R_{31} &:= ((Resource_1, Resource_1), \\ &\quad (Processor_1, Processor_1))_{(1)} . \\ R_{32} &:= ((Resource_1, Resource_1), \\ &\quad (Processor_0, Processor_1))_{(2)} . \\ R_{33} &:= ((Resource_1, Resource_1), \\ &\quad (Processor_0, Processor_0))_{(1)} . \end{aligned}$$

На рис.1. дугам графа, описывающего агрегированное пространство состояний в канонической форме, соответственно присвоены значения :

$$R_{11} \cup R_{12} := (task\ 2, 2r_2)$$

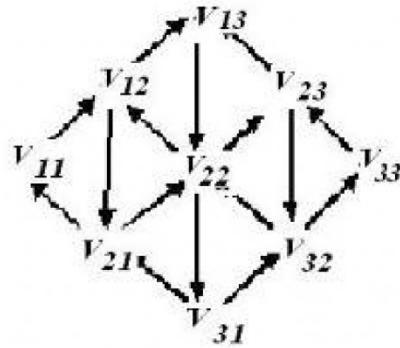


Рис. 2. Векторное пространство состояний

$$\begin{aligned} R_{12} \cup R_{13} &:= (task\ 2, r_2) \\ R_{21} \cup R_{22} &:= (task\ 2, 2r_2) \\ R_{22} \cup R_{23} &:= (task\ 2, r_2) \\ R_{31} \cup R_{32} &:= (task\ 2, 2r_2) \\ R_{32} \cup R_{33} &:= (task\ 2, r_2) \\ R_{21} \cup R_{11} &:= (reset, s) \\ R_{31} \cup R_{21} &:= (reset, 2s) \\ R_{22} \cup R_{12} &:= (reset, s) \\ R_{32} \cup R_{22} &:= (reset, 2s) \\ R_{23} \cup R_{13} &:= (reset, s) \\ R_{33} \cup R_{23} &:= (reset, 2s) \\ R_{12} \cup R_{21} &:= (task\ 1, r_1) \\ R_{13} \cup R_{22} &:= (task\ 1, 2r_1) \\ R_{22} \cup R_{31} &:= (task\ 1, r_1) \\ R_{23} \cup R_{32} &:= (task\ 1, r_1) \end{aligned}$$

На рис.2. вершинам графа, описывающего векторное пространство состояний, поставлены в соответствие:

$$\begin{aligned} V_{11} &:= (2, 0, 0, 2) \\ V_{12} &:= (2, 0, 1, 1) \\ V_{13} &:= (2, 0, 2, 0) \\ V_{21} &:= (1, 1, 0, 2) \\ V_{22} &:= (1, 1, 1, 1) \\ V_{23} &:= (1, 1, 2, 0) \\ V_{31} &:= (0, 2, 0, 2) \\ V_{32} &:= (0, 2, 1, 1) \\ V_{33} &:= (0, 2, 2, 0) \end{aligned}$$

На рис.2. дугам графа, описывающего векторное пространство состояний, присвоены те же значения, что и дугам графа, описывающего агрегированное пространство состояний в канонической форме на рис. 1.

Числовой вектор, формируемый для произвольной модели системы РЕРА, определяется следующим образом:

Определение 2.1 (Числовая векторная форма [7]) Для произвольной модели РЕРА М с n составляющими типами $C_i, i = 1, \dots, n$, каждый из которых имеет различные производные N , число-

вая векторная форма $M, \nu(M)$, является вектором $N = \sum_{i=1}^n N_i$ входами. Вход V_{i_j} показывает сколько экземпляров j -го класса частной производной составляющего типа С заданы в текущем состоянии.

3 Пространственные представления непрерывных состояний. Числовая векторная форма даёт альтернативный путь для представления пространства состояний модели РЕРА, но может также использоваться как привлечение обычного марковского анализа в терминах объединенной модели. Соединение частей основано на сильной эквивалентности.

Известно, что объединенный СТМС – укрупнённый эквивалент оригиналу СТМС, полученный из графа образования [17]. Объединенная модель изоморфна по отношению к канонической форме и автоматическому объединению частей, представленному в [16].

Однако, цель этой работы - использование формирования этого направления как основа для динамичного приближения, когда каждый компонент модели может быть скопирован достаточно большое число раз.

Дело обстоит так, что область значений каждого входа в $\nu(M)$ является достаточно обширной. Если K_i число компонентов типа C_i в начальной конфигурации модели, то каждый вход в i -ый подвектор будет из области $0, \dots, K_i$.

Рассмотрим произвольное состояние M' модели M , которое имеет числовое векторное представление $\nu(M')$. Когда происходит изменение состояния, это может случиться в двух различных случаях :

- когда единственный последовательный компонент является экземпляром класса компонент типа C_i и может участвовать в индивидуальном действии. В этом случае действие на $\nu(M')$ реализуется i -ым подвектором, один вход этого подвектора будет увеличен, а другой уменьшен на единицу.

- альтернативно общедоступное действие может быть выполнено, если участвуют одновременно два или больше последовательных компонент различных типов (так как предполагается, что повторяемые компоненты независимы друг от друга). Таким образом, подвекторы определены в $\nu(M')$. В этом случае один вход будет увеличен, а другой вход уменьшен на единицу.

Система входов дискретна, числовой вектор формируется из неотрицательных целых чисел. Когда число компонентов является достаточно большим, то шаги приближений являются относительно маленькими, и поведение можно рассматривать не как дискретное, а как непрерывное

состояние. Таким образом, наша цель состоит в том, чтобы заменить дискретную систему непрерывной, представленной графом образования системы РЕРА в виде системы дифференциальных уравнений. Представление числовой векторной формы состояний есть промежуточный шаг в достижении цели. Сформулируем несколько предварительных определений. Рассмотрим локальную производную D последовательного компонента. Деятельность (α, r) означает выходные действия D , если D соответствует (α, r) , т. е. допускает переход $D \rightarrow D$ в отмеченной системе перехода D . Обозначим набор выходных действий D через $Ex(D)$. Обозначим множество локальных производных, для которых (α, r) есть выходные действия, $Ex(\alpha, r)$.

