무선 이동 네트워크 환경에서 다단계 보안 데이터베이스를 위한 분산 이타적 잠금 기법

김 희 완†·박 동 순††·이 혜 경†††·김 응 모††††

요 약

본 논문에서는 무선 이동 네트워크 환경에서 다단계 보안 데이터베이스의 동시성 제어를 위한 향상된 트랜잭션 스케쥴링 프로토콜을 제안한다. 무선 통신은 잦은 접속단절의 특성을 가지고 있다. 따라서 단기 트랜잭션은 장기 트랜잭션으로 인한 지연이 없이 데이터베이스를 빨리 접근하여야 한다. 전통적인 직렬성 표기를 가진 두단계 잠금 기법을 무선 이동 네트워크 환경에서 다단계 보안 데이터베이스에 적용했다. 이타적 잠금기법은 기부를 통하여 트랜잭션이 더 이상 그 객체를 요구하지 않을 때 다른 트랜잭션들이 객체를 로크할 수 있도록 미리 객체에 대한 로크를 해제함으로써 트랜잭션들의 대기시간을 줄이기 위해서 제안된 것이다. 확장형 이타적 잠금기법은 처음에 기부되지 않는 객체까지도 처리하는 좀 더 완화된 기법이다. 본 프로토콜은 확장형 잠금 기법을 기초로 한 다단계 보안 데이터베이스를 위한 양방향 기부 잠금 규약(MLBiDL)으로 보안 요구와 동시성 제어를 동시에 만족한다. 시뮬레이션 결과 MLBiDL은 다른 잠금 기법들 보다 처리율과 트랜잭션의 평균 대기시간에서 우수한 성능을 보여주었다.

A Distributed Altruistic Locking Scheme For Multilevel Secure Database in Wireless Mobile Network Environments

Hee-Wan Kim† · Dong-Soon Park†† · Hae-Kyung Rhee††† · Ung-Mo Kim††††

ABSTRACT

We propose an advanced transaction scheduling protocol for concurrency control of multilevel secure databases in wireless mobile network environment. Wireless communication is characterized by frequent spurious disconnections. So short-lived transaction must quickly access database without any delay by long-lived one. We adapted two-phase locking protocol, namely traditional syntax-oriented serializability notions, to multilevel secure databases in wireless mobile network environment. Altruistic locking, as an advanced protocol, has attempted to reduce delay effect associated with lock release moment by use of the idea of donation. An improved form of altruism has also been deployed for extended altruistic locking. This is in a way that scope of data to be early released is enlarged to include even data initially not intended to be donated. Our protocol is based on extended altruistic locking, but a new method, namely bi-directional donation locking for multilevel secure databases (MLBiDL), is additionally used in order to satisfy security requirements and concurrency. We showed the Simulation experiments that MLBiDL outperforms the other locking protocols in terms of the degree of throughput and average waiting time.

키워드 : 양방향 기부 잠금(Bi-Directional Donation Locking), 이타적 잠금기법(Altruistic Locking), 무선 이동 네트워크(Mobile Network), 다단계 보안 데이터베이스(Multilevel Secure Database)

1. Introduction

Recent advances in technology have provided portable computers with wireless interfaces that allow networked communication even while a user is mobile. Lower bandwidths, higher error rates, and more frequent spurious disconnections characterize wireless communication [1]. So Short-lived transaction must quickly access database without any delay by long-lived transaction. A Multilevel secure database in wireless mobile network is a secure system which is shared by users from more than one clearance levels and contains data of more than one sensitivity levels [3]. When the database scheduler use the scheduling protocol to multilevel secure database, it must satisfy both the concurrency and the security requirements at the same time.

A data items correctness is guaranteed by standard transaction scheduling schemes like *two-phase locking* (2PL)

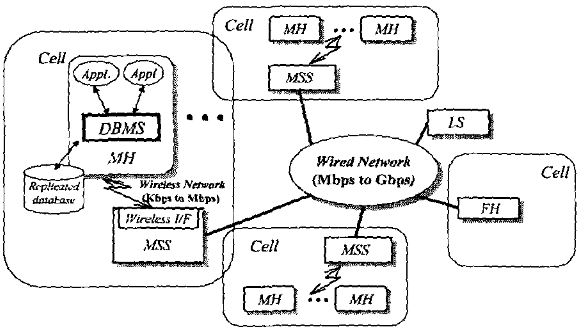
※ 본 연구는 한국과학재단 목적기초연구(과제번호 : R01-2000-00250) 지원으로 수행되었음. (This work was supported by grant No. R01-2000-00250 from the Korea Science & Engineering Foundation.)
† 정 회 원 : 삼육대학교 컴퓨터과학과 교수
†† 정 회 원 : 동원대학 컴퓨터정보과 교수
††† 정 회 원 : 용인송담대학 컴퓨터소프트웨어과 교수
†††† 종신회원 : 성균관대학교 전기전자및컴퓨터공학부 교수
논문접수 : 2001년 10월 23일, 심사완료 : 2001년 12월 26일

