УДК 519.21

Разрешение конфликтов в компьютерных системах

Можаров Г. П.^{1,*}

[*mojarov_g@mail.ru](mailto:mojarov_g@mail.ru)

¹МГТУ им. Н.Э. Баумана, Москва, Россия

Тупиковые ситуации, возникающие в сети при передаче данных препятствуют поступлению пакетов данных в узлы назначения. Поэтому проблема разрешения тупиковых ситуаций является одной из важнейших при проектировании передачи данных (особенно с буферизацией) в компьютерных сетях. Существуют два традиционных подхода к разрешению тупиковых ситуаций: проектируются сети и схемы, которые исключают возникновение тупиковых ситуаций; разработка распределённого алгоритма, который обнаруживает тупиковую ситуацию и затем выводит сеть из этой ситуации при помощи некоторого разрешающего распределённого алгоритма. Предложенный подход базируется на том, чтобы предоставлять неделимые ресурсы процессам так, чтобы минимизировать потери из-за конфликтов. Исследуются многокритериальная задача предоставления неделимых ресурсов процессам, причём принцип оптимальности выражается известным бинарным отношением на множестве средних векторов штрафов за конфликты по каждому из ресурсов. Показано, что совместное использование аппарата теории выбора и классического аппарата позволяет расширить известные постановки задач за счёт использования более общих принципов оптимальности; приводятся примеры решения многокритериальных задач оптимального управления разрешением конфликтов в компьютерных системах. Получены количественные оценки выигрыша при выбранной оптимальной стратегии разрешения конфликтов в многопроцессорных компьютерных системах.

Ключевые слова: компьютерная система, конфликты в компьютерных системах, многокритериальные задачи, оптимальная стратегия, тупики, предотвращения тупиковых ситуаций

Введение

Конфликтной ситуацией в компьютерных системах (КС) называется явление, возникающее при мультидоступе процессов к общим ресурсам, когда ни один из процессов, вовлеченных в эту ситуацию, не может продолжаться из-за ожидания определенных ресурсов, захваченных другими процессами, которые, в свою очередь,

находятся в аналогичном положении. Конфликтная ситуация имеет и иное название – тупик, что достаточно чётко отражается на состоянии КС.

Поиск практически применимых алгоритмов разрешения тупиковых ситуаций имеет большое прикладное значение для обеспечения информационной безопасности вычислительного процесса и в этой связи представленная статья посвящена решению важной и актуальной задаче.

Серьёзность ситуации зависит от типов процессов, находящихся в тупике, от типов используемых ресурсов, числа процессов и многих других факторов. Конфликты сопряжены с задержками процессов и снижают производительность КС. При этом, несмотря на относительную простоту данного явления, тупиковые ситуации ещё недостаточно изучены, а известные методы «борьбы» с тупиками не всегда эффективны, и, следовательно, требуют определённых исследований [1].

Например, недостаток метода предотвращения тупиковых ситуаций, используемого во многих современных операционных системах и основанного на предварительном планировании необходимых процессу ресурсов, очевиден – время ожидания может оказаться чрезмерно большим. Метод предотвращения с прерыванием работы процесса и освобождением его ресурсов очень специфичен и мало эффективен, когда имеется множество разнотипных ресурсов, запрашиваемых динамически. Недостаток еще одного метода, предотвращающего тупики путём упорядочения ресурсов, заключается в ограничении возможных последовательностей запросов на ресурсы [1-4].

Иной способ «борьбы» с тупиками – недопущение тупиковых ситуаций. Предполагается прогнозирование появления тупиковых ситуаций в будущем. Известные методы [1,2,4,5], основанные на определении и недопущении состояний, при которых могут возникать тупики. При этом используется предварительная информация о том, какие ресурсы может запросить процесс во время выполнения. Прежде чем выделить свободный ресурс процессу, осуществляется проверка на условие «безопасности» состояния. Состояние является «безопасным», если выделение ресурса процессу не может привести к появлению тупиковых ситуаций в будущем. Иначе состояние считается «опасным», и выделение ресурса откладывается. Один из вариантов данного метода предложен в [1]. Очевидный недостаток недопущения тупиковых ситуаций заключается в необходимости иметь априорную информацию о будущих потребностях в ресурсах, а это не всегда возможно.

Один из способов «борьбы» с тупиковыми ситуациями при отсутствии априорной информации о потребностях процесса в ресурсах – обнаружение тупиков. Обнаружение тупиковых ситуаций (еще не приводящее к их разрешению) – это периодическое использование алгоритма, который проверяет текущее распределение ресурсов с целью

указания того, существует ли тупиковая ситуация, и если существует, то какие процессы в нее вовлечены [3,5,6].

