On Delay Optimization in Quorum Consensus

Xuemin Lin

School of Computer Science and Engineering,

The University of New South Wales, Sydney, NSW 2052, Australia.

e-mail: lxue@cse.unsw.edu.au.

Abstract

The management of replicated data in distributed database systems is a classic problem with great practical importance. Quorum consensus is one of the popular methods, combining with eager replication, for managing replicated data. In this paper, we will investigate the problems of delay-optimal quorum consensus. Firstly, we will show that the problem of minimizing the total-delay (or mean-delay) restricted to a ring can be solved in a constant time in contrast to the existing approximation results. Secondly, we will show that the problem of minimizing the total-delay (or mean-delay) is NP-hard. Thirdly, we will present an approximate algorithm with an approximate ratio 2; and the approximate algorithm can guarantee the exact solutions for some specific network topology, such as trees and meshes. Finally, we will present an improvement on the existing algorithm to solve the problem of minimizing the maximal-delay; this will reduce the time complexity from $O(n^3 \log n)$ to $O(n^3)$ where n is the number of nodes.

Keywords: Quorum Consensus, Replicated Data Management, and Optimizations.

1 Introduction

The replicated data management in distributed databases is a classic problem with great practical importance. Distributed data warehouses and data marts contain a huge amount of replicated data distributed among a number of sites. Therefore, in recent developments [3, 4, 6, 10] of the area there is always a trade-off among system efficiency, data availability, data freshness, and data consistency.

Two replicated data management methods are available in the literature: *eager* and *lazy*. Eager replication management gives the data *consistency* and the highest data *freshness*. However, it suffers from the system efficiency due to an application of 2-phase commit protocol [19]. On the other hand, lazy replication management provides high system efficiency but does not necessarily provide the data freshness. Moreover, pure lazy replication management does not generally guarantee the data consistency. Recent research results [3] reveal that it is the best if these two methods can be combined.

To achieve a high system efficiency, "quorum consensus" is often adopted in eager replication. In this paper, we will investigate the quorum consensus method. A quorum consensus algorithm is based on the design of a "coterie" (to be formally defined in section 2). In a quorum consensus algorithm, a data processing (read or write) operation can proceed only if permission is granted by a group (*quorum* - an element in the coterie) of the data copies (sites) over the network. Permission from all copies from any one of the quorums in a coterie is sufficient for an operation to proceed, and to ensure the data consistency.

Recent developments in quorum consensus are mainly focused on 1) minimizing the total communication costs for processing a given set of transactions, and 2) minimizing the number of remote sites to be communicated while assembling a quorum. A number of quorum consensus protocols [5, 9, 11, 13, 14, 15, 16, 17, 20, 21] have been developed for these purposes.

Note that in quorum consensus, since messages are sent (possibly by the *multicast* mechanism [18]) to the multiple nodes in a quorum in order to ensure consistency of the operations the delays by passing messages through a long distance communication channel in a wide area network can create a bottleneck [19, 8] in the response time. In this paper we will investigate the problems of minimizing the"average" (or total) delay and minimizing the "maximal" delay.

These two problems were first investigated in [8]. It provides several algorithms for the problem of minimizing the maximal delay with respect to special classes of network topologies, such as rings, trees, and clustered graphs, while the paper [22] provides a $O(n^3 \log n)$ algorithm to solve the problem generally. Both papers also investigated the problem of minimizing the average (or total) delay. The paper [8] provides an approximate algorithm with approximation ratio 1.25 for rings with uniform links and uniform nodes and provides an exact algorithm for trees, while the paper [22] presents a way to reduce the mean delay when minimizing the maximum delay. In [12], a branch-and-bound algorithm is developed to minimize the mean delay. The experiment results demonstrated that the branch-and-bound algorithm is quite effective for moderate-sized networks though it runs in exponential time in the worst case. The complexity of minimizing mean (total) delay in general has been left open.

In this paper, we firstly show that there is a constant time algorithm that gives an exact solution to the problem of minimizing average (total) delay for rings with uniform links and nodes in contrast to the approximation result in [8]. Secondly, we show that the problem of minimizing the average (or total) delays is NP-hard. Thirdly, we provide an approximate algorithm with approximation ratio 2 in general. Note that the average-delay minimization problem, which we study in the paper, is more general than the problem in [8] where each node takes only a unit weight. While the NP-hardness we show in this paper is restricted to the case where each node takes a unit weight, the approximation ratio of our algorithm covers a general case where each node can take an arbitrary weight. Moreover, we can show that our approximate algorithm can guarantee exact solutions for the popular network topologies, such as meshes [19] and trees. These are the principal contributions of the paper. Finally, we show that the complexity of the algorithm in [22], for minimizing the maximum delay, may be reduced to $O(n^3)$ from $O(n^3 \log n)$ where n is the number of nodes.

Note that in [1, 18], the *file* allocation has been studied in an *on-line* environment with the assumption that each read is processed by reading one copy and each write has to be propagated to each copy - read one and write all policy. The duplicated data management, discussed in this paper, assumes that a file allocation is given, and investigates read/write policies instead of the policy of read one and write all. Therefore, the results and techniques in [1, 18] are not applicable.

The rest of the paper is organized as follows. In the second section, we provide the background knowledge, and precisely define the problems. In the third section, we present our results for the problem of minimizing the average (or total) delays. The fourth section presents our result for the problem of minimizing the maximum delays. This is followed by conclusions and remarks.

2 Preliminaries

A network is represented by a weighted graph $G = (V, E)$ where for each edge (physical link) $(u, v) \in E$, $d_{u,v}$ (non-negative) denotes the communication delay to send a unit message along the link from node u to v. A set S of subsets of V is a *coterie* for G if and only if the following conditions hold [9, 20]:

1. *Intersection*: $\forall Q_1, Q_2 \in S$, $Q_1 \cap Q_2 \neq \emptyset$.

2. *Non-redundancy*: $\forall Q_1, Q_2 \in S$, $Q_1 \not\subset Q_2$.

3. *Connectivity*: $\forall Q \in S$, the subgraph G_Q *induced* [7] by Q from G is connected.

