通信帯域に基づく状態分割を用いた 広域 State Machine Replication における状態転送手法

千葉泰理[1] 大村廉[1] 中村純哉[2]

概要： 本研究では，通信帯域に基づく状態分割を用いた広域 State Machine Replication (SMR) における状態転送手法を提案する．SMR はサービスを複数のレプリカに複製することで耐障害性を向上させる仕組みであり，レプリカの追加・回復時にレプリカ間でサービスの状態を転送する．本手法では，転送する状態をチャンクの集合として扱う．受信レプリカは，各レプリカとの通信帯域に応じた数のチャンクを送信側に要求する．送信レプリカは，要求された数のチャンクを転送する．送信レプリカごとに通信帯域に差がある場合は要求する転送チャンク数を増減する．これにより既存手法で見られた，広域 SMR の不均一な通信帯域による性能低下を改善する．Amazon EC2 に構築した広域 SMR 上で実施した評価実験から，提案手法は Bessani らの手法に対して最大 42%の状態転送速度の向上がみられたが，条件によっては Bessani らの手法に比べて状態転送速度が低下することがあった．この転送速度低下に対して検証を行った結果，転送速度の低下が起こる場合，事前に計測を行った通信帯域と比べて実際の通信帯域が低くなっていることを確かめた．

1. はじめに

State Machine Replication (SMR)[1] とは，サービスの耐障害性を向上させるための手法である．SMR では，サービスを複数のサーバー（レプリカ）に複製し，レプリカ間でリクエストの処理順序を合意することで，各レプリカの状態を同一に保ち続ける.これにより，一部のレプリカに故障が発生しても他のレプリカがサービスを継続することが可能となる．さらに，SMR を地理的に離れた複数のレプリカで行う場合，地震等の大規模災害への耐性を持たせることが可能になる. このようなシステムは広域 SMR と呼ばれる．

SMR では，レプリカの追加や回復時に，レプリケーションに参加している他のレプリカと状態を同一に保つために，他のレプリカからサービスの最新の状態を転送する．この処理を状態転送と呼ぶ．状態転送によって，SMR はレプリカの故障やネットワークの分断が起こって一部のレプリカを失ったとしても，システム全体を停止せずにレプリカを復帰させたり，新たなレプリカを追加したりできる．Schneider は状態転送の手法として，単一のレプリカから状態全体を取得する手法を紹介した[1]．また，状態転送の実行中はサービスの処理能力が低下するため, Bessani らは状態を等分割し，分割された状態（部分状態）を複数レプリカで分担して同時に転送することで状態転送時間を短縮する手法を提案した[2].

Bessani らの手法は，同一拠点内における SMR を想定しており，レプリカ間の通信帯域のばらつきが小さいことを前提としている．しかし，広域 SMR では，拠点間を結ぶ広域ネットワークを経由して通信を行うため，レプリカ間の通信帯域や通信遅延時間が大きく異なる場合が多い．このような状況下では，同一サイズの部分状態を転送したとしても，レプリカ間の通信帯域に応じて転送時間に差異が生じ，最も通信帯域が狭いレプリカ間の転送時間がボトルネックとなる．すなわち，Bessani の手法を広域 SMR に適用したとしても，期待する性能が得られない可能性が高い．

そこで，本論文では，各レプリカ間の通信帯域に合わせたサイズに状態を分割し，転送することを提案する．転送先レプリカが状態の分割サイズを指定して転送を要求するためには，事前に状態全体のサイズを転送元レプリカから取得する必要がある．ところが，広域 SMR では一般に通信遅延が大きい[3]ため，通信回数の増加は，状態転送時間の増加に大きく影響することとなる．そのため，提案手法では状態を一定の個数に分割されたデータ（チャンク）の集合として扱う．そして，各レプリカ間の通信帯域の比率に近い個数のチャンクを部分状態として要求する．これにより, 状態サイズを事前に取得することなく, 広域 SMR の状態転送時間を短縮する．

提案手法が広域 SMR の状態転送時間を短縮することを確認するため，Amazon EC2 上に広域 SMR システムを構築し，提案手法および既存手法の状態転送時間を測定し，比較を行った．比較の結果，提案手法は Bessani らの手法に対して状態転送速度が最大 42%向上することを確認した．一方，条件によっては Bessani らの手法に比べて状態転送速度が低下した．これは事前に計測した通信帯域と比べて実際の通信帯域が狭くなっていたことが原因である．また，総チャンク数Nを変化させて実験を行い，適切なNの値が受信レプリカごとに異なることを確かめた．

本論文は，2 章では先行研究および，比較実験で用いる既存手法を紹介する．3 章では提案手法について説明する．4 章では評価実験の方法および結果，考察を述べる．5 章では本論文のまとめを述べる．

