Ownership Transfer Towards Local Access in Geo-Replicated Database Systems

Paper Number: 344

A proof of correctness

We use following notations:

- Normally we use $A, B, \ldots, C$ or key to represent a key. Since a key may be deleted and then inserted, we use $A^{G,i}$ to refer the key A inserted at group G's i-th log entry. Thus, if on two nodes, A is observed, they may refer to different insertion.
- operation: change,insert,update,delete,read
- $C(A,G)$ means operation "change GID of key A to G". $C(A,G_1,G_2)$ means that the source group is G_2.
- $I(A,G)$ means operation "use GID G to insert key A". $I(A,B,G)$ means A is inserted using prev key B.
- $U(A)$ means operation "update key A".
- $D(A)$ means operation "delete key A".
- $R(A)$ means operation "read key A".
- Transaction, Log, belong, prev, lock
- We use $T[G]$ represents a transaction that started on the leader of group G. $OP \in T$ means transaction T has operation OP, which can be $C(A,G)$,$I(A,G)$,$U(A)$,$D(A)$,$R(A)$.
- $LK[T](A)$ means key A is locked by T on its start group.
- $BELONG[G](A,G_1)$ means on group G, key A belongs to G_1.
- $PREV[G](A,B)$ means on group G, key A's prev key is B.
- $LOG(G,i)$ means the i-th log entry in group G's log. The entry may have some dependency, and $LOG(G_1,i) \rightarrow LOG(G_2,j)$ requires that on any node, the latter log cannot apply until the first one applies. $T[G,i]$ means the transaction inside $LOG(G,i)$, and $\rightarrow$ can also be used on transaction T that has been replicated. A trivial conclusion is that $i \leq j$ leads to $LOG(G,i) \rightarrow LOG(G,j)$. The dependency is transitive using $\Rightarrow$.
- Suffix ^{R} for transaction T means "T is replicated".
- Suffix ^{A} for transaction T means "T is aborted".
- Suffix ^{C} for transaction T means "T is commited". A transaction can only be commited after replication.
- Suffix $\{G\}^E$ means "ends and releases all locks it acquired on the leader of group G". It can be either commit or abort.
- Sequence, happen before
- $X \triangleright Y$ represents the serialize order that X happens before Y. When using $\triangleright$ on a transaction, we let $T\{G_1\}$ to represent the following sequence:

$$T \ acuqires \ all \ its \ lock \ on \ G_1 \triangleright T\{G_1\}^E$$

Sometimes we only concern about T's operation on key A, thus we use $T(A)\{G_1\}$ to represent the following sequence:

$$LK[T](A)\{G_1\} \triangleright T\{G_1\}^E$$

Another acronym is that $T_1 \triangleright T_2$ means $T_1\{G\} \triangleright T_2\{G\}$ is true for all G.

Some trivial conclusions:

$$\frac{X \triangleright T[G]\{G\}}{X \triangleright T[G]\{G_1\}} \qquad \frac{T[G]\{G_1\} \triangleright X}{T[G]\{G\} \triangleright X}$$

$$\frac{LOG(G_1,i) \Rightarrow LOG(G_2,j)}{T_1[G_1,i] \triangleright T_2[G_2,j]} \qquad \frac{T_1[G_1]\{G_1\} \triangleright T_2[G_1]\{G_1\}}{T_1[G_1] \triangleright T_2[G_1]}$$

Define how an operation refer to an insertion

A transaction $T[G]$ may contains read, update, delete or change gid to key A. Since on the leader of G, all transactions' lock-unlock pair on A is serialized, thus T has its location in that serialization. Then there must exist a transaction $T_1[G_1,i]$ that updated A's applied index, and T_1 is either an insertion of A with $G_1 = G$ or a change of Gid from G_1 to G. In the first case, we say T operated on $A^{G_1,i}$. In the other case, we say T operated on the A that T_1 operated on.

