УДК 519.872.2

Ю. А. Троян, А. А. Рубчинский

Разрешение конфликтов в марковских цепях

Параллельные стохастические процессы в некоторых системах можно представлять как одновременное функционирование нескольких марковских цепей с одним и тем же множеством состояний. При этом состояния интерпретируются как неделимые ресурсы, поскольку в каждый момент времени в них может находиться не более одной цепи. Ситуация, при которой несколько цепей одновременно требуют перехода в одно и то же состояния, и представляет собой конфликт. Разрешение конфликта состоит в выборе той цепи, которой разрешается переход в конфликтное состояние, в то время как все остальные цепи, также желающие попасть в данное состояние, находятся в режиме ожидания, т.е. просто простаивают.

В работе предложен новый алгоритм разрешения подобного рода конфликтов, который может использоваться в ситуациях, когда известные ранее методы неприменимы.

Ключевые слова: системы массового обслуживания, управляемые марковские цепи с доходами, бернуллиевские процессы, разрешение конфликтов.

Об авторах

Троян Юрий Александрович — аспирант кафедры прикладной математики и информатики Государственного университета «Дубна».

Рубчинский Александр Анатольевич — кандидат технических наук, доцент кафедры прикладной математики и информатики Государственного университета «Дубна».

Цепь Маркова — такая последовательность случайных событий с конечным или счётным числом исходов, что, нестрого говоря, при фиксированном настоящем будущее независимо от прошлого. Названа в честь А.А. Маркова [3].

Обычно под марковской цепью имеют ввиду простую марковскую цепь, т.е. такую, где вероятности состояний в будущем зависят только от текущего состояния системы. Сложной же цепью Маркова называют такую, где состояния в будущем зависят от текущего состояния и от состояний на нескольких предыдущих шагах. В данной статье будут рассматриваться только простые цепи Маркова.

Процесс, работа которого описывается цепью Маркова, называется марковским. Наглядным примером такого процесса может служить поведение лягушки в пруду с кувшинками, где количество кувшинок ограничено, а лягушка время от времени перепрыгивает с одного листа кувшинки на другой по своему желанию в данный момент. Вероятность того, что она прыгнет на конкретный лист, зависит не от того, когда и на каких листьях лягушка побывала в прошлом, а от её текущего местоположения.

Марковские цепи являются одним из распространённых инструментов моделирования стохастических систем.

Пусть некоторая система может находиться в каждый момент времени $t = 0,1,\ldots$ в одном из состояний множества $S = \{1,\ldots,N\}$. Говорят, что поведение системы описывается марковской цепью с переходными вероятностями p_{ij}, если через единицу времени после того, как система находилась в состоянии $i \in S$, она с вероятностью p_{ij} оказывается в состоянии $j \in S$. Матрицу $\mathrm{P} = (p_{ij})$ называют матрицей переходов или матрицей переходных вероятностей.

Бернуллиевским марковским процессом называют такой, у которого вероятность перехода в любое состояние зависит только от самого желаемого для перехода состояния, но не от состояния, из которого процесс может перейти в него.

Вероятность нахождения цепи в случайный момент времени в состоянии $i \in S$ называется стационарной вероятностью состояния i и обозначается через π_i ($i = 1,\ldots,N$). Вектор $\pi = (\pi_1,\ldots,\pi_N)$ находится решением системы линейных уравнений:

$$\pi\mathrm{P} = \pi,\ \sum_{i=1}^{N} \pi_i = 1.$$

Приведём пример. Пусть некая фирма выпускает телевизоры, на которые спрос у населения может либо быть, либо нет. В зависимости от этого в конце каждого года фирма может находиться в одном из состояний: «спрос есть» (состояние 1) или «спроса нет» (состояние 2). Пусть вероятность сохранить уже имеющийся спрос в следующем году равна 4/5, а вероятность появления в следующем году отсутствующего спроса равна 3/5. Тогда работу фирмы можно описать марковской цепью с состояниями 1 и 2 и матрицей переходов

$$\mathrm{P} = \begin{pmatrix} 4/5 & 1/5 \\ 3/5 & 2/5 \end{pmatrix}.$$

Теперь усложним ситуацию. Введём возможность управления марковской цепью и доходы в зависимости от одной из выбранных стратегий. Пусть фирма, находясь в состоянии 1 (спрос есть), может увеличить спрос путём проведения рекламной компании. Но это потребует дополнительных затрат, что уменьшит доход. В состоянии 2 (спроса нет) фирма может увеличить вероятность появления спроса путём увеличения затрат на исследования, и это тоже потребует дополнительных расходов. Выделим две стратегии. Первая состоит в отказе от дополнительных затрат (на рекламу и исследования), вторая — в согласии на них. Пусть матрицы переходных вероятностей и матрицы доходов для данных стратегий имеют вид

