АНАЛИЗ И ПРОБЛЕМЫ АЛГОРИТМОВ ДИАГНОСТИРОВАНИЯ БЛОКИРОВОК В РАСПРЕДЕЛЕННЫХ СИСТЕМАХ

Андрюхин А.И., Ковалева Т.В.
Кафедра ПМиИ, ДонНТУ
andr@r5.dgtu.donetsk.ua

Abstract

Andruckin A.I., Kovaleva T.V. Analysis and problems of algorithms of deadlock diagnosis in distributed systems. *The rewiew of the of deadlock diagnosis in distributed systems is submitted.*

Введение

При параллельном исполнении процессов могут возникать такие ситуации, при которых два и более процесса всё время находятся в состоянии блокировки. О таких процессах говорят, что они находятся в состоянии взаимной блокировки, тупика, дедлока (deadlock) или клинча (clinch) ("смертельного объятия" - deadly embrace). С проблемой тупиков тесно связана проблема бесконечного откладывания, когда процесс, даже не находящийся в состоянии тупика, ожидает события, которое может никогда не произойти из-за "необъективных" принципов, заложенных в системе планирования ресурсов[1-3].

Теоретической моделью при анализе блокировок в распределенных системах является направленный граф распределения (ожидания) ресурсов-WFG, вершинами которого являются процессы и ресурсы, а ребра графа определяют запрос конкретным процессом определенного ресурса. Заметим, что в литературе по распределенным базам данных он именуется графом ожидания транзакций (TWG), где транзакция понимается как последовательность запросов, выполняющих чтение или запись объектов данных. На рис.1 представлен простейший случай deadlock-ситуации, когда P1, P2-процессы, R1, R2-ресурсы, ребра графа распределения ресурсов P2→R1, P1→R2 интерпретируем как запросы ресурсов процессами, а ребра R1→P1, R2→P2 определяем, как захват (распределения) ресурсов процессами. Ясно, что если в графе существует ориентированный цикл, в котором за ребром захвата ресурса следует ребро запроса ресурса, то система находится в deadlock-состоянии.

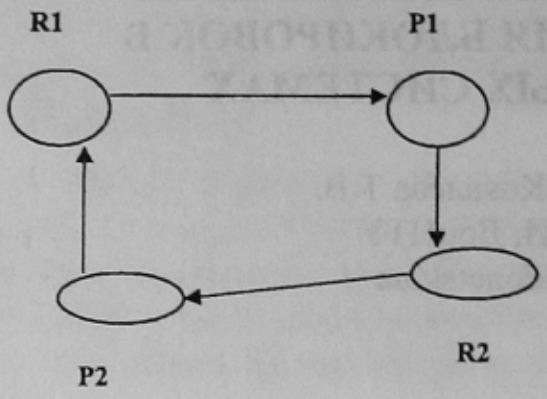

Рисунок 1 - Пример простейшей deadlock-ситуации

Существуют 4 вида отношения к deadlock- проблеме такие как, игнорирование ее, определение deadlock-ситуации, профилактика и избегание deadlock-ситуации.

Основные положения и обзор предыдущих работ

В распределенные базах данных возможны запросы различного вида. К примеру, транзакции могут требовать комбинации ресурса A и (AND) ресурса B или(OR) ресурса C. В [2] рассматриваются модели запросов AND, OR, AND-OR, C(n,k). В модели AND транзакция разрешена при наличии набора свободных ресурсов и она блокирована до тех пор, пока не доступны все ресурсы, которые ею запрошены. В OR-модели запрос множества ресурсов считается удовлетворенным, если гарантирован доступ к какому-либо ресурсу множества (так при требовании чтения множества копий данных нам достаточно возможности чтения какой-либо копии). Модели вида AND-OR, C(n,k) являются расширениями двух предыдущих моделей AND и OR. В наиболее общей модели C(n,k) (в [2] обозначается как $\binom{n}{k}$) транзакция разрешена при условии получения любых k ресурсов из набора ресурсов размерности n. Ясно, что запросы вида AND (OR) для n ресурсов описываются моделью C(n,n) (C(n,1)) соответственно.

Согласно принятой классификации алгоритмы определения и разрешения deadlock можно условно разделить на три группы: алгоритмы централизованного, распределенного и иерархического управления [2-4].

