1990

Generalized Quasi Serializability Theory

Weimin Du

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

Report Number:
90-1019

GENERALIZED QUASI SERIALIZABILITY THEORY

Weimin Du
Ahmed K. Elmagarmid

CSD-TR-1019
September 1990

Generalized Quasi Serializability Theory*

Weimin Du and Ahmed K. Elmagarmid
Department of Computer Sciences
Purdue University
W. Lafayette, IN 47907

Keywords: Databases, multidatabases, serializability, quasi serializability

1 Introduction

Quasi serializability is a correctness criterion for global concurrency control in multidatabase systems (MDBSs) [1]. A global execution is quasi serializable if it is equivalent to a quasi serial execution in which global transactions are executed sequentially. The main motivation for quasi serializability is the difficulties in maintaining global serializability in MDBSs, due to local autonomy and incompatibility between serialization order and execution order of global transactions [3]. It is generally impossible for the global concurrency controller to maintain global serializability by controlling submission of global transactions only. Local transactions may introduce indirect conflicts between global transactions of which the global concurrency controller is not aware. Quasi serializability is different from serializability in that it coordinates executions of global transactions only and ignores those indirect conflicts that do not introduce interactions between global transactions. As a result of the relaxation, quasi serialization order of global transactions is compatible with their execution order. Thus, quasi serializability can be effectively maintained at the global level without violating local autonomy. The price for this is, however, the possible mutual interactions between remote local transactions. Therefore, quasi serializability is most suited to those MDBSs where local transactions at different sites do not interact with each other. It may also be used in more general MDBSs. Undesirable remote interactions among local transactions can be prevented by controlling information flow between sites through global transactions [4].

*Sponsored by the Indiana Corporation for Science and Technology (CST), a PYI Award from the NSF, grants from the AT&T Foundation, Tektronix, SERC, Mobil Oil, and a David Ross Fellowship from the Purdue Research Foundation.

The concept of conflict quasi serializability has been introduced in [1]. A scheduler which generates quasi serializable executions was given in [2]. In this paper, we study the generalized quasi serializability theory: view quasi serializability and final-state quasi serializability. A brief comparison between serializability and quasi serializability will also be given.

2 Notations

We assume that readers are familiar with the basic concepts and notations of serializability [5].

An MDBS 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. In this paper, we assume that local databases are disjoint. In other words, there is no replication at the global level. 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. We use $\mathcal{L}$ to denote the set of all local transactions, $\mathcal{L}_1 \cup \mathcal{L}_2 \cup ... \cup \mathcal{L}_n$. 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 $LDBS_i$.

A *transaction* T_i is a partially ordered, finite set of operations. Each operation is either a *read operation* reading a data item x, denoted $r_i(x)$, or a *write operation* writing a data item x, denoted $w_i(x)$. We use $\mathcal{R}(T_i)$ and $\mathcal{W}(T_i)$ to denote the sets of read and write operations of T_i, respectively, and $\mathcal{O}(T_i) = \mathcal{R}(T_i) \cup \mathcal{W}(T_i)$ the set of all operations of T_i. We also use $\mathcal{O}_l = \cup_{T \in \mathcal{L}} \mathcal{O}(T)$ to denote the set of all local operations.

Let $o_1 \in \mathcal{R}(T)$ and $o_2 \in \mathcal{W}(T)$. We say that o_2 *depends* on o_1 (or there is value dependency between o_1 and o_2) if o_1 precedes o_2 in the partial order. The interpretation of value dependency between read and write operations is that the value written by the write operation may be a function of the value read by the previous read operation.

A local execution E_l in LDBS_l is an interleaved sequence of operations of transactions in $\mathcal{T}_l$. $\forall o_1, o_2 \in \mathcal{O}(T)$, if o_1 precedes o_2 in the partial order of T it must also precede o_2 in E_l. In particular, o_1 must precede o_2 in E_l if o_2 depends on o_1. The order in which operations are executed in E_l is called the execution order of E_l and is denoted as $\prec_{E_l}$.