$$\mathrm{P}_1 = \begin{pmatrix} 0{,}5 & 0{,}5 \\ 0{,}4 & 0{,}6 \end{pmatrix}, \mathrm{R}_1 = \begin{pmatrix} 9 & 3 \\ 3 & -7 \end{pmatrix}$$
$$\mathrm{P}_2 = \begin{pmatrix} 0{,}8 & 0{,}2 \\ 0{,}7 & 0{,}3 \end{pmatrix}, \mathrm{R}_2 = \begin{pmatrix} 4 & 4 \\ 1 & -19 \end{pmatrix}.$$

Смысл матриц P_1, P_2, R_1, R_2 таков. При отказе от дополнительных затрат (стратегия 1) вероятности переходов задаются матрицей P_1, а доходы, в зависимости от состояния, — матрицей R_1. Так, если система находится в состоянии 2, то при стратегии 1 с вероятностью 0,6 она в нём так и останется (спрос так и не появится), а потери составят 7 единиц (например, 7 миллионов долларов). При стратегии 2 (т. е. при попытках активных действий) вероятность отсутствия спроса (при его отсутствии в данный момент) будет заметно меньше (0,3 вместо 0,6), но и потери будут значительно больше — 19 вместо 7.

В рассматриваемой ситуации имеет место управляемая марковская цепь. Управление состоит в выборе стратегии. Стратегией называется последовательный выбор управлений в последовательные моменты времени. Стратегии, которые зависят только от состояния системы, называют стационарными.

Оптимальные стратегии — стратегии, максимизирующие средний доход. По крайней мере одна из стационарных стратегий является оптимальной [4].

Задача разрешения конфликтов в марковских цепях

Во время работы системы из нескольких процессов некоторым из них может одновременно потребоваться один и тот же ресурс, который может быть предоставлен только одному из процессов. Такой ресурс является неделимым, а сама конфликтная ситуация требует оптимального разрешения.

Подобного рода конфликты возникают многократно в самых разных системах, скажем, во время работы компьютера. Внутри каждого процесса можно выделить некоторые критические участки, когда ему необходим некий неделимый ресурс. Так как в каждый момент времени неделимый ресурс может быть предоставлен не более чем одному процессу, то все остальные претендующие на этот ресурс процессы временно блокируются, и должны ждать, пока он не освободится. Конфликты, таким образом, сопряжены с задержками процессов, что снижает производительность системы.

Возникает задача поиска оптимального правила разрешения конфликтов — такого порядка предоставления неделимых ресурсов процессам, при которых потери от конфликтов минимальны.

Опишем поведение процесса марковской цепью. Пусть имеется n процессов, при этом k-й процесс может находиться в одном из m+1 состояний 0,1,…,m, которые изменяются в соответствии с переходными вероятностями $p_{ij}^k (i, j = 0,1, \dots, m;\ k = 1, \dots, n)$.

Все состояния, кроме состояния 0, соответствуют неделимым ресурсам, т. е. в любом из них может находиться не более одного процесса одновременно. В состоянии 0 может одновременно находится любое число процессов.

Конфликтом в такой системе будет желание более чем одного процесса перейти в одно и то же состояние из множества {1, …, m}, разрешением такого конфликта будет пропуск в соответствующее «конфликтное» состояние ровно одного процесса из конфликтующих, чей выбор определяется правилом разрешения конфликтов. Остальные конфликтные процессы до получения требуемого ресурса находятся в состоянии ожидания. Не участвующие в конфликте процессы изменяют свои состояния в соответствии со своими переходными вероятностями.

Рассмотрим простейший случай с двумя процессами и двумя состояниями. Матрицы переходных вероятностей отдельных процессов таковы:

$$P_1 = \begin{pmatrix} p_{00}^1 & p_{01}^1 \\ p_{10}^1 & p_{11}^1 \end{pmatrix},\ P_2 = \begin{pmatrix} p_{00}^2 & p_{01}^2 \\ p_{10}^2 & p_{11}^2 \end{pmatrix}.$$

Одновременная работа обоих процессов может быть описана одной марковской цепью с четырьмя состояниями: (0,0), (0,1), (1,0), (1,1). Здесь первая компонента соответствует первому процессу, вторая — второму.