[8]. We adapted two-phase locking protocol to multilevel secure databases in wireless mobile network environment. To reduce starvation or livelock in 2PL, altruism has been suggested. *Altruistic locking* [5] is an extension to 2PL in the sense that several transactions may hold locks on an object simultaneously under certain conditions. Such conditions are signaled by an operation *donate*. *Extended altruistic locking* [5] attempted to expand the scope of donation in a way that data to be early disengaged is augmented by extra data originally not conceived to be rendered. Our protocol is based on extended altruistic locking but a new method, namely bi-directional donation locking, is additionally used in order to satisfy security and concurrency to multilevel secure databases in wireless mobile network environments.

2. Related Work

2.1 Basic mobile system architecture

Advances in computing and networking technologies have made extensive use of portable computers possible and enabled on-line information sharing via wireless communication channels.



(Figure 1) Basic mobile system architecture

Mobile computing, allows users to perform on-line transaction processing independent of their physical location [2]. Generally, a mobile computing architecture includes two distinct sets of entities : mobile hosts (MHs) in the wireless network and fixed hosts (FHs) in the wired network (Figure 1).

The MHs can dynamically move within a radio coverage area called a cell or between two cells while retaining their network connection. The FHs are steadily connected to the wired network and some of them, called mobile support stations (MSSs), are augmented with a wireless interface to communicate with the MHs. Normally, a single MSS is able to support a number of MHs, and is engaged to provide services such as data passing and message interpretation to the MHs positioned only within its cell. Each MH includes several applications such as groupwork tool and one small DBMS which performs basic tasks to manage database consistency regarding transactions issued by the local applications. In replicated mobile database environments, multiple MHs maintain replicated data and they use replication control tools for data synchronization. We expand our locking protocol in distributed database systems [10] to multilevel secure database system in wireless mobile network environments.

2.2 Multilevel Security

Each data item in multilevel secure database is labeled with its security classification and each user is assigned a clearance level. In example, we will use the following hierarchical levels ordered as follows :

$$\text{Top Secret} \geq \text{Secret} \geq \text{Confidential} \geq \text{Unclassified}$$

A security model is an abstract model of how a secure system enforces the security policy. One popular model was developed by Bell and LaPadula model [4]. The BLP model requires that the system satisfy the following properties.

Simple Security Condition

A subject may have read access to an object only if the subject's classification level dominates the object's sensitivity level.

*-Property (Star Property)

A subject may have write access to an object only if the object's sensitivity level dominates the subject's classification level.

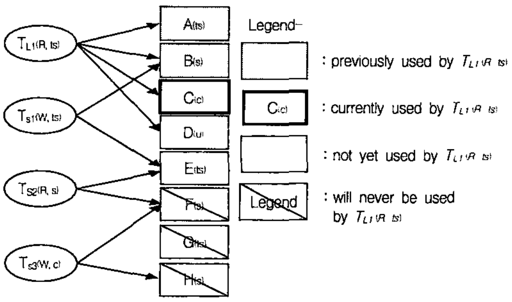
We used ts, s, c and u to denote the hierarchical level for transaction and data item orderly in this paper.

2.3 Applying Extended Altruistic Locking to MLS

MLXAL's rule is that wake expansion comes true only after a short transaction has already accessed data in its predefined wake list. So, the presumption could be called wakelist-first/other-later access. *MLXAL* performs badly if others- first/wake-later access paradigm is in fact to be observed. Example 1 shows this.

Example 1(Delay Effect Caused by Donation Extension in Short-Lived Transaction) : Suppose that the long-lived transaction $T_{LI}(R, ts)$ attempts to access data items, $A(ts)$.

$B(s)$, $C(c)$ and $D(u)$, orderly in multilevel secure database. Note that data items, $E(ts)$, $F(s)$, $G(ts)$, and $H(s)$ shall not be accessed by $T_{L1}(R, ts)$ at all. Presume that $T_{L1}(ts)$ has already locked and successfully donated $A(ts)$, $B(s)$ and $C(c)$. $T_{L1}(R, ts)$ now is supposed in the stage of accessing D(u). Suppose also that the short-lived transactions T_{S1}(W, s) wishing for $B(s)$ and $E(ts)$, $T_{S2}(R, s)$ wishing for $E(ts)$and $F(s)$, and $T_{S3}(W, c)$ wishing for $F(s)$ and H(s) (Figure 2).



