Journal scientific and applied research, vol. 4, 2013
Association Scientific and Applied Research
International Journal

Original Contribution ISSN 1314-6289

MODELLING OF CENTRALIZED TWO-PHASE LOCKING WITH INTEGRATED MECHANISM OF TIMESTAMPS BY THE "WAIT – DIE" METHOD

Aleksandar Milev, Svetlana Vasileva

KONSTANTIN PRESLAVSKI UNIVERSITY OF SHUMEN, SHUMEN 9712, 115, UNIVERSITETSKA STR.
e-mail: alex_milev@yahoo.com, svetlanaeli@abv.bg

Abstract. *This paper presents an algorithm for two-phase locking (2PL) in which deadlocks of distributed transactions for distributed database management systems (DDBMS) are avoided. The method of timestamps is chosen for solving deadlocks and the centralized 2PL algorithm is implemented in DDBMS. The „wait - die" strategy of timestamps mechanism for deadlocks avoiding is presented in this paper. The simulation results of modeling "wait-die" algorithm are given by using GPSS World Personal Version for two and three elements length of distributed transaction.*

Keywords: *distributed databases, distributed transactions, concurrency control, centralized two-phase locking, deadlock, timestamp ordering*

I. Introduction.

The concurrency control algorithms in the database management systems (DBMS) are considered as pessimistic (using a lock mechanism), optimistic (with deadlock avoidance) and their hybrid solutions. According to many authors [1-5], [7] and [10] two-phase locking – 2PL methods are more effective when the system is saturated by conflicts.

The main disadvantage of pessimistic method is the possibility of deadlocks between transactions [1], [2], [4], [5] and [7]. The deadlocks (DL) should be discovered and rejected or avoided in that case [5]. The first strategy relies on discovering and permission of DL. The basic method used in that case is creation and supporting of Wait-For Graph (WFG) transactions. The second strategy could be implemented by time-out and timestamps (TS) methods.

There are two advantages of timestamp ordering method. It is easier for implementation and deadlocks are not permitted [1], [6]. Moreover, the method does not allow cycle restart of rejected transaction due to retaining of time stamp and concurrency resource missing [1].

Two algorithms of deadlock avoidance could be used when timestamp ordering is implemented [1] and [7]:

-„Wait – die" method

In that case when a resource conflict occurs if a transaction Ti is "older" than a transaction holding the element locking Tj, (TS(Ti)<TS(Tj)), then Ti waits its releasing, and if Ti is “younger” than Tj (TS(Ti) > TS(Tj)), transactionTi is restarting.

- „Wound – wait” method

In that case when a resource conflict occurs if requesting for it transaction Ti is “older” than retaining transaction Tj, then Ti “wounds” Tj [1]. The “wound” is usually “deadly”. If the transaction Tj is not finished at the moment of “wound” it restarts. In that case the transaction Tj “becomes alive” and a rollback is not performed.

If the transaction Ti is “younger” than transaction Tj, then transaction Ti is given permission for waiting locking state.

The TS ordering algorithm is used by lock manager for handling transaction Ti which requests locking for the element X according to the „wound – wait” [9].

The methods for lock handling are not quite simple for implementation in distributed DBMS (DDBMS) as compared to centralized DBMS. Moreover, the algorithms for discovering and resolving of deadlocks require system resources.

Two cases are possible for the period between the algorithms for discovering and resolving of deadlocks.

If this period is very long then “deadlock” transactions could wait too much until deadlock has been discovered. In that case “the victim” will restart. Therefore, average system response time is increasing.

In the case of too short starting period it’s not necessary to use many system resources which have to perform checking for not rising up deadlock.

The deadlock handling in distributed databases (DDB) [4, pp.878-879], [7] use the method of creation and maintaining of WFG. It leads to restart of global transactions - „victims” of deadlocks.

Resolving the deadlock problem could be done by using timestamps ordering or time out [1], [6], [7], [8].