Первые три состояния означают, в каком из состояний находится каждый из процессов. Но одновременное их нахождение в состоянии 1 невозможно, поэтому состояние (1,1) означает, что оба процесса только намереваются перейти в это состояние.

Первые три строки матрицы переходов P имеют вид:

	(0,0)	(0,1)	(1,0)	(1,1)
(0,0)	$p_{00}^1 p_{00}^2$	$p_{00}^1 p_{01}^2$	$p_{01}^1 p_{00}^2$	$p_{01}^1 p_{01}^2$
(0,1)	$p_{00}^1 p_{10}^2$	$p_{00}^1 p_{11}^2$	$p_{01}^1 p_{10}^2$	$p_{01}^1 p_{11}^2$
(1,0)	$p_{10}^1 p_{00}^2$	$p_{10}^1 p_{01}^2$	$p_{11}^1 p_{00}^2$	$p_{11}^1 p_{01}^2$

Так как в состоянии 1 может находиться лишь один из процессов, то другой должен ждать освобождения ресурса, не меняя своего намерения перейти в это состояние (без получения требуемого ресурса дальнейшая работа процесса невозможна). Таким образом, если при разрешении конфликта мы отдаём предпочтение первому процессу, что последняя строка матрицы имеет вид:

(1,1)	0	p_{10}^1	0	p_{11}^1

Если же предпочтение отдаётся второму процессу, последняя строка имеет вид:

(1,1)	0	0	p_{10}^2	p_{11}^2

Итак, имеется управляемая марковская цепь с четырьмя состояниями. Попадание в состояние (1,1) означает «простой» одного из процессов, потому будем считать доход системы в состоянии (1,1), равным -1 (можно также считать потери системы в том состоянии, равными 1), а во всех остальных состояниях равным 0. Наша задача свелась к поиску оптимальной стратегии управления полученной марковской цепью.

Заметим, что работа системы должна быть максимально простой и очень быстрой, поэтому и критерий предпочтения того или иного ресурса должен считаться быстро и не требовать решения задачи линейного программирования большой размерности. Заметим также, что число состояний цепи в случае n процессов и двух ресурсов (m = 1) равно 2^n, а в случае двух процессов и (m + 1) ресурсов – 2^m.

Давно известны рекуррентный и итерационный методы поиска оптимальных стратегий в марковских системах с заранее известными матрицами переходов и доходов [5]. Оба они не подходят нам ввиду требования как можно более быстрых вычислений. Кроме того, некоторые самые простые случаи были рассмотрены ещё в 1980-х годах. Так, для случая двух бернуллиевских процессов и двух ресурсов, из которых один — неделимый, была доказана теорема, что в таком случае следует отдавать предпочтение процессу с наименьшей вероятностью обращения к конфликтному ресурсу [2]. Однако уже тот же самый случай, но с небернуллиевскими процессами (формула 1), рассмотрен не был.

$$P_1 = \begin{pmatrix} c_0 & 1-c_0 \\ c_1 & 1-c_1 \end{pmatrix}, P_2 = \begin{pmatrix} c_2 & 1-c_2 \\ c_3 & 1-c_3 \end{pmatrix}. \quad (1)$$

Применение того же самого подхода к вышеуказанному случаю (предпочтение процесса с наименьшей вероятностью обращения к конфликтному ресурсу) привело к тому, что для порядка 14 % случаев конфликты разрешаются неоптимальным образом. В более точной формулировке: правило разрешения конфликтов, при котором в конфликтное состояние переходит та из марковских цепей, для которой стационарная вероятность данного состояния меньше, примерно в 14 % случаев (при случайном задании цепей) оказывается неоптимальным, т.е. общее время простоев оказывается больше, чем при противоположном решении. То же самое верно и для цепей большей размерности. Хорошо известно (см., например, [5]), что задача определения оптимальной стратегии сводится к задаче линейного программирования, однако размерность задачи для определения оптимальных стратегий разрешения конфликтов растет с ростом числа состояний и процессов экспоненциально, так что ни о какой быстроте решения, что критично при разрешении конфликтов в вычислительных системах, говорить не приходится.

Таким образом, единственным приемлемым решением проблемы становится поиск приближённого решения, требующего минимум вычислений, но дающего небольшой процент неоптимальных решений.

Одним из наиболее простых путей поиска такого решения стала классификация большого числа разных систем, для которых заранее было точно подсчитано оптимальное решение. В данной работе эта идея для простоты изложена для рассматриваемого случая двух процессов и одного конфликтного состояния.