(Figure 2) Four Transactions, T_{L1} through T_{S3}, Competing for Same Data Donated

If we apply *MLXAL* for this situation, T_{S1}(W, s) could be allowed to access both $B(s)$ and $E(ts)$ without any delay. In case T_{S1}(W, s) initially requests $B(s)$ first rather than $E(ts)$, T_{S1}(W, s) is able to access not only $B(s)$ but $E(ts)$ as well, since T_{S1}(W, s) is fully in the wake of $T_{L1}(R, ts)$. So T_{S1}(W, s) succeeds to commit. $T_{S2}(R, s)$ then could not acquire $E(ts)$ because of $*$-property in BLP security model released by T_{S1}(W, s). $T_{S3}(W, c)$ could thereafter acquire $F(s)$ released by $T_{S2}(R, s)$.

In case, however, if $T_{S1}(W, s)$ initially requests $E(ts)$ first rather than $B(s)$, $T_{S1}(W, s)$ can certainly acquire $E(ts)$ but it fails for $B(s)$ because wake relationship cannot honor $E(ts)$ as a member of the wake list. Once this sort of wake dependency is detected, $T_{S1}(W, s)$ can be allowed to access $B(s)$ only after it is finally released by $T_{L1}(R, ts)$. $T_{S1}(W, s)$ in this case is therefore blocked. $T_{S2}(R, s)$ must then be blocked for $E(ts)$ to be released by $T_{S1}(W, s)$. $T_{S3}(W, c)$ as well must be blocked for $F(s)$ to be released by $T_{S2}(R, s)$, forging a chain of blockage. End of Example 1.

To resolve this sort of chained delay, others first/wake later approach could be made viable in a way of including others to a wake list. This enhancement is one of substances which could be considered as *backward donation*, compared to *MLXAL* which is based on *forward donation*. *MLXAL* can be viewed as *one donation* scheme in that it deals with donation principle involving only one long transaction. One other major substance is to let more than one long transaction donate while serializability is preserved in multilevel secure database. Our protocol allows more donation than one long transaction, but for the sake of presentation simplicity, degree of donation is limited to two in this paper.

3. Proposed Protocol

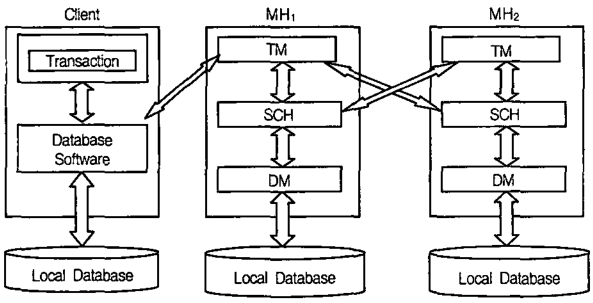
3.1 Algorithm

Bi-directional donation locking for multilevel secure database, MLBiDL for short, can be pseudo-coded as follows(Algorithm Wake Expansion).

```
Algorithm(Wake Expansion Rule of MLBiDL)
Input : TL1 ; TL2 ; TS
/* TS : short-lived trans ; TL1, TL2 : long-lived trans */
BEGIN
  FOREACH LockRequest
    IF(LockRequest.TS.data = Lock) THEN
        Reply := ScheduleWait(LockRequest) ;
    ELSE IF(LockRequest.TS.data = Donated) THEN
        FOREACH (ST.wake ∈ TL1 OR TL2)
          IF(ST.wake = TL1) THEN
            IF(ST.data ∈ TL1.marking-set) THEN
              Reply := ScheduleWait(LockRequest)
            ELSE
              Reply := SecurityCheck(LockRequest)
            ENDIF
          ELSE
            IF(ST.data ∈ TL2.marking-set) THEN
              Reply := ScheduleWait(LockRequest)
            ELSE
              Reply := SecurityCheck(LockRequest)
            ENDIF
          ENDIF
        ENDFOR
  ELSE
        Reply := SecurityCheck(LockRequest)
  ENDIF
    IF(Reply = Abort) THEN
    Abort Transaction(Transactionid) ; Send(Abort) ;
    Return() ;
    ENDIF
  ENDFOR
END
SecurityCheck(TRAN, DATA, GUBUN)
BEGIN
IF((TRAN.R = True) AND (TRAN.level ≥ Data.level)) OR
  ((TRAN.W = True) AND (TRAN.level ≤ Data.level))
  IF( GUBUN = Lock ) THEN
        Reply := ScheduleLock(LockRequest)
  ELSE
        Reply := ScheduleDonated(LockRequest)
  ENDIF
ELSE                    /* No read up or No write down */
        Reply := DiscardData(LockRequest)
  ENDIF
END
```

