

応答時間を考慮した Quorum プロトコルを利用するストレージの配置最適化

石川 宗寿[†] 杉木 章義[‡] 品川 高廣[†] 加藤和彦^{†‡}

[†] 筑波大学大学院 システム情報工学研究科

[‡] 科学技術振興機構 CREST

要 旨

データセンターの多拠点化やエッジサーバなど、広域分散環境でインターネットサービスの提供を行う機会が増加している。それに伴い、ストレージも広域分散化しており、データの一貫性を適切に管理する必要がある。Quorum プロトコルは、一定の制約のもとで書き込み、読み込みを行う複製の台数を柔軟に変えることができるプロトコルであり、サービスの性質に応じて可用性や応答性を適切に調整することができる。本研究では、応答時間を考慮した Quorum プロトコルの複製の台数を自動設定する機構を提案する。本論文は、提案アルゴリズムについて説明し、その議論を行う。

A Flexible Latency-Aware Optimization of Quorum-based Protocol

Munetoshi Ishikawa[†] Akiyoshi Sugiki[†] Takahiro Shinagawa[†]
Kazuhiko Kato^{†‡}

[†] Graduate School of Systems and Information Engineering, University of Tsukuba

[‡] CREST, Japan Science and Technology Agency

Abstract

A growing number of Internet services are provided through geographically distributed servers such as edge servers or clusters. In such environments, a certain level of replica consistency must be preserved. A Quorum-based protocol is a kind of replication protocol which can dynamically change the number of replicas that clients write to or read from depending on various circumstances. By changing the number of replicas, each service can satisfy a specific requirements for service availability and latency. In this paper, we present a flexible, latency-aware optimization method for quorum-based protocol. This paper specifically deals with the details of our protocol and discusses basic assumptions and future directions.

1 はじめに

近年、インターネットサービスの規模が大きくなるにつれて、サービスに用いるストレージも複数の

計算機で構成されることが多くなっている。現在、これらのストレージは一カ所のデータセンターに集約されることが多いが、サービスの規模が拡大した場合、複数のデータセンターにまたがって広域分散

配置されることが予想される。また、Akamai[1]をはじめとするエッジサーバも広く利用されており、広域分散環境でストレージを利用する機会が増加している。

ストレージの複製を広域分散環境上に配置する場合、遠隔地に配置されたストレージ計算機の応答時間が増加したり、一時的なネットワーク障害の影響を受ける。その結果、ストレージ全体としての応答時間や可用性といった性能が低下する可能性がある。そのため、高い性能が必要となる状況では、広域分散環境に適した複製の一貫性管理方法が要求される。

広域分散環境に適した複製の一貫性管理方式として Quorum プロトコル [5, 11] が知られている。Quorum プロトコルとは、書き込み、読み込みの操作ごとに投票を行い、定足数 (Quorum) 以上の計算機から応答が得られた場合に操作を行うプロトコルである。従来のプロトコル (ROWA (Read-One Write-All) と呼ばれる) が 1 台の計算機から読み込みを行い、全計算機から書き込みを行っていたのに対して、Quorum プロトコルはこれらの操作を行うのに必要な定足数を任意の組み合わせに変えることができる。この定足数を環境に応じて適切に設定することで、高い可用性を保つことができるという利点がある。

従来、定足数の調整は手動で行われていたが、定足数の自動設定に関する研究がいくつか進められている。これらの研究は、可用性 [3] や応答時間 [6, 9, 10]、ネットワーク全体の通信コスト [7, 8] を最適化するものなど様々な研究がある。特に、応答時間に関して最適化する研究では、応答時間の最大値や平均値を基準として最適化している。

しかし、インターネットサービスでは、応答時間の最大値の条件を満たしつつ、できるだけ応答を早く応答を返して欲しい場合や、反対に、応答は基準を満たしていればよいので、複製の台数をできるだけ減らしたい場合など、より柔軟に対応したい場合がある。

