Regular Paper

Concurrent Database Updates during Disconnection in Mobile Computing Environments

Yin-Huei Loh,† Takahiro Hara,† Masahiko Tsukamoto† and Shojiro Nishio†

In mobile computing environments, the limitations of mobile computers and wireless networks cause frequent disconnections among the hosts. Consequently, transactions on databases at disconnected sites cannot be executed smoothly without creating problems of inconsistency among copies of data. In this paper, we discuss how to minimize this problem. We assume that a database can be divided into clusters according to the data access pattern and that, for each cluster, the probability of transactions occurring is constant and known. We also assume that most of the disconnections are intentional and that no two disconnections can coexist. On the basis of the probability that transactions occur and the duration of the disconnection between the sites, our approach chooses whichever is more appropriate, the token method or the optimistic method, to control database updates. The token method enables a single site to execute transactions during disconnection, and thus ensures that no conflicts occur between the transactions. The optimistic method lets multiple disconnected sites execute transactions simultaneously. Conflicts are checked upon reconnection, and roll-back of transactions is performed if necessary. The evaluation functions of both methods are derived on the basis of the number of transactions expected to succeed during the disconnection period and the waiting time of these transactions before they can be committed. Whichever method gives a higher evaluation function value is chosen.

1. Introduction

In mobile computing environments, small portable computers are carried by users who can move around at will. These computers, also known as personal digital assistants (PDAs) or mobile hosts (MHs), have a wireless connection capability that enables their users to travel freely everywhere without being limited by the length of the wired cable. However, there are some limitations on these mobile hosts. Generally, mobile hosts run on batteries with a limited lifetime; a wireless network, though it is convenient and requires no physical connection, is costly and has limited bandwidth; and the connection has lower quality with more interference than a conventional wired network. When a mobile host is beyond the range of the wireless network, communication becomes impossible, and disconnection is said to have taken place. In other cases, to conserve energy, save network costs and reduce the network traffic in the wireless network with limited bandwidth, a mobile host may be disconnected intentionally, even though its location is within the range of the wireless network.

† Department of Information Systems Engineering, Graduate School of Engineering, Osaka University

The unique features of mobile environments are that the bandwidth for communication is limited and disconnections may occur. Therefore, if they are to serve as optimal platforms for database transaction execution, a few points need to be considered in such environments. Firstly, communication among mobile hosts should be minimized, since wireless communication is costly. This can be done by making every mobile host carry a data copy, a practice which is quite common in mobile environments. By means of this data copy, any transactions can be executed locally except during the initialization (to inform the network that it is updating the data) and the commit phase (to propagate the results).

Secondly, during disconnection, a mobile host is usually prohibited from initiating a transaction where updates are involved, to prevent inconsistency between data copies. Also, if disconnection occurs during the execution of a transaction, the transaction may need to be aborted, since communication capability is lost and the transaction cannot be committed.

Since conventional approaches disallow transactions in mobile hosts during periods of disconnection, the update ability of mobile hosts is greatly limited and this causes great inconvenience, especially to mobile hosts with high update frequency. Hence, there emerges a need to be able to update these data effectively and concurrently, even during disconnection.

As an example of an application in which data needs to be updated during disconnection, consider the schedule of a company manager. When the manager is out of the office, he may meet with clients who would like to fix a time to have a meeting with him the following week. At the same time, his secretary sitting in the office may receive phone calls from other clients who would also like to make appointments with the manager for the following week. In this scenario, an effective way is needed to allow both parties to update the schedule without having to contact each other, especially if the manager is overseas.

Much work has been done to solve the above problem. References 3)∼5) focus on file systems. For database systems, several approaches have been proposed in Refs. 1) and 11). However, transactions frequently have to be rolled back as conflicts may occur among them, resulting in a heavy workload.

References 6), 8)∼10) proposed strategies that partition the data values and allocate them to different sites for data items that are partitionable and in which transactions executed on those items are commutative. For example, a data value of 100 can be partitioned into 60 and 40 and allocated to two sites. This idea has been further extended for mobile computing environments in Ref. 7). One drawback of this approach is that it can only be applied to partitionable data items, and thus its applicability is limited.

In Ref. 12), several update propagation protocols for replicated databases with basically a lazy approach using graph theory were proposed. In that paper, it is assumed that for each data item, there is a primary copy and several secondary copies, and that only the site which holds the primary copy can update the data. In our paper, we eliminate the limitation that only the primary site can update the data by not fixing a primary site statically.

We try to raise the data update ability of transactions in mobile hosts while minimizing the rollback cost. We investigate the characteristics of two commonly used token and optimistic methods and derive a mathematical formula for determining which method has the better performance, according to the situation. Our approach uses a trade-off between the number of successful transactions and the work efficiency to choose the appropriate strategy.

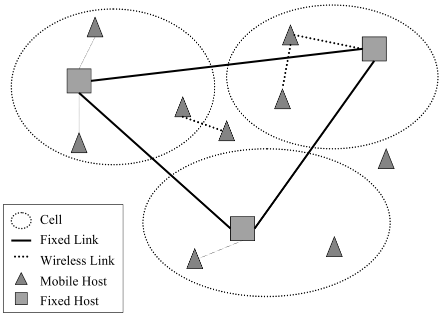

Fig. 1 Example of a system architecture consisting of fixed and mobile hosts.

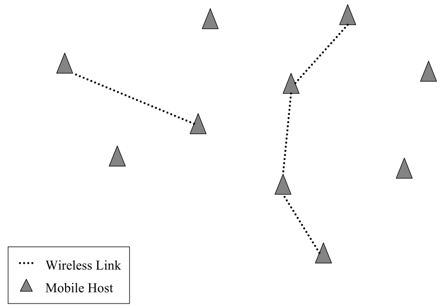

Fig. 2 Example of a system architecture: an ad hoc network consisting of only mobile hosts.

The remainder of the paper is organized as follows. In Section 2, we describe the system architecture. In Section 3, we explain the approach of our proposed algorithm in detail. We evaluate and discuss the proposed algorithm in Sections 4 and 5, and conclude the paper in Section 6 with some discussion of future work.