3.2 Transaction Processing Model

In distributed computing environments, a TM of the MH in mobile network environment receives transactions from terminals and passes them SCH queue or other MH's SCH queue in the MSS by disconnection. TM could receive a message informing abortion from SCH or an acknowledgement informing completion of a requested operation from DM. DM analyzes an operation from SCH to determine which data item the operation is intended to access, and then sends the operation to the disk where the requested data item is stored. The server executes operations in its own FIFO queue one at a time. Whenever an operation is completed at the server, it sends to TM the message informing that the requested operation has been completed successfully.



(Figure 3) *MLBiDL* Transaction Processing Model

3.3 Operation Instance of MLBiDL

In case we apply *MLBiDL* in previous Example 1, if $T_{S1}(W, s)$ initially requests $E(ts)$ first rather than $B(s)$, $T_{S1}(W, s)$ can certainly acquire not only $E(ts)$ but $B(s)$ according to other-first/wake-later policy. And $T_{S2}(R, s)$ can acquire $E(ts)$ to be released by $T_{S1}(W, s)$. $T_{S3}(W, c)$ as well can acquire $F(s)$ to be released by $T_{S2}(R, s)$. If there are many transactions like $T_{S1}(W, s)$, the scheduler has a burden to maintain enlarged wakes. This sort of deficiency would fortunately not incur a substantial burden to the system because the access time of short transactions usually commit promptly.

3.3 Correctness of MLBiDL

In this section, we will show that *MLBiDL* satisfy both serialization and security requirement. To do so, we will make use of the serializability theorem [6] and a lemma used in proving the correctness of *MLAL* [5].

The notations used in this correctness proof are as follows. We use oi[x], pi[x] or qi[x] to denote the execution of either read or write operation issued by a transaction, Ti, on a data item, x. Reads and writes of data items are denoted by ri[x] and wi[x], respectively. Locking operation for either read or write is also represented by oli[x], pli[x], qli[x], rli[x] or wli[x]. Unlock and donate operations are denoted by ui[x] and di[x] respectively. H represents a history which may be produced by *MLBiDL* and O(H) is a history obtained by deleting all operations of aborted transactions from H. The characteristics which may be produced by *MLBiDL* are as follows.

Property 1 (Two-Phase) : If oli[x] and ui[y] are in O(H), oli[x] < ui[y].

Property 2 (Lock) : If oi[x] is in O(H), oli[x] < oi[x] < ui[x].

Property 3 (Donate) : If oli[x] and di[x] is in O(H), oi[x] < di[x].

Property 4 (Unlock) : If di[x] and ui[x] is in O(H), di[x] < ui[x].

Property 5 (Altruism) : If $o_i[x]$ and $o_j[x]$ (i ≠ j) are conflicting operations in H, and $o_i[x] < o_j[x]$, then either $u_i[x] < ol_j[x]$, or $d_i[x]$ exists in H and $d_i[x] < ol_j[x]$.

Property 6 (Security) : If level(T_i) ≥ level($r_i[x]$) in O(H), $rl_i[x] < u_i[x]$, and If level(T_i) ≤ level($w_i[x]$) in O(H), $wl_i[x] < u_i[x]$.

Property 7 (Lower Level Transaction First) : If level(Ti) < level(Tj) in O(H), dj[x] < oli[x].

Property 8 (Indebtedness) : If Tj is indebted to Ti for every oj[x] in O(H), either oj[x] is in the wake of Ti or there exists ui[y] in O(H) such that ui[y] < oj[x].

Lemma 1 (Complexity-In-Wake) : If $T_1 \rightarrow T_2$ is in SH(G), then either $T_1 \rightarrow_u T_2$ or $T_1 \rightarrow_d T_2$.

Proof : We assume that T_2 is not completely in the wake of T_1 and show that this implies $T_1 \rightarrow_u T_2$. Because of the arc $T_1 \rightarrow T_2$, there must be conflicting operations $o_1[x] < o_2[x]$ in H. By Property 1, both transactions locks and unlock x. By Property 5, T_1 has either donated or unlocked x before T_2 locks it. In the first case we have $T_1 \rightarrow_u T_2$. In the second case, object x is donated by T_1 when it is locked by T_2, so T_2 is in the wake of T_1. Since T_2 is not completely in the wake, by Property 8 some lock of T_2 must follow some unlock of T_1 in H. End of Lemma 1.