Точно так же обозначим деятельность (β, s) , входных действий D , если существует производная D' , которая допускает (β, s) и D - первый шаг β – производная D' , т. е. допускает переход $D \rightarrow D$ в отмеченной системе перехода D . Используем $En(D)$ для обозначения набора входных действий D .

Эта классификация действий необходима для того, чтобы сделать запись действий каждой деятельности по каждой локальной производной C_{i_j} , в модели. Напомним, что в векторе сформированы числа этих производных $N(C_{i_j})$, которые стали переменными состояний.

Рассмотрим числовой вектор состояний. Пусть $V_{i_j}(t) = N(C_{i_j}, t)$ обозначает j -ий вход i -го подвектора по времени t , т. е. число экземпляров j -го класса локальных производных последовательного составляющего C_i . Для достаточно малого промежутка времени δt имеем :

$$\begin{aligned} N(C_{i_j}, t + \delta t) - N(C_{i_j}, t) &= \\ &- \underbrace{\sum_{(\alpha, r) \in Ex(C_{i_j})} r \times \min_{C_{k_j} \in Ex(\alpha, r)} (N(C_{k_j}, t)) \delta t}_{\text{для выходных действий}} \\ &+ \underbrace{\sum_{(\alpha, r) \in En(C_{i_j})} r \times \min_{C_{k_j} \in Ex(\alpha, r)} (N(C_{k_j}, t)) \delta t}_{\text{для входных действий}} \end{aligned}$$

Первый член делает запись влияния выходных действий. Если выходное действие - индивидуальная деятельность этого компонента $Ex(\alpha, r) = \{C_{i_j}\}$ и $\min(N(C_{i_j})) = N(C_{i_j}, t)$, то $N(C_{i_j}, t)$ являются экземплярами класса локальной производной, каждый из которых одновременно продолжает индивидуальную деятель-

ность. Когда действие $Ex(\alpha, r) \neq \{C_{i_j}\}$, то происходит разделенное вовлечение действий локальных производных от двух или более компонентов по типу многоканальной синхронизации. Норма в РЕРА определяется следующим образом: если N есть экземпляры класса компонента действий (α, r) , то норма действий будет $N \times r$. По семантике языка, норма синхронизированной деятельности есть минимум нормы кооперирующих компонентов. Второй член, который точно так же в замечании объяснен тем, что норма входной деятельности будет определена как число компонентов, для которой в соответствии с семантикой языка это и есть выходная деятельность.

Разделив на δt и взяв предел, при $\delta t \rightarrow 0$, получаем

$$\frac{dN(C_{i_j}, t)}{dt} = - \sum_{(\alpha, r) \in Ex(C_{i_j})} r \times \min_{C_{k_l} \in Ex(\alpha, r)} (N(C_{k_l}, t)) + \sum_{(\alpha, r) \in En(C_{i_j})} r \times \min_{C_{k_l} \in Ex(\alpha, r)} (N(C_{k_l}, t))$$

В следующем подразделе будет показано, как эти уравнения могут быть получены прямым путём из основного определения автоматически. Для полного определения системы обыкновенных дифференциальных уравнений (ОДУ) остается только задать начальные значения переменных состояний, т. е. $N(C_{i_j}, 0)$ при $i_j = 1, \dots, N$. Они легко получаются из начальной конфигурации.

4 Автоматически выводимые системы обыкновенных дифференциальных уравнений (ОДУ). Влияние действий производных может быть зафиксировано или с помощью графа, или в матричной форме, что легко получается из синтаксического представления модели.

Определение 4.1 (Граф действий [7]) Граф действий a является двусторонним графом (N, A) . Узлы N разделены в N_a действий и N_d производных. $A \subseteq (N_a \times N_d) \cup (N_d \times N_a)$, где $a = (n_a, n_d) \in A$, если n_a выходная деятельность для производной n_d , а $a = (n_d, n_a) \in A$, если n_a - входная деятельность для производной n_d .

Та же самая информация может быть представлена в т.н. матрице действий.

Определение 4.2 (Матрица действий [7]) Для модели с действиями N_a и N_d локальные производные различны, матрица действий M_a - матрица $N_d \times N_a$, и входные действия определены следующим образом:

$$(d_i, a_j) = \begin{cases} +1, & \text{если } a_j - \text{входная деятельность } d_i, \\ -1, & \text{если } a_j - \text{выходная деятельность } d_i, \\ 0 & \text{иначе.} \end{cases}$$

//Сформируем одно ОДУ для каждой локальной производной (переменные состояний)

For $i = 1, \dots, N_d$

//Произведём действия, привлекающие эту производную

For $i = 1, \dots, N_a$

If $M_a(i, j) \neq 0$

//Выходное множество формы $Ex(j)$

для деятельности j

$Ex(j) = 0$

For $k = 1, \dots, N_a$

If $M_a(k, j) = -1$

$Ex(j) = Ex(j) \cup \{k\}$

//Делаем запись влияния каждой такой деятельности

If $M_a(k, j) = +1$

добавим

$+ r_j \times \min_{k \in Ex(j)} (n_k(t))$

к уравнению

If $M_a(k, j) = -1$

добавим

$- r_j \times \min_{k \in Ex(j)} (n_k(t))$

к уравнению

Рис. 3. Алгоритм создания системы ОДУ

В матрице действий каждая строка соответствует единственной локальной производной. В представлении модели как системы обыкновенных дифференциальных уравнений найдётся одно уравнение для каждой переменной состояний, поскольку показано текущее значение каждой локальной производной. Это детали влияния уравнения остальной части системы на значения переменной состояний. Это может быть получено автоматически из матрицы действий, когда связываются переменная состояний n_i с каждой строкой матрицы, и постоянной нормой r_j с каждым столбцом матрицы. Число членов в системе ОДУ равно числу ненулевых записей в соответствующей строке, каждый член связывается с тем столбцом, где указана норма действий.

Как ранее было объяснено по семантике РЕРА, изменение фактической нормы, вызванное каждым действием, будет нормой умноженной на минимум текущего числа локальных производных.