そこで本研究では、応答時間の要求に応じて柔軟に定足数を調整する Quorum プロトコルの自動設定を提案する。まず、提案手法ではストレージの利用者に応答時間の評価関数を定義してもらう。この評価関数は、応答時間が与えられるとそれに応じた評価値を出力する。次に、この評価関数を用いて、考える定足数の組み合わせを評価し、最大となるものを出力する。

本論文では、まず 2 章で Quorum ベースプロトコルとその自動設定に関する研究を説明する。次に 3 章で提案アルゴリズムの詳細について説明する。さらに 4 章で、アルゴリズムの計算量や今後の改善方法などの議論を行う。最後に 5 章で、本論文のまとめを述べる。

2 背景と関連研究

2.1 Quorum プロトコル

Quorum プロトコル [5, 11] は、書き込みや読み込みなどの操作毎に、各計算機に投票 (vote) を行い、一定数 (Quorum) 以上の計算機から応答が得られた場合に操作を行うプロトコルである。

この Quorum プロトコルを広域分散環境で用いる利点は、大きく 2 つある。1 つ目は、データのアクセスパターンに応じて、書き込みや読み込みの定足数を適切に設定することで、ストレージ全体の可用性を高めることができる点である。2 つ目は、いくつかの計算機の中から、定足数以上から応答が得られればよいので、小規模なネットワーク分断や計算機の故障を隠蔽することができる点である。

本研究では、Quorum プロトコルの一つである Read-Write Coterie[2, 4] を利用する。このプロトコルでは図 1 のように、書き込み操作を行うグループ (書き込みグループ) を作成し、それを用いて書き込みの定足数を表現する。図 1 では、8 つの計算機 n_{1-8} で 3 つの書き込みグループ G_{1-3} が作成されており、書き込みグループ G_1 の計算機 n_{1-4} から応答が得られている様子を示している。読み込みの場合は図 2 のように、書き込みグループの制約に基づいて操作を行う。書き込みグループは、計算の無駄を省くという理由と、一貫性を満たす必要があるという理由により、以下の 2 つの条件を満たす必要がある。

1. 全ての異なる 2 つの書き込みグループが包含関係にない (Minimize property)
2. 全ての異なる 2 つの書き込みグループが交差する (Intersection property)

図 1 では、グループ G_{1-3} 全てにおいて、他のグループと包含関係になく、かつ他のグループと共通の計算機を持つので、これらの条件を満たしていると言える。これらの条件は、書き込みグループの集合 Q

を用いて、以下のように形式的に示せる。

$$(\forall G_1, G_2 \in Q) (G_1 \not\subseteq G_2) \quad (1)$$

$$(\forall G_1, G_2 \in Q) (G_1 \cap G_2 \neq \phi) \quad (2)$$

(1) は、ある書き込みグループの基準を満たせば、内側のグループの基準も満たされるので、無駄を取り除くためにある。(2) は、書き込みや読み込みが排他的に行われることを保証するためにある。この条件がないと、複数の操作が並列に行われる危険がある。これらの条件を満たすと、One-copy Serializability (ISR) を満たすことができるため、多くのインターネットサービスに必要となる一貫性を満たすことができる。One-copy Serializability とは、ストレージの利用者からは、単一のオブジェクトのように見えるという一貫性である。

書き込み操作は図 1 のように、書き込みグループの集合 Q のうち、どれか一つのグループ内の全ての計算機が応答がある場合に行う。この図では、グループ G_1 の全ての計算機 n_{1-4} の各計算機からの応答が得られたので、これらの計算機に対して書き込みを行っている。ここで (2) の条件により、 G_2, G_3 は G_1 と交差しているので、同時に書き込みが行われないことが保証できる。書き込みが行える条件を、投票が得られた計算機の集合 V を用いて形式的に示すと以下ようになる。