Для централизованных алгоритмов необходим глобальный граф ожиданий всей сети, что является практически невозможным в настоящее время ввиду размерностей реальных сетей. Имеются проблемы для такого подхода, как большая нагрузка на центр обработки собираемой информации по определению циклов в глобальном графе, большая вероятность определения фальшивой deadlock-ситуации и др.

Представителями централизованных алгоритмов являются однофазный и двуфазный Ho-Ramamoorhy-алгоритмы [5].

Поле распределенных алгоритмы ввиду их практической значимости наиболее исследуемо и нем выделяют четыре типа [2-4].

1. Алгоритмы проталкивания пути (Path-pushing), в которых передается информация о необходимом пути продвижения от узлов ожидания к блокированным узлам. Представителями этого типа являются Obermarck's алгоритм [6], Menasce-Muntz -алгоритм[7], Badal- алгоритм [8].

2. Алгоритмы прогонки (Edge-chasing), в которых определение циклов инициируется узлом, который не может выполнять операции из-за блокировки своих действий и который генерирует пробные сообщения. Эти сообщения изменяют характеристики заблокированных узлов при своем продвижении. Представителями этого типа являются Mitchell-Meritt-алгоритм [9], Chandy-Misra-Haas- алгоритм (AND-модель)[10].

3. Диффузионные алгоритмы, в которых инициация диффузионных вычислений определения циклов выполняются незаблокированными процессами и используются более сложные модели deadlock, нежели в алгоритмах прогонки. Представителями этого типа являются Chandy-Misra-Haas алгоритм (OR-модель)[[10], Hermann- Chandy -алгоритм[11].

4. Алгоритмы определения глобального состояния распределенной сети. В них выполняется попытка получения дампа глобального WFG. Представителями этого типа являются Kshemkalyani-Singhal- алгоритм [12], Bracha-Toueg -алгоритм[13].

Рассмотрим кратко наиболее известные распределенные алгоритмы, в частности Chandy-Misra-Haas-алгоритм, который оперирует с сообщениями специального вида, называемыми зондами (probe).

Они представляют собой тройки целых чисел (I,J,K) [10]. Здесь I-номер процесса, который инициировал определение deadlock и зонд передается со страницы J на страницу K.

Зондовые сообщения передаются по ребрам глобального графа ожиданий и deadlock обнаруживается, когда зондовое сообщение возвращается к инициирующему его процессу. Пример работы показан на рис. 2.

Для представителя иерархических алгоритмов Ho-Ramamoorhy-алгоритма[5] процессы группируются в независимые кластеры. Периодически определяется центральный контрольный кластер, который динамически выбирает управляющий центр для каждого кластера.

Схема его работы представлена на рис. 3.