Это даёт возможность для параллельных действий при определении типа компонента для каждой кооперации. Идентичность этих производных

может быть найдена в столбце, для соответствующих действий, а отрицательный элемент указывает, что производная участвует в этой деятельности. Каждой строке матрицы будет соответствовать одно уравнение системы ОДУ.

5 Пример. Снова обратимся к маленькому примеру, рассмотренному ранее, предположим теперь, что есть большие количества процессоров и ресурсов:

$$\begin{aligned} Processor_0 &\stackrel{\text{def}}{=} (task\ 1, r_1).Processor_1 \\ Processor_1 &\stackrel{\text{def}}{=} (task\ 2, r_2).Processor_0 \\ Resource_0 &\stackrel{\text{def}}{=} (task\ 1, r_1).Resource_1 \\ Resource_1 &\stackrel{\text{def}}{=} (reset, s).Resource_0 \\ (Resource_0 \parallel \dots \parallel Resource_0) &\triangleright \triangleleft_{(task\ 1)} \\ (Processor_0 \parallel \dots \parallel Processor_0) \end{aligned}$$

Граф действий изображен на рис. 4. Матрица действий изображена с помощью табл. 2.

Из матрицы получаем каждое дифференциальное уравнение и наоборот. Для каждой переменной состояний η_i , рассмотрим строку i . Каждый ненулевой элемент в строке будет соответствовать одному члену уравнения.

$$\begin{aligned} \frac{dn_1(t)}{dt} &= -r_1 \min(n_1(t), n_3(t)) + r_2 n_2(t) \\ \frac{dn_2(t)}{dt} &= r_1 \min(n_1(t), n_3(t)) - r_2 n_2(t) \\ \frac{dn_3(t)}{dt} &= -r_1 \min(n_1(t), n_3(t)) + s n_4(t) \\ \frac{dn_4(t)}{dt} &= r_1 \min(n_1(t), n_3(t)) - s n_4(t) \end{aligned} \quad (1)$$

Таблица 2 Матрица действий для модели ресурс-процессор

	task 1	task 2	reset	
Resource 0	-1	+1	0	n_1
Resource 1	+1	-1	0	n_2
Resource 0	-1	0	+1	n_3
Resource 1	+1	0	-1	n_4

Отметим, что включенная в начальную конфигурацию модели форма системы уравнений не зависит от числа компонентов. Только воздействия изменения числа экземпляров класса для каждого составляющего типа должны изменить начальные условия.

Таким образом, если есть первоначально 200 процессоров, начинаем с $Processor_0$ и 120 ресурсов, распределяются поровну между $Resource_0$ и $Resource_1$, тогда имеем начальные условия:

$$n_1(0) = 200 \quad n_2(0) = 0 \quad n_3(0) = 60 \quad n_4(0) = 60 \quad (2)$$

Приступаем к аналитическому решению сис-

темы (1) с начальными условиями (2).

Складывая первое и второе уравнения системы (1), а также третье и четвёртое той же системы получим

$$\frac{dn_1(t)}{dt} + \frac{dn_2(t)}{dt} = 0, \quad \frac{dn_3(t)}{dt} + \frac{dn_4(t)}{dt} = 0$$

С учётом (2) имеем

$$\begin{aligned} n_1(t) + n_2(t) &= 200 \\ n_3(t) + n_4(t) &= 120 \end{aligned} \quad (3)$$

Исключая из первого и третьего уравнений системы (1) с помощью (3) функции $n_2(t)$ и $n_4(t)$, получаем

$$\begin{aligned} \frac{dn_1(t)}{dt} &= -r_1 \min(n_1(t), n_3(t)) + r_2(200 - n_1(t)) \\ \frac{dn_3(t)}{dt} &= -r_1 \min(n_1(t), n_3(t)) + s(120 - n_3(t)) \end{aligned} \quad (4)$$

Вычитая из первого уравнения системы (4) второе, приходим к уравнению

$$\frac{d(n_1(t) - n_3(t))}{dt} = 200r_2 - r_2 n_1(t) - 120s + sn_3(t) \quad (5)$$

Введём вспомогательную функцию

$$y(t) = n_1(t) - n_3(t) \quad (6)$$

Из (5) с учётом (6) имеем

$$\frac{dy(t)}{dt} + sy(t) = 200r_2 - 120s + (s - r_2)n_1(t) \quad (7)$$

5.1. Вывод решения системы обыкновенных дифференциальных уравнений при $n_1(t) < n_3(t)$.

В случае

$$n_1(t) < n_3(t) \quad (8)$$

первое уравнение системы (4) принимает вид

$$\frac{dn_1(t)}{dt} = -r_1 n_1(t) + r_2(200 - n_1(t)) \quad (9)$$

Решая его относительно $n_1(t)$, получим

$$n_1(t) = \frac{200r_1}{r_1 + r_2} \exp(-(r_1 + r_2)t) + \frac{200r_2}{r_1 + r_2} \quad (10)$$

Из (10) и (3) следует

$$n_2(t) = \frac{200r_1}{r_1 + r_2} - \frac{200r_1}{r_1 + r_2} \exp(-(r_1 + r_2)t) \quad (11)$$

Исключая из (7) и (10) функцию $n_1(t)$, приходим к уравнению

$$\begin{aligned} \frac{dy(t)}{dt} + sy(t) &= \frac{200r_1r_2 - 120sr_1 + 80sr_2}{r_1 + r_2} + \\ &+ (s - r_2) \frac{200r_1}{r_1 + r_2} \exp(-(r_1 + r_2)t) \end{aligned} \quad (12)$$

Решая это линейное дифференциальное уравнение относительно функции $y(t)$, получим

$$y(t) = \alpha_1 \exp(-st) + \alpha_2 \exp(-(r_1 + r_2)t) + \alpha_3, \quad (13)$$