Цель работы состоит в том, чтобы разработать методы и алгоритмы, позволяющие минимизировать потери из-за тупиковых ситуаций в КС за счёт использования оптимальной стратегии разрешения конфликтов. Предлагаемый подход особенно эффективен для устранения тупиков в управляющих КС, набор программ которых фиксирован.

Математическое описание многокритериальной задачи нахождения оптимальных стратегий

При решении задач на КС несколько процессов могут одновременно потребовать одного и того же ресурса (процессора, канала, таблицы), который предоставляется только одному из них. Или, например, в процессе функционирования современных многопроцессорных КС могут возникать очереди запросов на выделение свободной памяти. Они образуются из-за конфликтов при обращении к общим данным (списку свободной памяти и т.д.), а также при реализации механизмов перераспределения и реорганизации памяти. Такие ситуации называются конфликтами, а ресурсы – неделимым.

Пусть имеется k типов ресурсов. В общем случае каждый тип i характеризуется объёмом ресурсов k_i . Под КС будем понимать множество процессов $P = \{p_1, p_2, \dots, p_m\}$, которые могут одновременно функционировать в КС и множество ресурсов $R = \{r_1, r_2, \dots, r_n\}$, которые процессы могут запрашивать и использовать.

Выделим основные типы неделимых ресурсов КС: информационные (R_1), программные (R_2), аппаратные (R_3). Примером ресурса типа R_1 является таблица распределения памяти в операционной системе, таблица распределения внешних устройств; примером ресурса R_2 – нереентерабельные программы: симплекс-метод, вычисление логарифма; аппаратные типа R_3 – диск, канал ввода-вывода, устройство печати.

Конфликты могут возникать многократно. Внутри каждого процесса можно выделить критические участки, где ему необходим неделимый ресурс. В каждый момент времени не более чем один процесс может использовать неделимый ресурс; при этом остальные процессы, претендующие на этот ресурс, временно блокируются и должны ждать, пока он не освободится [7,8,9].

Таким образом, задача состоит в том, чтобы предоставлять неделимые ресурсы процессам так, чтобы минимизировать потери из-за конфликтов. Порядок предоставления неделимых ресурсов процессам назовём стратегией S разрешения конфликтов [1,2].

Одним из возможных методов математического описания задачи разрешения конфликтов в многопроцессорной КС может являться применение марковских случайных процессов. Пусть i -й процесс в КС ($i = \overline{1, n}$) имеет $m+1$ состояний A_j^i ($j = \overline{0, m}$) (здесь $j = \overline{1, m}$ – номер неделимого ресурса). В отсутствие других процессов поведение i -го процесса ($i = \overline{1, n}$) опишем цепью Маркова с вероятностями переходов p_{qt}^i ($q = \overline{0, m}, t = \overline{0, m}$). Состояние A_0^i не соответствует ни одному из критических участков i -го процесса; состояние A_j^i ($j = \overline{0, m}$) соответствует j -му критическому участку. Множество процессов, находящихся в состоянии, соответствующем критическому участку относительно одного и того же неделимого ресурса, назовём группой конфликтующих процессов.

Пусть M_i – множество состояний i -го процесса. Состояние всей совокупности процессов в любой момент времени представляет собой точку множества $M = M_1 \cdot \dots \cdot M_n$. Рассмотрим подмножество $K_j \subset M$:

$$K_j = \left\{ s \in M \mid (\exists i \neq i') \left[s_i = A_j^i, s_{i'} = A_j^{i'} \right] \right\},$$

где $j \in \{1, \dots, m\}$. Подмножество K_j задаёт те состояния из M , в которых по крайней мере два процесса претендуют на j -й ресурс.

Введём величину

$$\chi(s, K_j) = \begin{cases} 1, & s \in K_j, \\ 0, & s \notin K_j, \end{cases} \quad (1)$$

и рассмотрим вектор

$$\chi = \langle \chi(s, K_1), \dots, \chi(s, K_m) \rangle, \quad \chi = \langle \chi(s, K_1), \dots, \chi(s, K_m) \rangle,$$

определяющий штрафы в состоянии $s \in M$ за конфликты по каждому из неделимых ресурсов. Штраф равен времени ожидания каждого из ресурсов.