Among the all 2PL protocols used in DDBMS like Centralized *2PL*, Primary copy *2PL*, Distributed *2PL* and Voting *2PL* most suitable for implementation of TS ordering method is centralized 2PL protocol [1], [2], [5] and [7].

In this paper we suggest a model of algorithm for two-phase locking in DDBMS with integrated mechanism of timestamps use timestamp method for deadlock avoidance. In our algorithms the *„wait-die*” method is considered.

II. Materials

A model of Centralized 2PL protocol with integrated timestamps mechanism

In our model we consider 6 flows of distributed transactions. The modeling algorithm is constructed according to diagrams in [9] for distribute transaction with 2PL protocol in DDBMS.

Our model generates transactions which handle 1, 2 or 3 data elements. In that case the possibility for longer transaction is great than the possibility for transaction which handles 1 element. Every data element has two copies. Diagram of modeling algorithm of DDBMS with centralized 2PL protocol and with TS mechanism is given at fig.1.

Parameters of the GPSS transactions

P1 – Number of transaction. The value is a sum of System Numeric Attribute MP2 (The subtraction between the relative model time and the content of the second parameter of GPSS transaction) and the number of the site;

P2 – Number of the site, where the transaction is generated. The value is a number from 1 to <number of stream transactions>;

P$Nel – Length of the modelled transaction. The value of that parameter in the constructed models is 1 or 2 or 3 chosen by probability defined by the function FN$BrEl respectively 0.25, 0.35 and 0.40. It is supposed that long transactions get in the system more frequently then short ones in that model;

P$El1 – Number of the first element, which the generated transaction will read or write. The value is a random number and is uniformly distributed in the interval [1, NumEl];

P$El2 (P$El3) – Number of the second (third) element, which will be processed by the generated transaction;

P3(P4,P5) – Type of the requested lock for the first (second, third) element, which will be processed by the generated transaction;

P6 – Value 0, if the transaction is in 1st phase – occupation of the locks and value 1, if the transaction finishes its work and has to release the locks;

P$CHTN1, P$CHTS1 – In the situation when P3 = 1 the transaction only "reads" the element with number P$El1. This is possible if the element is not free and the lock is permissible;

P$CHTN2, P$CHTS2 – In the situation when P4 = 1 the transaction only "reads" the element with number P$El2. This is possible if the element is not free and the lock is permissible;

P$CHTN3, P$CHTS3 – In the situation when P5 = 1 the transaction only "reads" the element with number P$El3. This is possible if the element is not free and the lock is permissible;

P7, P8, P9, P10 – In them there are correspondingly recorded the number of the site, where it is the nearest copy of the data element and the number of the site, where it is the second replica of the first data element, processed by transaction.

P11, P12 – In them there are correspondingly recorded the number of the site, where it is the nearest copy of the data element and the number of the site, where it is the second replica of the third data element, processed by transaction.

The basic steps in the synthesized Centralized 2PL with TS (method "wait-die") algorithm are as follows:

1. When the transaction T_{P2}^{P1} comes in the transaction manager TM_{P2} its length is checked (1 or 2 or 3 data elements will be processed) - operation 1 on fig.1 and the

transaction is prepared to be split (operations 11 on fig. 1).

2. With the operations 12 values of the parameters of the sub-transactions are acquired – the numbers of the data managers DM$_{P7}$, (DM$_{P8}$), (DM$_{P9}$ and DM$_{P10}$), (DM$_{P11}$ and DM$_{P12}$), where the sub-transactions $T_{P2,P7}^{P1}$, ($T_{P2,P8}^{P1}$), ($T_{P2,P9}^{P1}$ and $T_{P2,P10}^{P1}$), ($T_{P2,P11}^{P1}$ and $T_{P2,P12}^{P1}$) have to execute the operations of reading/recording of the copies of data elements El1, El2 and El3.