1 豊橋技術科学大学 情報・知能工学系
2 豊橋技術科学大学 情報メディア基盤センター

2. 関連研究

2.1 State Machine Replication

State Machine Replication (SMR)[1]は，サービスを複数のサーバー（レプリカ）に複製し，全てのレプリカが同一の順序で届くリクエストを処理することで，レプリカの状態を同一に保ち，サービスの耐障害性を向上させる手法である．SMR を実現するためには，全てのレプリカがリクエストを同一の順序で届く必要があるが，実際のネットワークでは各レプリカ間の通信に異なる遅延があるため，各レプリカに同時にリクエストを送った場合でも，リクエストが各レプリカに到着するタイミングには差が生まれる．

リクエストの到着タイミングに差がある場合にも，ある集団の全参加者に正しい順序でリクエストを届ける仕組みを Total Order Broadcast または Atomic Broadcast[4]と呼ぶ．Total Order Broadcast によって，全てのレプリカにリクエストを同一の順序で届くように送信できるため，SMR を実現できる．

SMR プロトコルは対象とする故障モデルによって CFT-SMR と BFT-SMR に分けられる．CFT-SMR では，レプリカは故障後一切動作しなくなる（クラッシュ故障）．BFT-SMR は，Byzantine 故障[10]に耐性を持つ SMR である．Byzantine 故障レプリカはプロトコルに従わず，任意に振る舞う．CFT-SMR を実現するプロトコルとして Paxos[7]や Raft[8], ZAB[9], BFT-SMR を実現するプロトコルとして，PBFT[11], Zyzzyva[12], BFT-SMaRt[13] などがある．

レプリカ同士の距離を大きく離して配置する SMR は広域 SMR と呼ばれ，地震など大災害に対する耐性を持つ．広域 SMR は従来の SMR と異なる特性を持つことから，レプリカ間のレイテンシ[14][15]や各レプリカの性能[15]の不均一性などに着目した様々な改善手法が提案されている．

本論文にて提案する状態転送手法は，SMR における状態転送を対象にしており，クラッシュ故障を対象とする CFT-SMR，Byzantine 故障を対象とする BFT-SMR のどちらにも適用することができる．

2.2 SMR における状態転送

SMR における状態転送の手法として，Schneider によって紹介された状態を単一のレプリカから状態全体を取得するという手法[1]がある．この手法を 4 個のレプリカが存在する広域 SMR にて行った場合を図 1 に示す．図 1 において，緑色の円柱で表されているのがレプリカの状態であり，右下のレプリカは追加または障害からの復旧を行った後に状態転送を開始し，状態転送先となったレプリカである．転送先レプリカは，一切の状態を持っておらず，状態のサイズやハッシュ値も知らない状態で状態転送を開始する．

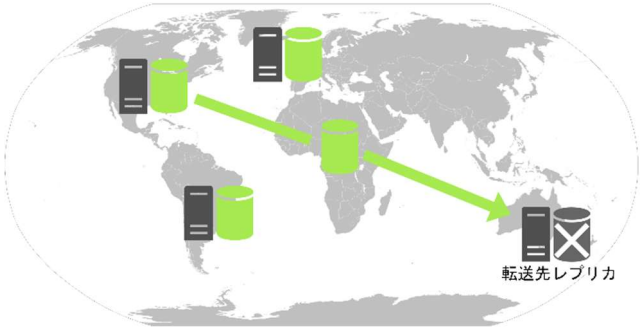


図 1 Schneider の手法を広域 SMR に適用

Schneider の手法の状態転送の流れを説明する．まず，転送先レプリカは，他のレプリカのうち一台に対して状態の転送を要求するメッセージを送信する．状態の転送要求を受けたレプリカは，転送先レプリカに対して自分の持つ状態全体を転送する．この時，BFT-SMR として動作する場合は，転送先レプリカは状態を要求したレプリカとは別のf台のレプリカに対して状態全体のハッシュ値を要求する．図 1 の場合は$f = 1$であるため，1 台のレプリカにハッシュ値を要求する．転送元レプリカから状態が転送し，BFT-SMR の場合はハッシュ値の比較も成功した場合は，転送先レプリカは自身に状態を適用し，状態転送を完了する．転送元レプリカから状態が転送されなかった場合や BFT-SMR の場合はハッシュ値の比較に失敗した場合は，別のレプリカに対して状態の転送を要求し，同一の処理を繰り返す．

図 1 において，状態転送処理を行っているレプリカは状態転送先のレプリカおよび状態転送元の単一のレプリカのみである．このため，状態転送を行っている最中に SMR システムにリクエストが送られた場合，状態転送を行っていないレプリカは通常の処理能力を維持しているが，状態転送元のレプリカは状態転送処理に CPU の処理能力と通信帯域を圧迫されているため，SMR システム全体のリクエスト処理のスループットは低下する[2]．なお，CFT-SMR であれば 5 個，BFT-SMR であれば 7 個以上のレプリカが存在する場合，SMR システムは 2 個のレプリカの故障を許容できるため，状態転送先のレプリカおよび状態転送元のレプリカの応答を待つ必要がなく，SMR システム全体のリクエスト処理のスループットは低下しない．