Each element Q in a coterie is called a *quorum*. The intersection property guarantees that any pair of quorums in a coterie have at least one common node. The connectivity of the nodes (vertices) in each quorum of a coterie is an important and practical restriction [11, 20]. If a vertex u is connected to vertex v only through z in a network, then we consider $\{u, z, v\}$ as a quorum rather than $\{u, v\}$; this is because a message between u and v must go through z.

Below is the notation to be used in the paper. We use, $V(G)$ to denote the vertex set of G and $l_{u,v}|_G$ to denote the length of the *shortest path* between u and v in G.

Suppose that S is a coterie for a network G; and that a vertex u uses $Q \in S$ as a quorum to execute a data operation issued from u. If the network adopts a multicast mechanism, then the *delay* of sending a message from u to every vertex (site) in Q is the length of the longest shortest path from u to a vertex of Q in G_Q; and it can be defined as: $\text{delay}(u, Q) = \max_{v \in Q}\{l_{u,v}|_G\}$. Note that for notation simplification, in this paper $l_{u,v}|_G$ sometimes is abbreviated to $l_{u,v}$ when no ambiguity is involved.

Given a coterie S, for each node u we define the smallest value of its delay with respect to all quorums as the *delay* of u in S; that is, $\text{delay}(u, S) = \min_{Q \in S}\{\text{delay}(u, Q)\}$. Further, for a coterie S we define the *max-delay* as:

$$\text{max-delay}(S) = \max_{\forall u \in V(G)}\{\text{delay}(u, S)\}. \tag{1}$$

The problem of minimizing the maximum delay (MMD) is to find a coterie for a given network G such that (1) is minimized.

Note that to apply a quorum consensus method, each data object must have a coterie designed. Different data objects may have different coteries designed. The problems discussed in this paper are about an optimal design of coterie for a given data object. Without loss of generality, we may assume a full replication of each data object, that is, a data object is replicated to every node (vertex) over the network. (For a data object partially replicated over the network, the results in this paper are also applicable; and we will discuss this in the last section.)

To define the problem of minimizing the average delay for a given data object, we assume that each vertex u is also associated with a weight w_u (non-negative) to represent the number of the messages issued at this site for reading/writing the data object, which have to be passed to an entire quorum. The average delay of a coterie S is defined as:

$$\text{avg-delay}(S) = \frac{\sum_{u \in V(G)} w_u \text{delay}(u, S)}{\sum_{u \in V(G)} w_u} \tag{2}$$

The problem of minimizing the average delay for a given data object over a network with weights specified on each vertex is to find a coterie such that (2) is minimized. As $\sum_{u \in V(G)} w_u$ is a constant, the problem of minimizing the average delay is equivalent to the problem of minimizing the total delays:

$$\text{tot-delay}(S) = \sum_{u \in V(G)} w_u \text{delay}(u, S) \tag{3}$$

In this paper we will present our results for the problem of minimizing the total delays (MTD).

In a coterie S, it is clear that for each vertex u there is a quorum $Q_u \in S$ such that $\text{delay}(u, Q_u) = \text{delay}(u, S)$, and $\{Q_u : u \in V\}$ also forms a coterie. This implies that both MMD and MTD problems can be simplified as:

Minimizing Max-Delays (MMD)
INSTANCE: A weighted graph $G = (V, E)$, and $\forall (u, v) \in E$ $d_{u,v}$ is an integer.
QUESTION: Find a set $\Pi = \{G_u : u \in V\}$ of connected subgraphs of G such that

- $S = \{V(G_u) : G_u \in \Pi\}$ is a coterie of G;
- $\forall u \in V$, $delay(u, V(G_u)) = delay(u, S)$;
- $\max_{u \in V}\{\text{delay}(u, V(G_u))\}$ is minimized among all such possible sets Π of connected subgraphs.

Note that in such a set Π as specified above, G_u is not necessarily different from G_v when $u \neq v$. For instance, in Figure 1 the 4 subgraphs (depicted by polylines), right-upper corner, right-bottom corner, left-upper corner, and left-bottom corner form a coterie for the graph in the figure. In the example, if each edge takes the weight 1, each circle node uses the right-upper corner subgraph as a quorum, each hexagon node uses the right-bottom corner subgraph as a quorum, each square node uses the left-upper corner subgraph as a quorum, and each triangle node uses the left-bottom corner subgraph as a quorum, then the coterie is an optimal solution for MMD since the *graph diameter* is 6 in this example.

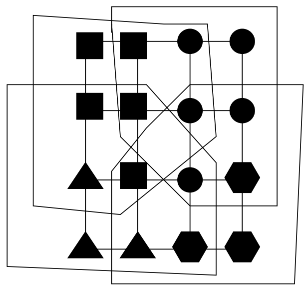


Figure 1: Max-delays

Minimizing Total-Delays (MTD)
INSTANCE: A weighted graph $G = (V, E)$, $\forall (u, v) \in E$ $d_{u,v}$ is an integer, and $\forall u \in V$ w_u is an integer.
QUESTION: Find a set $\Pi = \{G_u : u \in V\}$ of connected subgraphs such that

- $S = \{V(G_u) : G_u \in \Pi\}$ is a coterie of G;
- $\forall u \in V$, $delay(u, V(G_u)) = delay(u, S)$;
- tot-delay(S) is minimized among all such possible sets Π of connected subgraphs.

Note that the problems of MMD and MTD were studied in [8]. However, our definition of MTD is more general than that in [8] which studied only MTD restricted to the case where each vertex takes a unit weight.

In the next two sections, we will present our results for MTD and MMD problems.

3 Minimizing Total-Delays

The section is organized as follows. It starts with our results for rings with uniform links and nodes. Then we will investigate the complexity of the problem. Finally we will present an approximate algorithm to approach the problem. It can be shown that the proposed algorithm has approximation ratio 2; and the bound is quite tight. Moreover, we can show that the algorithm can guarantee the exact solutions for quite a large class of graphs, such as trees and meshes.

3.1 Rings

In this subsection, we investigated MTD restricted to a *uniform ring* G where the network G is a ring, all vertices of G are equally weighted, and all links of G are equally weighted. Note that in [8], MTD restricted to uniform rings was approached by an approximation algorithm with the approximate ratio 1.25. Below, we present our exact algorithm for this case. The algorithm runs in a constant time.