3. After the primary processing in the transaction coordinator TC$_{P2}$ the requests for locking El1, El2 and El3 are transmitted through the net to the central lock manager LM$_0$ (operations 2, 5 and 8 on fig. 1).

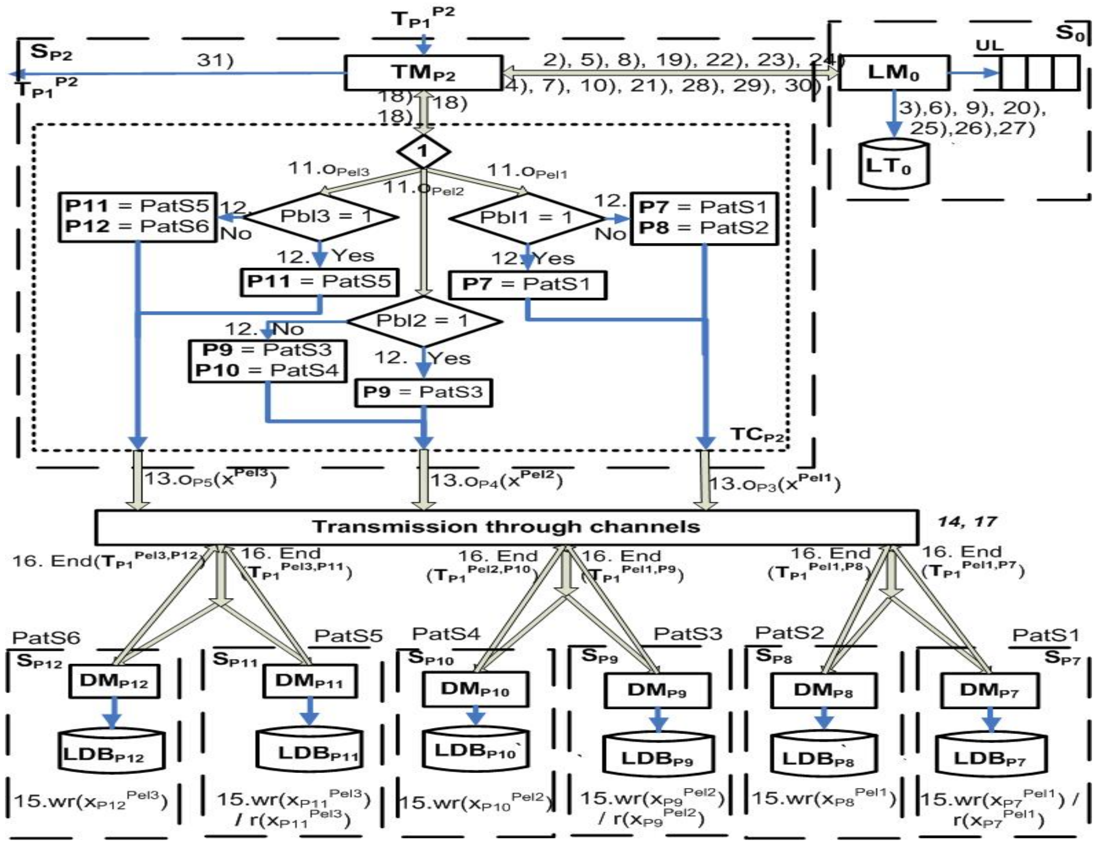

Fig. 1. *Simulation model на обслужване на distributed transaction by Centralized 2PL protocol with integrated timestamp ordering*

4. LM$_0$ checks in the lock table LT$_0$ if the lock of El1, El2 and El3 is allowed (operations 3, 6 and 9 on fig.1). If the lock of El1 (and El2 (and El3)) is allowed, the corresponding record is put opposite the number of the element in the lock table LT$_0$.