где

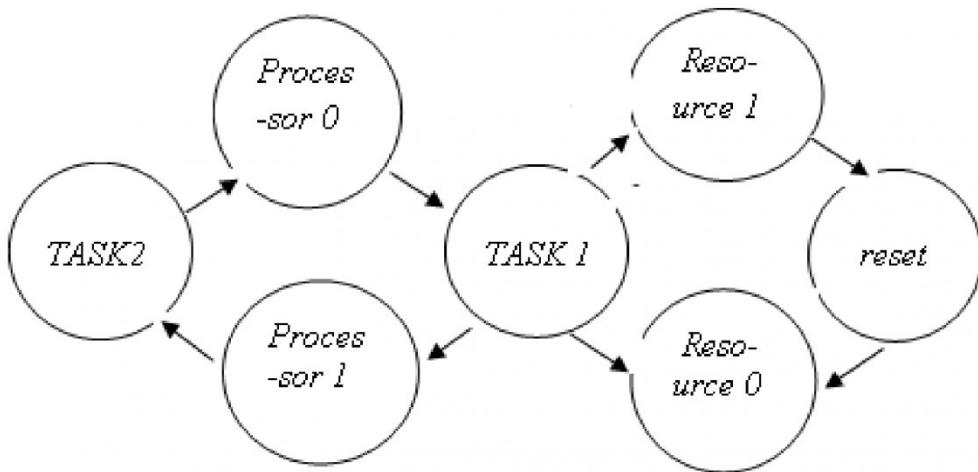


Рис. 4. Граф действий простой модели ресурс-процессор

$$\begin{aligned}\alpha_1 &= 260 + \frac{200 r_2^2}{s(r_1 + r_2 - s)} + \frac{200 s - 400 r_2}{(r_1 + r_2 - s)} - \frac{200 r_2^2}{s(r_1 + r_2)}, \\ \alpha_2 &= \frac{200 r_1(s - r_2)}{(s - r_1 - r_2)(r_1 + r_2)}, \\ \alpha_3 &= \frac{200 r_2^2 - 120 s r_1 + 80 s r_2}{s(r_1 + r_2)}.\end{aligned}$$

Из (13) и (6) следует

$$\begin{aligned}n_3(t) &= -\alpha_1 \exp(-st) + \frac{200 r_1^2}{(r_1 + r_2)(r_1 + r_2 - s)} \times \\ &\quad \times \exp(-(r_1 + r_2)t) + 120 - \frac{200 r_2^2}{s(r_1 + r_2)}.\end{aligned}\quad (14)$$

Из (14) и второго уравнения (3) следует

$$\begin{aligned}n_4(t) &= \alpha_1 \exp(-st) - \frac{200 r_1^2}{(r_1 + r_2)(r_1 + r_2 - s)} \times \\ &\quad \times \exp(-(r_1 + r_2)t) + \frac{200 r_2^2}{s(r_1 + r_2)}.\end{aligned}\quad (15)$$

5.2. Вывод решения системы обыкновенных дифференциальных уравнений при $n_1(t) > n_3(t)$.

Рассмотрим теперь случай $n_1(t) > n_3(t)$

Тогда второе уравнение системы (4) с учетом (3) принимает вид

$$\frac{dn_3(t)}{dt} = -r_1 n_3(t) + s(120 - n_3(t))\quad (17)$$

Решая его относительно $n_3(t)$, получим

$$n_3(t) = (n_3(0) - \frac{120s}{r_1 + s}) \exp(-(r_1 + s)t) + \frac{120s}{r_1 + s}\quad (18)$$

С учетом начальных условий имеем

$$n_3(t) = \frac{60(r_1 - s)}{r_1 + s} \exp(-(r_1 + s)t) + \frac{120s}{r_1 + s}\quad (19)$$

Тогда первое уравнение системы (4) с учетом (3) принимает вид

$$\frac{dn_1(t)}{dt} = -r_1 n_3(t) + r_2(200 - n_1(t))\quad (20)$$

Из (20) и (19) следует

$$\begin{aligned}\frac{dn_1(t)}{dt} + r_2 n_1(t) &= 200 r_2 - \frac{60r_1(r_1 - s)}{r_1 + s} \times \\ &\quad \times \exp(-(r_1 + s)t) - \frac{120r_1s}{r_1 + s}.\end{aligned}\quad (21)$$

$$n_1(t) = \beta_1 \exp(-r_2 t) + \beta_2 \exp(-(r_1 + s)t) + \beta_3,\quad (22)$$

где

$$\begin{aligned}\beta_1 &= \frac{60r_1(2s - r_2)}{r_2(r_1 - r_2 + s)}, \quad \beta_2 = \frac{60r_1(r_1 - s)}{(s + r_1 - r_2)(r_1 + s)}, \\ \beta_3 &= 200 - \frac{120r_1s}{r_2(r_1 + s)},\end{aligned}$$

Из (22) и (3) следует

$$\begin{aligned}n_2(t) &= \frac{120r_1s}{r_2(r_1 + s)} - \beta_1 \exp(-r_2 t) - \\ &\quad - \beta_2 \exp(-(r_1 + s)t).\end{aligned}\quad (23)$$

Из (19) и второго уравнения (3) следует

$$n_4(t) = \frac{120r_1}{r_1 + s} - \frac{60(r_1 - s)}{r_1 + s} \exp(-(r_1 + s)t).\quad (24)$$

6 Пример 1. Рассмотрим модель, которая демонстрирует использование ОДУ как процедуры анализа и связывает с существующими методами анализа характеристик для моделей системы PEPA.

Пусть Веб-служба имеет два типа клиентов; первые клиенты, которые обращаются к Веб-службе через безопасную внутреннюю сеть, и вторые клиенты браузера, которые обращаются к Веб-службе через внешнюю сеть. Вторые клиенты направляют свои запросы сервиса через посредников, которым доверяют.

