M-005

メッセージブローカの冗長化における複製オーバヘッドの評価
Evaluation of Replication Overhead in Message Broker Redundancy

藤本 剛瑠[1)]　乃村 能成[2)]
Takeru Fujimoto　Yoshinari Nomura

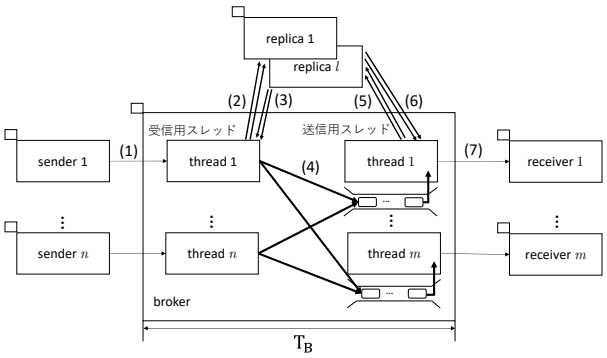


図 1: メッセージの流れ

1. はじめに

分散システムの冗長化の一手法としてレプリカの作成がある．この手法は冗長性向上と性能低下のトレードオフとなる．ここで，性能低下の要因として CPU 処理と通信処理の増加が考えられる．これらは並行処理を用いることで一部が圧縮されるため，冗長化によるオーバヘッドの増加はレプリカ数の増加に伴う単純な累加よりも小さいことが期待されるが，定量的な予測は難しい．

そこで，本稿では分散システムとしてのメッセージブローカについて，レプリカ作成を行う際のオーバヘッドを評価し，報告する．ブローカの性能や可用性がシステム全体の性能や信頼性に影響する [1] ため，求めるサービスに応じてこれらの程度が選択されることが重要である [2]．具体的な評価としては，レプリカ数を 0 から順に増やした際の，システム全体の処理時間の増加傾向を調査する．調査結果から，レプリカ数に対する全体の処理時間を導出するための各オーバヘッド要因どうしの関係を明らかにする．

2. 評価システム

評価のために実装したシステムを**図 1** に示す．ブローカを 1 プロセス，センダ，レシーバ，レプリカを**表 1** に応じたプロセス数ずつ用意する．センダは，メッセージパケットをブローカに送信する動作を繰り返す．ブローカは，受信したメッセージをキューに保存しつつ，レシーバに送信する．センダ，レシーバ，レプリカは 1 スレッドで動作する．ブローカはメインスレッド，センダ，レシーバの数と同数のスレッド数で動作する．

ブローカはセンダからメッセージを受信しキューにためる直前とキューから取り出したメッセージをレシーバに送信する直前にメッセージのコピーをレプリカに送信し複製させる．全てのレプリカから応答を受信すると処理を再開する．実際の評価環境を**表 2** に示す．

1） 岡山大学大学院環境生命自然科学研究科，Graduate School of Environment, Life, Natural Science and Technology, Okayama University
2） 岡山大学環境生命自然科学学域，Faculty of Environment, Life, Natural Science and Technology, Okayama University

表 1: 測定条件

通番	項目	値
1	sender 数	1, 10, 100
2	receiver 数	1, 10, 100
3	replica 数	0, 1, 2, 3, 4
4	message 数	100000

表 2: 評価環境

通番	種類	内容
1	OS	Ubuntu 22.04.3 LTS
2	Kernel	6.5.0-14-generic
3	CPU	AMD Ryzen 9 5950X (32) @3.4GHz
4	memory	64GB
5	compiler	rustc 1.71.0 (8ede3aae2 2023-07-12)