5. The transaction receives confirmation messages about the lock of El1 (operation 4) and if two data elements are being processed, TM$_{P2}$, through the transaction coordinator TC$_{P2}$ sends the request for lock of El2 to LM$_0$ (operation 5). And if three

data elements are being processed, TM_{P2}, through the TC_{P2} sends the request for lock of El3 to LM_0 (operation 8).

6. If the lock of the corresponding element is not possible, the number of the transaction is check if it is smaller than the number of the transaction which has put the lock:

- if the sub-transaction is not going to continue and is not going to restart, it waits the release of the element in user chain, whose number is the number of the element;

- if the sub-transaction has not received the lock of the element it restarts (operations 4, 7, 10 are restart operations). After it has arrived in TM_{P2}, the restarted lock request (operation 19) is transmitted to LM_0 (the repeated (successful) by operations 19, 20 and 21)

7. Transaction which has finished with the operation read/write releases the element in LT_0 – operations 25, 26 and 27 on fig.1. The requests for release of the lock of the elements are transmitted to the lock manager with operations 22, 23 and 24.

8. After the release of the lock of an element, the transaction which is first in the waiting list heads to the lock manager. If it is a group of sub-transactions then they receive a shared lock of the element.

9. Receiving a confirmation for a lock of the elements of the GPSS transaction being allowed, a modeling global transaction splits. After that the sub-transactions are transmitted through the net to the data managers for executing the read/write operations (operations 13 and 14 on fig.1).

10. The sub-transactions of T_{P1}^{P2} execute read/write in local databases LDB_{P7}, LDB_{P8}, LDB_{P9} and LDB_{P10}, LDB_{P11} and LDB_{P12} with the corresponding replicas of El1, El2 and El3 (operations 15 on fig.1). After that they are transmitted to the transaction manager TM_{P2} (operations 16 and 17). If a transaction renews a data element, the sub-transactions recording the corresponding copies wait for each other and get united (operations 18), before a request for release of the lock of the element is sent to LM_0.

11. Transaction T_{P1}^{P2} quits the system (operation 31 on fig. 1) as soon as sub-transactions T_{P1}^{Pel1} and T_{P1}^{Pel2} and T_{P1}^{Pel3} finish their process (modeled with operations 28, 29 and 30 and shown in fig.1).

The traffic and its transferring through the network to the central lock manager LM_0 and to the sites-executors, where are the data managers are simulated with retention.

III. Simulation Results

The parameters and indexes of the simulations of the considered model are as follows: *NumTr* – general number of the generated transactions for the time of incoming modeling; *FixTr* – general number of the completed (committed) transactions for the same period; *X=FixTr/Tn* – throughput of the queuing system; Tn – time interval in which the system is being watched; *Ps=FixTr/NumTr* – probability for transaction service. The results are received in 6 streams of concurrent transactions with different intensity. The copies of the data elements are distributed evenly and random by 6 sites in the system.

The results for throughput of algorithm simulation for centralized 2PL with TS (method "wait-die") are given at fig.2.

In that simulation the following is considered:

- equal intensity of input flows depending on period of observation (в seconds):

- minimum loading with intensity of summary stream 25 tr/s - 6 flows with average intensity for every one 4,17 tr/s;

- average loading with intensity of summary stream 50 tr/s - 6 flows with average intensity for every one 8,33 tr/s;

- max loading with intensity of summary stream 100 tr/s - 6 flows with average intensity for every one 16,67 tr/s.

The graphics in fig.3 show the results of probability service for simulation of centralized 2PL with TS (method "wait-die") algorithm in the same intensities of the incoming streams of global transactions (as the graphic shown in fig.2).

The graphics in fig.2 and fig.3 represent the processing in the model for different loads: min, average and max. The graphics in fig.3 show, that the probability service has the same behavior when the monitoring time is increased and the intensity of input streams accepts three different values. Moreover, the probability service reaches the max possible value when the static mode is used.