Чтобы гарантировать модульное наращивание, Веб-служба тиражируется с помощью главных компьютеров. Доступны многочисленные посредники. Многочисленные первые клиенты

для системы сетевой защиты используют сервис с помощью безопасных внутренних сетей. Вне системы сетевой защиты найдутся вторые клиенты. Вторые клиенты используют кодирование, чтобы гарантировать подлинность и свойства конфиденциальности. Посредники добавляют расшифровку, чтобы выстроить непрерывную

безопасность). Сначала посыпает запрос к Веб-службе, а затем ждет ответа. Передача, расшифровка и решифровка выполняются еще до получения ответа клиентом:

$$\text{Broker } \text{idle} \stackrel{\text{def}}{=} (\text{request } b, \perp). \text{Broker } \text{dec_input}$$

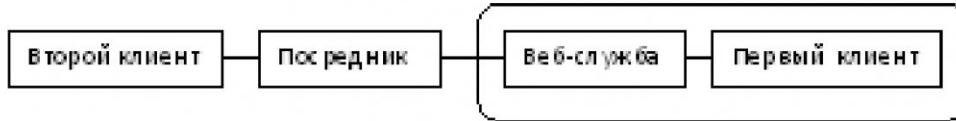


Рис. 5. Топология коммуникаций системы.
Веб-служба с первым и вторым клиентом и посредником.

безопасность из двухточечной безопасности. Обрабатывая запрос от второго клиента посредники расшифровывают запрос прежде, чем зашифровать его для Веб-службы еще раз. Когда ответ на запрос возвращен к посреднику, то он расшифровывает ответ прежде, чем повторно зашифровать его для второго клиента.

Схематическая топология коммуникаций системы показана на рис. 5.

Эталонные файлы обработаны описательными идентификаторами, такими как *FPC*, являющимися инструментальными средствами моделирования. Обозначим "первого клиента" через *FPC*, "второго клиента" через *SPC*, "Веб-службу" через *WS*.

7 Модель системы РЕРА. Второй клиент во время существования циклически обслуживает запросы, шифруя их и посыпая их посреднику. И он ждет ответа от посредника. Скорость, с которой происходят первые три действия, находится под управлением клиента. Скорость, с которой произведены ответы, определена при взаимодействии посредника и конечной точки сервиса. Как обычно, в моделях РЕРА этот компонент содержит некоторые индивидуальные действия, в которых он непосредственно участвует (оформление закономерностей и кодирование), и некоторые действия, которые выполнены в кооперации с другим компонентом (запрос и ответ находятся в кооперации с посредником):

$$\begin{aligned} SPC \text{ idle} &\stackrel{\text{def}}{=} (\text{compose } sp, r_{sp_cmp}). SPC \text{ enc} \\ SPC \text{ enc} &\stackrel{\text{def}}{=} (\text{encrypt } b, r_{sp_encb}). SPC \text{ sending} \\ SPC \text{ sending} &\stackrel{\text{def}}{=} (\text{request } b, r_{sp_req}). SPC \text{ waiting} \\ SPC \text{ waiting} &\stackrel{\text{def}}{=} (\text{response } b, \perp). SPC \text{ dec} \\ SPC \text{ dec} &\stackrel{\text{def}}{=} (\text{decrypt } b, r_{sp_decb}). SPC \text{ idle}. \end{aligned}$$

Посредник пассивен, пока он не получает запрос. Тогда он расшифровывает запрос прежде, чем повторно зашифровать его для Веб-службы (чтобы гарантировать непрерывную безопас-

$$\begin{aligned} \text{Broker } \text{dec_input} &\stackrel{\text{def}}{=} (\text{decrypt } sp, r_{b_dec_sp}) \times \\ &\times \text{Broker } \text{enc_input} \\ \text{Broker } \text{enc_input} &\stackrel{\text{def}}{=} (\text{encrypt } ws, r_{b_enc_ws}) \times \\ &\times \text{Broker } \text{sending} \\ \text{Broker } \text{sending} &\stackrel{\text{def}}{=} (\text{request } ws, r_{b_req}). \text{Broker } \text{waiting} \\ \text{Broker } \text{waiting} &\stackrel{\text{def}}{=} (\text{response } ws, \perp). \text{Broker } \text{dec_resp} \\ \text{Broker } \text{dec_resp} &\stackrel{\text{def}}{=} (\text{decrypt } ws, r_{b_dec_ws}) \times \\ &\times \text{Broker } \text{enc_resp} \\ \text{Broker } \text{enc_resp} &\stackrel{\text{def}}{=} (\text{encrypt } sp, r_{b_enc_sp}). \text{Broker } \text{replying} \\ \text{Broker } \text{replying} &\stackrel{\text{def}}{=} (\text{response } b, r_{b_resp}). \text{Broker } \text{idle}. \end{aligned}$$

Время существования первого клиента отражает время существования второго клиента за исключением того, что кодирование не должно использоваться, когда коммуникация производится через безопасную внутреннюю сеть. Методы вызова Веб-службы могут также быть различны, ибо сервис может быть вызван при удаленном запросе на главный компьютер вместо запроса HTTP (гипертекстовый транспортный протокол). Таким образом, первый клиент использует Веб-службу как блокирующий удаленный запрос:

$$\begin{aligned} FPC \text{ idle} &\stackrel{\text{def}}{=} (\text{compose } fp, r_{fp_cmp}). FPC \text{ calling} \\ FPC \text{ calling} &\stackrel{\text{def}}{=} (\text{invoke } ws, r_{fp_inv}). FPC \text{ blocked} \\ FPC \text{ blocked} &\stackrel{\text{def}}{=} (\text{result } ws, \perp). FPC \text{ idle}. \end{aligned}$$

Моделируется выполнение потока обслуживания во внешней сети. Существует два пути, которыми выполняется обслуживание. Выбор процесса обслуживания модели алгебры процесса производится путем выполнения одного из двух его режимов. Продолжительность выполнения обслуживания неизменна. Различие имеется только в том необходимо ли кодирование и представляет ли данный результат как ответ HTTP