Зададим бинарное отношение R на множестве средних векторов штрафов за конфликты по каждому из ресурсов за время функционирования КС (или векторов средних штрафов за единицу времени, если время функционирования КС бесконечно). Задача разрешения конфликтов состоит в нахождении R -оптимальной стратегии.

Установим соответствие между рассматриваемой задачей и задачей поиска R - оптимальных стратегий управления цепью Маркова. Построим цепь Маркова. Примем M за множество состояний цепи. Состояния $s \in M$ являются векторами: $s = \langle s_1, \dots, s_m \rangle$, где $s_i \in \{1, \dots, m\}$ – состояние i -го процесса в данный момент времени t ; $s_i = j$ означает, что i -му процессу в момент t требуется j -й ресурс. Пусть $n = 2, s_1(t) = j$ и $s_2(t) = j$. Тогда j -й ресурс может быть предоставлен только одному из процессов. Процессу, которому j -й ресурс в момент t не был предоставлен, тот же ресурс требуется и в $(t+1)$ -й момент. Поэтому переход из состояния $\langle i, j \rangle$ в состояние $\langle j', j'' \rangle$, где $j' \neq j, j'' \neq j$, невозможен. Таким образом, переходные вероятности зависят от того, какому процессу предоставлен j -й ресурс. Если предоставить ресурс 1-му процессу, то состояние $\langle j, j \rangle$ перейдёт в состояние $\langle j', j \rangle$ с вероятностью $p_{jj'}^1$ или в состояние $\langle j, j \rangle$ с вероятностью $1 - \sum_{j' \neq j} p_{jj'}^1 = p_{jj}^1$. Если предоставить ресурс 2-му процессу, то $\langle j, j \rangle$ перейдёт в $\langle j, j' \rangle$ с вероятностью $p_{jj'}^2$. Аналогично определяются переходные вероятности в общем случае n процессов ($n > 2$).

В данном случае удобнее говорить о минимизации штрафов, чем о максимизации доходов. Эквивалентность задач при этом очевидна. Вектор штрафа не зависит от принятого решения, а зависит только от состояния $s: r_s^k = r_s$; компоненты m -мерного вектора r_s определяются по формуле (1). Задача состоит в том, чтобы найти R - оптимальную стратегию для такой цепи. Таким образом, требуемое соответствие установлено. Это позволяет сводить исходную задачу к однокритериальной [3,9-11].

Пример решения задачи минимизации потерь процессорного времени из-за конфликтов

Рассмотрим следующие неделимые ресурсы: ρ_1 – канал ввода-вывода; ρ_2 – общая таблица; ρ_3 – устройство печати.

Пусть ресурсы ρ_1 и ρ_3 важнее, чем ресурс ρ_2 . Зададим критерии t_1, t_2 и t_3 , оценки по которым являются средними временами ожидания каждого ресурса. Зададим на множестве всех возможных векторов времён ожидания $\bar{t} = \langle \chi_1, \chi_2, \chi_3 \rangle$ иерархическое отношение T , учитывающее указанную неравноценность. Критерии t_1 и t_3 важнее, чем t_2 ; критерии t_1 и t_3 считаются равноценными. Логическая форма отношения T имеет вид

$$f_{123}(\beta_1, \beta_2, \beta_3) = \beta_1\beta_3, f_{12}(\beta_1, \beta_2) = \beta_1,$$

$$f_{13}(\beta_1, \beta_3) = \beta_1\beta_3, f_{23}(\beta_2, \beta_3) = \beta_3, f_i(\beta_i) = \beta_i \quad (i = \overline{1, 3}).$$

Мажорирующая последовательность $\langle i^\sigma \rangle(T)$ имеет вид $\langle i_1^{\sigma_1}, i_2^{\sigma_2}, i_3^{\sigma_3} \rangle = \langle 1^0, 2^0, 3^0 \rangle$.

Стратегия, минимизирующая среднее время ожидания ресурса ρ_1 , является T -оптимальной [5,6,10].

Найдём количественную оценку выигрыша в частном случае задачи нахождения оптимальной стратегии разрешения конфликтов. Пусть процессы в КС слабо связаны. При этом

$$p_{0j}^i = \varepsilon_j^i \quad (i = \overline{1, n}, j = \overline{1, m}), \quad p_{jk}^i = 0 \quad (j \neq k, j \neq \bar{0}, k \neq 0), \quad p_{ij}^i \in (0, 1) \quad (i = \overline{1, n}, j = \overline{1, m}).$$

Итак, пусть R – любое рациональное отношение. Существует $\varepsilon > 0$ такое, что при любых $\varepsilon_j^i < \varepsilon \quad (i = \overline{1, n}, j = \overline{1, m})$ для конечного и для бесконечного времени:

а) R -оптимальная стратегия единственна;

б) ресурс предоставляется процессу, имеющему меньшую вероятность p_{ij}^i остаться в состоянии A_j^i среди всех процессов данной группы.

