1990

Preserving Data Integrity in HDDBSs Using Quasi Serializable Executions

Ahmed K. Elmagarmid
Purdue University, ake@cs.purdue.edu

Weimin Du

Report Number:
90-970

#970

Preserving Data Integrity in HDDBSs Using Quasi Serializable Executions[1]

Ahmed Elmagarmid and Weimin Du
Department of Computer Sciences
Purdue University
West Lafayette, IN 47907
{ake,du}@cs.purdue.edu

[1]This work is supported by a PYI Award from NSF under grant IRI-8857952 and grants from AT&T Foundation, Tektronix, SERC and Mobil Oil.

1 Introduction

Data integrity is one aspect of database consistency concerned with the validity of databases. Integrity requirements are expressed by means of a set of integrity constraints. A database state is consistent (with respect to a set of constraints) if all these constraints are evaluated to be true in it. An execution preserves integrity if it transfers a database from an consistent state to another consistent state and each transaction sees an consistent state in the execution.

Database integrity is usually enforced through integrity enforcement mechanisms [BM88] or correct scheduling of transactions. The latter approach, however, works only for those constraints that one can determine statically (i.e., independently of database state) whether they are preserved by a single transaction. In this paper, we are interested in the second approach and assume that each transaction, when executed alone, preserves all constraints.

Serializability is the conventional correctness criterion for scheduling transactions. The problem with serializability approach in heterogeneous distributed database systems (HDDBSs), however, is that it is very difficult to maintain, due to both heterogeneity and autonomy of local databases [DELO89] [DEK90].

Another approach in HDDBSs is quasi serializability [DE89]. Quasi serializability is attractive because it can be effectively maintained at global level without violating local autonomy [ED90] [DE90a]. We showed in [DE90b] that quasi serializable executions also maintain transaction consistency of HDDBSs. In this paper, we study the strength and weakness of quasi serializability with respect to HDDBS integrity.

2 Background

An HDDBS consists of a set $\mathcal{D}$[1] of data items and a set $\mathcal{T}$ of transactions. The data item set $\mathcal{D}$ consists of n subsets, $\mathcal{D}_1, \mathcal{D}_2, ..., \mathcal{D}_n$, called local databases[2]. The transaction set $\mathcal{T}$ consists of $n+1$ subsets, $\mathcal{G}, \mathcal{L}_1, \mathcal{L}_2, ..., \mathcal{L}_n$, where $\mathcal{L}_i$ is a set of local transactions that access $\mathcal{D}_i$ only, while $\mathcal{G}$ is a set of global transactions that access more than one local database. A global transaction G_i consists of a set of subtransactions $\{G_{i,1}, G_{i,2}, ..., G_{i,n}\}$, where the subtransaction $G_{i,j}$ accesses $\mathcal{D}_j$ only. The data item set $\mathcal{D}_i$, together with the transaction set $\mathcal{T}_i = \mathcal{L}_i \cup \mathcal{G}_i$ where $\mathcal{G}_i = \{G_{j,i} \mid G_j \in \mathcal{G}\}$, forms the local database system $LDBS_i$.

[1] In the paper, we use italic letters to denote instances, e.g., lower case for data items and upper case for transactions, calligraphic letters to denote sets, and roman letters to denote acronyms.

[2] We assume that local databases are disjoint. In other words, there is no replication at global level.

2.1 Quasi Serializable Executions

Definition 2.1 (Quasi serial executions) *A global execution $E = \{E_1, E_2, ..., E_n\}$ is quasi serial if*

- *each local execution E_l is serializable; and*
- *there exists a total ordering of all global transactions such that $\forall G_i, G_j \in \mathcal{G}$ and G_i precedes G_j in the ordering, o_i precedes o_j in E_l for all o_i of G_i and all o_j of G_j, $(1 \leq l \leq n)$.*

Definition 2.2 (Quasi serializable executions) *An execution is quasi serializable if it is equivalent to a quasi serial execution of the same set of transactions.*

The order in which global transactions are executed in the equivalent quasi serial execution is called *quasi serialization order* of the execution. The quasi serialization order of an execution may not be unique.