Bessani らによって，状態を等分割し分割された状態（部分状態）を複数レプリカで分担して同時に転送することで，状態転送時間を短縮する手法が提案されている[2]．Bessani らの手法を 4 個のレプリカが存在する広域 SMR にて行った場合を図 2 に示す．図 2 では，三等分に分割された部分状態が赤，緑，青の三色の円柱によって表されている．各レプリカは図 1 のように単一の状態を持っているが，状態転送する際に三等分に分割し，各レプリカが異なる部分状態を同時に転送先レプリカに転送する．BFT-SMR として動

作している場合は，各レプリカは部分状態に加えて，状態全体のハッシュを同時に送信する．転送先レプリカが全ての部分状態を受信し，BFT-SMR の場合はハッシュ値の比較も成功した場合は，転送先レプリカは自身に状態を適用し，状態転送を完了する．どれかの部分状態の受信に失敗した場合や BFT-SMR の場合はハッシュ値の比較に失敗した場合は，Schneider の手法に切り替える．

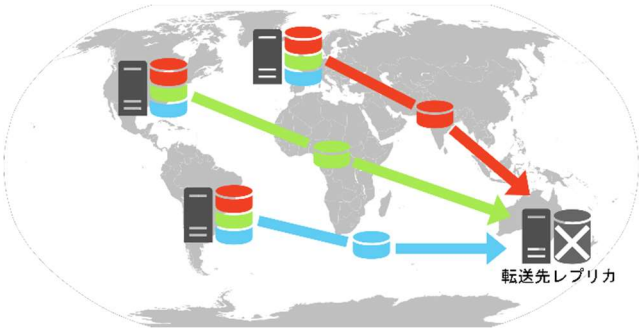


図 2 Bessani らの手法を広域 SMR に適用

3. 提案手法

3.1 提案手法のアルゴリズム

本手法では，状態をN個のチャンクに分割する．以降では，全チャンクの集合をC，レプリカtが状態転送を要求するときにレプリカiが送信するチャンクの集合をC_i^tと表記する．また，レプリカiとレプリカjの間の通信帯域をw_{ij}，レプリカtと他レプリカ間の通信帯域の総和を$\mathrm{w}_{\mathrm{all}}^{\mathrm{t}} = \sum_{\mathrm{i}} \mathrm{w}_{\mathrm{ti}}$と表記する．これらの通信帯域は，各レプリカ間の通信帯域が既知である場合はその値を用いる．未知である場合，各レプリカがレプリケーション中に行う通信から計測する．

提案手法の擬似コードを，図 3 に示す．転送先レプリカtは，状態転送が必要になると各転送元レプリカiのチャンク集合C_i^tを次の条件を満たすように計算する．

$$|C_i^t| = N \cdot w_{ti} / w_{all}^t,$$
$$\cup_i C_i^t = C,$$
$$\forall_j, C_i^t \cup C_j^t = \emptyset$$

ここで，$\forall_j, C_i^i = \emptyset$である．その後，各転送元レプリカに計算したチャンク集合を送信し，部分状態の転送を要求する．要求した全ての部分状態が転送された後，提案手法はN個のチャンクを結合し，得られた状態を転送先レプリカ tに適用する．

転送元レプリカの故障などによって部分状態の受信に失敗すると，提案手法は Schneider の手法に切り替える（図 3，6 行目）．これにより，レプリカ故障に耐性を備える．

BFT-SMR の場合はf台のレプリカがビザンチン故障による偽の応答を返す可能性があるため，3 行目にて状態全体のハッシュ値を同時に要求し，8 行目にてチャンクの結合後，ハッシュ値の検証を行うことで転送元レプリカが故障していないことを確かめる．なお，ここでハッシュ値の検証に失敗した場合も，部分状態の受信失敗と見なして Schneider の手法に切り替える．

1. **For** $i = 1$ to n do
2. **If** $i = t$, continue
3. $|C_i^t| = N \cdot w_{ti} / w_{all}^t$，$\cup_i C_i^t = C$，$\forall_j, C_i^t \cup C_j^t = \emptyset$を満たす C_i^t を i に要求
4. **End For**
5. $\forall_i, C_i^t$の転送が終了する，あるいはタイムアウトまで待つ
6. **If** タイムアウト，提案手法は失敗したため return し，Schneider の手法を行う
7. $C \leftarrow \cup_i C_i^t$
8. Cを結合し，転送先レプリカに適用する．