Lemma 2 (Correctness of *MLAL*) : Consider a path $T_1 \rightarrow \cdots T_{n-1} \rightarrow T_n$ in SG(H). Either :

- $T_1 \rightarrow_u T_2$, or

- There exists some T_i on the path such that $T_1 \rightarrow_u T_i$.

Proof : We will use induction on the path length n. By Lemma 1, the lemma is true for n = 2. Assume the lemma is true for paths of length n-1, and consider a path of length n. By the inductive hypothesis, there are two cases :

1). There is a T_l between T_1 and T_{n-1} such that $T_1 \rightarrow_u T_k$. The lemma is also true for paths of length n.
2). $T_1 \rightarrow_d T_{n-1} \rightarrow T_n$ and T_{n-1} conflicts on at least one object, x. Since $T_{n-1}$is completely in the wake of T_1, we must have $d_1[x] < ql_{n-1}[x]$ in O(H). By Property 1, T_n must lock x. By Property 4, T_1 must unlock x. Either $u_1[x] < ol_n[x]$ or $ol_n[x] < u_1[x]$. In the first case, we have that $T_1 \rightarrow_u T_n$, i.e., T_n is the T_k of the lemma. In the second case, T_n is indebted to T_1. By Property 8, T_n is completely in the wake of $T_1(T_1 \rightarrow_d T_n)$ or $T_1 \rightarrow_u T_n$.

Theorem 1 (Serializability of MLBiDL) : If O(H) is acyclic, O(H) is serializable and satisfies security rules.

Proof : Assume that there exists a cyclic $T_1 \rightarrow \cdots T_{n-1} \rightarrow T_n$ in serialization graph. By Lemma 2, $T_1 \rightarrow_d T_1$, or $T_1 \rightarrow_u T_i$. By Property 3, only $T_1 \rightarrow_u T_i$ is possible. By Property 6, T_i in H satisfies security property. Since T_i is prohibited to lock any more data items once T_1 unlocks any one, T_i cannot be T_1. Again, by applying Lemma 2 to the same cycle $T_i \rightarrow T_{i+1} \rightarrow \cdots T_i$, we get $T_i \rightarrow_u T_k$.for the same reason and thus we get $T_1 \rightarrow_u T_i \ _u T_k$ in all. Since the relation $_u$ is transitive, $T_1 \rightarrow_u T_k$ is satisfied. Thus, T_k cannot be any of T_1 and T_i. If we are allowed to continue to apply Lemma 2 to the given cycle n-3 times more in this manner, we will get a path $T_1 \rightarrow_u T_{iu} \rightarrow T_k \rightarrow_u \cdots \rightarrow_u T_m$ containing all transactions, i.e., T_1 through T_n. If we apply Lemma 2 to the given cycle starting from T_m one more time, we are enforced to get a cycle $T_1 \rightarrow_u T_i \rightarrow_u T_k \rightarrow_u \cdots \rightarrow_u T_m \rightarrow_u T_1$ and we get a contradiction of violating Property 1 or Lemma 2. Thus serialization graph is acyclic and by the serializability theorem O(H) is serializable and satisfies security rules. End of Theorem 1.

Theorem 2 (Security Satisfaction of *MLBiDL*) : If H is a history with Property 6 and 7, then H satisfies security requirements.

Proof : By Property 6, a transaction can read data items at its own or lower level, and write data items at its own or higher level. Let Ti and Tj be two transactions such that L(Ti) > L(Tj). If Ti and Tj are conflicting with each other, then we can see that Ti read down the data item *x* while Tj writes into *x*. Then, there are two possible cases :

(i) Tj holds a lock on *x* before Ti requests a read lock on *x*, and
(ii) Ti holds a read lock on x before Tj requests a lock.

In the first case, Ti must wait for the data item x until Tj's donation of data *x* by Property 7. Therefore, the lower level transaction Tj is not delayed by the higher level one Ti.

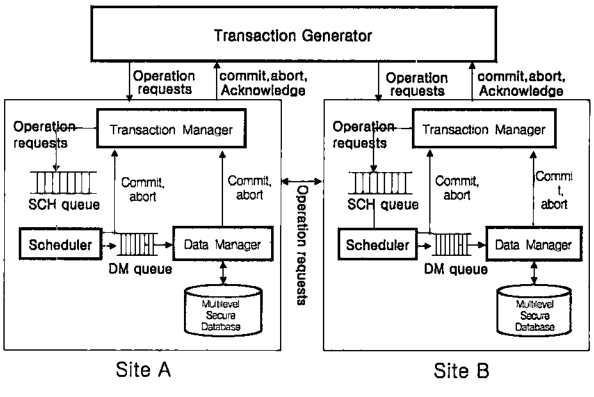
In the second case, in order to prevent covert channels, Tj can lock *x* without delaying by Property 7. Thus, Tj is neither delayed nor aborted by Ti. According to the above cases, the proposed protocol satisfies security requirements. End of Theorem 2.