3. 評価

3.1 評価項目

評価項目を以下に示す．

1. 複製により発生するオーバヘッド．
2. オーバヘッドにおけるブローカとレプリカ間の通信処理時間とレプリカの CPU 処理時間の割合．

本評価では全てのプロセスを単一計算機内に配置して測定を行った．設計したブローカはメッセージを各レプリカに順番に送信，その後レプリカから Ack を受信する．ここで，複製処理は大別するとブローカとレプリカ間の通信処理とレプリカの CPU 処理に分けられる．通信処理時間はレプリカ数に比例すると考えられる．また，レプリカの CPU 処理はレプリカによる受信が完了した直後から始まる．このことから，レプリカ数の増加がオーバヘッド増加の直接的な原因になると考えられる．また，メッセージ複製のオーバヘッドを T，ブローカとレプリカ間の通信処理時間を送信処理時間 T_{send} と受信処理時間 T_{recv} とし，レプリカの CPU 処理時間を T_{rep}，レプリカ数を n とする．考えられるオーバヘッドの発生例を**図 2** に示す．レプリカ数 1 のとき，オーバヘッドは図 2(a) と次の式で表される．

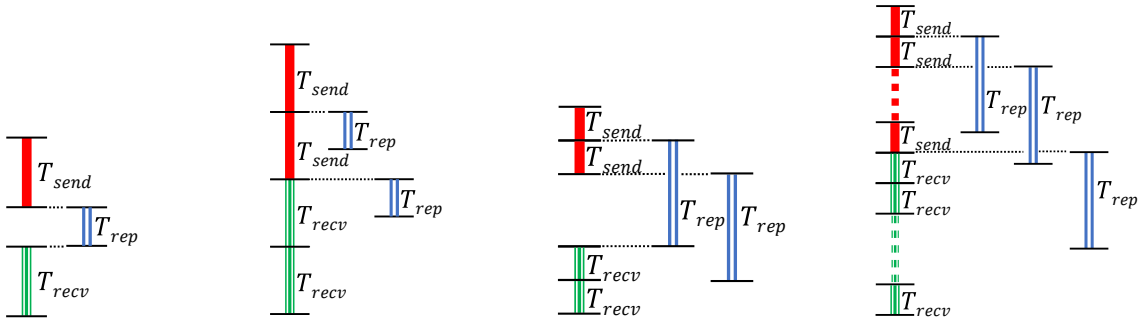
$$T = T_{send} + T_{recv} + T_{rep} \tag{1}$$

次に，レプリカ数 2 以上の場合に考えられるオーバヘッドの発生例を通信時間と CPU 処理時間を考慮に入れて次項以降で説明する．

3.1.1 通信処理時間が CPU 処理時間よりも大きい場合

各通信処理時間がレプリカの CPU 処理時間よりも大きい場合，オーバヘッドは図 2(b) と次の式で表される．ここで，CPU 処理時間は並行性により隠蔽される．

$$T = n(T_{send} + T_{recv}) \tag{2}$$



(a) $n = 1$ の場合，
$T = T_{send} + T_{recv} + T_{rep}$

(b) $min(T_{send}, T_{recv}) > T_{rep}$ かつ $n > 1$ の場合，
$T = n(T_{send} + T_{recv})$

(c) $max(T_{send}, T_{recv}) < T_{rep}$ かつ $(n-1)T_{send} < T_{rep}$ の場合，
$T = T_{send} + nT_{recv} + T_{rep}$

(d) $max(T_{send}, T_{recv}) < T_{rep}$ かつ $(n-1)T_{send} > T_{rep}$ の場合，
$T = n(T_{send} + T_{recv})$

図 2: レプリカ数を n としたときのブローカとレプリカ間の通信処理時間 (送信処理時間 T_{send}，受信処理時間 T_{recv}) とレプリカの CPU 処理時間 T_{rep} の大小関係による複製オーバヘッド T の予測

3.1.2 通信処理時間が CPU 処理時間よりも小さい場合

通信処理時間がレプリカの CPU 処理時間よりも小さい場合，次の 2 つが考えられる．

1. 2 回目以降の送信処理時間の和が CPU 処理時間より小さい場合，図 2(c) と次の式で表される．
$$T = T_{send} + nT_{recv} + T_{rep} \quad (3)$$
2. 2 つ目以降の送信処理時間の和が CPU 処理時間より大きい場合，図 2(d) と式 (2) で表される．

3.2 測定手法

センダ，レシーバ，レプリカ，メッセージの数を表 1 の条件で測定する．センダから 100,000 件のメッセージをブローカ経由でレシーバに送信し，全メッセージが送信されるまでの時間を求める．測定区間を図 1 の T_B，すなわち処理 (1) から (7) の区間で表す．全体の処理時間は最初のメッセージの受信時刻から最後のメッセージの送信時刻までの時間とする．