図 3 提案手法の擬似コード

提案手法を 4 個のレプリカが存在する広域 SMR にて行った場合の例を図 4 に示す．図 4 の各レプリカは中央上のレプリカから反時計回り順にレプリカ 1 から 4 という連番を持ち，各レプリカの右側にある複数の立方体はチャンクとして分割したレプリカの状態，矢印に挟まれた立方体は転送されているチャンクをそれぞれ表している．図 4 の各レプリカの状態分割数は$N = 27$，レプリカ間の通信帯域は$w_{14} = 18$ Mbps, $w_{24} = 6$ Mbps, $w_{34} = 3$ Mbpsである．また，転送先レプリカはレプリカ 4 である．図 4 では，レプリカ 4 が通信帯域から要求チャンク数を$|C_1^4| = 18$，$|C_2^4| = 6$，$|C_3^4| = 3$とした要求を各レプリカに送信し，状態の転送を待っている状態であり，各レプリカからチャンクが送信されている．

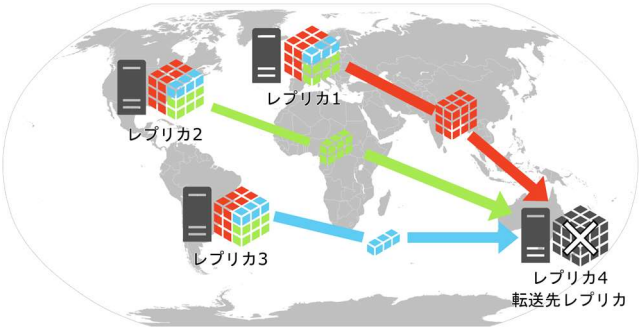


図 4 広域 SMR に対する提案手法の適用例

3.2 提案手法の理論的な状態転送時間

提案手法と既存手法の理論的な状態転送時間を比較すると，提案手法では，状態サイズ$Size$をNで割り切れる場合に転送効率が最も高くなり，この場合の転送時間$TransferTime$は$TransferTime = Size / w_{all}^t$となる．Schneider の手法では，転送先レプリカtに対して，他のレプリカの中で最大の通信帯域w_{max}を持つレプリカi_{max}からのみ状態を転送するため，$TransferTime = Size / w_{max}$となる．Bessani らの手法では，状態を総レプリカ数n個に等分割し，転送先レプリカtへ他のレプリカが同時に同一サイズの分割した状態を転送する．よって，転送先レプリカtに他のレプリカとの最小の通信帯域w_{min}を持つレプリカi_{min}

から等分割した状態の一部を転送する通信が最後に終了する．このため，$TransferTime = (Size/n)/w_{min}$となる．

提案手法と Schneider の手法を比較すると，$w_{all}^{t} \leq w_{max}$であるため，$Size/w_{all}^{t} \leq Size/w_{max}$であり，提案手法の転送時間は Schneider の手法よりも短くなる．また，提案手法と Bessani らの手法を比較すると，$w_{all}^{t} \leq n \cdot w_{min}$であるため，$Size/w_{all}^{t} \leq (Size/n)/w_{min}$であり，提案手法の転送時間は Bessani らの手法よりも短くなる．

4. 評価実験

提案手法の性能を評価するため， Amazon EC2 上にオープンソースの SMR ライブラリ BFT-SMaRt 1.2[13]を用いて広域 BFT-SMR システムを構築し，状態転送時間を測定した．

4.1 実験方法

状態転送手法として，提案手法とともに，Schneider の手法[1]，Bessani らの手法[2]を実装した．Schneider の手法では転送元レプリカの決定方法を規定していないため，事前計測したレプリカ間の通信帯域が最も広いレプリカを転送元レプリカとした．Bessani らの手法は，提案された手法のうち，状態転送に関係する“Optimizing CST”のみを実装した．

実験ではレプリカの配置として 2 つのグループを用いる．グループ A では，レプリカを Sydney, São Paulo, North Virginia, Ireland の 4 リージョンに配置し，グループ B では，Frankfurt, Ireland, London, Paris の 4 リージョンに配置する．グループ A は 4 レプリカが 4 大陸のリージョンで構成するのに対し，グループ B はヨーロッパ地域のみのリージョンで構成することで，大陸間をまたがる広域 SMR と同一の大陸内で構成される広域 SMR での提案手法の性能を検証する．

各レプリカは，表 1 に示す性能を持つ t3.xlarge インスタンスを用いる．t3.xlarge インスタンスでは，Amazon Linux 2 上で Docker 18.09.2 を用い，openjdk:14-alpine コンテナによって BFT-SMaRt を起動した．

表 1 t3.xlarge インスタンスの能力[16]

CPU	Intel Xeon Platinum 8000 series
vCPUs	4
Memory [GiB]	16.0
Network burst bandwidth [Gbps]	5

BFT-SMaRt では，状態をログとチェックポイントという 2 種類の方法を用いて保持する．チェックポイントは，定期的に取得される現在の状態の複製であり，ログは最新のチェックポイント取得後に実行されたリクエストのリストである．本実験ではチェックポイントのサイズを固定し，これを状態サイズとした．なお，ログのサイズは事前実験において今回固定した状態サイズの 1/1000 未満であることを確認しており，結果に影響を及ぼさないと考えた．