$$(\exists G \in Q) (G \subseteq V) \quad (3)$$

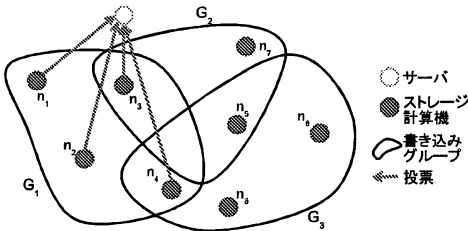


図 1: 書き込み操作が許可される状態

一方、読み込み操作は図 2 のように、全ての書き込みグループで一台以上の計算機から応答がある場合に行う。この図では、グループ G_1 に含まれる計算機 n_2 と、グループ G_2, G_3 の両方に含まれる計算機 n_5 からの応答が得られたので、これらの計算機から読み込みを行っている。どのグループに最新の

書き込みが行われているか判らないので、全ての書き込みグループからの応答を待ち、タイムスタンプを比較して最新のデータを取得する。読み込みが行える条件を、投票が得られた計算機の集合 V を用いて形式的に示すと以下ようになる。

$$(\forall S \in Q) (S \cap V \neq \phi) \quad (4)$$

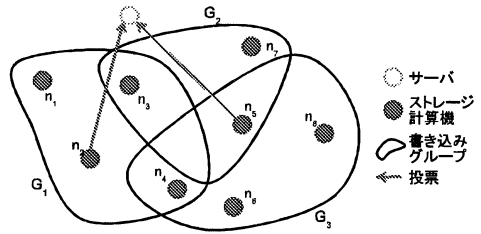


図 2: 読み込み操作が許可される状態

2.2 Quorum プロトコルの自動設定に関する研究

従来より、Quorum プロトコルの定足数の自動設定に関する研究がいくつか進められている。Quorum プロトコルを利用するアプリケーションによって、最適化の基準が異なるため、応答時間を最適化する研究 [6, 9, 10]、可用性を最適化する研究 [3]、ネットワーク全体の通信コストを最適化する研究 [7, 8] などがある。本論文では、インターネットサービスを想定し、応答時間に関して最適化を行うため、応答時間に対して最適化している関連研究について説明する。

応答時間に対して最適化している従来の研究では、応答時間の最大値や平均値を基準として最適化している。最大値を基準としている研究に、Lin らの研究 [9] がある。しかし、インターネットサービスでは、応答時間の最大値の条件を満たしつつ、できるだけ応答を早く応答を返して欲しい場合や、反対に、応答は基準を満たしていればよいので、複製の台数をできるだけ減らしたい場合など、より柔軟に最適化を行いたい場合がある。その場合、最大値のみを基準とした場合には、最悪の状況のみを考慮して最適化しているため対応が難しい。

Gupta ら [6] や Oprea ら [10] は、応答時間の平均値を基準として最適化している。しかし、インターネットサービスにおいては、応答時間の平均値

が同じであっても、ほとんどの応答時間が平均値に集まっている場合や、反対に、それぞれの応答時間を見ると大きな広がりがある場合がある。平均値のみを基準とした場合には、平均的な振る舞いのみを考慮しているため、応答時間の広がりについては十分な対応が難しい。

インターネットサービスには様々なサービスがあるので、応答時間について最適化する場合でも、サービスによって基準が異なる可能性がある。本論文では、様々なサービスに柔軟に対応できる、応答時間の最適化手法を提案する。

3 提案手法

3.1 提案手法の概要

本研究では、様々なサービスの要求を満たすために、応答時間の評価関数をサービスの種類に合わせてサービスの提供者に定義してもらう。この評価関数を用いて、それぞれのリクエストごとの応答時間を評価することで最適化を行う。