Требовалось, по сути дела, определить разделяющую функцию в четырёхмерном пространстве. В целях поиска наиболее быстрого с точки зрения вычислений решения предполагалось искать линейную функцию.

Использовался следующий классический алгоритм поиска разделяющей функции (см, например, его описание в [1]):

1. Для каждой из множества четырёхмерных точек $(c_0, c_1, c_2, c_3), c_i \in (0, 1)$ рассчитать оптимальное правило разрешения конфликтов (отдавать предпочтение первому либо второму процессу) во время работы системы (1) и соответственно классифицировать точку как принадлежащую к одному из двух классов — 1-му или 2-му (какому процессу отдавать предпочтение).

2. Положить i равным 0, j_A и j_B равными 1, C_{A}^0 равным любой точке из класса 1, C_{B}^0 равным любой точке из класса 2, $S_A^0 = \frac{(C_{\mathrm{A}}, C_{\mathrm{A}})}{2}$, $S_{\mathrm{B}}^0 = \frac{(C_{\mathrm{B}}, C_{\mathrm{B}})}{2}$, $F(x) = \left(C_{\mathrm{A}}^0 - C_{\mathrm{B}}^0, x\right) - (S_{\mathrm{A}}^0 - S_{\mathrm{B}}^0)$, где (p,q) означает скалярное произведение векторов p и q.

3. Положить $i = i + 1$.

4. Выбрать следующую точку x^i, если больше точек нет, остановить работу.

5. Если $F(x^i) > 0$, а x^i принадлежит классу 1, или же если $F(x^i) < 0$, а x^i принадлежит классу 2, то положить $\mathrm{C}_{\mathrm{A}}^{\mathrm{i}} = \mathrm{C}_{\mathrm{A}}^{\mathrm{i}-1}$, $S_{\mathrm{A}}^i = S_{\mathrm{A}}^{i-1}$, $C_{\mathrm{B}}^i = C_{\mathrm{B}}^{i-1}$, $S_{\mathrm{B}}^i = S_{\mathrm{B}}^{i-1}$ и перейти к пункту 3.

6. Если $F(x^i) < 0$, то положить $\mathrm{C}_{\mathrm{A}}^{\mathrm{i}} = C_{\mathrm{A}}^{i-1} + \gamma^{j_{\mathrm{A}}}(x^i - C_{\mathrm{A}}^{i-1})$, $S_{\mathrm{A}}^i = S_{\mathrm{A}}^{i-1} + \gamma^{j_{\mathrm{A}}}\left[\left(C_{\mathrm{A}}^i, x^i\right) - 2S_{\mathrm{A}}^{i-1}\right]$, $C_{\mathrm{B}}^i = C_{\mathrm{B}}^{i-1}$, $S_{\mathrm{B}}^i = S_{\mathrm{B}}^{i-1}$, $j_{\mathrm{A}} = j_{\mathrm{A}} + 1$ и перейти к пункту 3.

7. Иначе положить $C_{\mathrm{B}}^i = C_{\mathrm{B}}^{i-1} + \gamma^{j_{\mathrm{B}}}(x^i - C_{\mathrm{B}}^{i-1})$, $S_{\mathrm{B}}^i = S_{\mathrm{B}}^{i-1} + \gamma^{j_{\mathrm{B}}}\left[\left(C_{\mathrm{B}}^i, x^i\right) - 2S_{\mathrm{B}}^{i-1}\right]$, $C_{\mathrm{A}}^i = C_{\mathrm{A}}^{i-1}$, $S_{\mathrm{A}}^i = S_{\mathrm{A}}^{i-1}$, $j_{\mathrm{B}} = j_{\mathrm{B}} + 1$ и перейти к пункту 3.

В пунктах 6 и 7 $\gamma^j = \frac{1}{2j}$.

Найденная функция имеет следующий вид:

$$f(c_0, c_1, c_2, c_3) = c_0 + 3c_1 - c_2 - 3c_3. \quad (2)$$

Как видно, критерий крайне быстро вычисляется — требуются лишь два умножения и три сложения.

Проверим эффективность критерия. Для случайно сгенерированной системы попытаемся предсказать её оптимальное разрешение, используя только что найденный критерий, а потом проверим правильность предсказания, проведя численный эксперимент и сравнив количество конфликтов на итерацию в случаях предпочтений одного или другого процесса. Результаты показаны в таблице.