Schneider の手法において，転送元レプリカは他のレプリカのうち一つに状態転送を要求するが，本実験では転送元レプリカが状態転送を要求するレプリカは，転送元レプリカとの通信帯域が最も広いレプリカとした．

提案手法における各レプリカに要求するチャンク数の決定および Schneider の手法における転送元レプリカの決定に用いる各レプリカ間の通信帯域は，iperf 3.1.3 を用いて使用する Amazon EC2 のリージョン間の通信帯域を 5 回測定した平均値を用いた．測定した 2 組のリージョン間通信帯域を表 2 および表 3 に示す．なお，各インスタンスの通信はインターネットを経由して行った．

表 2 グループ A のリージョン間通信帯域

受信リージョン [Mbps] ＼ 送信リージョン [Mbps]	Sydney	São Paulo	North Virginia	Ireland
Sydney		33.6	57.6	41.9
São Paulo	33.7		100	63.0
North Virginia	57.1	100		172
Ireland	41.5	64.4	165	

表 3 グループ B のリージョン間通信帯域

受信リージョン [Mbps] ＼ 送信リージョン [Mbps]	Frankfurt	Ireland	London	Paris
Frankfurt		557	809	1420
Ireland	517		1080	738
London	930	1230		1680
Paris	1490	746	1770	

4.2 提案手法と既存手法との転送時間の比較

ここでは，提案手法と既存手法の状態転送にかかる時間について，比較する．グループ A，グループ B でそれぞれ広域 SMR を構築し，グループの各レプリカがそれぞれ転送先レプリカになる場合について，状態サイズを 10 MB, 50 MB, 100 MB, 500 MB と変更して各 5 回ずつ状態転送時間を計測し，平均値を求めた．なお、提案手法では，全体のチャンク数$N = 256$とした．グループ A による広域 SMR を測定した提案手法および既存手法の平均状態転送時間を転

送先レプリカ毎に図 5 に，グループ B による広域 SMR についても同様に図 6 に示す．

グループ A（図 5）では，転送先レプリカ，状態サイズを問わず提案手法の状態転送時間が最も短くなった．また，状態サイズが大きくなるほど他の手法に対する提案手法の状態転送時間の改善率が向上した．最も改善率が向上したのは転送先レプリカが Ireland，状態サイズが 500MB の場合であり, Bessani らの手法の結果に対して 42%の向上がみられた．3.2 節より，Bessani らの手法に対する改善率は一定であるはずだが，実際には状態サイズが大きくなるほど状態転送時間の改善率が向上した．この結果の理由を明らかにするため, 4.3 節では状態転送時間の内訳を分析した．

グループ B（図 6）では，転送先レプリカが Frankfurt および Paris の場合は，すべてのケースで提案手法の状態転送時間が最も短かった．一方，Ireland が転送先レプリカとなった時は 500MB 以外，London が転送先レプリカとなった時では全ての状態サイズにおいて提案手法は Bessani らの手法より状態転送速度が長くなった．3.2 節より，提案手法の状態転送時間は Bessani らの手法と同一か下回るはずであるが, 実際には状態転送時間が長くなることがあった．この結果に対しても，理由を明らかにするために 4.3 節では状態転送時間の内訳を分析した．

4.3 状態転送の各ステップにかかる時間

前節で状態サイズが大きくなるほど状態転送時間の改善率が向上したこと，また，提案手法の状態転送時間が Bessani らの手法よりも長くなることがあったことの理由を明らかにするために，状態サイズが大きくなるほど状態転送時間の改善率が向上したグループ A の Ireland への状態転送（図 5(d)），Bessani らの手法と比べて状態転送時間が長くなったグループ B の London への状態転送（図 6(c)）の 2 パターンの状態転送について，状態転送の各ステップにかかる時間を測定した．測定では，状態サイズが 10MB および 500 MB の場合で各 5 回ずつ状態転送の各ステップにかかる時間を計測し，平均値を求めた．なお、提案手法では， 全体のチャンク数$N = 256$とした．グループ A による Ireland への状態転送の結果を図 7，グループ B による London への状態転送の結果を図 8 に示す．図 7 および図 8 では，状態転送の処理を 3 ステップに分けて処理時間を示している．3 ステップは，転送先レプリカからの状態転送要求の送信（req 送信），各転送元レプリカからの状態チャンクの転送（受信(1/3)から受信(3/3)），受信した状態の結合とハッシュ値による状態の検証（ハッシュ計算）となっており，全体という棒グラフは状態転送全体の時間を示している．また，各転送元レプリカからの状態チャンクの転送は，受信完了が早い順に 1/3, 2/3, 3/3 という番号を付けている．