Without loss of generality, we assume that the vertex weight and the link weight are all normalized to 1. The theorem below is the key to the correctness of the algorithm.

Theorem 1 *Suppose that G is a uniform ring with n nodes. Then, the minimum total delay T_{OPT} has lower bounds below with respect to the 3 different cases, respectively.*

Case 1: $T_{OPT} \geq \frac{n^2}{4}$ if n is even.

case 2: $T_{OPT} \geq \lambda(4\lambda + 1)$ if n is odd and $n = 4\lambda + 1$ for an integer λ.

case 3: $T_{OPT} \geq (2\lambda + 1)^2 + (\lambda + 1)$ if n is odd and $n = 4\lambda + 3$ for an integer λ.

Proof: Suppose that the vertex set of G is V and $S = \{Q_v : v \in V\}$ is a coterie that leads to an optimal solution of MTD, where for each $v \in V$, $delay(v, Q_v) = delay(v, S)$. Suppose the vertices in G are sub-indexed by $\{1, 2, ..., n\}$ counter-clock wise according to adjacency; that is, for $1 \leq i \leq n-1$, v_i and v_{i+1} are adjacent, and v_n and v_1 are also adjacent. Below, we prove the theorem with respect to each case.

Case 1: Clearly the n nodes can be paired in to $n/2$ pairs $\{(v_i, v_{\frac{n}{2}+i}) : 1 \leq i \leq \frac{n}{2}\}$ and the shortest distance between the two nodes in each pair is $n/2$. For every such pair v_i and $v_{\frac{n}{2}+i}$ of vertices, $delay(v_i, S) + delay(v_{\frac{n}{2}+i}, S) \geq \frac{n}{2}$ because Q_{v_i} must intersect $Q_{v_{\frac{n}{2}+i}}$. Consequently, $T_{OPT} \geq \frac{n^2}{4}$.

Case 2: We first prove that there must be at least one vertex u such that $delay(u, S) \geq \lambda$. Suppose that for all vertices $v \in V$, $delay(v, S) < \lambda$. Then, Q_{v_n} cannot intersect $Q_{v_{2\lambda+1}}$ since the shortest distance between v_n and $v_{2\lambda+1}$ in G is 2λ; this is contradictory.

Without loss of generality, we assume that $delay(v_n, S) \geq \lambda$. The remaining 4λ $(= n-1)$ vertices are paired into 2λ pairs $\{(v_i, v_{2\lambda+i}) : 1 \leq i \leq 2\lambda\}$ and the shortest distance between each pair v_i and $v_{2\lambda+i}$ (for $1 \leq i \leq 2\lambda$) of vertices is 2λ. Therefore, for $1 \leq i \leq 2\lambda$, $delay(v_i, S) + delay(v_{2\lambda+i}, S) \geq 2\lambda$. Consequently, $T_{OPT} \geq 2\lambda \times 2\lambda + \lambda$.

Case 3: We first prove that there must be one vertex u such that $delay(u, S) \geq \lambda + 1$. Suppose that for all vertices $v \in V$, $delay(v, S) < \lambda+1$. Then, Q_{v_n} cannot intersect $Q_{v_{2\lambda+2}}$ since the shortest distance between v_n and $v_{2\lambda+2}$ in G is $2\lambda + 1$; this is contradictory.

Without loss of generality, we assume that $delay(v_n, S) = \lambda$. The remaining $4\lambda + 2$ $(= n-1)$ vertices are paired into $2\lambda + 1$ pairs $\{(v_i, v_{2\lambda+1+i}) : 1 \leq i \leq 2\lambda + 1\}$ and the shortest distance between each pair v_i and $v_{2\lambda+1+i}$ (for $1 \leq i \leq 2\lambda+1$) of vertices is $2\lambda+1$. Therefore, for $1 \leq i \leq 2\lambda + 1$, $delay(v_i, S) + delay(v_{2\lambda+1+i}, S) \geq 2\lambda + 1$. Consequently, $T_{OPT} \geq (2\lambda + 1) \times (2\lambda + 1) + (\lambda + 1)$.

□

For an integer n, n can fall only in one of the 3 cases as described in Theorem 1. Our algorithm OPT-RING, presented below, for solving MTD restricted to uniform rings is developed respect to these 3 different cases of n, where n is the number of vertices in a uniform ring G.

Algorithm OPT-RING

Case 1: n is even. Use a coterie with only one-node; that is, randomly choose a vertex u from G, and then for each vertex in G, use u as its quorum.

Case 2: n is odd and $n = 4\lambda + 1$ for an integer λ. In this case, for each vertex u, assign Q_u as the set of the vertices each of which is at most λ away from u.

Case 3: n is odd and $n = 4\lambda + 3$ for an integer λ. We design a coterie as follows.

- For $1 \leq i \leq 2\lambda + 1$, Q_{v_i} consists of the vertices which are within λ away from v_i.
- For $2\lambda + 3 \leq i \leq 4\lambda + 2$, Q_{v_i} consists of the vertices which are within $\lambda + 1$ away from v_i.
- For vertices $v_{2\lambda+2}$ and $v_{4\lambda+3}$, $Q_{v_{2\lambda+2}} = Q_{v_{2\lambda+1}}$ and $Q_{v_{4\lambda+3}} = Q_{v_1}$ respectively. $\square$

It can be immediately verify that the algorithm OPT-RING gives a set of Q_vs which can form a coterie of G; that is, the set satisfies the 3 properties, Intersection, Non-redundancy, Connectivity, as specified in the last section. Further, it can also be verified that the total delay of such a coterie produced by the algorithm is the corresponding lower bound in Theorem 1 with respect to each case. Therefore, the algorithm OPT-RING provides an optimal solution for MTD restricted to uniform rings. Clearly, the algorithm OPT-RING runs in a constant time if n is known.

3.2 Complexity of Minimizing Total-Delays

In this subsection, we prove the NP-hardness of MTD problem; this is done by proving the NP-completeness of the corresponding decision problem.

MTD decision (MTDD) problem

INSTANCE: A weighted graph $G = (V, E)$, $\forall (u, v) \in E$ $d_{u,v}$ is an integer, $\forall u \in V$ w_u is an integer, and an integer N.