Рассмотрим однокритериальный вариант задачи со слабосвязанными процессами.

Сравним среднее число конфликтов при данной стратегии со средним числом конфликтов при равновероятном выборе процесса, которому предоставляется ресурс. Значения параметров:

$$n = 2, m = 2, p_1^1 = 0,9, p_2^1 = 0,75, p_1^2 = 0,7, p_2^2 = 0,95;$$

при этом значения ε_1^1 меняются от 0,035 с шагом 0,04, значения ε_2^1 – от 0,04 с шагом 0,06, значения ε_1^2 – от 0,035 с шагом 0,03 и значения ε_2^2 – от 0,025 с шагом 0,04. В таблице приведены результаты расчётов. Строка соответствует фиксированному начальному состоянию (запись 12 означает, что первый процесс находится в состоянии A_1^1 , а второй – в состоянии A_2^2 ; аналогичный смысл имеют обозначения 00, 01 и т.д.).

Столбец соответствует времени функционирования КС T (числу шагов); на пересечении строки и столбца записано в верхней части клетки среднее число конфликтов при оптимальной стратегии, а в нижней – при стратегии, предоставляющей ресурс равновероятно тому или иному процессу. Таблица разбита на части, каждая из которых соответствует фиксированным значениям $\varepsilon_j^i \quad (i, j = \overline{1, 2})$.

Расчеты количественной оценки выигрыша в задаче нахождения оптимальной стратегии разрешения конфликтов (в приведенном примере) свидетельствуют об адекватности математического моделирования задачи разрешения конфликтов в многопроцессорной КС цепью Маркова.

Таблица. Результаты расчётов количественной оценки выигрыша в задаче нахождения оптимальной стратегии разрешения конфликтов

Начальное состояние	Число шагов				
	5	10	20	25	30
$\varepsilon_1^1 = 0,035; \varepsilon_2^1 = 0,04; \varepsilon_1^2 = 0,035; \varepsilon_2^2 = 0,25$					
00	0,04	0,193	0,662	0,929	1,207
	0,043	0,229	0,871	1,256	1,663
01	0,134	0,327	0,783	1,043	1,315
	0,150	0,404	1,041	1,414	1,809
11	2,832	3,559	4,226	4,507	4,788
	3,405	4,736	5,911	6,334	6,743
12	0,038	0,247	0,897	1,239	1,574
	0,040	0,290	1,188	1,693	2,197
22	3,119	4,170	5,215	6,610	5,974
	3,750	5,657	7,660	8,338	8,933
$\varepsilon_1^1 = 0,075; \varepsilon_2^1 = 0,1; \varepsilon_1^2 = 0,065; \varepsilon_2^2 = 0,065$					
00	0,170	0,700	2,060	2,781	3,514
	0,183	0,832	2,675	3,690	4,730
01	0,296	0,820	2,115	2,819	3,540
	0,329	0,994	2,753	3,737	4,754
11	2,888	3,823	5,140	5,815	6,516
	3,444	4,982	6,959	7,902	8,877
12	0,085	0,524	1,881	2,624	3,375
	0,090	0,612	2,424	3,467	4,535
22	3,213	4,644	6,571	7,399	8,194
	3,809	6,031	9,054	10,307	11,440
$\varepsilon_1^1 = 0,115; \varepsilon_2^1 = 0,160; \varepsilon_1^2 = 0,095; \varepsilon_2^2 = 0,105$					
00	0,345	1,237	3,274	4,310	5,350
	0,374	1,467	4,200	5,633	7,081
01	0,457	1,293	3,250	4,269	5,298
	0,506	1,555	4,175	5,575	7,000
11	2,937	4,060	5,894	6,858	7,855
	3,481	5,208	7,827	9,145	10,507
12	0,129	0,747	2,606	3,625	4,660
	0,136	0,867	3,299	4,694	6,129
22	3,298	5,005	7,515	8,624	9,705
	3,862	6,324	10,025	11,646	13,255
$\varepsilon_1^1 = 0,155; \varepsilon_2^1 = 0,220; \varepsilon_1^2 = 0,25; \varepsilon_2^2 = 0,145$					
00	0,535	1,713	4,214	5,457	6,698
	0,582	2,027	5,357	7,061	8,771
01	0,608	1,702	4,129	5,357	6,589
	0,673	2,035	5,256	6,929	8,617
11	2,984	4,263	6,473	7,634	8,828
	3,517	5,404	8,493	10,065	11,683
12	0,168	0,927	3,141	4,349	5,576
	0,178	1,070	4,929	5,559	7,137
22	3,373	5,284	8,184	9,486	10,760
	3,911	6,554	10,720	12,586	14,372

Заключение

Разработана и предложена эффективная стратегия управления информационными процессами в многопроцессорных КС, обнаруживающая и запрещающая тупиковые ситуации в компьютерных системах.