図 7, 8 より，状態転送時間の大部分を占めるのは状態の受信であり，リクエストの送信やハッシュ計算に費やす時間は小さいことがわかる．

図 7(a)と図 7(b)を比較すると，提案手法において状態サイズ 10MB では，受信(1/3)と受信(3/3)では転送時間の差が 2.14 倍あるのに対して，状態サイズ 500MB では受信(1/3)と受信(3/3)との時間差が 1.17 倍に小さくなっている．このため，状態サイズが大きいほど提案手法の効果が高くなっている．状態サイズが大きいほど状態チャンクの受信にかかる時間の差が小さくなる理由については, 4.4 節で実際の通信帯域を求めることで検証する．

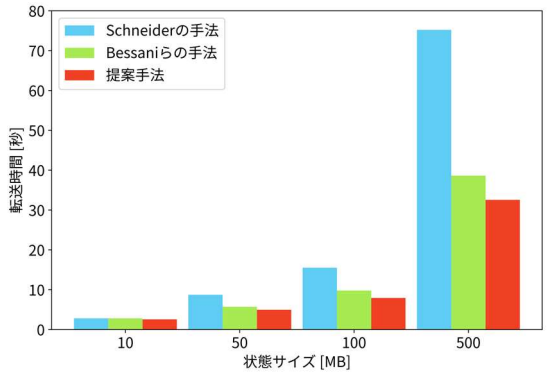
図 8(a)と図 8(b)より，グループ B による London への状態転送では，状態サイズ 500MB の場合で，受信(1/3)と受信(3/3)との転送時間の差が 1.70 倍あり，Bessani らの手法の受信(1/3)と受信(3/3)との転送時間の差である 1.51 倍に比べて大きくなっている．提案手法は各レプリカからの転送時間の差を Bessani らの手法と比べて小さくなるように状態チャンクを送信するため，正常な動作の結果 Bessani らの手法と比べて転送時間の差が大きくなることは考えにくいため，グループ B による London への状態転送では，提案手法の通信帯域の予測に何らかの問題があり，本来の通信帯域とは異なる比でチャンクを割り振ってしまっていると予想する．この予想に関しても, 4.4 節で実際の通信帯域を求めることで検証する．

4.4 状態転送時間から算出した通信帯域

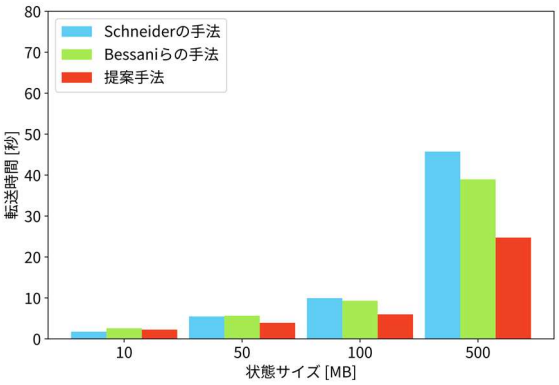
前節で，グループ A の Ireland への状態転送において提案手法によって各レプリカから状態のチャンクを転送する時間の差が，状態サイズが大きいほど小さくなった理由および，グループ B による London への状態転送において提案手法によって各レプリカから状態のチャンクを転送する時間の差が, Bessani らの手法によって各レプリカから等分割した状態を転送する時間の差よりも大きくなった理由を明らかにするために，グループ A での Ireland への状態転送およびグループ B での London への状態転送の 2 パターンの状態転送について, 状態サイズ 10MB および 500MB の場合でチャンクの個数と状態転送時間から実際の通信帯域を算出し，各 5 回ずつの平均値を求めた．算出した通信帯域を表 4 から表 7 に示す．

表 4 と表 5 を比較すると, 表 4 では表 2 に示した iperf による計測値の実際の通信帯域からの誤差率が最小 140%, 最大 509%となっており，表 2 の計測値が実際の通信帯域から大きく外れている．これに対し，表 5 では表 2 の計測値からの誤差率が最小-0.240%, 最大 19.6%と小さくなっている．

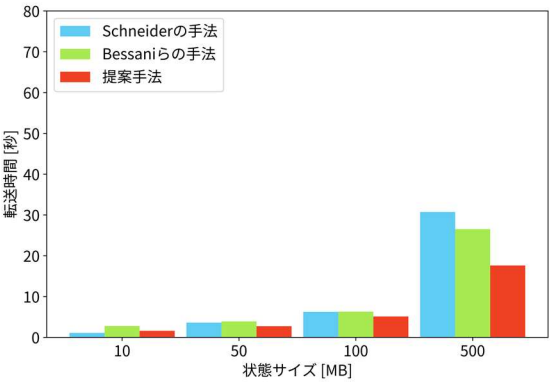
表 6 と表 7 では，表 6 では表 3 の計測値の実際の通信帯域からの誤差率が最小 1210%，最大 1280%であり，表 7 でも表 3 の計測値の実際の通信帯域からの誤差率が最小 243%，最大 359%であるため，状態サイズ 500MB の方が誤差率は小さくなってはいるものの，どちらも表 3 の計測値が実際の通信帯域から大きく外れていることがわかる．