QUESTION: Is there a set $\Pi = \{G_u : u \in V\}$ of connected subgraphs such that

- $S = \{V(G_u) : G_u \in \Pi\}$ is a coterie of G;
- $\forall u \in V$, $delay(u, V(G_u)) = delay(u, S)$;
- tot-delay$(S) \leq N$

Theorem 2 *MTDD is NP-complete.*

To prove Theorem 2, below we transform the vertex cover problem to a special case of MTDD.

Vertex Cover

INSTANCE: A graph $G = (V, E)$, positive integer $K \leq |V|$.
QUESTION: Is there a vertex cover of size K or less for G?

For each instance $I_{VC} = \{G, K\}$ in the vertex cover, we construct the corresponding instance I_{VC}^{MTDD} of MTDD, say $I_{VC}^{MTDD} = \{G', N'\}$, as follows.

- $V(G) \subset V(G')$, and the vertices in $V(G') - V(G)$ are added along with the new edges created below.
- In G', replace each edge (u, v) in G by 4 edges (u, x), (x, v), (u, y), and (y, v), together with two new vertices x and y, such that $d_{u,x} = 1.5$, $d_{x,v} = 1$, $d_{u,y} = 1$, and $d_{y,v} = 1.5$.
- For each pair $\{u, v\}$ of vertices in G such that $(u, v) \notin E(G)$, add two edges (u, x) and (x, v) to G', together with one new vertex x, such that $d_{u,x} = 1$ and $d_{x,v} = 1$.
- For each pair $\{u, v\}$ of vertices in $V(G') - V(G)$, G' has one edge (u, v) connecting u and v, and $d_{u,v} = 1$. Note that the induced graph on $V(G') - V(G)$ (with respect to G') forms a clique.
- In G', for each vertex $u \in V(G)$ ($V(G) \subset V(G')$) let $w_u = 2n^4$ where n is the number of vertices in G; and for each vertex $u \in V(G') - V(G)$ let $w_u = 1$.

- Let $N' = 2n^5 + Kn^4 + n^2$.

An example of such a transformation is depicted in Figure 2, where square vertices represent the newly added vertices, dotted edges carry the weight 1, and square vertices are connected to form a complete subgraph in which each edge carry the weight 1.

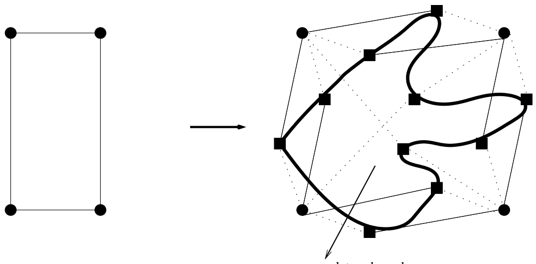


Figure 2: a transformation from vertex cover to MTDD

Clearly, the transformation is polynomial with respect to n. Now we prove that for each instance I_{VC} in the vertex cover problem, it has a solution if and only if the corresponding instance I_{VC}^{MTDD} has a solution. First we prove "only if" part.

Lemma 1 *If I_{VC} has a vertex cover of size K or less for G, then I_{VC}^{MTDD} also has a "yes" answer for MTDD.*

Proof: Suppose that I_{VC} has a cover C for G such that $|C| \leq K$. Then in G',

- $\forall v \in C$, construct a subgraph G_v such that it contains all vertices the shortest distance of which to v is not greater than 1.5;
- $\forall v \in V(G') - C$, construct a subgraph G_v such that it contains all vertices whose shortest distance to v is 1.

Note that $V(G') - C = (V(G') - V(G)) \cup (V(G) - C)$, $|V(G)| = n$, $|C| \leq K$, and $|V(G') - V(G)| \leq n(n-1)$. It can be immediately verified that:

$$\begin{aligned} & \sum_{v \in V(G')} w_v \times delay(v, V(G_v)) \\ \leq\ & 2n^4 \times (n - |C|) + 1.5 \times |C| \times 2n^4 + |V(G') - V(G)| \\ \leq\ & 2n^5 + Kn^4 + n^2. \end{aligned}$$

Moreover, it can be immediately shown that each pair of G_vs have at least one common vertex in $G(V') - G(V)$. Therefore, it implies that $\{V(G_v) : v \in V(G')\}$ form a coterie. $\square$

Next, we prove the "if" part. We proceed it by the two lemmas below.

Lemma 2 *Suppose that $I_{VC}^{MTDD} = (G', N')$ has a "yes" answer to MTDD and $n \geq 5$. Then in any set $\Pi = \{G_v : v \in V(G')\}$ of connected subgraphs that gives the "yes" answer to MTDD, for each vertex $u \in V(G)$ ($V(G) \subset V(G')$) $delay(u, V(G_u)) \neq 0$. Note G is the graph in the corresponding instance I_{VC} of the vertex cover problem.*

Proof: Suppose that there is a G_u in Π such that $delay(u, V(G_u)) = 0$ and $u \in V(G)$. Then to meet the restriction that G_u must intersect with another G_v ($\forall v \in V(G) - \{u\}$), $delay(v, G_v) \geq 2$ ($\forall v \in V(G) - \{u\}$) according to the structure of G'. Thus, when $n \geq 5$,

$$\sum_{v \in V(G')} w_v \times delay(v, V(G_v)) \geq 2 \times 2n^4(n-1) > N'.$$

Contradiction. $\square$

Lemma 3 *Suppose that $I_{VC}^{MTDD} = (G', N')$ has a "yes" answer to MTDD and $n \geq 5$. Then the corresponding instance I_{VC} also has a "yes" answer to the vertex cover problem.*

Proof: Suppose that $\Pi = \{G_v : v \in V(G')\}$ is a set of connected subgraphs of G' that leads to a "yes" answer to MTDD with respect to I_{VC}^{MTDD}.

According to Lemma 2, for each vertex $v \in V(G)$ (where G is the graph in the corresponding I_{VC}) $delay(v, V(G_v)) \neq 0$, that is, $delay(v, V(G_v)) \geq 1$. Let:

$$C = \{u : u \in V(G) \;\&\; delay(u, V(G_u)) > 1\}.$$

According to the structure of G', it is clear that for each vertex u in C, $delay(u, V(G_u)) \geq 1.5$. Now we prove the following two properties for C by contradictory method, p1: $|C| \leq K$.

p2: C is a vertex cover of G.