Отличительной особенностью предлагаемых эффективных методов и алгоритмов ликвидации тупиков является возможность использования их при разработке и эксплуатации многопроцессорных КС, работающих в реальном масштабе времени. Приведенный пример подтверждает работоспособность, эффективность и перспективность предлагаемых метода и алгоритма разрешения тупиковых ситуаций в многопроцессорных КС.

Предложенные метод и алгоритм обеспечивает уменьшение среднего числа конфликтов в КС на 30-40%.

Список литературы

1. Андреев А.М., Можаров Г.П., Сюезв В.В. Многопроцессорные вычислительные системы: теоретический анализ, математические модели и применение. М.: Изд-во МГТУ им. Н.Э. Баумана, 2011. 334 с.
2. Королёв В.Ю., Бенинг В.Е., Шоргин С.Я. Математические основы теории риска. 2-е изд., перераб. и доп. М.: ФИЗМАТЛИТ, 2011. 620 с.
3. Ногин В.Д. Принятие решений при многих критериях. СПб.: Изд-во «ЮТАС», 2007. 104 с.
4. Подиновский В.В., Ногин В.Д. Парето-оптимальные решения многокритериальных задач. 2-е изд. испр. и доп. М.: ФИЗМАТЛИТ, 2007. 256 с.
5. Ногин В.Д. Введение в оптимальное управление. СПб.: Изд-во «ЮТАС», 2008. 92 с.
6. Райгородский А.М. Вероятность и алгебра в комбинаторике. М.: МЦНМО, 2011. 48 с.
7. Райгородский А.М. Теория вероятностей и комбинаторная геометрия // Математика в задачах: сб. ст. М.: МЦНМО, 2009. С. 381-384.
8. Ширяев А.Н. Вероятностно-статистические методы в теории принятия решений. М.: ФМОП, МЦНМО, 2011. 144 с.
9. Eggemann N., Noble S.D. The clustering coefficient of a scale-free random graph // Discrete Applied Mathematics. 2011. Vol. 159, no. 3. P. 953-965. DOI: [10.1016/j.dam.2011.02.003](https://doi.org/10.1016/j.dam.2011.02.003)
10. Kolountzakis M.N., Miller G.L., Peng R., Tsourakakis Ch.E. Efficient Triangle Counting in Large Graphs via Degree-Based Vertex Partitioning // Internet Mathematics. 2011. Vol. 8, no. 1. P. 161-185.
11. Peskir G., Shiryaev A. Optimal stopping and free-boundary problems. Basel: Birkhäuser, 2006. 500 p.

Conflict Resolution in Computer Systems

G.P. Mojarov^{1,*}

[*mojarov_g@mail.ru](mailto:mojarov_g@mail.ru)

¹Bauman Moscow State Technical University, Moscow, Russia

Keywords: computer system, conflicts in computer systems, multicriterion problems, optimum strategy, deadlocks, prevention of impasses

A conflict situation in computer systems (CS) is the phenomenon arising when the processes have multi-access to the shared resources and none of the involved processes can proceed because of their waiting for the certain resources locked by the other processes which, in turn, are in a similar position. The conflict situation is also called a deadlock that has quite clear impact on the CS state.

To find the reduced to practice algorithms to resolve the impasses is of significant applied importance for ensuring information security of computing process and thereupon the presented article is aimed at solving a relevant problem.

The gravity of situation depends on the types of processes in a deadlock, types of used resources, number of processes, and a lot of other factors.

A disadvantage of the method for preventing the impasses used in many modern operating systems and based on the preliminary planning resources required for the process is obvious - waiting time can be overlong. The preventing method with the process interruption and de-allocation of its resources is very specific and a little effective, when there is a set of the polytypic resources requested dynamically. The drawback of another method, to prevent a deadlock by ordering resources, consists in restriction of possible sequences of resource requests.