2.2 Integrity Constraints

Integrity constraints are means to enforce integrity of databases. Generally, integrity constraints are formulas in predicate calculus that express relationships of data that a database must satisfy. For example, the balance of an account in a bank should be greater than or equal to zero.

Definition 2.3 (Simple predicate) *A simple predicate on $\mathcal{D}$ is any predicate having the form $P = x \bowtie y$, where*

- *x has the form $f(x_1, x_2, ..., x_p)$, where f is a function, $x_1, x_2, ..., x_p \in \mathcal{D}$ and $p \geq 1$;*
- *$\bowtie$ is one of the following operators: $>, \geq, =, \neq, <$ and $\leq$;*
- *y has the form $f'(y_1, y_2, ..., y_q)$, where f' is a function, $y_1, y_2, ..., y_q \in \mathcal{D}$ and $q \geq 0$* [3].

Definition 2.4 (Integrity constraint) *An integrity constraint on $\mathcal{D}$, IC, is any predicate having the form $IC = \bigwedge_{i=1}^{p}(\bigvee_{j=1}^{q} P_{i,j})$, where $P_{i,j}$ is a simple predicate on $\mathcal{D}$ (called base predicates of IC), $p \geq 1$ and $q \geq 1$.*

Given an integrity constraint IC, its data set, denoted $\mathcal{DS}(IC)$, is a set of data on which IC is defined. Let $\mathcal{C}$ be the set of all integrity constraints in an HDDBS. An integrity constraint in $\mathcal{C}$ can be classified into one of the following three categories, according to the composition of its data set.

[3] If $q = 0$ then y is a constant.

- **A local constraint** involves data at a single site only.
  $\mathcal{LC} = \{IC \in \mathcal{C} \mid \exists i \text{ such that } \mathcal{DS}(IC) \subseteq \mathcal{D}_i\}$
- **A global constraint** involves data that are updatable by global transactions only[4]. Let $\mathcal{D}_i^g$ be the subset of $\mathcal{D}_i$ that consists only of data that are updatable by global transactions only.
  $\mathcal{GC} = \{IC \in \mathcal{C} \mid \mathcal{DS}(IC) \subseteq \cup_{i=1}^{n} \mathcal{D}_i^g\}$
- **A distributed constraint** involves data at more than one site and at least one data item is updatable by local transactions.
  $\mathcal{DC} = \{IC \in \mathcal{C} \mid \nexists i \text{ such that } \mathcal{DS}(IC) \subseteq \mathcal{D}_i \text{ and } \mathcal{DS}(IC) \subseteq \cup_{i=1}^{n} \mathcal{D}_i^g\}$

Clearly, $\mathcal{C} = \mathcal{LC} \cup \mathcal{GC} \cup \mathcal{DC}$.

Local constraints represent old ones that existed before integration, while global and distributed constraints represent new ones introduced in/or after integration.

3 (G + L)-Integrity

In this section, we show that quasi serializable executions preserve all local and global constraints.

Given an HDDBS H with the data set $\mathcal{D} = \{d_1, d_2, ..., d_m\}$, a state of H is defined as an instantiation of $\mathcal{D}$.

$\mathcal{S} = \{s_i \mid s_i \text{ is an instantiation of } d_i,\ i = 1, 2, ..., m\}$

Let $\mathcal{C}$ be a set of integrity constraints of H, we say that $\mathcal{S}$ is *$\mathcal{C}$-consistent* if it is consistent with respect to $\mathcal{C}$.

Definition 3.1 ((G + L)-integrity of HDDBSs) *An HDDBS state is (G + L)-consistent if it is both $\mathcal{GC}$-consistent and $\mathcal{LC}$-consistent, where $\mathcal{GC}$ and $\mathcal{LC}$ are the sets of all global and local constraints of the HDDBS, respectively.*

Given an execution E of transactions $\mathcal{T}$ in H and $T \in \mathcal{T}$, we define the *view* seen by T in E, denoted $V(E,T)$, to be the set of all values read by T. For each T, let $\mathcal{R}(T)$ be the set of all data read by T. Then, there exists an HDDBS state $\mathcal{S}$, such that $V(E,T) = \Pi_{\mathcal{R}(T)}\mathcal{S}$, the projection of $\mathcal{R}(T)$ on $\mathcal{S}$.