Suppose that $|C| \geq K + 1$. Then,

$$\begin{aligned}
& \sum_{v \in V(G')} w_v \times delay(v, V(G_v)) \\
\geq\ & (n - |C|) \times 2n^4 + 1.5|C| \times 2n^4 \\
\geq\ & 2n^5 + (K+1)n^4 \\
>\ & N'.
\end{aligned}$$

It is contradictory; and thus $|C| \leq K$.

Suppose that C is not a vertex cover of G; this means there is an edge $(u_1, u_2) \in E(G)$ such that $u_1 \notin C$ and $u_2 \notin C$. Consequently, in Π $delay(u_1, V(G_{u_1})) = 1$ and $delay(u_2, V(G_{u_2})) = 1$. According to the structure of G', this implies G_{u_1} does not intersect with G_{u_2}. It is contradictory again. Thus, p2 also holds.

Therefore, C leads to a "yes" answer to the corresponding I_{VC} of the vertex cover problem. □

It can be immediately verified that MTDD is in NP. From Lemma 1 and Lemma 3, the Theorem 2 immediately follows. Further, we can have a stronger version of Theorem 2.

Theorem 3 *MTDD is NP-complete even when restricted to the case when each vertex has a unit weight.*

Proof: For notation simplification, we denote the problem of MTDD, restricted to the case when each vertex has weight 1, by MTDD1. To prove the theorem, we also transfer the vertex cover problem to a special case of MTDD1. For each instance $I_{VC} = (G, K)$, the transformation of I_{VC} to an instance $I_{VC}^{MTDD1} = (G'', N'')$ of MTDD1 may be viewed as a modification of $I_{VC}^{MTDD} = (G', N')$ (constructed as above from I_{VC}):

- *Modification of G' to produce G'':* For each vertex u in G' with weight $2n^4$, add $2n^4 - 1$ new vertices which are connected to u as a star where those new added edges have weight $\frac{1}{n^5}$, while others in G' remain unchanged. Figure 3(b) illustrates such a modification from the graph as depicted in Figure 3(a) regarding the example shown in Figure 2. Note that in G'', each vertex takes weight 1.

- *N'':* $N'' = 2n^5 + Kn^4 + n^2 + 2$ may be also viewed as a modification from $N' = 2n^5 + Kn^4 + n^2$.

Similar arguments to those in Lemmas 1 - 3 can lead to a proof of this theorem. □

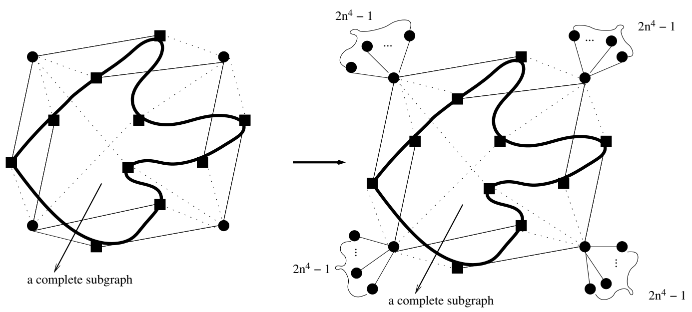


Figure 3: a modified transformation ($n = 4$)

3.3 An Approximate Algorithm

In this subsection, we present a heuristic for MTD. We will show that for a class of graphs, the heuristic can deliver the optimal solution; this class includes many popular networks [19], such as meshes and trees. Further, we will provide a tight worst case performance ratio for the heuristic.

The algorithm consists of three steps as follows. Suppose that a weighted network G is given.

Algorithm MTD

Step 1 Compute all pairs of shortest paths $\{L_{u,v} : \forall u, v \in V(G)\}$.

Step 2 Choose a vertex u^* such that $\sum_{v \in G} w_v \times l_{u^*,v}$ is minimized. (Recall that $l_{u^*,v}$ denotes the length of the shortest path $L_{u^*,v}$).

Step 3 $\forall v \in V(G)$, let $G_v = \{u^*\}$. $\square$

Note that the algorithm MTD always gives a coterie consisting of only one vertex $\{u^*\}$. The algorithm can be easily implemented in time $O(n^3)$. This is because that the step 1 can be implemented in $O(n^3)$ time by the algorithm Floyd-Warshall [7], while step 2 and step 3 can be trivially implemented in $O(n^2)$ time.

Next we show the approximation behaviour of the algorithm.

Theorem 4 *The approximation ratio of the algorithm MTD is* 2*.*

Proof: Suppose that $G = (V, E)$ is the weighted network and $S = \{Q_v : v \in V\}$ is a coterie that leads to an optimal solution for MTD. Further, suppose that for each $v \in V$, $delay(v, Q_v) = delay(v, S)$, and l_v denotes $delay(v, Q_v)$. The minimal total delay is

$$T_{OPT} = \sum_{v \in V(G)} w_v l_v. \tag{4}$$

Assume that u^* is the vertex chosen in the algorithm MTD as the coterie. Clearly, the total delay produced by the algorithm MTD is:

$$T_{MTD} = \sum_{u \in V(G)} w_u l_{u,u^*}. \tag{5}$$

Suppose that u_0 is the vertex such that l_{u_0} has the minimum value among all l_v. For each vertex $z \in V(G)$ other than u_0, Q_z must intersect with Q_{u_0}. Consequently,

$$l_{z,u_0} \le l_z + l_{u_0} \le 2l_z$$

Hence,

$$\sum_{z \ne u_0} w_z l_{z,u_0} \le \sum_{z \ne u_0} 2 w_z l_z \le 2T_{OPT}$$

Note that $T_{MTD} \leq \sum_{z \neq u_0} w_z l_{z,u_0}$. The theorem immediately follows. $\square$

We can show that the ratio, obtained by our algorithm, is fairly tight by the following example. As depicted in Figure 4, a class of graphs $G = (V, E)$ have the following structure:

- $V = V_1 \cup V_2$ where V_1 has n vertices (circle vertices in V_1) and V_2 (square vertices in V_2) has $\frac{n(n-1)}{2}$ vertices;
- each pair of vertices in V_1 correspond to a vertex in V_2, and are connected by two edges via the vertex in V_2;
- the induced graph on V_2 is complete;
- each edge has a unit weight;
- the weight of a vertex in V_1 is $N = n^k$ (k is sufficiently large) and the weight of a vertex in V_2 is 1.