A different way of "struggle" against deadlocks is a prevention of impasses. In the future a prediction of appearing impasses is supposed. There are known methods [1,4,5] to define and prevent conditions under which deadlocks may occur. Thus the preliminary information on what resources a running process can request is used. Before allocating a free resource to the process, a test for a state "safety" condition is provided. The state is "safe" if in the future impasses cannot occur as a result of resource allocation to the process. Otherwise the state is considered to be "hazardous", and resource allocation is postponed. The obvious shortcoming in preventing impasses is a need to have a priori information on the future demand for resources, and it is not always possible.

One of ways to "struggle" against impasses when there is no a priori information on the process demand for resources is to detect deadlocks. Detection of impasses (without leading to

their resolution yet) is a periodical use of the algorithm which checks current distribution of resources to reveal whether there is an impasse and if it exists what processes are involved in it.

The work objective is to develop methods and algorithms allowing us to minimize losses because of impasses in CS using the optimum strategy of conflict resolution. The offered approach is especially effective to eliminate deadlocks in management (control) computer systems having a fixed set of programmes.

The article offers a developed efficient strategy of the information processes management in multiprocessing CS, which detects and prevents impasses. The strategy is based on allocation of indivisible resources to computing processes so that losses caused by conflicts are minimized. The article studies a multi-criterion problem of indivisible resources allocation to the processes, with the optimality principle expressed by the known binary relation over set of average vectors of penalties for conflicts in each of resources. It is shown that sharing a decision theory tool and a classical one allows more efficient problem solution to eliminate deadlock. The feature of suggesting effective methods and algorithms to eliminate deadlocks is that they can be used in CS development and operation in real time. The article-given example shows that the proposed method and algorithm for the impasse resolution in multiprocessing CS are capable and promising.

The offered method and algorithm provide reducing the average number of CS conflicts by 30-40 %.

References

1. Andreev A.M., Mozharov G.P., Syuzev V.V. *Mnogoprotsessornye vychislitel'nye sistemy: teoreticheskii analiz, matematicheskie modeli i primenenie* [Multiprocessor computer systems: theoretical analysis, mathematical models and application]. Moscow, Bauman MSTU Publ., 2011. 334 p. (in Russian).
2. Korolev V.Yu., Bening V.E., Shorgin S.Ya. *Matematicheskie osnovy teorii riska* [Mathematical foundations of risk theory]. Moscow, Fizmatlit Publ., 2011. 620 p. (in Russian).
3. Nogin V.D. *Prinyatie reshenii pri mnogikh kriteriyakh* [Mathematical foundations of risk theory]. St. Petersburg, YuTAS Publ., 2007. 104 p. (in Russian).
4. Podinovskii V.V., Nogin V.D. *Pareto-optimal'nye resheniya mnogokriterial'nykh zadach* [Pareto-optimal solutions of multicriteria problems]. Moscow, Fizmatlit Publ., 2007. 256 p. (in Russian).
5. Nogin V.D. *Vvedenie v optimal'noe upravlenie* [Introduction to optimal control]. St. Petersburg, YuTAS Publ., 2008. 92 p. (in Russian).
6. Raigorodskii A.M. *Veroyatnost' i algebra v kombinatorike* [Probability and algebra in combinatorics]. Moscow, MTsNMO Publ., 2011. 48 p. (in Russian).
7. Raigorodskii A.M. Probability theory and combinatorial geometry. *Matematika v zadachakh: sb. st.* [Mathematics in problems: collected articles]. Moscow, MTsNMO Publ., 2009, pp. 381-384. (in Russian).

8. Shiryaev A.N. *Veroyatnostno-statisticheskie metody v teorii prinyatiya reshenii* [Probabilistic and statistical methods in decision making theory]. Moscow, FMOP Publ., MTsNMO Publ., 2011. 144 p. (in Russian).
9. Eggemann N., Noble S.D. The clustering coefficient of a scale-free random graph. *Discrete Applied Mathematics*, 2011, vol. 159, no. 3, pp. 953-965. DOI: [10.1016/j.dam.2011.02.003](https://doi.org/10.1016/j.dam.2011.02.003)
10. Kolountzakis M.N., Miller G.L., Peng R., Tsourakakis Ch.E. Efficient Triangle Counting in Large Graphs via Degree-Based Vertex Partitioning. *Internet Mathematics*, 2011, vol. 8, no. 1, pp. 161-185.
11. Peskir G., Shiryaev A. *Optimal stopping and free-boundary problems*. Basel, Birkhäuser, 2006. 500 p.