An obvious conclusion is that all transactions that operated on $A^{G,i}$ have a total order in their $\triangleright$ relationship. Consider two different transaction $T_{a1}[G_{a1}]$ and $T_{b1}[G_{b1}]$ that operated on $A^{G,i}$. If $G_{a1} = G_{b1}$, their positions in G_{a1}'s log trivially defined T_{a1} and T_{b1}'s $\triangleright$ order.

Consider the apply index that $T_{a1}[G_{a1}]$ reads in DB: $[G_{a2}, i_{a2}]$, we have that $T_{a2}[G_{a2}, i_{a2}]$ is $T[G,i]$, or is a change of $A^{G,i}$'s gid from G_{a2} to G_{a1}. Both leads to $T_{a2} \triangleright T_{a1}$ (change gid case is due to constraint 2). By doing the same tracing on T_{a2} and T_{b1} till we meet $T[G,i]$, their will must be a first $T_{an} = T_{bm}$.

If $m,n > 1$, we have $G_{an-1} = G_{bm-1}$, and T_{an-1}, T_{bm-1}'s indexes in G_{an-1} must be different. Consider $T_{an-1} \triangleright T_{bm-1}$, T_{an-1} does not update the apply index in db, which means $n = 2$ and $T_{a1} \triangleright T_{b1}$. Else if one of m,n is 1, we still have $T_{a1} \triangleright T_{b1}$ or $T_{b1} \triangleright T_{a1}$.

Thus we have a trivial conclusion that the insertion $T[G,i]$ is at the head of the sequence, and if there is a deletion of $A^{G,i}$, it must be at the tail of that sequence.

Claim the trivial case for two leader reads same insertion is consistent. The first simple case is that it's impossible to have $A^{G,i}$ belong to two different group at the same time. Consider two transaction $T_1[G_1]$ and $T_2[G_2]$ respectively reads $BELONG[G_1](A,G_1)$ and $BELONG[G_2](A,G_2)$ at the same time t_0. Since T_1 and T_2 are not commited yet, they must be both at the tail of the $\triangleright$ order of all those transactions that operated on $A^{G,i}$:

$$\frac{OP(A^{G,i}) \in T[G_0]}{T\{G_1\} \triangleright LK[T_1](A) \qquad T\{G_2\} \triangleright LK[T_2](A)}$$

Thus consider the last commited transaction T_3 in $A^{G,i}$'s order, we have $T_3\{G_1\} \triangleright T_1\{G_1\}$ and $T_3\{G_2\} \triangleright T_2\{G_2\}$. However A cannot be both belong to G_1 and G_2 after T_3, which proves the case.

Here we claim a theorem.

Theorem 1. *Consider two transaction $T_1[G_1']$, $T_2[G_2']$ with $G_1' \neq G_2'$. It's impossible to have T_2 read*

$$BELONG[G_2'](key_2, G_2'), key_1 = key_2$$

or

$$BELONG[G_2'](key_2, G_2'), PREV[G_2'](key_1, key_2), key_1 \ absense \ on \ G_2'$$

at the same time when T_1 reads $BELONG[G_1'](key_1, G_1')$.

Proof. Let $T_{A_0}[G_{A_0}]$ and $T_{B_0}[G_{B_0}]$ represent T_1 and T_2, and A_1, B_1 represnet the key_1 and key_2 they read(the order is not defined).

If we trace A_1 and B_0's insertion prev key, then we get sequences

$$A_1^{G_{A_1},I_{A_1}}, A_2^{G_{A_2},I_{A_2}}, \cdots$$
$$B_1^{G_{B_1},I_{B_1}}, B_2^{G_{B_2},I_{B_2}}, \cdots$$

There must exist key $A_n^{G_{A_n},I_{A_n}}$ and$B_m^{G_{B_m},I_{B_m}}$ that $A_n = B_m$ and $G_{A_n}, I_{A_n} = G_{B_m}, I_{B_m}$, so let m,n to be the smallest pair that fits the statement. The case $m = n = 1$ has been discussed, thus we may assume $mn \neq 1$.