例として図3の3つの評価関数を用いて説明する。応答時間の評価関数は、応答時間が与えられるとそれに応じた操作可用性を評価値として出力する。操作可用性とは、データが消失しないだけでなく、操作までが完了できる割合である。この操作可用性は0から1までの値をとり、このグラフでは横軸が評価関数に与える応答時間を示し、縦軸がそのときの操作可用性を示す。評価関数1は、応答時間に関係なく操作が完了する割合をできるだけ高めるという場合である。そのため、どのような応答時間となっても、常に1を返している。評価関数2は、操作にタイムアウトを設けたい場合である。ここでは t_1 をタイムアウトとして、 t_1 までに応答があれば1を返し、それ以上となれば0を返す。評価関数3は、操作にタイムアウトを設けて、それを超えると評価を徐々に下げる場合である。タイムアウト t_1 までの応答は1を返し、それ以上となれば t_2 までで徐々に下げていく。このように評価関数を変えることで様々なサービスの要件に対応させることができる。

しかし、応答時間のみを考慮すると、データの複製数が極端に少なくなることがある。この複製数がサービスの基準を満たしているかどうかは、データ可用性を用いることで評価できる。データ可用性とは、データが消失しない割合である。サービスの提供者にデータ可用性の目標値も同時に与えてもらい、

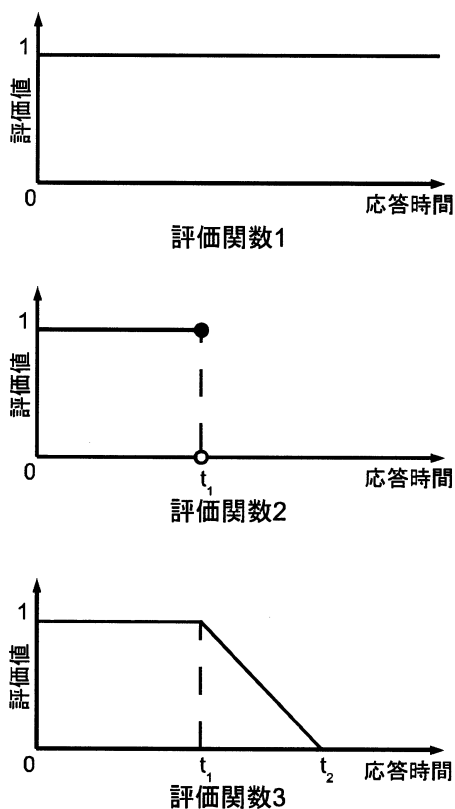


図 3: 評価関数の例

その目標を満たしつつ、応答時間の評価が良くなる書き込みグループの組を選択する。

本論文で提案するアルゴリズムの概要を示す。本アルゴリズムは2段階で行う。まず、全ての計算機の中からデータ可用性の目標値を満たす書き込みグループを列挙する。次に、それぞれの書き込みグループの組を応答時間の評価関数を用いて評価し、最も評価値の高い組み合わせを選択し、出力する。

3.2 入力情報

本提案手法では、最適化に用いる情報として、サービス提供者が入力する4つの情報と、本手法が自動的に環境から収集する2つの情報の2種類を用いる。

サービス提供者が入力しなくてはならない情報は、次の4つである。

- 全計算機の集合 U : ストレージの複製先として

利用可能な計算機を定義する。

- 各計算機 $n \in U$ のデータ可用性 A_{d_n} : ハードウェア仕様書などを参照して大まかな値を入力したり、過去の長期間にわたる計測から推測して決定する。
- データ可用性の目標値 A_d : サービスごとに満たしたいデータ可用性の目標値を定義する。
- 応答時間 l の書き込み・読み込みの応答時間の評価関数 $f_w(l), f_r(l)$: 3.1 節で示した評価関数を、書き込み操作、読み込み操作ごとに与えてもらう。

提案手法は、以下の3つの情報を自動的に環境から収集し、利用する。

- 各計算機 $n \in U$ ごとの書き込み・読み込み操作の応答時間の確率分布 $d_{w_n}(l), d_{r_n}(l)$
- 書き込み読み込みのリクエストが完了した割合 A_{w_n}, A_{r_n}
- 書き込み・読み込み操作の割合 $p, 1-p$