It can be immediately shown that the Algorithm MTD will randomly choose a square vertex as the coterie; and it produces the total delay is $T_{MTD} = 2(n-1)N + \frac{n(n-1)}{2} - 1$. On the other hand, for this class of graphs the minimal total delay will be the coterie where the delay of a quorum for each vertex is 1; and thus $T_{OPT} = Nn + \frac{n(n-1)}{2}$. Therefore,

$$\lim_{n \to \infty} \frac{T_{MTD}}{T_{OPT}} = 2 \tag{6}$$

This means that the approximation bound in Theorem 4 is asymptotically tight. Figure 4 illustrates such a graph when $n = 4$. Moreover, the asymptotically lower bound illustrated in (6) also holds even restricted to the case when each vertex has a unit weight; this can be shown by adding N vertices "star-like" attached to each vertex with weight N in the way as depicted in Figure 3(b) and the new edges all take a very small weight ϵ. Furthermore, we can obtain a tighter expression of the approximation ratio restricted to the case when all vertices have a unit weight.

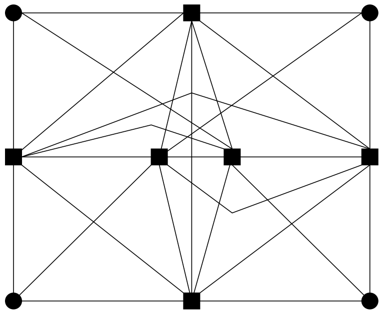


Figure 4: to shown the tightness of the ratio

Theorem 5 *Algorithm MTD has approximation ratio* $2 - \frac{3}{n}$ *if each vertex is equally weighted and* $n \geq 3$*. Here,* n *is the number of vertices in the network.*

Proof: Suppose that $G = (V, E)$ is the weighted network and $S = \{Q_v : v \in V\}$ is a coterie that leads to an optimal solution for MTD. Further, suppose that for each $v \in V$, $delay(v, Q_v) = delay(v, S)$, l_v denotes $delay(v, Q_v)$, and the delay l_0 of the quorum of vertex u_0 is minimal. There are two cases:

case 1: In the optimal coterie S, the quorum of u_0 intersects all the other quorums at a common vertex v_0.

case 2: In the optimal coterie, the quorum of u_0 does not have a common intersection vertex with other quorums; that is, it intersects other quorums at several (at least 2) vertices, respectively.

For case 1, the algorithm MTD should be able to find the optimal coterie; for instance, choosing v_0 as the coterie in Algorithm MTD will lead to an optimal coterie. The theorem holds for this case.

In case 2, assume that the quorum of u_0 intersects, respectively, other quorums at k ($k \geq 2$) vertices v_i for $1 \leq i \leq k$. Note that there are two sub-cases in case 2:

case 2a: u_0 is in $V' = \{v_i : 1 \leq i \leq k\}$.

case 2b: u_0 is not in $V' = \{v_i : 1 \leq i \leq k\}$.

In case 2a, without loss of generality we assume that $v_k = u_0$. Clearly, $l_{v_i,u_0} \leq l_0$ for $1 \leq i \leq k-1$ and

$$l_{z,u_0} \leq l_z + l_0 \ \text{ for } z \notin V'. \tag{7}$$

With respect to the optimal coterie, let $\delta_z = l_z - l_0$ for each vertex z. Clearly, $\delta_z \geq 0$. From (7), it follows that

$$\begin{aligned}
\sum_{z \neq u_0} l_{z,u_0} &\leq \sum_{z \notin V'} (2l_z - \delta_z) + (k-1)l_0 \\
&\leq 2\sum_{z \in V}(l_z) - \sum_{z \notin V'}(\delta_z) - 2\sum_{j=1}^{k-1} \delta_{v_j} - (k+1)l_0 \\
&\leq 2\sum_{z \in V} l_z - ((k+1)l_0 + \sum_{z \in V} \delta_z) \\
&\leq (2 - \frac{k+1}{n}) \sum_{z \in V} l_z
\end{aligned}$$

Let T_{MTD} denote the total delay produced by the algorithm MTD.

$$T_{MTD} \leq \sum_{z \neq u_0} l_{z,u_0} \leq (2 - \frac{k+1}{n}) \sum_{z \in V} l_z.$$

As $k \geq 2$, the theorem also holds for case 2a.

In case 2b, by similar arguments as those in case 2a we can show that

$$T_{MTD} \leq (2 - \frac{k+2}{n}) \sum_{z \in V} l_z.$$

Therefore the theorem holds for case 2b. □

Next we show that the algorithm MTD actually can produce optimal solutions for quite a large class of graphs, including meshes and trees.

Theorem 6 *Suppose that G is a weighted graph with n vertices, where*

1. *Each vertex in G is equally weighted.*

2. *The vertex set V can be divided into m ($n/2$ or $(n-1)/2$) disjoint pairs of vertices:*

   - *if $n = |V|$ is even, $V = \{u_i, v_i : 1 \leq i \leq \frac{n}{2}\}$, and*
   - *if $n = |V|$ is odd, $V = \{u_i, v_i : 1 \leq i \leq \frac{n-1}{2}\} \cup \{z_0\}$,*

such that a shortest path between each pair of v_i and u_i shares the common vertex; and the common vertex is z_0 if n is odd. Then the algorithm MTD will produce the optimal solution for G.

Proof: Let m denote $\frac{n}{2}$ when n is even, and denote $\frac{n-1}{2}$ when n is odd. Let l_i denote the length of the shortest path between v_i and u_i for $1 \leq i \leq m$.

