構造化オーバーレイにおけるデータの複製による マルチクエリに対する応答の高速化

Speeding Up of Response to a Multi-query Using Replication of Data in Structured Overlay Networks

小泉 悠介 Yusuke Koizumi　渡部 康平 Kohei Watabe　中川 健治 Kenji Nakagawa

長岡技術科学大学 大学院 工学研究科
Graduate School of Engineering, Nagaoka University of Technology

1 概要

オーバーレイネットワークにおいてデータ間の関係性をデータ配置へ反映することは，クエリに対する応答を高速化する上で重要である．文献 [1] で提案された構造化オーバーレイネットワークでは，隣り合う ID を持つデータを同じノードに配置している．要求するデータを ID の範囲で指定するレンジクエリに対応するためだが，隣り合う ID を持つデータは同時に取られやすいという関係性に基づき，データを配置していると捉えられる．しかし，各データ間の関係性は複雑であり，ID によって表現できるとは限らない．特に，特定の条件に該当するデータを複数指定するマルチクエリでは多くの関係性を含む．

そこで本研究では，ID に反映されない各データ間の関係性を，データの複製によってデータ配置へ反映する．これにより，マルチクエリに含まれる全てのデータを 1 つのノードで取得できるようにすることで，マルチクエリに対する応答を高速化する手法を提案する．

2 提案法について

提案法では，各ノードにおいて，保存されているデータと関係性が深いと考えられるデータを他ノードから複製して保存する．提案法を用いない場合に比べて，多くの保存領域が必要となるが，各ノードでマルチクエリに含まれるデータが全て保存されている場合が多くなるため，応答を高速化することができる．関係性が深いと考えられるデータは，過去に送られてきたクエリの集合 A を利用することで推定する．各ノードは，1 つのノードで全てのデータを取得できるマルチクエリの数を最大化する．

上記の問題は，整数計画法により下記のように定式化することができる．各ノードにおいて下記の問題の解を求めることにより，どのデータを保存するか決定する．

$$\begin{cases} \text{maximize} & \sum_{i=1}^{n} x_i \\ \text{subject to} & \sum_{j=1}^{m} y_j \leq c, \\ & y_j \geq d_{ij} x_i, \quad \forall i, j \end{cases}$$

$$\begin{pmatrix} c & : & \text{他ノードから複製したデータの保存領域のサイズ} \\ n & : & \text{利用するクエリの集合 } A \text{ の要素数} \\ m & : & A \text{ に含まれるデータの集合の要素数} \\ d_{ij} \in \{0,1\} & : & i \text{ 番目のクエリに } j \text{ 番目のデータを含む場合は 1，それ以外は 0 となる指示関数} \\ x_i \in \{0,1\} & : & i \text{ 番目のクエリに含まれるデータを保存する場合は 1，それ以外は 0 となる指示関数} \\ y_j \in \{0,1\} & : & j \text{ 番目のデータを保存する場合は 1，それ以外は 0 となる指示関数} \end{pmatrix}$$

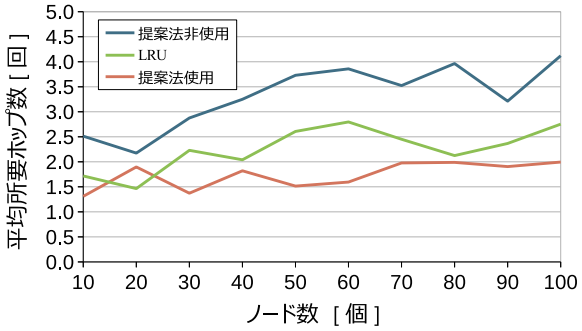

図 1　シミュレーション結果

3 シミュレーション

文献 [1] で提案された構造化オーバーレイネットワークにおいて，ノード数を変化させたとき，マルチクエリの平均所要ホップ数がどのように変化するかシミュレーションを行った．今回シミュレーションで用いたクエリ生成モデルでは，データは 2 次元トーラス上で関係性を表現できるとし，任意の長方形で囲む範囲のデータがマルチクエリとして要求されるとする．長方形の各辺はパラメータ $s = 1.4, N = 10$，左下頂点の座標はパラメータ $s = 1.4, N = 100$ のジップ分布に従うとする．ネットワーク全体で保存してあるデータ数を 10000 個，他ノードから複製したデータの保存領域のサイズを 30 個として，ノード数を 10 個から 100 個まで変化させた．このときマルチクエリ処理を 1000 回行い平均所要ホップ数を算出した．各ノードで前述の問題に対する解を算出するのに要する時間は数秒程度であった．提案法を用いない場合，提案法を用いた場合でシミュレーションを行った．提案法を用いない場合と提案法を用いた場合では，各ノードに保存されるデータ数が異なる．比較のため，他ノードから複製したデータの保存領域を LRU で管理した場合についてもシミュレーションを行った．

シミュレーション結果を図 1 に示す．図 1 より，提案法を用いない場合に比べ，提案法を用いた場合の方が平均所要ホップ数が減少していることが分かる．これは，提案法を用いることによって，各ノードでマルチクエリに含まれるデータが全て保存されている場合が多くなるためである．

4 今後の方針

提案法における定式化した問題の解を探索的に求める方法を検討する．

参考文献

[1] T. Schütt, F. Schintke, A. Reinefeld, “Range queries on structured overlay networks”, Computer Communications Vol. 31, No. 2, 2008.