これらは、十分な期間にわたってリクエスト情報を収集し、計算する。

3.3 アルゴリズムの詳細

本アルゴリズムは、処理を2段階に分けて行う。まず、データ可用性の条件を満たす書き込みグループを列挙する。次に、そのグループの組の中から最も評価値の高いグループの組を出力する。これらについて、それぞれ3.3.1節、3.3.2節で説明する。

3.3.1 書き込みグループの列挙

利用可能な計算機の集合 U が与えられた場合、考えられる書き込みグループは冪集合 2^U の要素で表すことができる。

これらの書き込みグループの中で、データ可用性の目標値 A_d を満たす書き込みグループを列挙する。書き込みグループを列挙した集合を S とすると、以下の式で表すことができる。 A_{d_G} はある書き込みグループ G のデータ可用性である。

$$S = \{G | G \in 2^U, A_{d_G} \geq A_d\} \quad (5)$$

A_{d_G} は次のように計算する。 A_{d_G} は G 内で、同じ値を書き込んだ計算機の内、どれか1つ以上の計算機が生存していればよいので、以下のように計算

することができる。各計算機 $n \in G$ のデータ可用性は A_{d_n} である。

$$A_{d_G} = 1 - \prod_{n \in G} (1 - A_{d_n}) \quad (6)$$

書き込みグループの集合 S の組み合わせ 2^S の中で、2節の式1, 2を満たしていることを $C(Q)$ とすると、これを満たす組み合わせ T は以下のように表すことができる。

$$T = \{Q | Q \in 2^S, C(Q)\} \quad (7)$$

3.3.2 操作可用性の算出

書き込みグループの組 $Q \in T$ の中で、応答時間を評価した操作可用性 $A_{op}(Q)$ が最も高い組み合わせを出力する。これを数式で表すと、以下になる。

$$\arg \max_{Q \in T} A_{op}(Q) \quad (8)$$

操作には、書き込みと読み込みが含まれるので、これらの操作可用性 $A_{op_w}(Q), A_{op_r}(Q)$ として、 $A_{op}(Q)$ を以下のように計算する。 p は3.2節の書き込み操作の割合である。

$$A_{op}(Q) = p \cdot A_{op_w}(Q) + (1-p) \cdot A_{op_r}(Q) \quad (9)$$

書き込みの操作可用性 $A_{op_w}(Q)$ は次のように計算する。

$$A_{op_w}(Q) = A_{w_Q} \int_0^\infty f_w(l) \cdot d_{w_Q}(l) dl \quad (10)$$

A_{w_Q} は、組み合わせ Q で書き込みが完了する割合である。積分部分はの応答時間の評価を行っており、応答時間の分布 $d_{w_Q}(l)$ と評価関数 $f_w(l)$ を利用している。

読み込みの操作可用性 $A_{op_r}(Q)$ も同様に、次のように計算する。

$$A_{op_r}(Q) = A_{r_Q} \int_0^\infty f_r(l) \cdot d_{r_Q}(l) dl \quad (11)$$

ここで、 A_{r_Q} は、組み合わせ Q で読み込みが完了する割合であり、 $d_{r_Q}(l)$ は、 Q の応答時間の分布である。

以降で A_{w_Q}, A_{r_Q} と $d_{w_Q}(l), d_{r_Q}(l)$ の計算方法について説明する。まず、 A_{w_Q}, A_{r_Q} を説明する。書き込み操作はいずれかの書き込みグループから応答があればよく、読み込み操作は全ての書き込みグループから応答がなくてはならない。まず、単純な場合と

して各書き込みグループが交差をしていない場合について説明する。 A_{w_Q}, A_{r_Q} は、書き込みグループ $G \in Q$ で操作が完了する割合を A_{w_G}, A_{r_G} として、次のように計算できる。

$$A_{w_Q} = \sum_{G \in Q} A_{w_G} \quad (12)$$

$$A_{r_Q} = 1 - \sum_{G \in Q} (1 - A_{r_G}) \quad (13)$$

式 12 は、いずれかの書き込みグループからの応答があればよいので和となる。式 13 は、全ての書き込みグループからの応答がなくてはならないので、1 からいずれかの書き込みグループからの応答がない割合を引いている。