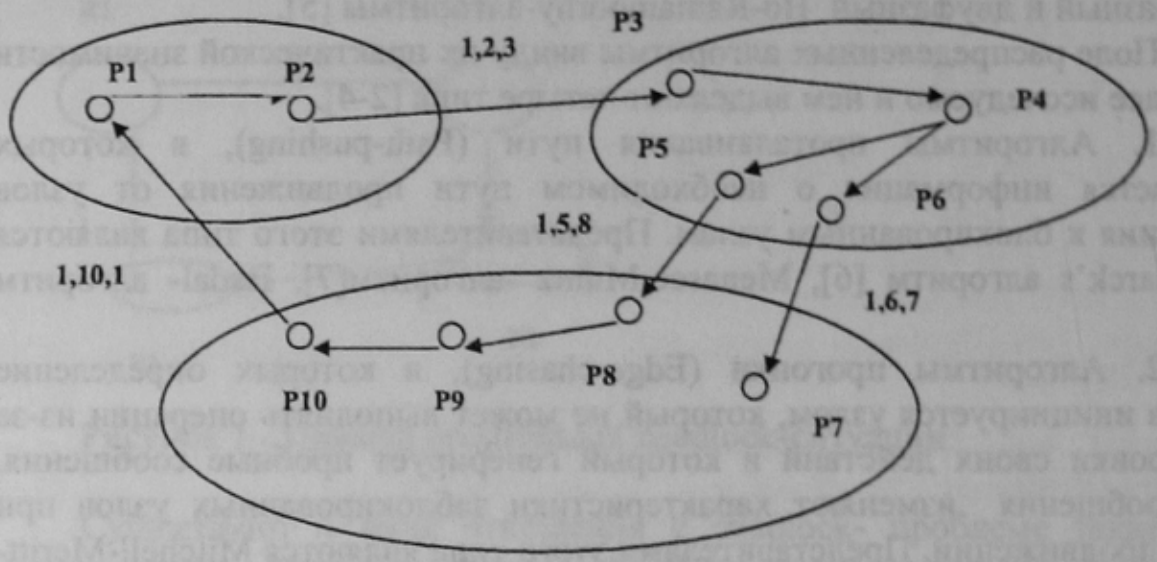

Рисунок 2. - Пример работы Chandy-Misra-Haas-алгоритм (AND-модель).

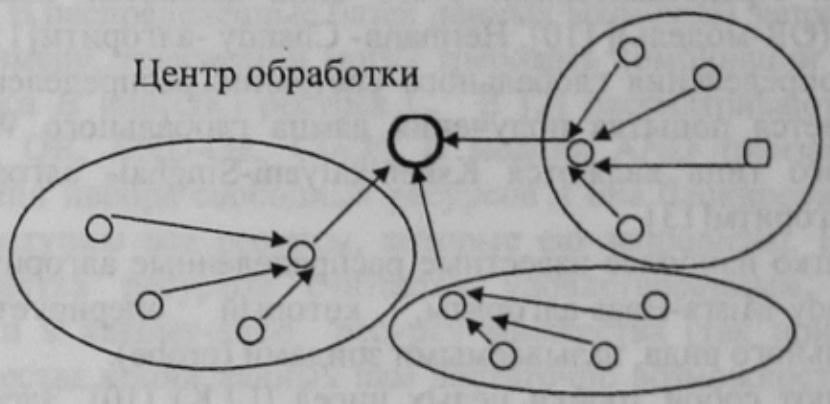

Рисунок 3 - Схема работы Ho-Ramamoorhy -алгоритма.

Определим следующие параметры для характеристик алгоритмов:

- N-число областей,
- n(m)-число областей (процессов) участвующих в deadlock-ситуации;
- T-коммуникативная задержка при взаимодействии областей;
- p - период между двумя модификациями глобального графа ожиданий;
- d - диаметр глобального графа ожиданий.

Размер(длину) сообщения определяем параметрами V - переменная, C - константная, L - большая, M -, S - малый, B - наилучший случай, W - наихудший случай. Согласно [3,4, 14] получаем таблицу 1.

Таблица 1. Характеристики алгоритмов диагностирования блокировок

Алгоритм	Число сообщений	Задержка	Размер сообщений	Сложность запросов
I.Централизованные Алгоритмы				
Ho-Ramamoorhy (двуфазный)	4N	B: 4T W:p+4NT	V,L	And
Ho-Ramamoorhy (однофазный)	2N	B: 2T W:p+2NT	V,L	And
II.Распределенные алгоритмы				
1.Проталкивание пути				
Menasce-Muntz	m(n-1)	nT	V,S	And
Obermarck	m(n-1)/2	nT	V,M	And
Badal (1)	M-1	(M-1)T	V,M	And
Badal(2)	n-1	(n-1)T	V,M	And
2.				
Mitchell-Merritz (без приоритета)	m(n-1)/2	(n-1)T/2	C,S	Одиночный ресурс
Mitchell-Merritz (с приоритетом)	m(n-1)	(n-1)T	C,S	Одиночный ресурс
3.Диффузионные алгоритмы				
Chandy-Misra-Haas	W:2m(n-1)	2dT	C,S	Or
Hermann- Chandy	$W:2m^2(n-1)$	2dT	V,M	And-Or
4.Алгоритмы глобального состояния				
Bracha-Toueg	4m(N-1)	4dT	V,M	C(n,k)

Проблемы

Основными проблемами для предлагаемых алгоритмов определения и разрешения deadlock являются

1) определение их корректности;
2) высокая эффективность определения deadlock;
3) высокая эффективность разрешения deadlock;
4) фальшивые deadlock-ситуации.

Для корректности алгоритма определения и разрешения блокировок, необходимо, чтобы он удовлетворял двум критериям:

А) алгоритм должен определять все существующие deadlock-ситуации за конечное время;

Б) алгоритм не должен генерировать сообщения о несуществующих deadlock-ситуации.