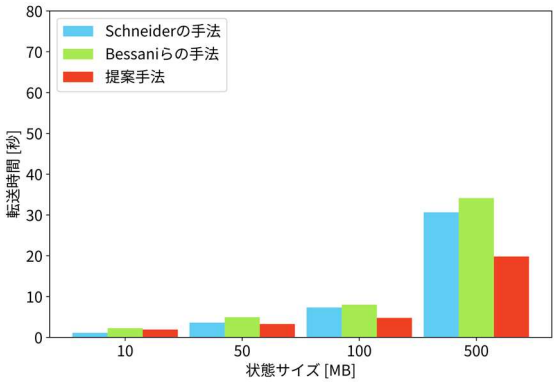
(a) 転送先：Sydney



(b) 転送先：São Paulo

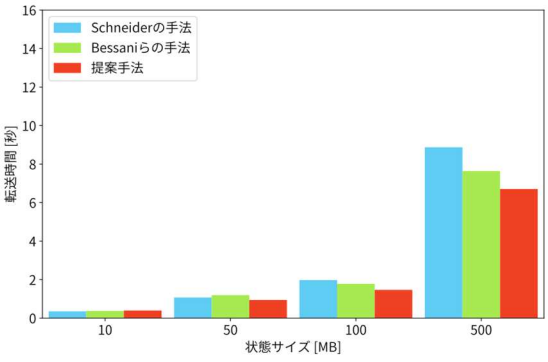


(c) 転送先：North Virginia

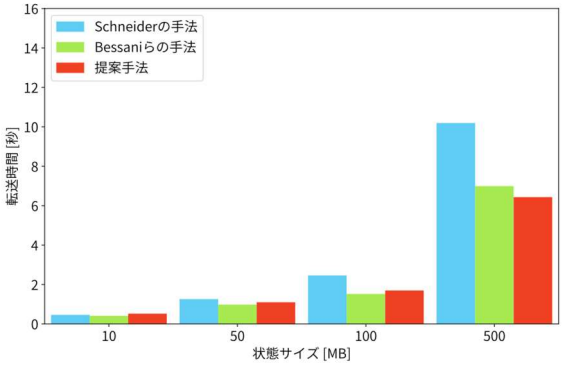


(d) 転送先：Ireland

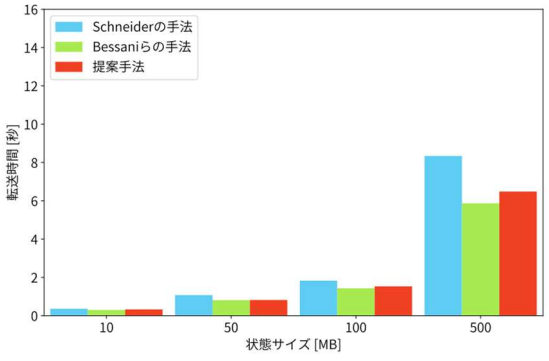
図 5 状態サイズと転送時間の関係（グループ A）



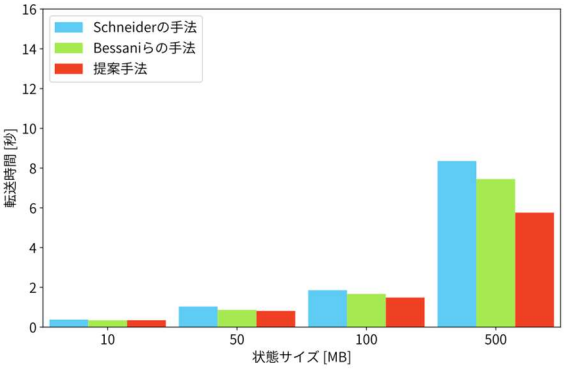
(a) 転送先：Frankfurt



(b) 転送先：Ireland

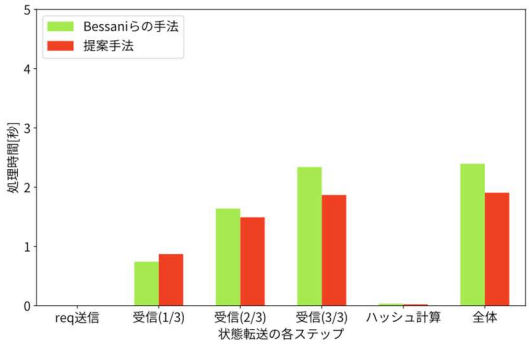


(c) 転送先：London

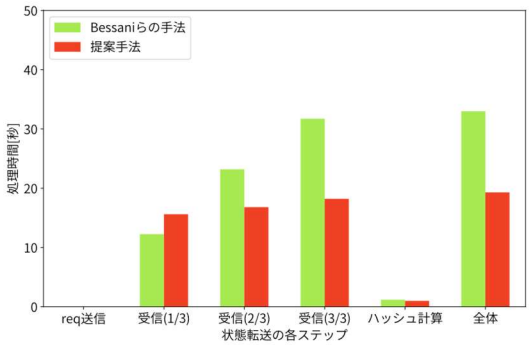


(d) 転送先：Paris

図 6 状態サイズと転送時間の関係（グループ B）

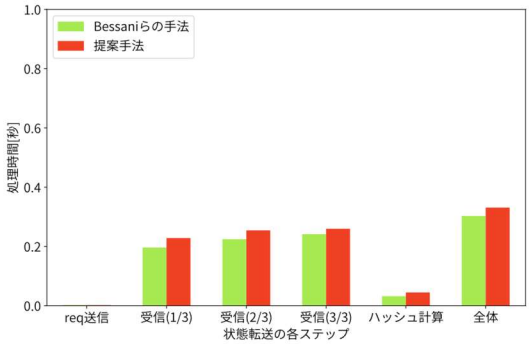


(a) 10MB

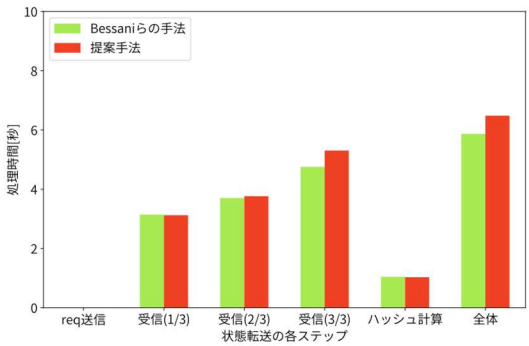


(b) 500MB

図 7 各ステップにかかる時間（転送先：Ireland）



(a) 10MB



(b) 500MB

図 8 各ステップにかかる時間（転送先：London）

これらの結果より，グループ A の Ireland への状態転送において提案手法によって各レプリカから状態のチャンクを転送する時間の差が，状態サイズが大きいほど小さくなった理由，グループ B による London への状態転送において提案手法によって各レプリカから状態のチャンクを転送する時間の差が，Bessani らの手法によって各レプリカから等分割した状態を転送する時間の差よりも大きくなった理由のどちらも，実際の通信帯域に対して iperf による計測値が大きく外れている場合が原因であることがわかった．誤差率が大きい表 4, 6, 7 において実際の通信速度はすべて iperf による計測値よりも低下しているため，何らかの原因で速度低下が起こっていると予想している．しかし具体的な原因についてはわかっておらず，詳細な調査は今後の課題である．