しかし、実際の書き込みグループは交差しているので、その重複部分を取り除く計算をする必要がある。式 12, 13 の A_{w_Q}, A_{r_Q} は次のようになる。

$$A_{w_Q} = \sum_{R \in 2^Q} (-1)^{|R|-1} \cdot A_{w_R} \quad (14)$$

$$A_{r_Q} = 1 - \sum_{R \in 2^Q} \left(1 - (-1)^{|R|-1} \cdot A_{r_R}\right) \quad (15)$$

$$\text{ただし } G = \bigcap_{H \in R} H$$

式 14, 15 は、書き込みグループが奇数個重なっている場合は和をとり、偶数個重なっている場合は差をとっている。

A_{w_G}, A_{r_G} は次のように計算する。書き込みの場合の A_{w_G} はグループ内の全ての計算機に行わなくてはならないので積となり、読み込みの場合の A_{r_G} は読み込み操作はグループ内のいずれかの計算機に行えばよいので、次のように計算できる。 A_{w_n}, A_{r_n} は、3.2 節で説明した各計算機 $n \in G$ の操作が完了する割合である。

$$A_{w_G} = \prod_{n \in G} A_{w_n} \quad (16)$$

$$A_{r_G} = 1 - \prod_{n \in G} (1 - A_{r_n}) \quad (17)$$

次に、応答時間の確率分布 $d_{w_Q}(l), d_{r_Q}(l)$ の計算方法について説明する。これらは、応答時間が l 以下になる確率 $P_{w_Q}(X \leq l), P_{r_Q}(X \leq l)$ を一度計算し、それらを微分することで求めることができる。さらに $P_{w_Q}(X \leq l), P_{r_Q}(X \leq l)$ は、式 14-17 と同様に計算できる。 $P_{w_G}(X \leq l), P_{r_G}(X \leq l)$ は、書

き込みグループ $G \in Q$ の応答時間が l 以下になる確率であり、 $P_{w_n}(X \leq l), P_{r_n}(X \leq l)$ は、計算機 $n \in G$ の応答時間が l 以下になる確率である。

$$P_{w_Q}(X \leq l) = \sum_{R \in 2^Q} (-1)^{|R|} \cdot P_{w_R}(X \leq l) \quad (18)$$

$$P_{r_Q}(X \leq l) = 1 - \sum_{R \in 2^Q} \left(1 - (-1)^{|R|} \cdot P_{r_R}(X \leq l)\right) \quad (19)$$

$$\text{ただし } G = \bigcap_{H \in R} H$$

$$P_{w_G}(X \leq l) = \prod_{n \in G} P_{w_n}(X \leq l) \quad (20)$$

$$P_{r_G}(X \leq l) = 1 - \prod_{n \in G} (1 - P_{r_n}(X \leq l)) \quad (21)$$

$P_{w_n}(X \leq l), P_{r_n}(X \leq l)$ は、3.2 節の各計算機 $n \in G$ の応答時間の分布関数 $d_{w_n}(l), d_{r_n}(l)$ を用いて計算が可能であるため、 $d_{w_Q}(l), d_{r_Q}(l)$ も計算可能となる。

4 議論と今後の予定

現在、研究がまだ初期段階にあり、大きく 3 つの改良を行う必要がある。まずは、計算量の削減である。提案アルゴリズムは、現在、全数探索を行っており、書き込みグループの列挙のために $O(2^n)$ 、書き込みグループの組の探索のために $O(2^{2n})$ の計算量を必要とし、複製の数が増加した場合、計算が現実的な時間で終了しない可能性がある。そのため、近似解を利用する手法 [8] やグリッド構造 [11] などのネットワーク構造を持たせプロトコルを簡略化する手法などと組み合わせることで、計算量を削減することを目指す。