2. System Architecture

In this section, we describe the system architecture of our proposed algorithm.

We assume an environment consisting of mobile hosts with or without fixed hosts. We do not differentiate between mobile hosts and fixed hosts. Each host carries a copy of the database and can be connected to any other hosts. As the example in **Fig. 1** shows, even if a mobile host is in a cell within wireless communication range of a fixed host (mobile support station), it can voluntarily disconnect from the fixed host. The mobile hosts can also communicate freely among themselves. In the case shown in **Fig. 2**, the network may be an ad hoc one containing only mobile hosts that move freely and communicate freely with any other mobile hosts around.

Disconnections is said to have taken place between two hosts when these two hosts disconnect from each other and there is no other route connecting them; that is, when there is no direct or indirect way in which they can communicate with each other.

Disconnections are assumed to occur one at a time, sequentially, but not simultaneously. Thus, upon disconnection, the network is separated into two different networks, which we consider as two sites. Each site may consist of one or more mobile hosts and may be further separated into two different sites. This process may be repeated recursively, resulting in many disconnected sites. For the purpose of simplicity, we consider the case in which no two disconnections coexist. Thus, there are at most two sites in the whole network.

We assume that all disconnections are intentional and that the duration of disconnection is fixed and known. Later in this paper, we also show that the algorithm can be applied with little error when the duration of disconnection varies from the expected value. Even if a disconnection is unintentional, the reconnection time can be predicted by using historical data or the speeds and directions of motion of the mobile hosts. Thus, the algorithm can also be applied to unintentional disconnections.

Let us assume that the database is partitioned into one or more clusters according to the data access patterns of the transactions. Data items that are often accessed together in the same transactions are clustered together. The probability of the occurrence of transactions for each cluster at each host can be determined beforehand by referring to historical data or a schedule. We assume that the probability is constant.

Our algorithm focuses on only one data cluster. This is to enable independent transactions to be handled separately so that more transactions can be executed at the same time. The whole database can be handled by running the algorithm on all the data clusters. All the descriptions below apply to a single data cluster.

We also assume that transactions are executed by using a two-phase commit protocol as in a distributed environment. When a site consists of multiple hosts where more than one host hold the same data copy, two-phase commit is used for any updates to this data copy in this site. When a transaction is executed at a host, locks must be obtained from all connected hosts that hold the data copies updated in this transaction, and after the transaction has been committed, the result is propagated to all hosts that hold the data copies to ensure data consistency. Thus, even though many hosts may own the same data copy in a site and transactions may be executed at different hosts, no conflicts will occur.

Assume that in a very small unit of time, at most one single transaction can occur at a single site. Thus, the number of transactions, x, that can occur in a unit of time at any site is such that $0 \le x \le 1$, where

x = *the probability of the occurrence of one transaction per unit of time at a particular site.*

Let $P(k)_i^t$ denote the probability that i transactions occur at site k in t units of time ($0 \le i \le t$). Since at most one transaction can occur at a site in one unit of time, the probability that one transaction occurs at site k in one unit of time is

$$P(k)_1^1 = x,$$

and the probability that no transaction occurs at site k in one unit of time is

$$P(k)_0^1 = 1 - P(k)_1^1.$$

In general, the probability that i transactions occur at site k during time t is:

$$P(k)_i^t = {}^tC_i(P(k)_1^1)^i(P(k)_0^1)^{(t-i)}. \qquad (1)$$

(There are ${}^tC_i$ ways to arrange i transactions in t units of time.)

To decide whether the two disconnected sites will be allowed to run transactions on the data cluster or not during disconnection, an algorithm is used to choose whichever is better, the token method or the optimistic method, before the disconnection. In Ref. 2), the concepts of optimistic strategies and pessimistic strategies are surveyed as means for maintaining consistency in partitioned networks. The optimistic method basically allows execution of transactions in different partitions and uses version vectors or precedence graphs (by detecting cycles) to ensure database consistency. "Tokens," one of the pessimistic strategies, allows only the partition that holds the token (which is determined dynamically) to access the data. The definitions of both methods in this paper are as follows:

- Token method: A "token" is used to represent the right to execute transactions. Upon disconnection, the site with a higher probability of the occurrence of transactions will receive the token, and only this site with the token is granted permission to run transactions on the data cluster (followed by a commit action), while the other site (without token) is prohibited from running transactions on the data cluster. We assume that each host holds a unique priority value which can be decided in many ways, such as comparing the host ID. When the probabilities of the occurrence of transactions are the same for both sites, the priority values of the two hosts involved in the disconnection are compared and the site that contains the host with the higher priority value receives the token.

- Optimistic method: Both sites can run transactions on the data cluster, but it is not guaranteed that these transactions can be committed. In other words, during the disconnection, if transactions occur at only one of the sites, these updates will succeed when the sites reconnect. On the other hand, if conflicts occur; that is, if transactions occur at both sites, rollback has to be performed at the site where fewer transactions occurred.

Note that in this paper, the term "conflict of transaction" can mean not only a conflict caused by multiple "write" transactions, but also a conflict caused by conflicting "read" and "write" transactions. We do not intend to discuss this in further detail, but in principle conflict is defined by the serializability of the transactions.

Disconnection occurs when the connection between two hosts is broken. These two hosts will then decide what method to use. Since we assume that disconnection is intentional, the method can be decided before disconnection. It can alternatively be decided after disconnection as long as the two hosts know the probability of the transactions occurring at the other disconnected site and the priority value of the other host. This can be easily achieved by exchanging data on the probability that transactions occur and the priority value once before disconnection. The two hosts are also responsible for propagating the result, that is, the method chosen, to the other hosts at the same site which hold copies of the involved data cluster.

In Fig. 1, if a mobile host disconnects from a fixed host, this mobile host and the fixed host are responsible for deciding the method to be used. If both parties are mobile hosts (in Fig. 1 or Fig. 2), the mobile hosts will be responsible for that. Note that if a mobile host disconnects from a fixed host but can still access the fixed host through other mobile hosts or fixed hosts, they are not disconnected, as they are still in the same site (connected network), and are able to communicate with each other.