Suppose that $S = \{Q_v : v \in V\}$ is a coterie that leads to the optimal solution for MTD, where for each $v \in V$, $delay(v, Q_v) = delay(v, S)$. We have that for $1 \leq i \leq m$ $delay(u_i, Q_{u_i}) + delay(v_i, Q_{v_i}) \geq l_i$; and thus

$$T_{OPT} \geq \sum_{i=1}^{m} l_i \tag{8}$$

On the other hand, if we choose this common vertex v (it is z if n is odd) of G as a coteries, the total delay is $\sum_{i=1}^{m}(l_{v_i,v} + l_{u_i,v})$. Therefore,

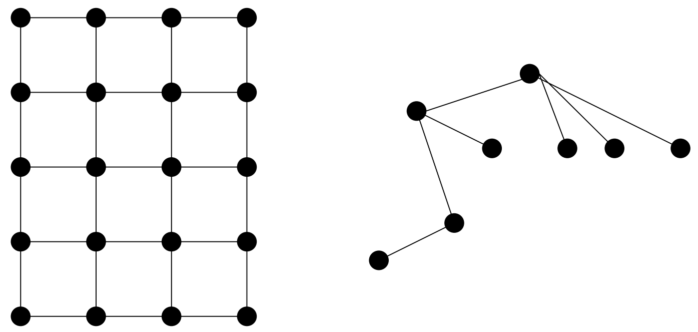
$$T_{MTD} \leq \sum_{i=1}^{m}(l_{v_i,v} + l_{u_i,v}) \tag{9}$$

Note that $l_{v_i,v} + l_{u_i,v} = l_i$. This implies that

$$T_{MTD} \leq \sum_{i=1}^{m} l_i. \tag{10}$$

From (8) and (10), it follows that $T_{MTD} \leq T_{OPT}$; and thus $T_{MTD} = T_{OPT}$. □

Theorem 6 implies that for those popular network topologies [19], such as *meshes* (Figure 5(a)) and *trees* (Figure 5(b)), the algorithm MTD can produce the optimal solution



(a) (b)

Figure 5: meshes and trees

for MTD when each vertex takes the same weight and each edge also takes the same weight. Note that this algorithm restricted to trees is similar to the algorithm in [8] for trees, and MTD does not provide optimal solution to MTD for uniform rings when the number of vertices is odd.

4 Minimizing Maximal-Delays

In this section, we investigate the problem of minimizing maximum-delays. Particularly, we present an efficient algorithm based on a mixture of dynamic programming and greedy paradigms. The algorithm uses the same optimization strategy as that in [22]; however our execution runs in $O(n^3)$ in contrast to $O(n^3 \log n)$ in [22]. It conceptually consists of three steps as follows. Suppose that $G = (V, E)$ is the network.

Algorithm MMD

Step 1 In G, compute all pair shortest paths $\{L_{u,v} : u, v \in V\}$ and their lengths $\{l_{u,v} : u, v \in V\}$.

Step 2 Sort $\{l_{u,v} : u, v \in V\}$ in an increasing order and store them in D.

Step 3 Scan D according to the increasing ordering of $l_{u,v}$, iteratively construct subgraphs by an expansion till each pair of subgraphs have an intersection. Then check non-redundancy and remove all redundant quorums if exist. □

As mentioned in last section, applying the algorithm Floyd-Warshall in [7], the step 1 can be implemented in $O(n^3)$ time where n is the number of vertices. Clearly, Step 2 can be implemented in $O(n^2 \log n)$ time as there are n^2 pairs of vertices. In fact, step 3 can also be implemented in $O(n^3)$ time; and it is described below.

In our implementation of Step 3, for each vertex v_k in G we use a linear data structure (for instance, a linked list) A_k to store the vertices w such that $l_{v_k,w}$ is less than the current element $l_{u,v}$ of D. Clearly, each induced [7] subgraph G_k by the vertices in A_k is connected, and $delay(v_k, A_k) \leq l_{u,v}$.

Moreover, suppose that A_k is constructed up to $l_{u,v}$ in D. Then A_k also gives us the information that for any pair $\{v_i, v_j\}$ of vertices and any two quorums Q_i and Q_j such that $delay(v_i, Q_i) \leq d_{u,v}$ and $delay(v_j, Q_j) \leq l_{u,v}$, Q_i intersects with Q_j at v_k if and only if v_i and v_j are both in A_k. Further, for each pair $\{v_i, v_j\}$ we use $B_{i,j}$ $(i < j)$ to indicate whether or not the current A_i and A_j already intersect. As mentioned earlier, Step 3 consists of two sub-steps:

Step 3a: construct a vertex set A_i for each vertex v_i, and then

Step 3b: check non-redundancy: if $A_i \subset A_j$ then replace A_j by A_i.

Below is the detailed implementation of Step 3a.

Step 3a

Set each A_k $(1 \leq k \leq n)$ to empty; and set each $B_{i,j}$ $(1 \leq i < j \leq n)$ to zero; $m \leftarrow 0$;
for each l_{v_i,v_j} **do**: (Scan D according to the increasing ordering of its elements)
  { $A_i \leftarrow A_i \cup \{v_j\}$; $A_j \leftarrow A_j \cup \{v_i\}$;
    for each vertex v_x in $A_i - \{v_j\}$ **do**
    if $B_{\min\{x,j\},\max\{x,j\}} = 0$ **then**
      { $B_{\min\{x,j\},\max\{x,j\}} \leftarrow 1$; $m \leftarrow m + 1$;
       if $m = \frac{n(n-1)}{2}$ **then** terminate the algorithm; }

for each vertex v_x in $A_j - \{v_i\}$ **do**

if $B_{\min\{x,i\},\max\{x,i\}} = 0$ **then**

{ $B_{\min\{x,i\},\max\{x,i\}} \leftarrow 1$; $m \leftarrow m + 1$;

if $m = \frac{n(n-1)}{2}$ **then** terminate the algorithm; } $\square$

It can be immediately shown that the algorithm runs in time $O(n^3)$. To implement Step 3b efficiently, we first sort all A_i in an increasing order according to the cardinalities of each A_i. Then we test the set inclusion property for A_i and A_j according to the ordering where $|A_i| < |A_j|$. In fact, the test of the set inclusion property for a pair A_i and A_j can be done by $O(\min\{|A_i|, |A_j|\})$ if all vertices A_i and A_j are stored in a global ordering applied to each A_k for $1 \leq k \leq n$. Therefore, Step 3b can also be implemented in $O(n^3)$. These imply that the algorithm MMD can be implemented in $O(n^3)$ time.