We use T_{A_i} to represent the transaction in $LOG(G_{A_i}, I_{A_i})$ that inserted $A_i^{G_{A_i},I_{A_i}}$. Thus we have

$$T_{A_i} \triangleright T_{A_{i-1}} \qquad i = n, n-1, \ldots, 1.$$

(sequence for B is similar).

Since $T_{A_{n-1}}$ and $T_{B_{m-1}}$ both did read operation on $A_n^{G_{IA_n},I_{IA_n}}$, then it's either

$$T_{B_{m-1}} \triangleright LK[T_{A_{n-1}}](A_n)$$

or

$$T_{A_{n-1}} \triangleright LK[T_{B_{m-1}}](B_m)$$

We can simply assume the first case, which makes $m > 1$.

or we can claim the case $m = n = 1$ here.

Now we try to group $A_{n-1}, A_{n-2}, \ldots, A_1$. Let k_i be the largest index having $A_{k_i} \geq B_i$(if equal, they refer to different insertion)($i = 1, \ldots, m-1$). If such k_i does not exist, we let $k_i = 0$. Note that k_i may be equal to k_{i-1}. for understanding, in other words, A_{k_i} is the smallest key having$A_{k_i} \geq B_i$. We also let $k_m = n-1$. Now we have

$$k_i \leq k_{i-1} \qquad \frac{X \triangleright T_{A_{k_i}}}{X \triangleright T_{A_{k_{i-1}}}} \qquad i = m, \ldots, 2$$

Thus we have $T_{A_{k_i}}$ can not see $B_i^{G_{B_i},I_{B_i}}$'s existcance.

Since $T_{B_{m-1}} \triangleright T_{A_{k_m}}$, we have $T_{B_{m-1}} \triangleright T_{A_{k_{m-1}}}$. Since $T_{A_{k_{m-1}}}$ does not see $B_{m-1}^{G_{B_{m-1}},I_{B_{m-1}}}$ exist, we have that $B_{m-1}^{G_{B_{m-1}},I_{B_{m-1}}}$ must have been deleted before $T_{A_{k_{m-1}}}$. Let B_{m-1} is deleted by $D(B_{m-1}) \in T_{DB_{m-1}}[G_{DB_{m-1}}, I_{DB_{m-1}}]$. According to constraint 3:

$$T_{DB_{m-1}} \triangleright T_{A_{k_{m-1}}}$$

Even if the deletion has been purged, this stands trivially since $T_{DB_{m-1}}$ has been applied on every node.

Another trivial fact is that

$$T_{B_{m-1}} \triangleright T_{B_{m-2}} \triangleright T_{DB_{m-1}}$$

Thus we have

$$T_{B_{m-2}} \triangleright T_{DB_{m-1}} \triangleright T_{A_{k_{m-1}}}$$

By using this method inductively on $T_{B_{m-2}} \triangleright T_{A_{k_{m-1}}}$, we finally have

$$T_{B_0} \triangleright T_{DB_1} \triangleright T_{A_{k_1}} \qquad T_{B_0} \triangleright T_{DB_1} \triangleright T_{A_0}$$

However T_{B_0} finds $B_1^{G_{B_1},I_{B_1}}$ exists, which means T_{DB_1} cannot even exist, thus we have contradiction. Or in other words, T_{B_0} and T_{A_0} can not exist at the same time. □

The correctness for failure case is based on Paxos:

Theorem 2. *At any point in time, for each record, there exists at most one Paxos group managing it*

Proof. For any change of group ID, consider the Paxos commit of it happens before or after the leader's failure. If the leader crashed before that changeGID commits, then no other region will learn that transaction. If after, Paxos will ensure that the changeGID transaction will be learned by the remaining region. Thus, there won't be more than one region managing the same record. □