3. Algorithm Description

In this section, we discuss how to choose the more appropriate method.

Let us consider the number of transactions that succeed in a range of time (i.e., the expected duration of disconnection) for each method. By "succeed," we mean that the transactions are committed either on the spot or upon reconnection. Even though it is predicted that the number of transactions that occur at the site with a higher probability of the occurrence of transactions would exceed that at the site with a lower probability of the occurrence of transactions, in real life, there is always a possibility that the reverse may happen.

In this case, use of the optimistic method will ensure a higher number of successful transactions, because it decides which transactions will succeed only after they have occurred, whereas the token method decides before the transactions occur. However, use of the optimistic method requires much more work to execute all the transactions on both sites and later roll back some of these transactions. Moreover, in the optimistic method, transactions executed have to wait for a certain amount of time (i.e., from the time at which the transaction is executed to the time at which the site is reconnected) before they can be committed. In the token method, this waiting time is zero, since the transactions are committed immediately after they are executed. Thus, when the expected numbers of successful transactions for both methods are the same, use of the token method is obviously more efficient, as it requires less work and no waiting time. Our algorithm quantifies the characteristics of both methods.

Since the optimistic method requires a waiting time, in contrast to the token method, we introduce a function $f(x)$ to define the satisfaction level of each successful transaction according to this waiting time. The satisfaction level of each successful transaction decreases when x increases, where x is the duration of the period between the time at which the transaction happens and the time at which it commits. For example, when a transaction occurs at time 1 and commits at time 4, $x = 4 - 1 = 3$. Thus, using parameter K $(0 < K < 1)$ as a constant, we define $f(x)$ as:

$$f(x) = \begin{cases} 1 - Kx & x < \frac{1}{K} \\ 0 & \text{otherwise.} \end{cases} \quad (2)$$

Integrating this function $f(x)$, we define the evaluation function as the summation of the satisfaction levels for all successful transactions relating to the duration of disconnection. We represent this evaluation function by the symbol $F(T)$ for the token method and $F(O)$ for the optimistic method.

For the token method, since transactions always commit soon after they are executed, the waiting time is always 0 and the satisfaction level is always 1. Thus, $F(T)$ equals the number of transactions that succeed in the corresponding duration of disconnection. This is equal to the number of transactions which occur at the site with a higher probability of the occurrence of transactions (which is the site that will receive the token), and thus is equal to the summation of the products of (i) the number of transactions, and (ii) the probability that this number of transactions occur. When two sites A and B exist, and when $P(A)^1_1 > P(B)^1_1$,

$$\begin{aligned} F(T) &= 1P(A)^t_1 + 2P(A)^t_2 + \ldots + tP(A)^t_t \\ &= \sum_{i=1}^{t} iP(A)^t_i \\ &= tP(A)^1_1. \end{aligned}$$

(We omit the proof here.)

Applying the same concept for $P(B)^1_1 > P(A)^1_1$, we obtain the following equation:

$$F(T) = \begin{cases} tP(A)^1_1 & P(A)^1_1 > P(B)^1_1 \\ tP(B)^1_1 & \text{otherwise.} \end{cases} \quad (3)$$

We now derive $F(O)$. For the optimistic method, the satisfaction level is always less than 1, since a successful transaction only commits when the sites are reconnected. Transactions that occur at different times have different satisfaction levels. Thus, we need to consider each of them separately.

From Eq. (1), we know that:

$$P(A)^t_i = {}^tC_i (P(A)^1_1)^i (P(A)^1_0)^{(t-i)}.$$

Thus, there are ${}^tC_i$ possible patterns in which i transactions occur in the duration of disconnection t. Thus, the probability of each pattern occurring is $P(A)^t_i / {}^tC_i$.

Table 1 Example of calculating the instances of $P(A)^1_1$ for $P(A)^t_i$ using $P(A)^4_2$.

Pattern \ Time	1	2	3	4	Total $P(A)^1_1$
1	1	1	0	0	2
2	1	0	1	0	2
3	1	0	0	1	2
4	0	1	1	0	2
5	0	1	0	1	2
6	0	0	1	1	2
Total $P(A)^1_1$	3	3	3	3	12

The formula of $P(A)^t_i$ can be expanded into ${}^tC_i$ terms, in which each term equals $(P(A)^1_1)^i (P(A)^1_0)^{(t-i)}$, and thus contains i instances of $P(A)^1_1$ and $(t-i)$ instances of $P(A)^1_0$. Hence, when we consider the total number of all ${}^tC_i$ terms, there are $i \times {}^tC_i$ instances of $P(A)^1_1$. These $i \times {}^tC_i$ instances of $P(A)^1_1$ are also evenly distributed over the time $1, 2, \cdots, t$. Thus, for each time $1, 2, \cdots, t$, the total number of instances of $P(A)^1_1$ is $(i \times {}^tC_i)/t$.

As an example, consider the case of $P(A)^4_2$ $(t = 4$ and $i = 2)$. There are ${}^4C_2$ $(= 6)$ patterns to arrange 2 transactions in 4 units of time as shown in **Table 1**. The probability that each of these 6 patterns occurring is $P(A)^4_2/6$. For each, there exist 2 instances of $P(A)^1_1$, which add up to a total of $(2 \times 6)P(A)^1_1$, that is, $12P(A)^1_1$. These 12 instances of $P(A)^1_1$ are evenly distributed over the time $1, 2, 3, 4$. Thus, for each time $1, 2, 3, 4$, there are $(12/4)P(A)^1_1$, that is, $3P(A)^1_1$.