We can also immediately show that the algorithm MMD guarantees providing an exact solution for MMD problem; this is because we use an iteration procedure and the algorithm immediately terminates once each pair of quorums have an intersection.

5 Conclusion and Remarks

In this paper, we investigated the complexity issues of minimizing the max-delay (MMD) and minimizing the total-delay (MTD) for designing coteries (quorum consensus methods). Firstly, we showed that MTD can be solved in a a constant time for uniform rings in contrast to the approximate results in [8]. Then, we showed that MTD is NP-hard even when restricted to the case where each vertex takes a unit weight. We also proposed a cubic time approximate algorithm for solving MTD. The approximate algorithm has an approximation ratio 2 in general, and has an approximation ratio $2 - \frac{3}{n}$ for the case when each vertex takes a unit weight. We showed that our approximation ratios are asymptotically tight. Moreover, we proved that the approximate algorithm can guarantee exact solutions for a quite large class of graphs such as "uniform" trees and uniform meshes. Finally, in the paper we reduced the time complexity of the algorithm in [22], for solving

MMD, to $O(n^3)$ from $O(n^3 \log n)$ where n is the number of vertices.

Note that for presentation simplicity, the results presented in the paper follow the assumption, made in [8], that each data object is fully replicated over the network. In case if a data object is partially replicated in the network, the design of coteries must be restricted to the vertices that hold the replicas. In this case, our results still hold if we add this restriction to our algorithms when generate quorums. However, we have to drop the connectivity property from a coterie; this is because that those sites holding replicas may not necessarily be adjacent.

As a possible future study, we are interested in MTD for non-uniform rings and meshes, that is, links and vertices do not necessarily take unit weight. Further, we are interested in MTD for design of "wr"-coteries [8, 11, 20].

Acknowledgement

Part of the work was done when the author took study leave in SEEM Department at the Chinese University of Hong Kong (CUHK). The author would also like to thank Prof. S. Zhang from CUHK for useful discussions.

References

[1] B. Awerbuch, Y. Bartal, and A. Fiat, Distributed paging for general networks, *Journal of Algorithms*, 28:67-104, 1998.

[2] P. Bernstein, V. Hadzilocs and N. Goodman, *Concurrency Control and Recovery in Database Systems*, Addison-Wesley, Reading, Mass., 1987.

[3] Y. Breitbart, R. Komondoor, R. Rastogi, and S. Seshadri, Update Propagation Protocols for Replicated Databases, *Proceedings of the 1999 ACM SIGMOD*, 97 - 108, 1999.

[4] Y. Breitbart and H. F. Korth, Replication and Consistency: Being lazy helps sometimes, *Proceedings of the ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, Tucson, Arizona, 1997.

[5] H. Chen and J. Tang, An Efficient Method for Mutual Exclusion in Truly Distributed Systems, *Int. Conf. on Distributed Computing Systems*, 97-104, 1994.

[6] P. Chundi, D.J. Rosenkratz, and S.S. Ravi, Deferred Updates and Data Placement in Distributed Databases, *Proceedings of the Twelveth International Conference on Data Engineering*, New Orleans, Louisiana, 1996.

[7] T. H. Cormen, C. E. Leiserson, and R. L. Rivest, *Introduction to Algorithms*, MIT press, 1990.

[8] A. W. Fu, Delay-Optimal Quorum Consensus for Distributed Systems, *IEEE Transactions on Parallel and Distributed Systems*, 8(1), 59-69, 1997.

[9] H. Garcia-Molina and D. Barbara, How to Assign Votes in a Distributed Systems, *J. ACM*, 32(4), 841-860, 1985.

[10] J. Gray, P. Helland, P. O'Neil, and Dennis Shasha, The Dangers of Replication and a Solution, *Proceedings of the ACM SIGMOD Conference*, Montreal, Quebec, Canada, 1996.

[11] T. Ibaraki, H. Nagamochi, and T. Kameda, Optimal Coteries for Rings and Related Networks, *Distributed Computing*, 8, 191-201, 1995.

[12] N. Kobayashi, T. Tsuchiya, and T. Kikuno, Minimizing the Mean Delay of Quorum-based Mutual Exclusion Schemes, *Journal of Systems and Software*, 58(1), 2001.

[13] A. Kumar, Hierarchical Quorum Consensus: A New Algorithm for Managing Replicated Data, *IEEE Transactions on Computers*, 40(9), 996-1004, 1991.

[14] A. Kumar and A. Segev, Cost and Availability Tradeoffs in Replicated Data Concurrency Control, *ACM Transactions on Database Systems*, 18(1), 102-131, 1993.

[15] X. Lin, A Fully Distributed Quorum Consensus Method with High Fault-Tolerance and Low Communication Overhead, *Theoretical Computer Science*, 185(2), 259-275, 1997.

[16] X. Lin and M. E. Orlowska, An Efficient Optimal Algorithm for Minimizing the Overall Communication Cost in Replicated Data Management, *J. Parallel and Distributed Comput.* 35, 97-103, 1996.

[17] M. Maekawa, A $\sqrt{N}$ Algorithm for Mutual Exclusion in Decentralized Systems, *ACM Transactions on Computer Systems*, 3(2), 145-159, 1985.

[18] B. Maggs, F. Mayer auf der Heide, B. Vocking, and M. Westermann, Exploiting Locality for Data Management in Systems of Limited Bandwidth, *Proc. of the 38th IEEE Symp. on Foundations of Computer Science (FOCS)*, 284-293, 1997.

[19] M. T. Ozsu and P. Valduriez, *Principles of Distributed Database Systems*, Prentice Hall, 1991.

[20] C. H. Papadimitriou and M. Sideri, Optimal Coteries, *Proceedings of ACM 10th Symposium of Distributed Computing*, 75-80, 1991.

[21] D. Saha, S. Rangarajan, and S. K. Tripathi, An Analysis of the Average Message Overhead in Replica Control Protocols, *IEEE Transactions on Parallel and Distributed Systems*, 7(10), 1026 - 1034, 1995.

[22] T. Tsuchiya, M. Yamaguchi, and T. Kikuno, Minimizing the Maximum Delay for Reaching Consensus in Quorum-Based Mutual Exclusion Schemes, *IEEE Transactions on Parallel and Distributed Systems*, 10(4), 1999.