Постановка задачи

В распределенных системах, где нет единой глобальной памяти и коммуникация осуществляется путем сообщений, трудно построить корректный алгоритм определения deadlock, так как динамика процессов приводит к ситуации, когда мы определяем цикл блокировок в построенном нами, но несуществующем в реальности графе ожиданий, хотя его различные части существовали в различное время.

Отдельно стоит задача определения вероятности появления deadlock, которая зависит от многих факторов, таких как состава множества процессов, среднего числа объектов используемых процессами, времени использования ресурса и т.п. Получены такие качественные результаты, что ожидание транзакций и deadlock-ситуации являются редкими событиями, но вероятность их появления линейно зависит от количесгва процессов, большинство deadlock-ситуаций имеют длину два (два процесса блокируют друг друга), увеличение числа deadlock-ситуаций (ожиданий) пропорционально четвертой (второй) степени размера транзакций соответственно. Эти результаты подтверждены статистическими экспериментами авторов в комплексе SimDeadlock, целью реализации которого является прогноз deadlock-ситуации в реальных сетях с различными характеристиками.

Выводы

Нет ясных победителей среди алгоритмов: Mitchell-Merritt-алгоритм превосходит своей элегантностью и простотой, Chandy-Misra- Haas алгоритм своим доказательством корректности, Hermann-Chandy-алгоритм своей идеей и техникой распараллеливанием вычислений на нескольких уровнях иерархии.

Каждый из этих алгоритмов имеют достоинства, но многие из них достигают простоты разрешения deadlock-ситуации, сильно ограничивая формы запросов ресурсов. Согласно [2] различия в сложности алгоритмов оказались менее значимыми, чем ожидалось. Различные модели запросов имеют неблагоприятную сложность порядка $O(N^2)$ или $O(N^3)$, где N - количеством узлов в WFG. Поэтому выбор схемы или алгоритма обнаружения тупика, который нужно применять в конкретном приложении требует дополнительного рассмотрения.

Вероятность появления deadlock-ситуации является важным параметром и дальнейшая работа авторов имеет целью определение этой величины для графов, имеющих статистические характеристика сети Internet.

Литература

1. Т.Бройнль. Паралельне програмування. К., Выща школа, 1997, 358 с.

2. E. Knapp. Deadlock detection in distributed databases.*ACM Computing Surveys*, 19(4):303–328, December 1987.

3. M. Singhal. Deadlock detection in distributed systems.*IEEE Computer*, 22(11):37–48, November 1989.

4. A. D. Kshemkalyani and M. Singhal. Distributed detection of generalized deadlocks. In *Proc. of the 17th Intl.Conf. on Distributed Computing System*, pages 553–560, 1997.

5.Ho, G. S., and Ramamoorthy, C. V. Protocols for deadlock detection in distributed database systems. IEEE Trans. Softw. Eng. SE-1982.8, 6 (Nov.), 554-557.

6. R. Obermarck. Distributed deadlock detection algorithm.*ACM Trans. on Database Systems*, 7(2):187–208, June 1982.

7. D A. Menasce and R. R. Muntz. Locking and deadlock detection in distributed data bases. *IEEE Trans.Software Eng.*, 5(3):195–202, May 1979.

8. D. Z. Badal. The distributed deadlock detection algorithm.*ACM Trans. Comp. Syst.*, 4(4):320–337, November 1986.

10. K. M. Chandy, J. Misra, and L. M. Haas. Distributed deadlock detection. *ACM Trans. Comp. Syst.*, 1(2):141–156, May 1983.

11. Hermann, T., and Chandy, K. M. 1983. A distributed procedure to detect AND/OR deadlock. Tech. Rep. TR LCS-8301, Dept. of Computer Sciences, Univ. of Texas, Austin, Tex.

12. Mitchell, D. P., and Merritt, M. J. 1984. A distributed algorithm for deadlock detection and resolution. In-Proceedings of the ACM Symposium on Principles of Distributed Computing. ACM, New York, pp. 282-284.

13. G. Bracha and S. Toueg. Distributed deadlock detection.*Distributed Computing*, 2:127–138, 1987.

14. M.Samadi. Survey of Deadlock Detection In Distributed Operating. Systems. http://mehr.sharif.edu/~msamadi

---

Дата надходження до редакції 12.06.2005 р.