The actual number of transactions that occur at each time $1, 2, \cdots, t$ is equal to the number of transactions multiplied by the probability that this number of transactions occur (as in the token method):

$$\frac{i \times {}^tC_i}{t} \times \frac{P(A)^t_i}{{}^tC_i} = P(A)^t_i \left(\frac{i}{t}\right).$$

We now consider the satisfaction level of the transactions that occur at each time $1, 2, \cdots, t$. For a duration of disconnection t, if disconnection occurs at time 1, then reconnection will occur at time $t + 1$. Recall from Eq. (2) that x is the duration of the period from the time at which the transaction occurs to the time at which it commits. In this case, we know that when a transaction occurs at:

time 1, $x = t$,
time 2, $x = t-1$,
$\vdots$
time t, $x = 1$.

Hence, by referring to Eq. (2), the satisfaction levels $f(x)$ for transactions that occur at each of the times 1 to t are:

$$f(t) = \begin{cases} 1 - tK & t < \frac{1}{K} \\ 0 & \text{otherwise}, \end{cases}$$

$$f(t-1) = \begin{cases} 1 - (t-1)K & t-1 < \frac{1}{K} \\ 0 & \text{otherwise}, \end{cases}$$

$$\vdots$$

$$f(1) = 1 - K.$$

Thus, the total satisfaction function, $S(A)^t_i$, is the summation of the satisfaction levels for the transactions that occur as follows:

$$S(A)^t_i = P(A)^t_i \left(\frac{i}{t}\right) \times \sum_{j=1}^{t} f(j)$$

$$= \begin{cases} P(A)^t_i \left(\frac{i}{t}\right) \times \sum_{j=1}^{t} (1 - jK) & t < \frac{1}{K} \\ P(A)^t_i \left(\frac{i}{t}\right) \times \sum_{j=1}^{\lfloor \frac{1}{K} \rfloor} (1 - jK) & \text{otherwise} \end{cases}$$

$$= \begin{cases} P(A)^t_i \left(\frac{i}{2}\right) (2 - (t+1)K) & t < \frac{1}{K} \\ P(A)^t_i \left(\frac{i}{t}\right) \times \sum_{j=1}^{\lfloor \frac{1}{K} \rfloor} (1 - jK) & \text{otherwise}. \end{cases}$$

For $F(T)$, the evaluation function corresponding to each i $(= 1, 2, \cdots, t)$ is $iP(A)^t_i$. For $F(O)$, the satisfaction level is integrated and the evaluation function for each i is $S(A)^t_i$.

Thus, when transactions occur at only one site,

$$F(O) = P(B)^t_0 \sum_{i=1}^{t} S(A)^t_i + P(A)^t_0 \sum_{i=1}^{t} S(B)^t_i.$$

When transactions occur at both sites, as mentioned earlier, transactions occurring at the site with more transactions will succeed, while transactions at the other site will have to undergo rollback. Thus, for this case, when the number of transactions at site A exceeds that at site B,

$$F(O) = \sum_{i=1}^{t} \sum_{j=1}^{i} S(A)^t_i P(B)^t_j.$$

When the reverse happens,

$$F(O) = \sum_{i=1}^{t} \sum_{j=i}^{t} P(A)^t_i S(B)^t_j.$$

Thus, we obtain the following result:

$$\begin{aligned} F(O) = {} & P(B)^t_0 \sum_{i=1}^{t} S(A)^t_i \\ & + P(A)^t_0 \sum_{i=1}^{t} S(B)^t_i \\ & + \sum_{i=1}^{t} \sum_{j=1}^{i} S(A)^t_i P(B)^t_j \\ & + \sum_{i=1}^{t} \sum_{j=i}^{t} P(A)^t_i S(B)^t_j. \end{aligned} \tag{4}$$

When $F(T) > F(O)$, the token method is used, and vice versa.

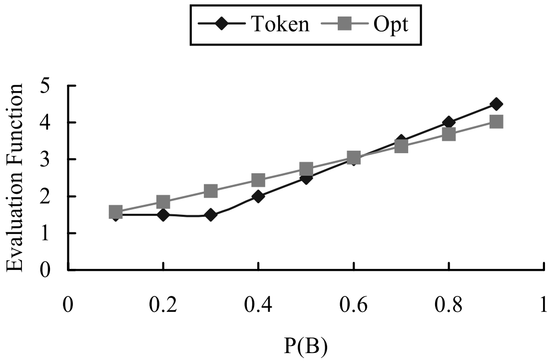

Fig. 3 Evaluation function versus $P(B)^1_1$ when $P(A)^1_1 = 0.3$, $K = 0.04$, and $t = 5$.

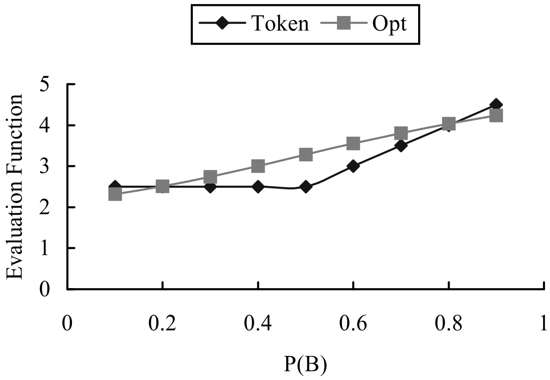

Fig. 4 Evaluation function versus $P(B)^1_1$ when $P(A)^1_1 = 0.5$, $K = 0.04$, and $t = 5$.

4. Evaluation

In this section, we will evaluate the algorithm we proposed in Section 3.

In **Figs. 3**, **4** and **5**, we show how the val-

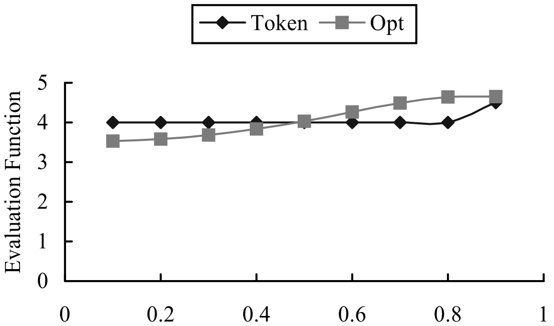

Fig. 5 Evaluation function versus $P(B)_1^1$ when $P(A)_1^1 = 0.8$, $K = 0.04$, and $t = 5$.