A global execution E in an MDBS consists of a set of local executions, $E = \{E_1, E_2, ..., E_n\}$, where E_l is the local execution at LDBS_l. The execution order of E is the sum of all local execution orders, denoted as $\prec_E = \cup_{l=1}^{n} \prec_{E_l}$.

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

Let $o_i \in \mathcal{R}(T_i)$ and $o_j \in \mathcal{W}(T_j)$ be two operations in a global execution E, where $T_i \neq T_j$. We say that o_i *reads* x *from* o_j in E if they both access data item x, $o_j \prec_E o_i$ and o_j is the last operation updating x before o_i in the corresponding local execution. We also say that T_i reads x from o_j and T_j.

3 View Quasi Serializability

3.1 View Quasi Serializable Executions

Given a global execution E, its *read from graph*, $RFG(E) =< \mathcal{V}, \mathcal{A}_r >$, is a directed graph defined as follows. The set of vertices $\mathcal{V}$ is the set of all operations in E, plus all operations in the "pseudotransaction" G_0 which writes the initial values of all data items, and G_∞ which reads the final results of the execution of E. The set of arcs $\mathcal{A}_r$ is defined as follows:

$\mathcal{A}_r = \{(o_j, o_i) \mid o_i \in \mathcal{R}(T_i), o_j \in \mathcal{W}(T_j)$, where $T_i \neq T_j$, and o_i reads from $o_j\}$

The *value dependency graph* of E, $VDG(E) =< \mathcal{V}, \mathcal{A}_v >$, is a directed graph, where

$\mathcal{A}_v = \{(v_1, v_2) \mid v_1 \in \mathcal{W}(T), v_2 \in \mathcal{R}(T)$ and $v_1 \prec_E v_2\}$.

Given two global executions E and E' of the same set of transactions, we say that E is *view convertible* to E' if and only if

- $RFG(E) = RFG(E')$; and
- $VDG(E) \subseteq VDG(E')$.

The intuitive interpretation of view convertibility from one execution to another is that the semantics of the former is totally preserved in the latter. The first condition guarantees that each transaction reads the same values in E' as in E, while the second condition says that all value dependency of transactions in E is preserved in E'. Since the execution order of E reflects all value dependency of related transactions, E and E' are computationally equivalent. Hence, if we consider E' as a correct execution, so does E. Note that the reverse may not be true because some value dependency that are correctly supported in E' may not be preserved in E.

Example 3.1 *Consider a banking database system. Suppose that a customer wants to deposit \$100 to account a and check the balance of account b. The request can be implemented as follows.*

$$E : ...r(a)w(a)r(b)...$$

Since $w(a)$ does not value depend on $r(b)$, E is computationally equivalent to the following execution.

$$E' : ...r(b)r(a)w(a)...$$

Note that $VDG(E) = \{(r(a), w(a))\} \subset VDG(E') = \{(r(a), w(a)), (r(b), w(a))\}$.

Definition 3.1 (View Quasi Serial Executions) *A global execution* $E = \{E_1, E_2, ..., E_n\}$ *is view quasi serial if*

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

Definition 3.2 (View Quasi Serializable Executions (VQSR)) *A global execution is view quasi serializable if it is view convertible to a view quasi serial execution.*

Example 3.2 *Consider global execution* E_1, *where*

$$E_1 = \begin{cases} w_{g_2}(a) w_{g_2}(b) w_{g_1}(a) r_{l_1}(a) r_{l_1}(b) w_{l_1}(a) & \text{at } \mathcal{D}_1 \\ w_{g_1}(x) r_{g_2}(x) & \text{at } \mathcal{D}_2 \end{cases}$$

E_1 *is view quasi serializable. It is view convertible to* E_1', *where*

$$E_1' = \begin{cases} w_{g_1}(a) r_{l_1}(a) w_{g_2}(a) w_{g_2}(b) r_{l_1}(b) w_{l_1}(a) & \text{at } \mathcal{D}_1 \\ w_{g_1}(x) r_{g_2}(x) & \text{at } \mathcal{D}_2 \end{cases}$$

It is not hard to see that E_1 *is not view serializable.*

3.2 View Quasi Serializability Theorem

The main difference between serializability and quasi serializability is that global and local transactions are treated differently in the latter. Since quasi serializability is a correctness criterion for global concurrency control in MDBSs, it focuses on interactions among global transactions. More specifically, in quasi serializability theory, interactions are modeled at transaction level for global transactions and at operation level for local transactions.