A view is *(G + L)-consistent* if it is the projection on a (G + L)-consistent state of H.

[4]Data may not be updatable by local transactions for reasons of database consistency and transaction recovery [DEK90] [WV90].

Theorem 3.1 ((G + L)-integrity theorem) *A quasi serializable execution preserves the (G + L)-integrity of an HDDBS and each transaction sees a (G + L)-consistent view of the HDDBS in the execution.*

Proof: Let $E = \{E_1, E_2, ..., E_n\}$ be a quasi serializable execution of transactions $\mathcal{T} = \mathcal{G} \cup \mathcal{L}$. We prove by induction on k, the number of global transactions in E.

Basis step: ($k \leq 1$) It is true because each quasi serializable execution is also serializable.

Induction hypothesis: Assume it is true for all quasi serializable executions of less than k global transactions.

Induction step: Let $\mathcal{G} = \{G_1, G_2, ..., G_k\}$. Without loss of generality, assume that the quasi serialization order of $\mathcal{G}$ is $G_1, G_2, ..., G_k$. For each E_i, there exist an equivalent serial execution E_i^s and an equivalent quasi serial execution E_i^q, respectively. They can be expressed as follows.

$E_i^s : L_{i,0}G_{i,i_1}L_{i,1}G_{i,i_2}...L_{i,j-1}G_{i,i_j}L_{i,j}...L_{i,k-1}G_{i,i_k}L_{i,k}$[5], where $1 \leq j \leq k$ and $L_{i,p} \in \mathcal{L}_i, (p = 0, 1, ..., k)$.

$E_i^q : E_{i,0}^q G_{i,1} E_{i,1}^q ... E_{i,k-1}^q G_{i,k} E_{i,k}^q$, where $E_{i,p}^q$ is the subexecution consisting of operations of those local transactions that follow $G_{i,p}$ and precede $G_{i,p+1}$ in $E_i^s, (p = 0, 1, ..., k)$.

Notation: Let E_0 be an execution and $\mathcal{T}_0$ a set of transactions. We use $(E_0 \setminus \mathcal{T}_0)$ to denote the subexecution resulted by taking away from E_0 those operations that belong to transactions in $\mathcal{T}_0$.

(1). $V(E, G_p)$ **is (G + L)-consistent,** $p = 1, 2, ..., k-1$.

Let us consider $(E \setminus G_k)$. Since G_p precedes G_k in the quasi serialization order, it does not read (neither directly nor indirectly) from G_k. Therefore, $V(E, G_p) = V((E \setminus \{G_k\}), G_p)$. Clearly, $(E \setminus \{G_k\})$ is a quasi serializable execution of $k-1$ global transactions. By induction hypothesis, $V((E \setminus \{G_k\}), G_p)$ is (G + L)-consistent and so is $V(E, G_p)$.

(2). $V(E, G_k)$ **is (G + L)-consistent.**

Let $E_i' = (E_i^q \setminus \{L_{i,j_k}, L_{i,j_k+1}, ..., L_{i,k}\})$, where j_k is the subscript in E_i^s such that $G_{i,i_{j_k}} = G_{i,k}$. Since G_k precedes $L_{i,j_k}, ..., L_{i,k}$ in E_i^s, it does not read (neither directly nor indirectly) from any of them. Therefore, $V(E, G_k) = V(E', G_k)$, where $E' = \{E_1', E_2', ..., E_n'\}$. Let us consider $(E_i' \setminus \{G_k\})$. Clearly $E_i' = (E_i' \setminus \{G_{i,k}\})G_{i,k}$. Since $(E' \setminus \{G_k\}) = \{(E_1' \setminus \{G_{1,k}\}), ..., (E_n' \setminus \{G_{n,k}\})\}$ is a quasi serializable execution of $k-1$ global transactions, it preserves (G + L)-integrity (by induction hypothesis). Therefore, $V(E', G_k)$ is (G + L)-consistent and so is $V(E, G_k)$.