4. Performance Evaluation

4.1 Simulation Model

4.1.1 Queuing System Model

The simulation model in (Figure 4), consists of subcomponents in charge of fate of a transaction from time of inception to time of retreat : *transaction generator* (TG), *transaction manager*(TM), *scheduler* (SCH), *data manager*(DM), *database*(DB).



(Figure 4) Simulation Model

TG generates user transactions one after another and sends their operations to TM one at a time in a way of interleaving. TM receives transactions from terminals and passes them SCH queue. Our simulation model is limited to two sites in wireless mobile network environment for the sake of simplicity in this paper.

This simulation model has been implemented using *Scheme* [7] discrete-event simulation (DEVS) language. In DEVS formalism one must specify basic models from which larger ones are built, and describe how these models are connected together in hierarchical fashion[9].

4.1.2 Experimental Methodology

<Table 1> summarizes the model parameters and shows the range of parameter values used in our experiments. Values for parameters were chosen by reflecting real world computing practices.

〈Table 1〉 Parameters Setting for Simulation

Parameters	Values
num_site	2
db_size	100
num_cpus	2
num_disks	4
num_security_levels	4
short_tran_size	2, 3, 4
long_tran_size	5, 6, 7, 8, 9
tran_creation_time	30
sim_leng	100, 300, 500, 700, 900, 1100, 1300, 1500

Database size matters if it affects the degree of conflict. If *db_size* is much larger than *short_tran_size* and *long_tran_size*, conflicts rarely occur. To see performance tradeoff between *ML2PL* and *MLBiDL*, average transaction length represented by number of operation in transaction were treated to vary.

The number of CPUs and disks, *num_cpus* and *num_disks*, are set to 2 and 4, respectively. The idea behind this status of balance by 1-to-2 ratio has been consulted from [8].

4.2 Simulation Results and Interpretations

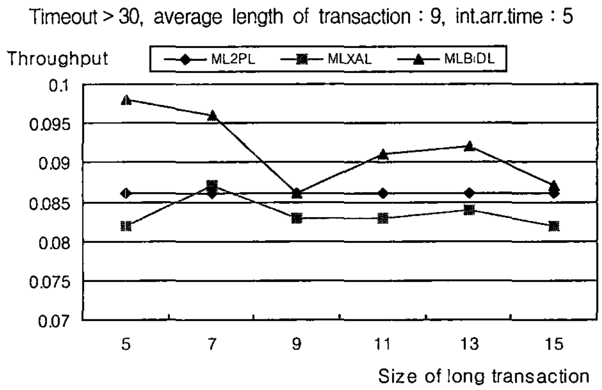
4.2.1 Effect of Security Requirement Level

This experiment has been revealed that *MLBiDL* satisfied the security requirement by Bell and LaPadula model. We have counted the processing ratio data item which satisfy the security requirement against total ones. Each transaction has Read/Write option, four clearance level, and data items which they process. Each data items have four sensitivity levels. If the transaction satisfy the security requirement which it wish to process the data item, it process the data item the next time slice. Otherwise, the transaction discards the data item, and it remains the current time slice of operating system. In this experimental, the entire processing ratio was 61.4 percent. So this model satisfies the security requirement by BLP model.

4.2.2 Effect of Multiprogramming Level

This experiment shows that *MLBiDL* generally appears to outperform *ML2PL* in terms of average waiting time. The best throughput performance is also exhibited by *MLBiDL* and the worst average waiting time is portrayed by *MLXAL*.

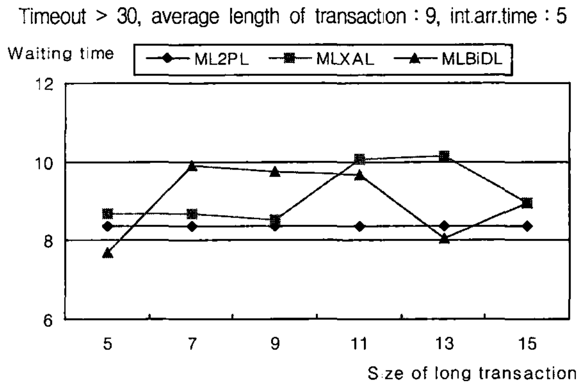
The major force behind prevalence of *MLBiDL* mainly comes from capitalizing advantage from maintaining two different transaction wakes. Performance gain of *MLBiDL* against *ML2PL* is from 100 to 114 percent increment in terms of throughput at every size of long transaction. This is because *MLBiDL* has the backward donation to reserve data objects to be accessed. In case the size of long transaction is 9 and 15, we can guess that there are no donation in *MLBiDL*'s scheduler because the throughput of *MLBiDL* similarly equal to the one of *ML2PL*.