3.3 測定結果

3.3.1 複製により発生するオーバヘッド

測定結果を**図 3** に示す．センダ数，レシーバ数ともに 1 のときに注目し，レプリカ数 0 と 1 を比較すると，全体の処理時間が約 3 倍に増加することが分かる．この増加分をオーバヘッドと見なすと，1 度でも複製すると性能が約 3 分の 1 以下になると考えられる．一方で，レプリカ数 1 と 2 以降を比較すると，オーバヘッドの増加量はレプリカ数 0 と 1 のそれより小さい．これは，並行性により一部の処理が重複するためだと考えられる．

3.3.2 オーバヘッドにおけるブローカとレプリカ間の通信処理時間とレプリカの CPU 処理時間の割合

センダ数，レシーバ数ともに 1 のときに注目する．仮に測定結果が図 2(b) の場合だと考えると，レプリカ数 0 から 1 のオーバヘッドの少なくとも約 3 分の 2 は通信処理時間であると考えられる．しかし，レプリカ数 1 から 2 以降のオーバヘッドの増加量はこの仮定に矛盾する．一方で，測定結果が図 2(c) に該当するとし，レプリカの CPU 処理時間が十分に大きいと仮定すると，各レプリカ数のオーバヘッドの増加量はこの仮定に矛盾しない．これらから，本稿の測定における複製オーバヘッドの予測は図 2(c) のモデルにより説明できると考えられる．また，以上を踏まえると，レプリカ数 0 と 1 の差をオーバヘッド T，1 と 2 の差を T_{recv} と考えられる．ここで，$T_{recv} = T_{send}$ と単純化すると $T_{rep} = T - 2T_{recv} = 2T_{recv}$ すなわち，レプリカの CPU 処理時間が通信処理時間の約 2 倍であると言える．レプリカをさらに増やすことで図 2(d) になると考えられる．

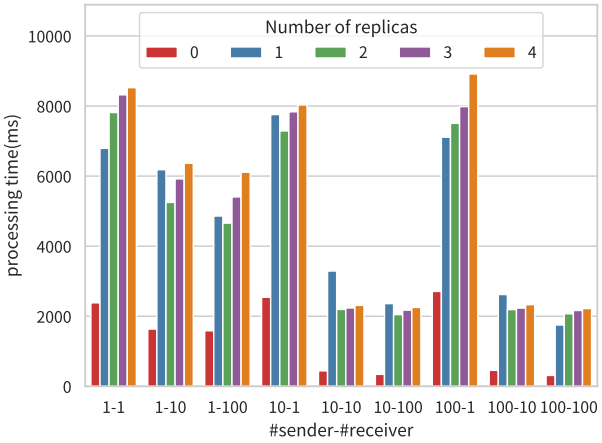


図 3: 各レプリカ数におけるセンダとレシーバの組ごとの平均処理時間 (ms)

また，センダ数，レシーバ数が 1 以外のとき，仮定に反する場合があり，送信，受信処理時間に関する単純な仮定も当てはまらない可能性がある．これらについては詳細な分析が必要である．

4. おわりに

本稿では分散システムとしてメッセージブローカを用いて，レプリカ作成による冗長化の性能評価を行った．レプリカの発展として Raft[3] をはじめとしたコンセンサスアルゴリズムによる冗長化が考えられる．

参考文献

[1] 中川雄介，乃村能成：通信プロトコルの違いによるメッセージブローカの処理時間の比較，第 20 回情報科学技術フォーラム (FIT2021) 講演論文集，Vol. 4, pp. 221–222 (2021).

[2] Fu, G., Zhang, Y. and Yu, G.: A Fair Comparison of Message Queuing Systems, *IEEE Access*, Vol. 9, pp. 421–432 (online), DOI: 10.1109/ACCESS.2020.3046503 (2021).

[3] Ongaro, D. and Ousterhout, J.: In Search of an Understandable Consensus Algorithm, Philadelphia, PA, USENIX Association, pp. 305–319 (2014).