(гипертекстового транспортного протокола) или как прямое значение:

$$\begin{aligned}
 WS_{idle} &= (request_{ws}, \perp).WS_{decoding} + \\
 &+ (invoke_{ws}, \perp).WS_{method} \\
 WS_{decoding} &\stackrel{\text{def}}{=} (\text{decrypt } Req_{ws}, r_{ws_dec_b}) \times \\
 &\times WS_{execution} \\
 WS_{execution} &\stackrel{\text{def}}{=} (\text{execute } ws, r_{ws_exec})WS_{securing} \\
 WS_{securing} &\stackrel{\text{def}}{=} (\text{encrypt } Resp_{ws}, r_{ws_enc_b}) \times \\
 &\times WS_{responding} \\
 WS_{responding} &\stackrel{\text{def}}{=} (\text{response } ws, r_{ws_resp_b})WS_{idle} \\
 WS_{method} &\stackrel{\text{def}}{=} (\text{execute } ws, r_{ws_exec})WS_{returning} \\
 WS_{returning} &\stackrel{\text{def}}{=} (\text{result } ws, r_{ws_res})WS_{idle}
 \end{aligned}$$

В начальном состоянии системной модели представлен каждый из четырех составляющих типа, находящихся первоначально в пассивном состоянии. Компоненты K , L и M синхронизируют на их общих видах действий,

$$\begin{aligned}
 System &\stackrel{\text{def}}{=} \left(\underset{K}{\left(SPC_{idle} \triangleright\triangleleft Broker_{idle} \right)} \right) \triangleright\triangleleft \times \\
 &\times \left(\underset{M}{\left(WS_{idle} \triangleright\triangleleft FPC_{idle} \right)} \right)
 \end{aligned}$$

где

$$\begin{aligned}
 K &= \{request_b, response_b\} \\
 L &= \{request_{ws}, response_{ws}\} \\
 M &= \{invoke_{ws}, result_{ws}\}.
 \end{aligned}$$

Эта модель представляет пример наименьшей

системы, где встречается по одному случаю каждого составляющего типа. Оцениваем систему по числу клиентов, посредников и улучшению качества обслуживания.

8 Оценка анализа. Характеристики модели допускают много различных типов анализа. Некоторые имеют более низкое качество оценки, но менее информативны, чем традиционный анализ. Другие имеют более высокую оценку качества и более информативны, как, например, анализ переходного процесса. Сравниваем оценку на основе решения ОДУ с другими методиками, которые могли бы использоваться для анализа модели.

Отметим, что в табл. 3 показан только единственный анализ модели, выполненный на основе переходных процессов. Практически, из-за стохастической природы исследований, анализ должен быть повторен многократно, для того чтобы привести к результатам, сопоставимым с результатом анализа ОДУ. Кроме того отметим, что число ОДУ является постоянным независимо от числа компонентов в системе, пока пространство состояний растет динамично.

Из табл. 3 видно, что продолжительность получения результатов при решении ОДУ сравнивается с другими подходами, которые были опробованы при увеличении числа компонентов.

9 Сравнение результатов. На рис. 6 - 9 показаны результаты решения модели Веб-службы РЕРА как системы ОДУ с числом клиентов обоих видов, посредников, и веб-службы до 1000 экземпляров каждого класса.

Результаты представлены для нашей систе-

Таблица 3. Продолжительность исследований

Второй клиент	посредник	Web-служба	Первый клиент	Число состояний в полном пространстве состояний	Число состояний в соединенном пространстве состояний	Редкая матрица установившихся вычислений (время в секундах)	Гибрид матрицы МТВДД установившихся вычислений (время в секундах)	Переходное решение в течение времени $t = 100$ (время в секундах)	Моделирование МС один управляемый ко времени $t = 100$ (время в секундах)	Решение ОДУ (время в секундах)
1	1	1	1	48	48	1.04	1.10	1.01	2.47	2.81
2	2	2	2	6304	860	2.15	2.26	2.31	2.45	2.81
3	3	3	3	1130.5	161.3	172.5	255.5	588.8	2.48	2.83
4	4	4	4	234.7	30251	-	-	-	2.44	2.85
100	100	100	100	-	-	-	-	-	2.78	2.78
1000	100	500	1000	-	-	-	-	-	3.72	2.77
1000	1000	1000	1000	-	-	-	-	-	5.44	2.77

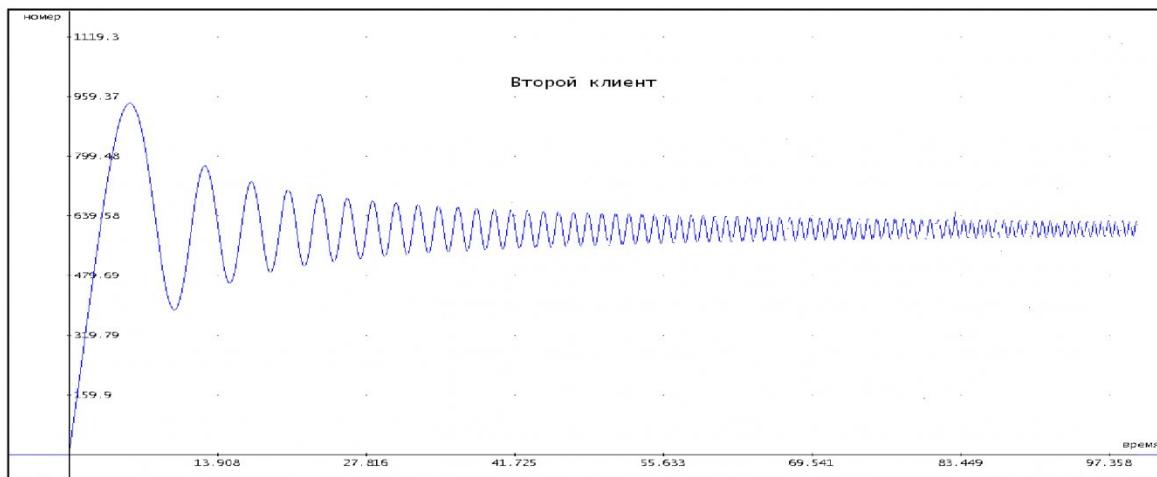


Рис. 6. График временного ряда компонента второй клиент.