Definition 3.3 (View Quasi Serialization Graph (VQSG)) *A view quasi serialization graph of global execution* E *is a polygraph* $VQSG(E) =< \mathcal{V}, \mathcal{A}, \mathcal{C} >$, *where* $\mathcal{V} = \mathcal{G} \cup \mathcal{O}_l$ *is a set of vertices,* $\mathcal{A}$ *is a set of arcs defined as follows:*

- $\forall v_i, v_j \in \mathcal{V}$, $(v_i, v_j) \in \mathcal{A}$ *if* $(v_i, v_j) \in \mathcal{A}_v(E)$ *(v_i and v_j are operations of a local transaction);*
- $\forall v_i, v_j \in \mathcal{V}$, $(v_j, v_i) \in \mathcal{A}$ *if* v_i *reads from* v_j *(v_i and v_j are either local operations or global subtransactions).*

and $\mathcal{C}$ *is a set of choices defined as follows:*

$\forall v_i, v_j, v_k \in \mathcal{V}$, $(v_i, v_k, v_j) \in \mathcal{C}$ *if* v_i *reads* x *from* v_j *and* v_k *writes* x

The sets $\mathcal{A}$ and $\mathcal{C}$ are completely defined once we have read from relation of the execution and value dependency of transactions. Thus, we have the following lemma.

Lemma 3.1 *$VQSG(E) \subseteq VQSG(E')$ if E is view convertible to E'.*

Proof: Since $\mathcal{V}(E) = \mathcal{V}(E')$, we only need to show that $\mathcal{A}(E) \subseteq \mathcal{A}(E')$ and $\mathcal{C}(E) \subseteq \mathcal{C}(E')$
(1). $\mathcal{A}(E) \subseteq \mathcal{A}(E')$

$\forall (v_i, v_j) \in \mathcal{A}(E)$, let us consider the following two cases.

Case 1 $v_i \in \mathcal{R}(T)$ and $v_j \in \mathcal{W}(T)$ for $T \in \mathcal{L}$ and $v_i \prec_E v_j$. By the definition of view convertibility, $v_i \prec_{E'} v_j$. Therefore, $(v_i, v_j) \in \mathcal{A}(E')$.

Case 2 v_j reads from v_i in E. Then there exist o_i, o_j such that o_j reads fromo_i in E, where $o_i = v_i$ if v_i is a local operation and $o_i \in \mathcal{O}(v_i)$ otherwise, and $o_j = v_j$ if v_j is a local operation and $o_j \in \mathcal{O}(v_j)$ otherwise, Since $\mathrm{R}(E) = \mathrm{R}(E')$, o_j also reads from o_i in E'. Thus, v_j reads from v_i in E' and therefore $(v_i, v_j) \in \mathcal{A}(E')$.

(2). $\mathcal{C}(E) \subseteq \mathcal{C}(E')$

$\forall (v_i, v_k, v_j) \in \mathcal{C}(E)$, where v_i reads x from v_j and v_k writes x in E. Similar to Case 2, there exist o_i and o_j such that o_i reads x from o_j. Since $RFG(E) = RFG(E')$, o_i also reads x from o_j in E'. Therefore v_i reads x from v_j in E'. In other words, $(v_i, v_k, v_j) \in \mathcal{C}(E')$. □

Lemma 3.2 *$VQSG(E)$ is acyclic if E is view quasi serial.*

Proof: Given a view quasi serial execution E, let us first transform $VQSG(E)$ to a directed graph. $\forall (v_i, v_k, v_j) \in \mathcal{C}(E)$, we replace the choice by (v_i, v_k) if $v_i \prec_E v_k$, and (v_j, v_k) otherwise. Note that $v_j \prec_E v_k$ in the latter case. It is impossible for v_k to follow v_i but precede v_j in E because v_i reads from v_j in E. Let the directed graph be $VQSG_d(E)$. For each (v_i, v_j) in $VQSG_d(E)$, $v_i \prec_E v_j$. We next show that $VQSG_d(E)$ is acyclic.