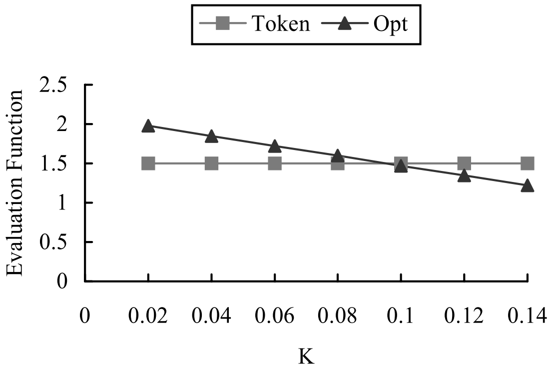

Fig. 6 Evaluation function versus K with the following parameters: $P(A)_1^1 = 0.3$, $P(B)_1^1 = 0.2$, $t = 5$.

ues of the evaluation functions of the token and optimistic methods change when we fix the parameters $P(A)_1^1$, K, and t (duration of disconnection) while varying the value of $P(B)_1^1$. K and t are fixed as 0.04 and 5, respectively, in all cases. In Figs. 3, 4 and 5, the values of $P(A)_1^1$ are set at 0.3, 0.5 and 0.8, respectively. From the graphs, it can be observed that the optimistic method gives a higher evaluation function value when the probabilities that transactions occur at two sites are closer; that is, in each figure, the optimistic method gives a higher value when the value of $P(B)_1^1$ is close to the value of $P(A)_1^1$ (when $P(B)_1^1$ is around 0.3 in Fig. 3, 0.5 in Fig. 4, and 0.8 in Fig. 5).

For both the token and optimistic methods, it is important that we consider the trade-off between the work and the results.

The advantage of the token method is that the success or failure of transactions is known on the spot without any delay. Only the site with the token is allowed to execute transactions. In other words, only transactions that will succeed are executed, while transactions that will not succeed are rejected straight away. Thus, there is no waste of work at all in executing the transactions. The efficiency is 100%.

However, there is no chance at all for a site without a token to execute a single transaction. Thus the site with the lower probability of the occurrence of transactions never has a right to execute transactions during disconnection.

In the optimistic method, transactions are tentatively executed in the hope that eventually they will succeed. In some cases, they may succeed if no conflict occurs. This happens mostly when the probability of the occurrence of transactions is not high and/or the duration of disconnection is short.

However, the transactions executed may fail if conflicts occur, mostly in cases where the probability of the occurrence of transactions is high and/or the disconnection period is relatively long. Further, if the disconnection period is long, many uncommitted transactions accumulate and eventually they may all fail, resulting in a considerable waste of time and work.

It is hard to quantify the work of transaction rollback in the optimistic method and compare it with the loss of not being able to execute transactions in the token method. There is no exact boundary between these two methods in which we can say that "the token method is better than the optimistic method" or vice versa. It all depends on the circumstances.

However, by understanding the above phenomena, that is, the merits and demerits of both methods, a user can set his own priority and decide a boundary to define his own evaluation function. This is where the satisfaction level $f(x)$ and the constant parameter K have roles to play.

The value of the constant K may be changed depending on which is more important: the number of transactions that succeed or the efficiency (result per work). The higher the value of K, the higher the probability that token method is applied. Users may set their own values of K to suit their own requirements.

As an example, in the graph shown in **Fig. 6**, the values of the probability of the occurrence of transactions at sites A and B, and the duration of disconnection t, are fixed at $P(A)_1^1 = 0.3$, $P(B)_1^1 = 0.2$, and $t = 5$, and K is changed to reflect the change in the values of the evaluation functions for the token and optimistic methods (refer to Eqs. (3) and (4)). As can be seen in the graph, the value of the evaluation

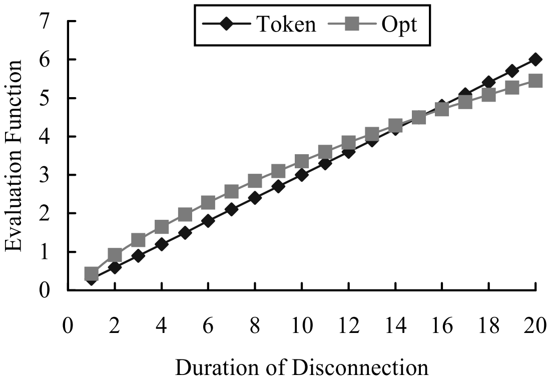

Fig. 7 Evaluation function versus duration of disconnection with the following parameters: $P(A)_1^1 = 0.3$, $P(B)_1^1 = 0.2$, $K = 0.02$.

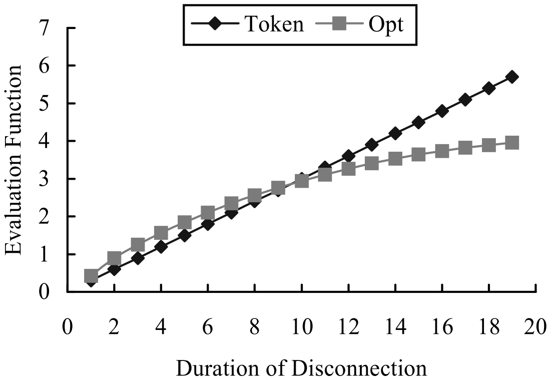

Fig. 8 Evaluation function versus duration of disconnection with the following parameters: $P(A)_1^1 = 0.3$, $P(B)_1^1 = 0.2$, $K = 0.04$.

function for the token method remains constant as K has no effect on it, while the value of evaluation function for the optimistic method decreases with the increase of the value of K. This graph shows us that for the same set of parameters, when $K \leq 0.095$ (approximately), the algorithm chooses the optimistic method, while when $K > 0.095$, it chooses the token method.