(Figure 5) Throughputs

And *MLBiDL* outperforms *ML2PL* from 92 to 96 percent decrease of performance at transaction waiting time at long transaction size is 5 or 13. At the other case, the waiting time of *MLBiDL* has longer time than the other scheme because *MLBiDL* has the bi-directional donation which contributes to give transactions more chance to use the objects than the other schemes.

This was the conclusion that *MLBiDL* outperforms theother schemes due to enhanced degree of freedom given

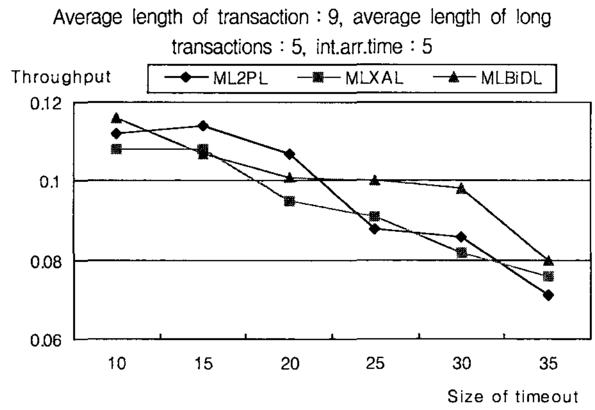


(Figure 6) Average Waiting Time

to *MLBiDL* in accessing donated data by extending to bi directional donation.

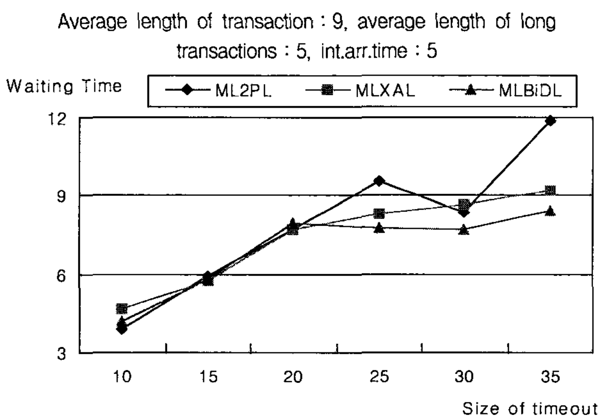
4.2.3 Effect of Timeout

At a higher range of *timeout*, *MLBiDL* shows a higher throughput and a lower transaction waiting time for three scheme. Throughput of *MLBiDL* outperforms *MLXAL* and *ML2PL* when timeout size is 10, 25, 30 or 35. We can observe that average waiting time curve of ML2PL rapidly increase from 30 to 35 in (Figure 8). As MLBiDLs result, This phe nomenon again shows us higher throughput gives lower average waiting time. *MLBiDL* performs better than *ML2 PL* between 103 percent to 113 percent of performance at transaction throughput at most case.



(Figure 7) Throughputs with Longer Timeout

As the timeout size is increased, the transaction waiting time of *MLXAL* is slowly increased. However, if the timeout size is far extended beyond a certain point, say 30, the average waiting time curve of *ML2PL* increase than other two scheme. *MLBiDL* outperforms *ML2PL* with 70.89% of performance at transaction waiting time when the timeout size is 35.



(Figure 8) Average Waiting Time with Longer Timeout

Overall behaviors have been revealed that as the size of timeout increases, *MLBiDL* generally outperforms in terms of throughput and waiting time. This shows a possibility that performance gain of *ML2PL* against *MLBiDL* could be deteriorated sharply if the timeout size is far extended beyond the size of timeout 30.

5. Conclusions

MLBiDL showed a more satisfying performance compared to any other scheme methods [5] for multilevel secure databases in wireless mobile network environment when long-lived transaction lead to abort overhead. As database access needs for multilevel secure database in wireless mobile network environment are adapted to a wide range of applications, transaction processing models require long-lived transactions needs. *MLBiDL* is definitely recommended in particular for environments where benefit of concurrency degree improvement exceeds overheads associated with aborts of long-lived transactions. Bi-directional donation altruism could be rendered to a simple-minded locking in which even database integrity is violated. *MLBiDL* is considered to be candidate for *ML2PL*, through *ML2PL* is dominant in many commercialized database engine. *MLBiDL* is considered to be a practical solution to take in real world environment where long-lived transactions naturally coexist with short-lived ones in wireless mobile network environments.