また、現在のプロトコルは全ての書き込みグループに均一にアクセスすることを仮定している。すべての計算機に対して操作の要求を行い、その中で速く応答したものに対して操作を行っている。この手法はブロードキャストが利用できない環境では、リクエスト送信のためのコストが大きい。そのため、特定の計算機や書き込みグループを選択し、操作を行うような改良を行う。この実現のためには、各式の計算方法も変わることからアルゴリズム自身の改良も必要である。

現在、シミュレーションによる評価を進めている最中であり、本手法の有効性について検証する。また、実計算機のストレージにも本手法を実装し、実機上でも評価を行うことを目指す。

5 まとめ

インターネットサービスではさまざまなサービスがあり、サービスごとに可用性や応答時間に対する要求が異なる。Quorum プロトコルは、一定の制約のもとで書き込み、読み込みを行う複製の台数を柔軟に変えるプロトコルであり、サービスの性質に応じて可用性や応答性を適切に調整することができる。本論文では、応答時間の要求に応じて柔軟に対応することができる Quorum プロトコルの複製数の自動設定機構を提案した。本方式はサービスの提供者に応答時間の評価関数を定義してもらい、個別のリクエストの応答時間ごとにその評価関数を用いて評価を行うことで、複製数の最適化を行う。また、応答時間のみを考慮すると複製の台数が極端に少なくなる可能性があることから、データ可用性の目標値を設け、その目標値を満たす設定の中から応答時間の評価が最も高くなるものを選択する。今後、シミュレーションや実機での評価や改良を進めていく。

参考文献

- [1] Akamai: <http://www.akamai.com/>.
- [2] Barbara, D. and Garcia-Molina, H.: Mutual Exclusion in Partitioned Distributed Systems, *Distributed Computing*, Vol. 1, No. 2, pp. 119–132 (1986).
- [3] Cheung, S. Y., Ahamad, M. and Ammar, M. H.: Optimizing Vote and Quorum Assignments for Reading and Writing Replicated Data, *Transactions on Knowledge and Data Engineering*, Vol. 1, No. 3, pp. 387–397 (1989).
- [4] Garcia-Molina, H. and Barbara, D.: How to Assign Votes in a Distributed System, *Journal of ACM*, Vol. 32, No. 4, pp. 841–860 (1985).
- [5] Gifford, D. K.: Weighted Voting for Replicated Data, *SOSP '79: Proceedings of the seventh ACM symposium on Operating systems principles*, pp. 150–162 (1979).
- [6] Gupta, A., Maggs, B. M., Oprea, F. and Reiter, M. K.: Quorum Placement in Networks to Minimize Access Delays, *PODC '05: Proceedings of the twenty-fourth annual ACM symposium on Principles of Distributed Computing*, pp. 87–96 (2005).
- [7] Kumar, A. and Segev, A.: Optimizing Voting-Type Algorithms for Replicated Data, *EDBT '88: Proceedings of the International Conference on Extending Database Technology*, pp. 428–442 (1988).
- [8] Kumar, A. and Segev, A.: Cost and Availability Tradeoffs in Replicated Data Concurrency Control, *ACM Transaction on Database and Systems*, Vol. 18, No. 1, pp. 102–131 (1993).
- [9] Lin, X.: Delay Optimization in Quorum Consensus, *Algorithmica*, Vol. 38, No. 2, pp. 397–413 (2003).
- [10] Oprea, . and Reiter, .: Minimizing Response Time for Quorum-System Protocols over Wide-Area Networks, *DSN '07: 37th Annual IEEE/IFIP International Conference on Dependable Systems and Networks*, pp. 409–418 (2007).
- [11] Thomas, R. H.: A Majority Consensus Approach to Concurrency Control for Multiple Copy Databases, *ACM Transaction on Database and Systems*, Vol. 4, No. 2, pp. 180–209 (1979).