The graphs in **Figs. 7** and **8** show how the value of the evaluation function for each method is affected by a change of the duration of disconnection. The values of the probability of the occurrence of transactions at site A and B are fixed at $P(A)_1^1 = 0.3$ and $P(B)_1^1 = 0.2$ in both cases, while the values of K are fixed at 0.02 and 0.04, respectively. From the graphs we can see that, with different values of K, the algorithm gives a different result for the choice of method, since the value of the evaluation function of the optimistic method is affected by the change of value K.

As a guideline for choosing the value of K, for applications that cannot tolerate waiting time and need to be committed urgently, it is appropriate to set the value of K higher. Examples include applications involving money, such as systems for business trading, budget management, and bank account management. On the other hand, a small value of K is suitable for applications such as schedule management systems, since these are not time-critical. Moreover, transactions in these systems are relatively simple, and thus rollback can be easily performed.

Compared with the approaches that use only one method, either token or optimistic, ours is certainly better, as it balances the merits and demerits of both methods and allows the users a choice to decide the more appropriate method by considering the probability that transactions occur and the duration of disconnection between the sites.

5. Further Discussion

So far, we have assumed that the duration of disconnection is known before the disconnection. In this section, we investigate the effectiveness of our algorithm when the duration of disconnection deviates from the expected value.

When the duration of disconnection varies, for an average duration of disconnection, T, the actual duration of disconnection may be more or less than T. In our approach, T is used as the approximate duration of disconnection to calculate the values of evaluation functions of each method, thus the same method will always be chosen. However, ideally, the exact duration of disconnection which may deviate from T should be used to calculate the values of evaluation functions, where a different method my be chosen. We evaluate the difference by comparing the total evaluation function values for both cases.

We first consider our approach. Using different distribution functions, we consider all possible values of duration of disconnection t. For each t, we multiply the probability that the duration of disconnection equals t by the evaluation function of the method chosen for this t (the same method is always chosen, since the average duration of disconnection T is used). The summation of the results of this calculation for all t will give us the "actual" value.

The "ideal" value is calculated as follows. For each t, we multiply the probability that the duration of disconnection equals t by the evalu-

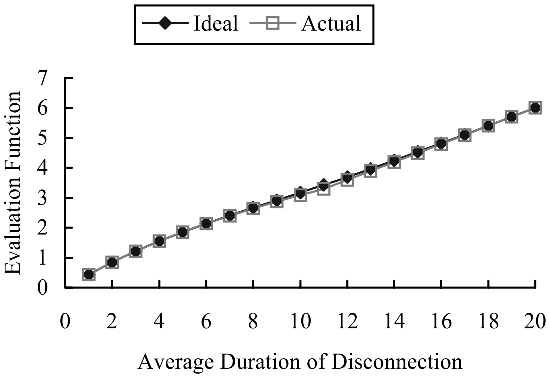

Fig. 9 Evaluation function versus average duration of disconnection for the Poisson distribution.

ation function of the method that should be chosen when this occurs (which may be different from the method chosen in our approach). The summation of the results of this calculation for all t gives us the "ideal" value.

The probability values for different durations of disconnection t are calculated by means of four kinds of different distribution functions—Poisson, normal, uniform, and exponential—using the mean as the average disconnection time.

In the Poisson distribution, the probability density function is

$$f(x) = \begin{cases} \dfrac{e^{-\lambda}\lambda^x}{x!} & x \in \{0, 1, \ldots\} \\ 0 & \text{otherwise} \end{cases},$$

where the mean is λ.

In the normal distribution, the probability density function is

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}},$$

where the mean is μ and the standard deviation is σ.

In the uniform distribution, the probability density function is

$$f(x) = \begin{cases} \dfrac{1}{b-a} & a \le x \le b \\ 0 & \text{otherwise} \end{cases},$$

where the mean is $(a+b)/2$ and the range of x is $[a, b]$.

In the exponential distribution, the probability density function is

$$f(x) = \begin{cases} \dfrac{1}{\beta} e^{-x/\beta} & x > 0 \\ 0 & \text{otherwise} \end{cases},$$

where the mean is β.

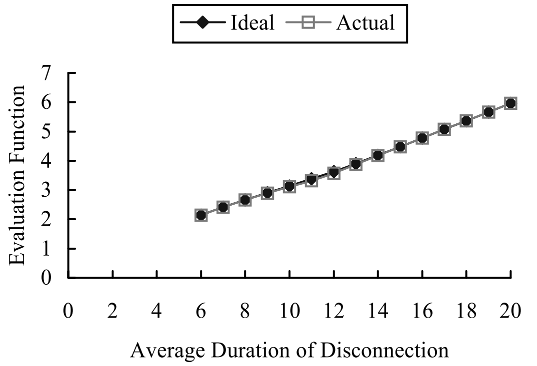

Fig. 10 Evaluation function versus average duration of disconnection for the normal distribution with a standard deviation of 2 units of time.

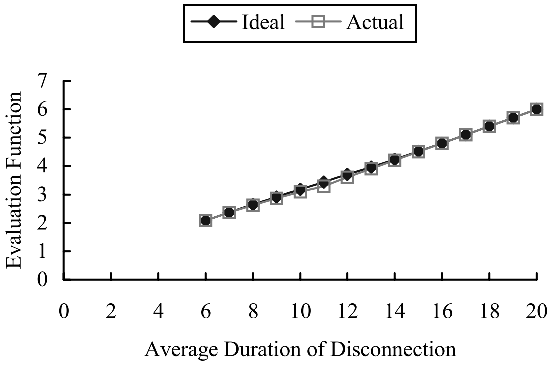

Fig. 11 Evaluation function versus average duration of disconnection (t) for the uniform distribution with a duration range of $[t-5, t+5]$.

Using these values, the actual and ideal values are calculated and shown in **Figs. 9** (Poisson distribution), **10** (normal distribution), **11** (uniform distribution) and **12** (exponential distribution). For Figs. 10 and 11, the values of the evaluation function for durations of disconnection below 0 are undefined, and thus the evaluation function is undefined for average duration of disconnection close to 0. The parameters are set as follows for all cases: $P(A)_1^1 = 0.3$, $P(B)_1^1 = 0.2$, $K = 0.03$.