The results of our simulations of the model for equal intensity of 6 input flows for 2 (element copies are summarized and presented graphically in fig.4. The results shown in fig.4 are under the same initial conditions as those in fig.2 and fig.3, The difference is that probability service transaction accessed 1 element is 0.30 and probability service transaction accessed 2 elements is 0.70.

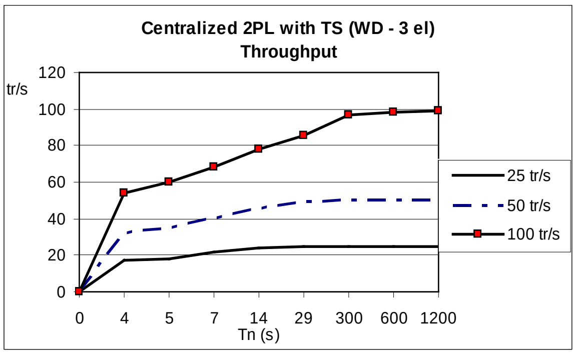

Fig. 2. *Throughput of the model in one and the same intensities of the incoming streams*

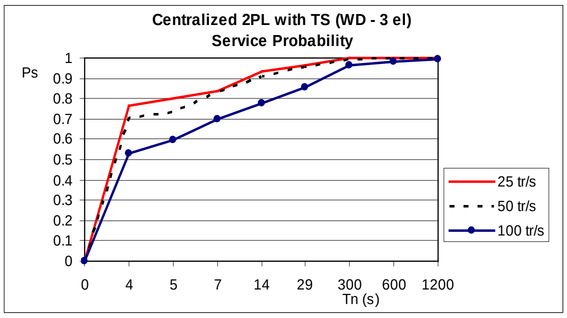

Fig. 3. *Probability service of the model in one and the same intensities of the incoming streams*

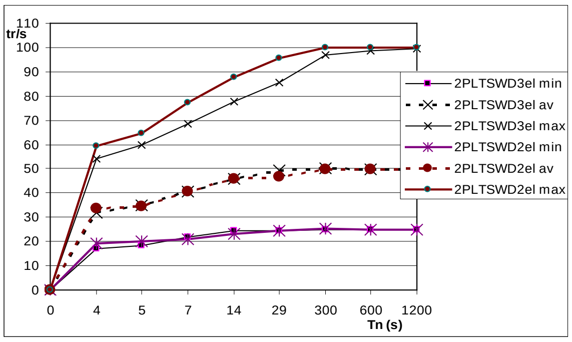

***Fig. 4.** Throughput in Centralized 2PL with TS (method "wait-die") and transactions handling up to 3 elements and in case of up to two elements*

From the graphs in fig.4 is observed that the throughput of systems in case of longer transactions (handling 3 elements) is very similar under the same systems workload as in stationary mode, and average and maximum systems load the graphics merge.

Fig.5 shows the diagram of frequency distribution of response time (RT) in case of: summary input intensity 100 tr/s, modeling time 28800 model units, i.e. before stationary regime. The diagram in fig.5 corresponds to the stereotyped graphic of response time, shown in [9, p.74].

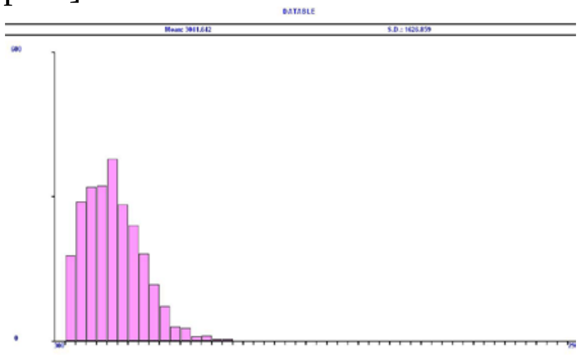

***Fig. 5** Frequency distribution of transaction RT in centralized 2PL with TS model for input intensity 100 tr/s. Time intervals on axis X have length 400 ms*

Fig.6 shows the diagram of frequency distribution of response time for transactions accessed up to 2 elements (in case of: summary input intensity 100 tr/s, modeling time 28800 model units).