This wake-dependency may cause a lot of burdens for performing the submitted transactions. This is because *MLXAL* and *MLBiDL* have a certain overheads to reserve data objects to be accessed.

References

[1] G. H. Forman, J. Zahorjan, "The Challenges of Mobile Computing," IEEE Computer, Apr., 1994.

[2] Siwoo Byun, Songchun Moon, "Resilient data management for replicated mobile database systems," Data & Knowledge Engineering 29, pp.43-55, 1999.

[3] T. F. Keefe, W. T. Tsai and J. Srivastava, "Multilevel Secure Database Concurrency Control," Proceedings of Sixth International Conference on Data Engineering, pp.337-344, 1990.

[4] D. E. Bell, and L. J. LaPadula, "Secure Computer Systems : Unified Exposition and Multics Interpretations," Technical Report MTR-2997, Mitre Corp., March, 1976.

[5] K. Salem, H. Garcia-Molina and J. Shands, "Altruistic Locking," ACM Transactions on Database Systems, Vol.19, No.1, pp.117-169, March, 1994.

[6] P. A. Bernstein, V. Hadzilacos and N. Goodman, Concurrency Control and Recovery in Database Systems, Addison-Wesley, Massachusetts, U.S.A., 1987.

[7] H. Bartley, C. Jensen and W. Oxley, " Scheme User's Guide and Language Reference Manual," Texas Instruments, Texas, U.S.A., 1988.

[8] R. Agrawal, M. J. Carey and M. Linvy, "Concurrency Control Performance Systems," Vol.12, No.4, pp.609-654, December, 1987.

[9] Zeigler, B. P., "Object-Oriented Simulation with Hierarchical, Modular Models : Intelligent Agents and Endomorphic Systems," Academic press, San Diego, CA, 1990.

[10] Hee-Wan Kim, Hae-Kyung Rhee, Chil Gee Lee, Chul-Hwan Kim, and Ung-Mo Kim, "A Double Donation Locking in Distributed Database Systems," Proceeding of the International Conference on Parallel and Distributed Processing Techniques and Applications : PDPTA 2001, Las Vegas, Nevada, U.S.A., June, 2001.

김 희 완

e-mail : hwkim@syu.ac.kr

1987년 광운대학교 전자계산학과 졸업(이학사)

1995년 성균관대학교 정보공학과 졸업(공학석사)

2002년 성균관대학교 전기전자 및 컴퓨터공학부 졸업(공학박사)

1991년 한국전력공사 정보처리처 근무

1996년 정보처리 기술사(정보관리) 취득

1999년 공인 정보시스템감리인 자격취득(한국전산원)

1996년 삼육의명대학 전산정보과 조교수

2001년~현재 삼육대학교 컴퓨터과학과 조교수

관심분야 : DB보안, 동시성제어, 분산DB, Mobile Computing

박 동 순

e-mail : dspark@tongwon.ac.krr

1979년 성균관대학교 전자공학과 졸업(공학사)

1981년 육군 사병 전역

1984년 삼성전자(주) 컴퓨터부문 근무

1990년 LG-EDS(주) 근무

1993년 현대정보기술(주) KPDC 근무

1995년 성균관대학교 정보공학과 졸업(공학석사)

2000년 성균관대학교 전기전자 및 컴퓨터공학부 박사과정 수료

1996년~현재 동원대학 컴퓨터정보과 조교수

관심분야 : 분산DB, DB보안, Web log, Mobile Computing

이 혜 경

e-mail : rheehk@dove.kyungin-c.ac.kr

1979년 숭실대학교 전자계산학과 졸업(공학사)

1985년 미국 일리노이대학교(Urbana-Champaign) 전산학과 졸업(공학석사)

2000년 성균관대학교 정보공학과 졸업(공학박사)

1988년 국립 천안공업전문대학 전자계산과 전임강사

1993년 경인여자대학 멀티미디어정보학부 조교수

2000년~현재 용인송담대학 컴퓨터소프트웨어과 조교수

관심분야 : 동시성제어, 분산DB, DB보안, Mobile Computing

김 응 모

e-mail : umkim@yurim.skku.ac.kr

1981년 성균관대학교 수학과 졸업(이학사)

1986년 Old Dominion University 전산학과 졸업(공학석사)

1990년 Northwestern University 전산학과 졸업(공학박사)

1990년~현재 성균관대학교 전기전자 및 컴퓨터공학부 교수

관심분야 : DB보안, 데이터마이닝, 웹DB, 공간DB, 동시성제어