As can be seen from the graphs, the difference between the actual values and the ideal values is very small. The difference increases as the duration of disconnection approaches the point at which the values of the evaluation functions for the token and optimistic methods intersect. By calculation, we found that the percentage of difference for each duration of disconnection ($(ideal - actual)/ideal$) is less than 5% for the Poisson, normal, and uniform distributions, and less than 10% for the exponential distribution.

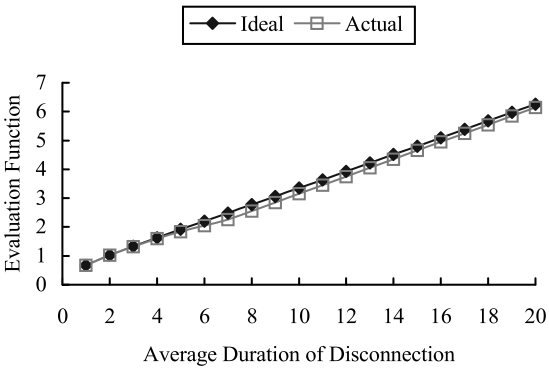

Fig. 12 Evaluation function versus average duration of disconnection for the exponential distribution.

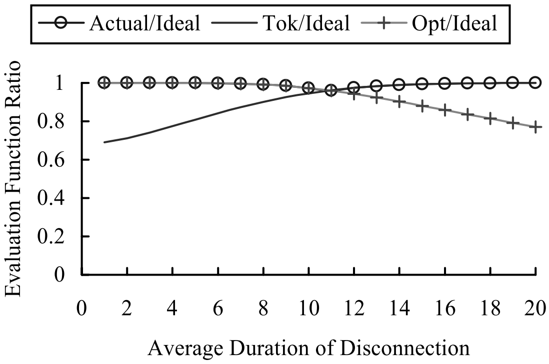

Fig. 13 Evaluation function ratio versus average duration of disconnection for the Poisson distribution.

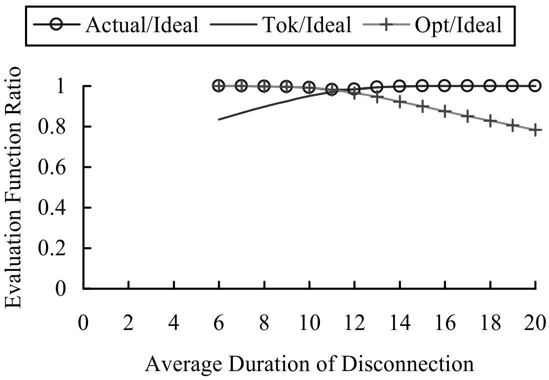

Fig. 14 Evaluation function ratio versus average duration of disconnection for the normal distribution with a standard deviation of 2 units of time.

Moreover, in **Figs. 13**, **14**, **15** and **16**, using the values when only the token method is used (Tok) and when only the optimistic method is used (Opt), the ratios of $Actual/Ideal$, $Tok/Ideal$ and $Opt/Ideal$ are calculated for each duration of disconnection. From the results shown in the graphs, it is verified that even when the proposed algorithm's performance deteriorates as a result of the deviation of the duration of disconnection, as long as the deviation is not extreme and follows a common probability distribution, it still outperforms approaches that use only a single method, either the token method or the optimistic method.

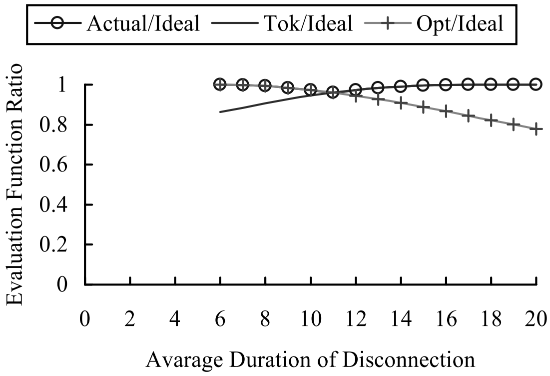

Fig. 15 Evaluation function ratio versus average duration of disconnection (t) for the uniform distribution with a duration range of $[t-5, t+5]$.

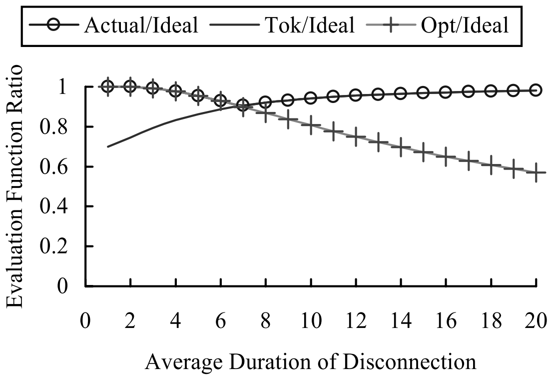

Fig. 16 Evaluation function ratio versus average duration of disconnection for the exponential distribution.

From the above, we can conclude that when the duration of disconnection is not fixed but deviates from the expected value, our algorithm can still work well and with little error, using the average value. In mobile computing environments, when a mobile host moves beyond the range in which wireless communication is possible, this mobile host is forced to disconnect from other hosts. Using available statistical information such as the movement speed, direction, and current location of the mobile host, we can estimate the expected reconnection time. By periodically exchanging these data with other hosts, a mobile host can calculate the estimated duration of disconnection and decide which data update method to use after disconnection. As long as the actual reconnection time does not vary too drastically from the average value, the algorithm can be applied.

Thus, we conclude that the proposed algorithm will work for unintentional disconnections.

6. Conclusions

In this paper, we have used two approaches, the token method and the optimistic method, to handle database updates during periods of disconnection. The former predicts the number of transactions that will occur based on the probability of the occurrence of transactions, and then decides which site is allowed to execute transactions. The latter uses an optimistic approach in which all the sites are allowed to run transactions, but rollback may occur if there are any conflicts. The latter always ensures that the maximum number of transactions succeed, but it requires more work.