 ISSN 1818-0744

Результаты проверки эффективности критерия (2)

№	Δ	№	Δ	№	Δ	№	Δ
1	0,00546	26	0,00952	51	0,00507	76	0,01654
2	0,06924	27	0,02132	52	0,0292	77	0,0114
3	0,00272	28	0,04244	53	0,01611	78	0,03135
4	0,06168	29	0,00202	54	0,01393	79	0,00799
5	0,00928	30	0,0988	55	0,00535	80	0,00328
6	0,07463	31	0,01816	56	0,03907	81	0,03244
7	0,02794	32	0,00655	57	0,00264	82	0,02439
8	0,00629	33	0,00785	58	0,00943	83	0,00446
9	0,0529	34	0,00512	59	0,01321	84	0,00549
10	0,02714	35	0,00467	60	0,04453	85	0,02444
11	0,00629	36	0,00264	61	0,00625	86	0,02952
12	0,07312	37	0,00503	62	0,01209	87	0,00628
13	0,00555	38	0,13575	63	0,00638	88	0,00223
14	0,06917	39	0,00171	64	0,01161	89	0,04015
15	0,00358	40	0,08311	65	0,0001	90	0,00985
16	0,00752	41	0,01191	66	0,00935	91	0,01495
17	0,01413	42	0,0297	67	0,00473	92	0,01875
18	0,0065	43	0,00441	68	0,00825	93	0,00852
19	0,01049	44	0,00354	69	0,08494	94	0,00905
20	0,02519	45	0,00666	70	0,01557	95	0,00219
21	0,02388	46	0,02326	71	0,00602	96	0,0363
22	0,01367	47	0,04656	72	0,00329	97	0,06301
23	0,00584	48	0,00647	73	0,04856	98	0,013
24	0,00573	49	0,02535	74	0,1182	99	0,0039
25	0,00375	50	0,03838	75	0,01273	100	0,00517
Сумма							2,20388
Процент							≈2,2
Процент неудачных предсказаний:							≈3,2

Примечание: № — номер неудачно предсказанного эксперимента;
Δ — разница числа конфликтов на итерацию.
Удачно предсказанные эксперименты в таблице не представлены.

Итак, при применении данного критерия неоптимальные решения получаются только примерно в 3 % случаев. Кроме того, конфликты при данных неоптимальных решениях случаются всего на 2 % чаще, что является вполне приемлемым результатом.

Заключение

Была рассмотрена задача оптимального разрешения конфликтов в управляемых марковских цепях, когда разным процессам одновременно требуется один и тот же неделимый ресурс. Существующие на настоящий момент точные методы не подходят для решения данный задачи в силу ограничений по скорости разрешения конфликтов. На примере простого случая с двумя общими марковскими процессами и двумя ресурсами (один неделимый) был продемонстрирован способ решения проблемы в виде поиска приближённого решения, которое не требует сложных расчётов, но достаточно точно. Предполагается продолжить исследования на системах больших размерностей. В частности, кроме приведенного алгоритма построения разделяющей плоскости предполагается рассмотреть метод Вапника, известный под названием *SVM* (*support vector machine*).

Не менее важно рассмотреть случай, когда матрицы переходных вероятностей неизвестны или известны не полностью. В этих случаях речь может идти об адаптивных правилах разрешения конфликтов. Вместо оптимальных в таких случаях рассматриваются так называемые целесообразные правила. По определению, целесообразными называются правила, которые приводят к результатам лучшим, чем при случайном выборе. В одной простой ситуации такое правило было найдено Генинсоном [2], однако в более общих случаях вопросы остаются открытыми.

Библиографический список

1. Браверман, Э. М. Структурные методы обработки эмпирических данных / Э.М. Браверман, И.М. Мучник. — Москва : Наука, 1983.
2. Генинсон Б.А. Разработка и исследование оптимальных правил разрешения конфликтов в многопроцессорных системах и алгоритмов их реализации : дисс. … канд. техн. наук. — Москва : Институт проблем управления, 1984.
3. Кельберт, М. Я. Вероятность и статистика в примерах и задачах. Т. 2. Марковские цепи как отправная точка теории случайных процессов и их приложения / М.Я. Кельберт, Ю.М. Сухов. — Москва : МЦНМО, 2009.
4. Рубчинский, А. А. Методы и модели принятия управленческих решений / А.А. Рубчинский. — Москва : Юрайт, 2015.
5. Ховард, Р. А. Динамическое программирование и марковские процессы / Р.А. Ховард. — Москва : Советское радио, 1964.

Поступила в редакцию
25.03.2016