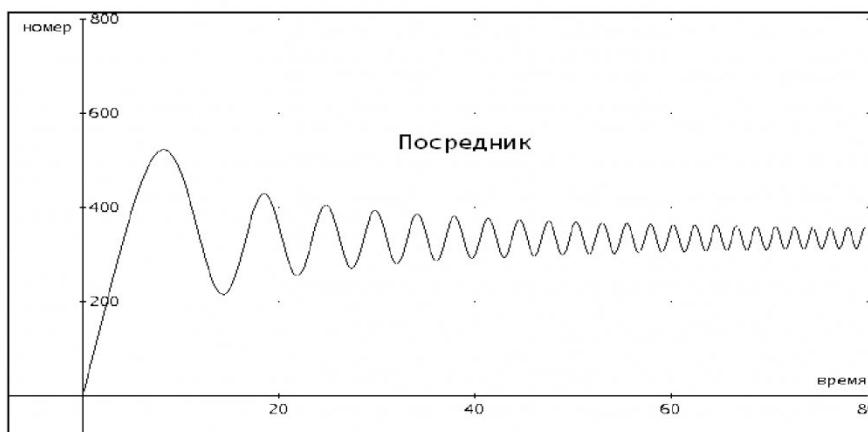


Рис. 7. График временного ряда компонента посредник.

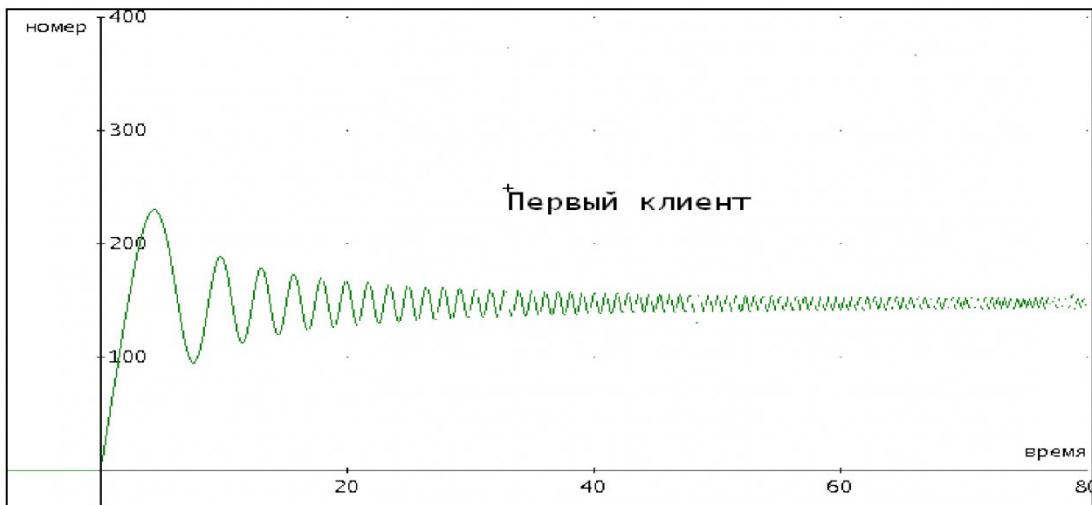


Рис. 8. График временного ряда компонента первый клиент

мы ОДУ интегратор – графики временных рядов, чисел для каждого типа изменяющихся как функции времени.

Показаны графы колебаний чисел компонентов в промежутке времени от $t = 0$ до $t = 100$ для оцениваемых значений.

Наблюдаем колебание деятельности в начальный период, пока система не стабилизируется в установившееся равновесие во времени (около

значения) $t = 50$.

Сравниваем вероятности типа компонентов с вероятностями, вычисленными с использованием системы РЕРА.

Эксперимент реализуем [18] на простейшем экземпляре модели (по одному компоненту каждого типа) и вычисление вероятностей с использованием коэффициента умножения 1000.

Такой подход законен в отсутствие разделения

на блоки, когда числа каждого компонента равны, но, возможно, он не использовался для модели с различным числом клиентов, посредников и служб. Хорошая согласованность результатов получилась при решении ОДУ после того, как система стабилизировалась в установившееся равновесие.

На рис. 6. представлен график временного ряда компонента второй клиент. Математическая модель временного ряда компонента второй клиент будет иметь следующий вид:

$$0.8632 \int_0^{246.356x} \cos\left(\frac{\pi t^2}{3920000}\right) dt.$$

На рис. 7 представлен график временного ряда компонента - посредник. Его математическая модель будет иметь вид:

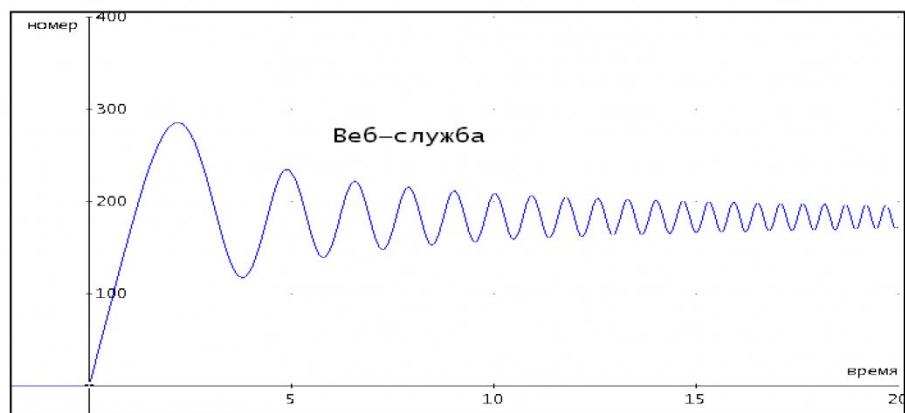


Рис. 9. График временного ряда компонента Веб-служба

$$4.11 \int_0^{19.695x} \cos\left(\frac{\pi t^2}{53113.85}\right) dt.$$

На рис. 8 представлен график временного ряда компонента первый клиент, математическая модель которого имеет вид:

$$0.6475 \int_0^{104.81x} \cos\left(\frac{\pi t^2}{414364}\right) dt.$$

На рис. 9. представлен график временного ряда компонента Веб-служба, математическая модель которого имеет вид:

$$0.77717 \int_0^{215.44x} \cos\left(\frac{\pi t^2}{444490}\right) dt.$$

Нормы изменения в модели обобщены так, чтобы позволить степеням активности управлять распределением вероятностей, заменив детерминированное описание системы набором случайных дифференциальных уравнений [19]. При дальнейшем обобщении появляется большая неопределенность, и поэтому возможно применение более сложных форм стохастических дифференциальных уравнений [20].