We then proposed an algorithm for determining which of these two methods is more effective during disconnection, depending on the probability of transactions occurring and the duration of disconnection. In order to choose the better method, the evaluation function in our algorithm uses the number of transactions that succeed and the satisfaction level of each transaction. In the formula of our proposed algorithm, users can set the appropriate parameter K to suit their own requirements depending on their priority.

We also showed that the algorithm works well and with little error when the duration of disconnection deviates from the average value. In either case, our approach outperforms approaches that use only a single method without considering the situation.

In this paper, we have considered only the case in which a network is divided into two sites as a result of disconnection. For multiple disconnected sites, we think the same concept can be applied by using the probability that transactions occur and the status of connections among hosts. When a transaction occurs, we assume that it can be committed if the site is connected to a majority of the other sites. By estimating the duration of the waiting time (from the time at which the transaction happens to the time at which the site will be able to commit the transaction), we can use the same formula to determine the method to be used.

In addition to the token and optimistic methods, we would like to include the Escrow method[6)~10)] for partitionable data in our algorithm, to make it more robust in the future.

Acknowledgments This research was supported in part by the Research for the Future Program of Japan Society for the Promotion of Science under the Project "Advanced Multimedia Content Processing (Project No.JSPS-RFTF97P00501)".

References

1) Breitbar, Y. and Korth, H.F.: Replication and Consistency: Being Lazy Helps Sometimes, *Proc. 16th ACM SIGACT-SIGMOD-SIGART Symposium on PODS*, pp.173–184 (1997).

2) Davidson, S.B., Garcia-Molina, H. and Skeen, D.: Consistency in Partitioned Networks, *Computing Surveys*, Vol.17, No.3, pp.341–370 (1985).

3) Demers, A., Petersen, K., Spreitzer, M., Terry, D., Theimer, M. and Welch, B.: The Bayou Architecture: Support for Data Sharing among Mobile Users, *Proc. Workshop on Mobile Computing Systems and Applications*, pp.2–7 (1994).

4) Heidemann, J.S., Page, T.W., Guy, R.G. and Popek, G.J.: Primarily Disconnected Operation: Experiences with Ficus, *Proc. Second Workshop on Management of Replicated Data*, pp.2–5 (1992).

5) Kistler, J.J. and Satyanarayanan, M.: Disconnected Operation in the Coda File System, *ACM Trans. Comput. Syst.*, Vol.10, No.1, pp.3–25 (1992).

6) Krishnakumar, N. and Bernstein, A.J.: High Throughput Escrow Algorithms for Replicated Databases (Extended Abstract), *Proc. 18th International Conference on VLDB*, pp.175–186 (1992).

7) Krishnakumar, N. and Jain, R.: Escrow Techniques for Mobile Sales and Inventory Applications, *Wireless Networks 3*, Vol.3, No.3, pp.235–246 (1997).

8) Kumar, A.: An Analysis of Borrowing Policies for Escrow Transactions in a Replicated Data Environment, *Proc. 6th International Conference on Data Engineering*, pp.446–454 (1990).

9) Kumar, A. and Stonebraker, M.: Semantics Based Transaction Management Techniques for Replicated Data, *Proc. ACM SIGMOD'88*, pp.117–125 (1988).

10) O'Neil, P.E.: The Escrow Transactional Method, *ACM Trans. Database Syst.*, Vol.11,

No.4, pp.405–430 (1986).
11) Pitoura, E. and Bhargava, B.: Maintaining Consistency of Data in Mobile Distributed Environments, *Proc. 15th International Conference on Distributed Computer System*, pp.404–413 (1995).
12) Breitbart, Y., Komondoor, R., Rastogi, R. and Seshadri, S.: Update Propagation Protocols for Replicated Databases, *Proc. ACM SIGMOD'99*, pp.97–108 (1999).

(Received May 8, 2000)
(Accepted March 9, 2001)

Yin-Huei Loh received her bachelor degree in Computer Science from University of Science, Malaysia, in 1997 and M.E. degree in Information Systems Engineering from Osaka University, Japan, in 2001. She is currently a Ph.D. candidate in the Department of Information Systems Engineering, Osaka University. Her research interests include database systems in mobile computing environments, virtual reality systems and ad hoc networks. She is a student member of the IEEE and ACM.

Takahiro Hara received his B.E., M.E. and Dr.E. degrees in Information Systems Engineering from Osaka University, Japan, in 1995, 1997 and 2000, respectively. Currently, he is a research associate in the Department of Information Systems Engineering, Osaka University. His research interests include distributed database systems in advanced computer networks such as high-speed ATM networks and mobile computing environments. He is a member of three learned societies, including IEEE.

Masahiko Tsukamoto received his B.E., M.E. and Dr.E. degrees from Kyoto University, Japan, in 1987, 1989 and 1994, respectively. From 1989 to February 1995, he was a research engineer of Sharp Corporation. From March 1995 to September 1996, he was an assistant professor in the Department of Information Systems Engineering of Osaka University, and since October 1996, he has been an associate professor in the same department. His current research interests include mobile computing and augmented reality. He is a member of eight learned societies, including ACM and IEEE.

Shojiro Nishio received his B.E., M.E. and Dr.E. degrees from Kyoto University, Japan, in 1975, 1977 and 1980, respectively. From 1980 to 1988 he was with the Department of Applied Mathematics and Physics of Kyoto University. In October 1988, he joined the faculty of the Department of Information and Computer Sciences, Osaka University, Japan. Since August 1992, he has been a full professor in the Department of Information Systems Engineering of Osaka University. He has been serving as the director of Cybermedia Center of Osaka University since April 2000. His current research interests include database systems, multimedia systems and distributed computing systems. Dr. Nishio has served on the Editorial Board of *IEEE Transactions on Knowledge and Data Engineering*, and is currently involved in the editorial board of *Data and Knowledge Engineering*, *New Generation Computing*, *International Journal of Information Technology*, *Data Mining and Knowledge Discovery*, *The VLDB Journal*, and *ACM Transactions on Internet Technology*. He is a member of eight learned societies, including ACM and IEEE.