First, there is no cycle in $VQSG_d(E)$ which contains more than one global transaction. This is true because global transactions are executed sequentially in E. Therefore, each cycle in $VQSG_d(E)$ is local, i.e., contains operations in a single local execution only. However, this is also impossible because all local executions are view serializable. □

Theorem 3.1 (View Quasi Serializability Theorem) *A global execution E is view quasi serializable if each local execution is view serializable and $VQSG(E)$ is acyclic.*

Proof: *(only if)* Given a global execution E, suppose that E is view quasi serializable. By Definition 3.2, each local execution is view serializable. In addition, there exists a view quasi serial execution E' to which E is view convertible. By Lemma 3.1, $VQSG(E) \subseteq VQSG(E')$. By Lemma 3.2, $VQSG(E')$ is acyclic, and so is $VQSG(E)$.

(if) Suppose that $VQSG(E)$ is acyclic. In other words, by replacing each choice (v_i, v_k, v_j) of $VQSG(E)$ with one of the arcs (v_i, v_k) and (v_k, v_j), we may get an acyclic directed graph $VQSG_d(E)$. $\forall v_i, v_j \in \mathcal{V}$, if v_j reads from or depends on v_i in E then arc (v_i, v_j) is in $VQSG_d(E)$.

Let $v_{i_1}, v_{i_2}, ..., v_{i_m}$ be a topologic sort of $VQSG_d(E)$. For each local execution E_l, we define E'_l to be the projection of global subtransactions and local operations of E_l on $v_{i_1} v_{i_2} ... v_{i_m}$. More specifically, E'_l is constructed by removing all local and global operations from $v_{i_1} v_{i_2} ... v_{i_m}$ that are not in E_l.

We now show that E_l is view convertible to E'_l. First, $\forall (o_i, o_j) \in \mathcal{A}_r(E)$, $\exists v_i, v_j \in \mathcal{V}$ where v_i (v_j) is o_i (o_j) if it is a local operation and corresponding global subtransaction otherwise. According to Definition 3.3, $(v_i, v_j) \in \mathcal{A}$. Therefore, v_i precedes v_j in E'_l. Since v_j reads from v_i, $\nexists v_k$ such that $v_i \prec_{E'_l} v_k$ and $v_k \prec_{E'_l} v_j$. In other words, v_j reads from v_i, and hence o_j reads from o_i in E'_l. Similarly, we can prove that $RFG(E') \subseteq RFG(E)$ and $VDG(E_l) \subseteq VDG(E'_l)$.

Let $E' = \{E'_1, E'_2, ..., E'_n\}$. Then E is view convertible to E'. Since each local execution is view serializable and E' is view quasi serial, E is view quasi serializable. □

4 Final-State Quasi Serializability

If we ignore the view seen by each transaction in an execution, quasi serializability can be further generalized, resulting in final-state quasi serializability. Let us first introduce some notions.

Given a global execution E, its *dependency graph* $\mathrm{D}(E) = < \mathcal{V}, \mathcal{A}_d >$ is a direct graph, where $\mathcal{V}$ is the same as that in read from graph and $\mathcal{A}_d$ is defined as follows.

$\mathcal{A}_d = \{(v_i, v_j) \mid v_i, v_j \in \mathcal{V}$, and v_i either reads from or depends on v_i $\}$

Let $\mathcal{V}' = \{v \mid v \in \mathcal{V}$, and $\exists v_\infty \in \mathcal{O}(G_\infty)$ such that $(v, v_\infty) \in D^*(E)\}$[2]. Then $\mathcal{V}'$ consists of only those operations whose effects can be seen by the final read "pseudotransaction" G_∞. The read from and value dependency graphs for $\mathcal{V}'$ is defined as follows.