(3). $\forall L \in \mathcal{L}, V(E, L)$ **is (G + L)-consistent.**

[5] We assume that there is at most one local transaction between two consecutive global transactions. More than one local transactions can always be merged into one local transaction without changing the semantics of the execution.

Given $\mathcal{S}$, a state of H, the state of $LDBS_i$, denoted $\mathcal{S}_i$, is defined as $\Pi_{\mathcal{D}_i}\mathcal{S}$. $\mathcal{S}_i$ is (G + L)-consistent if $\mathcal{S}$ is (G + L)-consistent.

Consider E_i^s. Let $\mathcal{S}_i^0$ be the initial state of $LDBS_i$. Assume that $\mathcal{S}_i^0$ is (G + L)-consistent, then $V(E, L_{i,0}) = \Pi_{\mathcal{R}(L_{i,0})}\mathcal{S}_i^0$ is also (G + L)-consistent. Suppose that $L_{i,0}$ and G_{i,i_1} transfer $LDBS_i$ from $\mathcal{S}_i^0$ to $\mathcal{S}_i^1$ and $\mathcal{S}_i^1$ to $\mathcal{S}_i^2$, respectively. Then $\mathcal{S}_i^2$ is (G + L)-consistent. Therefore, $V(E, L_{i,1}) = \Pi_{\mathcal{R}(L_{i,1})}\mathcal{S}_i^2$ is (G + L)-consistent.

Similarly, we can prove that $V(E, L_{i,2}), V(E, L_{i,3}), ..., V(E, L_{i,k})$ are all (G + L)-consistent.

(4). **E preserves (G + L)-integrity of H.**

E preserves all local constraints because each transaction sees an $\mathcal{LC}$-consistent view and local executions are serializable. It also preserves global constraints because the involved data are only updatable by global transactions which see (G + L)-consistent views and are executed sequentially in $E^q = \{E_1^q, E_2^q, ..., E_n^q\}$, the equivalent quasi serial execution of E. □

4 Static Integrity

Quasi serializable executions may violate distributed constraints because transactions (both local and global) that access data involved in a distributed constraint may not be executed in a serializable way. In this section, we identify those distributed constraints that are guaranteed to be preserved by quasi serializable executions. First, let us introduce the notation of static constraints.

Given a simple predicate on $\mathcal{D}$, $P : f(x_1, x_2, ..., x_p) \bowtie f'(y_1, y_2, ..., y_q)$. We say that it is static if, for each operation o which updates a data item by a constant, there exists a constant c_o, such that $\forall \mathcal{S}$, if $f(x_1^s, x_2^s, ..., x_p^s) \bowtie f'(y_1^s, y_2^s, ..., y_q^s) + c$ is true for constant c, then $f(x_1^{s'}, x_2^{s'}, ..., x_p^{s'}) \bowtie f'(y_1^{s'}, y_2^{s'}, ..., y_q^{s'}) + c + c_o$ is also true, where $x_1^s, ..., x_p^s, y_1^s, ..., y_q^s \in \mathcal{S}$, $x_1^{s'}, x_2^{s'}, ..., x_p^{s'}, y_1^{s'}, ..., y_q^{s'} \in \mathcal{S}'$ and o transfer H from $\mathcal{S}$ to $\mathcal{S}'$. Similarly, an integrity constraint is static if its base predicates are all static.

Definition 4.1 (Static integrity) *An HDDBS state is statically consistent if it is $\mathcal{SC}$-consistent, where $\mathcal{SC}$ is the set of all static integrity constrains of the HDDBS.*

Static integrity is another aspect of HDDBS integrity that can be effectively preserved by quasi serializable executions, as the following theorem shows.

Theorem 4.1 (Static integrity theorem) *A quasi serializable execution preserves static integrity of an HDDBS and each transaction sees a statically consistent view of the HDDBS in the*

execution.

Proof: Similar to that of (G + L)-integrity theorem except step (4).