Analysis of results given in fig.5 and fig.6 shows that response time in the model handling 3 elements is greater than similar for shorter transactions. Moreover, dispersion of average response time is greater in case of handling 3 elements. The diagrams of frequency distribution of response time (fig.5 and fig.6) are similar, which support steadiness of the algorithm of 2PL with integrated TS used method "wait-die".

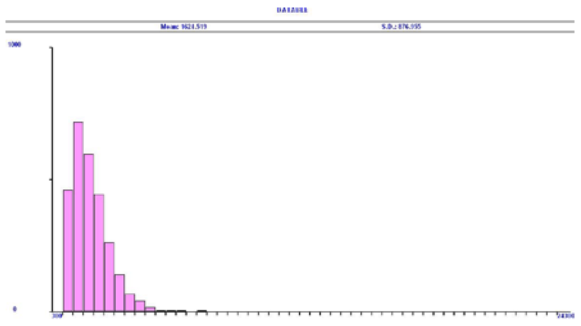

***Fig. 6** Frequency distribution of transaction RT in centralized 2PL with TS (2 elements) model for input intensity 100 tr/s. Time intervals on axis X have length 200 ms*

Conclusions

It is suggested a structural scheme of modeling algorithm to control the transactions for primary copy two phase protocol in distributed managed database systems, in which a mechanism of timestamps (method "wait-die") for evasion of deadlocks is embedded.

A program code for GPSS World is developed which can be used to model the processing of distributed transactions in pessimistic protocol in

DDB. The advantages of embedding the mechanism of timestamps (method “wait-die”) in the 2PL algorithms in DDBMS are proved experimentally.

We continue the gathering of statistics of simulations of the modeling algorithm in different parameters of the system and the coming streams [3],[10] and a comparative analysis of the results in different solutions for the data replication and the concurrency control.

References

[1] Bodyagin I. Deadlocks. Chto takoe vzaimoblokirovki i kak s nimi borotsya, RSDN Magazine #5, 2003, http://sasynok.narod.ru/index.htm?omvs.htm.

[2] Garsia-Molina, G., D. Ulman, D. Widom, Sistemi baz danni, Williams Moscow, 2003.

[3] Simeonova, N, Simeonov, S., Iliev, A. „Concepts for Creating Operating Systems with Special Purpose”, UNITECH’12, Gabrovo, 16-17 November 2012, pp 377-381, ISSN 1313-230X

[4] Konoli, T., К. Beg, Basi Danni, Williams Moscow – Sanct Peterburg - Kiev, 2003.

[5] Kudryavtsev, E. GPSS World Osnowi imitacionovo modelirovaniya razlichnih sistem, DMK Press, Moscow, 2004.

[6] Kuznetsov, S. Bazi dannih. Vodnih kurs, Moscow, CIT FORUM, 2008, http://www.citforum.ru/database/advanced_intro/43.shtml

[7] Bernstein, Ph., N. Goodman Concurrency Control in Distributed Database Systems, Computing Surveys, Vol. 13, No. 2, 1981, pp. 185-221, www.sai.msu.su/~megera/postgres/gist/papers/concurrency/p185bernstein.pdf

[8] Krivokapic, N., A. Kemper, E. Gudes Deadlock detection in distributed database systems: A new algorithm and a comparative performance analysis, http://masters.donntu.edu.ua/2005/fvti/kovalyova/library/d1.pdf.

[9] TPC – C Benchmark. http://www.tpc.org/tpcc/spec/tpcc_current.pdf. (2007)

[10] Simeonov, S., Tswetanov, Ts., Network Flow Security Baselining, IT-Incidents Management & IT-Forensics - IMF , pp. 143-156, 2008