$RFG_f(E) =< \mathcal{V}', \mathcal{A}'_r >$, where $\mathcal{A}'_r = \{(v_i, v_j) \mid v_i, v_j \in \mathcal{V}'$ and $(v_i, v_j) \in \mathcal{A}_r\}$

$VDG_f(E) =< \mathcal{V}', \mathcal{V}'_v >$, where $\mathcal{A}'_v = \{(v_i, v_j) \mid v_i, v_j \in \mathcal{V}'$ and $(v_i, v_j) \in \mathcal{A}_v\}$

Given two global executions E and E', we say that E is *final-state convertible* to E' if and only if

[2] $D^*(E)$ stands for the transitive closure of $D(E)$

- $RFG_f(E) = RFG_f(E')$; and
- $VDG_f(E) \subseteq VDG_f(E')$.

That E is final-state convertible to E' means that E transforms the database from an initial state to the same final state as E' does.

Definition 4.1 (Final-State Quasi Serial Executions) *A global execution $E = \{E_1, E_2, ..., E_n\}$ is final-state quasi serial if*

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

Definition 4.2 (Final-State Quasi Serializable Executions) *A global execution is final-state quasi serializable if it is final-state convertible to a final-state quasi serial execution.*

Example 4.1 (Final-State Quasi Serializable execution) *Consider execution E_2, where*

$$E_2 = \begin{cases} w_{g_1}(a)r_{l_1}(a)w_{g_2}(b)r_{l_1}(b)w_{l_1}(a)w_{g_1}(b) & at\ \mathcal{D}_1 \\ w_{g_2}(x)r_{g_1}(x)w_{g_1}(y)w_{g_1}(z)r_{g_2}(z)w_{l_2}(z) & at\ \mathcal{D}_2 \end{cases}$$

E_2 is final-state quasi serializable. It is final-state convertible to E_2', where

$$E_2' = \begin{cases} w_{g_2}(b)w_{g_1}(a)r_{l_1}(a)r_{l_1}(b)w_{l_1}(a)w_{g_1}(b) & at\ \mathcal{D}_1 \\ w_{g_2}(x)r_{g_2}(z)r_{g_1}(x)w_{g_1}(y)w_{g_1}(z)w_{l_2}(z) & at\ \mathcal{D}_2 \end{cases}$$

However, E_2 is not final-state serializable.

Definition 4.3 (Final-State Quasi Serialization Graph (FQSG)) *A final-state quasi serialization graph of global execution E is a polygraph $FQSG(E) =< \mathcal{V}, \mathcal{A}, \mathcal{C} >$, where $\mathcal{V} = \mathcal{G} \cup \mathcal{O}_l$ is the set of vertices, $\mathcal{A}$ is the set of arcs defined as follows:*

- *$\forall v_i, v_j \in \mathcal{V}$, $(v_i, v_j) \in \mathcal{A}$ if $(v_i, v_j) \in \mathcal{A}_v'(E)$;*
- *$\forall v_i, v_j \in \mathcal{V}$, $(v_j, v_i) \in \mathcal{A}$ if $\exists o_i, o_j$, such that $(o_j, o_i) \in \mathcal{A}_r'(E)$, where $o_i(o_j) = v_i(v_j)$ if $v_i(v_j)$ is a local operation and $o_i(o_j) \in \mathcal{O}(v_i)(\mathcal{O}(v_j))$ otherwise.*

and $\mathcal{C}$ is a set of choices defined as follows:

$\forall v_i, v_j, v_k \in \mathcal{V}$, $(v_i, v_k, v_j) \in \mathcal{C}$ if v_k writes x, and $\exists o_i, o_j$, such that o_j reads x from o_j, where $o_i(o_j) = v_i(v_j)$ if $v_i(v_j)$ is a local operation and $o_i(o_j) \in \mathcal{O}(v_i)(\mathcal{O}(v_j))$ otherwise.

Similar to the view quasi serializability theorem, we have

Theorem 4.1 (Final-State Quasi Serializability Theorem) *A global execution E is final-state quasi serializable if and only if each local execution is final-state serializable and $FQSG(E)$ is acyclic.*

5 Relationships to Serializability

Comparisons between quasi serializability and serializability in this section are based on their inclusiveness only. Generally speaking, quasi serializability is a more general notion. The set-theoretic difference is shown in the following theorem.