(4). ***E* preserves static integrity of *H*.**

Consider global transaction $G_i = \{G_{i,1}, G_{i,2}, ..., G_{i,n}\}$, where $G_{i,j} : g^1_{i,j} g^2_{i,j} ... g^{p_{i,j}}_{i,j}$. $\forall g^m_{i,j}, \exists gc^m_{i,j}$ (a constant) such that $\forall \mathcal{S}$, if $f(\mathcal{S}) \bowtie f'(\mathcal{S})$[6] $+c$ for constant c and $g^m_{i,j}$ transfers H from $\mathcal{S}$ to $\mathcal{S}'$, then $f(\mathcal{S}') \bowtie f'(\mathcal{S}') + c + gc^m_{i,j}$ is also true. Similarly, for a local transaction $L_{i,j} : l^1_{i,j} l^2_{i,j} ... l^{q_{i,j}}_{i,j} \in \mathcal{L}_i$, there exists a constant $lc^m_{i,j}$ for each $l^m_{i,j}$. Assume $\sum_{j=1}^{n} \sum_{m=1}^{p_{i,j}} gc^m_{i,j} = 0$ and $\sum_{m=1}^{q_{i,j}} lc^m_{i,j} = 0$.

Let $\mathcal{S}^0$ be the initial state of H. Then $\mathcal{S}^0$ is statically consistent: $f(\mathcal{S}^0) \bowtie f'(\mathcal{S}^0)$ is true. Let $\mathcal{S}^f$ be the final state of H (after executing E). Then $f(\mathcal{S}^f) \bowtie f'(\mathcal{S}^f) + \sum_{i=1}^{k} (\sum_{j=1}^{n} \sum_{m=1}^{p_{i,j}} gc^m_{i,j}) + \sum_{i=1}^{n} (\sum_{j=0}^{k} (\sum_{m=1}^{q_{i,j}} lc^m_{i,j}))$. Therefore, $f(\mathcal{S}^f) \bowtie f'(\mathcal{S}^f)$. □

In summary, quasi serializable executions preserve all local and global constraints, as well as the distributed constraints that are static.

References

[BM88] E. Bertino and D. Musto. Correctness of semantic integrity checking in database management systems. *Acta Informatic*, 26(1):25–57, 1988.

[DE89] W. Du and A. Elmagarmid. Quasi serializability: a correctness criterion for global concurrency control in InterBase. In *Proceedings of the International Conference on Very Large Data Bases*, pages 347–355, Amsterdam, The Netherlands, August 1989.

[DE90a] W. DU and A. Elmagarmid. Maintaining quasi serializability in HDDBSs. Technical Report CSD-TR-971, Purdue University, March 1990.

[DE90b] W. Du and A. Elmagarmid. Maintaining transaction consistency in HDDBSs using quasi serializable executions. Technical Report CSD-TR-969, Purdue University, March 1990.

[DEK90] W. Du, A. Elmagarmid, and W. Kim. Effects of local autonomy on heterogeneous distributed database systems. Technical Report ACT-OODS-EI-059-90, MCC, February 1990.

[6] We use $f(\mathcal{S})$ and $f'(\mathcal{S})$ as shorthands for $f(x_1, ..., x_p)$ and $f'(y_1, ..., y_q)$, respectively, where $x_1, ..., x_p, y_1, ..., y_q \in \mathcal{S}$.

[DELO89] W. Du, A. Elmagarmid, Y. Leu, and S. Ostermann. Effects of autonomy on global concurrency control in heterogeneous distributed database systems. In *Proceedings of the Second International Conference on Data and Knowledge Systems for Manufacturing and Engineering*, pages 113–120, Gaithersburg, Maryland, October 1989.

[ED90] A. Elmagarmid and W. Du. A paradigm for concurrency control in heterogeneous distributed database systems. In *Proceedings of the Sixth International Conference on Data Engineering*, Los Angeles, California, February 1990.

[WV90] A. Wolski and J. Veijalainen. 2PC agent method: Achieving serializability in presence of failures in a heterogeneous multidatabase. In *Proceedings of PARBASE-90*, Miami Beach, Florida, 1990.