Представлен новый метод анализа характеристик больших систем в виде стохастического

алгебраического процесса.

В отличие от известного подхода анализа с помощью непрерывных марковских цепей [1 - 5] предлагается основное математическое представление в виде системы обыкновенных дифференциальных уравнений. Этот процесс хорошо применим к системам с большим числом компонент.

СПИСОК ЛИТЕРАТУРЫ

1. Сорокин А. С. Применение полумарковских процессов к определению характеристик надежности технологических схем. // Вестн. Кузбасского гос. тех. унив., 2005. № 1. С. 3 - 9 .
2. Сорокин А.С. Применение методов теории вероятностей к исследованию некоторых процессов производства./Труды 4-ой междунар. Конф. Кибернетика и технологии XXI века. Воронеж, 2003. С. 312-323.
3. Сорокин А.С. Марковские процессы в теории надежности технологических систем гидродобычи угля // Вестн. Кузбасского гос. тех. унив., 2008. № 1. С.61-69.
4. Сорокин А.С. Структурное моделирование надежности технологических систем с использованием скелетонов// Вестн. Кузбасского гос. тех. унив., 2008. № 4(68). Кемерово, С. 31-45.
5. Сорокин А.С. Математическое моделирование оценки надежности технологических систем// Вестн. Кузбасского гос. тех. унив., 2008. № 5(69). Кемерово, С. 28-37.
6. Коэн Дж., Боксма О. Границные задачи в теории массового обслуживания. М.: МИР, 1987.
7. Hillston J. A Compositional Approach to Performance Modelling. Cambridge University Press, 1996.

8. Hermanns H., Meyer-Kayser J., Siegle M. Multiterminal binary decision diagrams to represent and analyse continuous time Markov chains. In Proc. of 3rd Intl. Workshop on the Numerical Solution of Markov Chains, pages 188–207, 1999.
9. Deavours D., Sanders W. An efficient disk-based tool for solving large Markov models. Performance Evaluation, 33:67–84, 1998.
10. Knottenbelt W.J. Parallel Performance Analysis of Large Markov Models. PhD thesis, Department of Computing, Imperial College, 1999.
11. Hillston J. Tuning systems: From composition to performance. The Computer Journal, 2005. To appear.
12. Bowman H., Bryans J., Derrick J. Analysis of a multimedia stream using stochastic process algebra. In C. Priami, editor, 6th Int. Workshop on Process Algebras and Performance Modelling, pages 51–69, Nice, September 1998.
13. Bradley J.T., Dingle N.J., Gilmore S.T., Knottenbelt W.J. Derivation of passage-time densities in PEPA models using ipc: the Imperial PEPA Compiler. In G. Kotsis, editor, Proc. of the 11th IEEE/ACM Int. Symp. on Modeling, Analysis and Simulation of Computer and Telecommunications Systems, pages 344–351, October 2003. IEEE Computer Society Press.
14. Thomas N., Bradley J.T., Knottenbelt W.J. Stochastic analysis of scheduling strategies in a grid-based resource model. IEE Software Engineering, 151(5):232–239, September 2004.
15. Hillston J., Kloul L., Mokhtari A. Towards a feasible active networking scenario. Telecommunication Systems, 27(2–4):413–438, 2004.
16. Gilmore S., Hillston J., Ribaud M. An efficient algorithm for aggregating PEPA models. IEEE Transactions on Software Engineering, 27(5):449–464, May 2001.
17. Hillston J. Compositional Markovian modelling using a process algebra. In W. Stewart, editor, Proc. of 2nd Int. Workshop on Numerical Solution of Markov Chains: Computations with Markov Chains, Raleigh, North Carolina, Jan. 1995. Kluwer Academic Press.
18. Gilmore S., Hillston J. The PEPA Workbench: A Tool to Support a Process Algebra-based Approach to Performance Modelling. In Proc. of 7th Int. Conf. on Modelling Techniques and Tools for Computer Performance Evaluation, number 794 in LNCS, pages 353–368, Vienna, May 1994. Springer-Verlag.
19. Soong T. Random Differential Equations in Science and Engineering. Academic Press, 1973.
20. Oksendal B. Stochastic Differential Equations. Springer, 2003.

□ Автор статьи:

Сорокин
Андрей Семенович
- канд. физ.-мат. наук, доцент, ст.н.с.
(филиал КузГТУ в г. Новокузнецке),
тел.8(3843)725007

УДК 519. 21

А.В. Бирюков

ГРАНУЛОМЕТРИЯ ДИСПЕРСНЫХ СИСТЕМ

В дисперсной системе, частицы которой заполняют некоторую трехмерную область, будем рассматривать множество A , состоящее из всех частиц дисперсной системы, и множество B частиц, расположенных на поверхности системы.

В основе гранулометрического анализа дисперсной системы лежат результаты измерений частиц из множества A , образующих репрезентативную выборку. Однако в большинстве случаев измерениям доступны лишь частицы из множества B , что не обеспечивает репрезентативность выборки.

Тем не менее, результаты таких измерений обычно распространяют на всю дисперсную систему, что приводит к существенному искажению гранулометрических характеристик.

Крупность частиц характеризуется диаметром x , т. е. наибольшим линейным размером.

Обозначим через $f(x)$ и $g(x)$ плотности распределения диаметра частиц из множеств A и B , а через M_k и N_k - моменты порядка k этих распределений, получаемых интегрированием функций $x^k f(x)$ и $x^k g(x)$ по всему диапазону значений.

Как показывают многочисленные наблюдения, доля крупных частиц в множестве B больше, чем в множестве A .

Для мелких частиц наблюдается обратная картина. Поэтому существует значение x_0 диаметра частиц, для которого выполняются условия: $f(x) < g(x)$ при $x > x_0$ и $f(x) > g(x)$ при $x <$