Theorem 5.1 *Let CSR, VSR and FSR be the sets of global executions that are conflict, view and final-state serializable respectively, and let CQSR, VQSR and FQSR be the sets of global executions that are conflict, view and final-state quasi serializable respectively. Then we have:*

1. $CSR \subset VSR \subset FSR$
2. $CQSR \subset VQSR \subset FQSR$
3. $CSR \subset CQSR$
4. $VSR \subset VQSR$
5. $FSR \subset FQSR$
6. $CQSR \not\subset VSR$
7. $CQSR \not\subset FSR$
8. $VQSR \not\subset FSR$
9. $VSR \not\subset CQSR$
10. $FSR \not\subset CQSR$
11. $FSR \not\subset VQSR$

Example 5.1 *Let E_1 and E_2 be the global executions in Example 3.2 and Example 4.1, respectively. As indicated in Figure 1, $E_1 \in (VQSR - (VSR \cup CQSR))$ and $E_2 \in (FQSR - (FSR \cup VQSR))$.*

Let E_3 and E_4 be global executions:

$$E_3 = \begin{cases} w_{g_1}(a) r_{l_1}(a) w_{g_2}(a) w_{g_2}(b) r_{l_1}(b) w_{l_1}(a) & at\ \mathcal{D}_1 \\ r_{g_1}(x) & at\ \mathcal{D}_2 \end{cases}$$

$$E_4 = \begin{cases} w_{l_1}(a) r_{g_1}(a) w_{g_1}(b) r_{g_2}(c) w_{g_2}(d) r_{l_1}(c) w_{l_1}(c) & at\ \mathcal{D}_1 \\ w_{g_1}(x) r_{g_2}(x) w_{g_2}(y) w_{g_2}(z) r_{g_1}(z) w_{l_2}(z) & at\ \mathcal{D}_2 \end{cases}$$

Then $E_3 \in (VSR - CQSR)$ and $E_4 \in (CQSR - FSR)$.

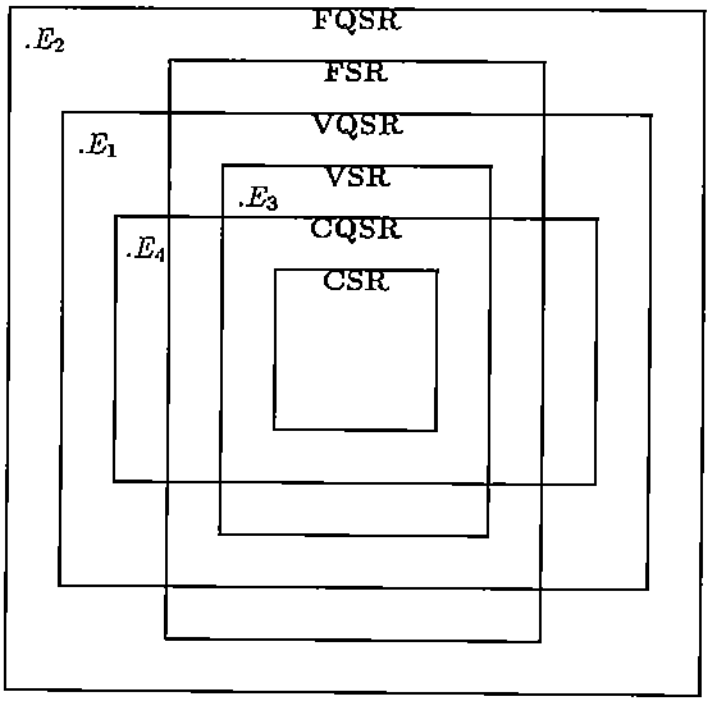

Figure 1: Relationships Between SR and QSR

References

[1] 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.

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

[3] 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.

[4] A. Elmagarmid and W. Du. Maintaining HDDBS consistency: The quasi serializability approach. Technical Report CSD-TR-1017, Purdue University, Sept. 1990.

[5] C. Papadimitriou. *The Theory of Database Concurrency Control.* Computer Science Press, 1986.