表 4 状態転送時間から算出した通信帯域
（10MB，転送先：Ireland）

転送元レプリカ	通信帯域[Mbps]	表 2 の誤差率[%]
Sydney	6.81	509
São Paulo	13.1	140
North Virginia	57.4	343

表 5 状態転送時間から算出した通信帯域
（500MB，転送先：Ireland）

転送元レプリカ	通信帯域[Mbps]	表 2 の誤差率[%]
Sydney	41.6	-0.240
São Paulo	58	11.0
North Virginia	138	19.6

表 6 状態転送時間から算出した通信帯域
（10MB，転送先：London）

転送元レプリカ	通信帯域[Mbps]	表 3 の誤差率[%]
Frankfurt	71.2	1210
Ireland	91.8	1240
Paris	122	1280

表 7 状態転送時間から算出した通信帯域
（500MB，転送先：London）

転送元レプリカ	通信帯域[Mbps]	表 3 の誤差率[%]
Frankfurt	271	243
Ireland	312	294
Paris	366	359

4.5 提案手法の全体のチャンク数ごとの転送時間

提案手法では，状態全体をいくつのチャンクに分割するかを全体のチャンク数Nとして固定し，通信帯域の比率に合わせてレプリカが転送するチャンクの個数をそれぞれ割り振っている．ここで，Nが大きくなるほど通信帯域の比率とレプリカが転送するチャンクの個数との誤差が小さくなるので，転送速度が理論値に近づくと予想する．ただ，チャンクの分割数が増えることで，チャンクの分割や結合にかかる時間も増えると予想するため，Nは無際限に大きくすれば効率が良くなるのではなく，最適な個数が存在すると予想される．これを確かめるために，全体のチャンク数ごとの転送時間を測定した．測定方法は，グループ A について，状態サイズ 500MB，$N = 16, 256, 65536$とした場合の状態転送時間を各レプリカで 5 回ずつ計測し，平均値を求めた．測定した平均状態転送時間を図 9 に示す．

図 9 では，提案手法においてチャンク数を変更した際，転送時間が変化すること，今回比較した 3 種類のチャンク数において，Sydney, São Paulo, Ireland の 3 リージョンでは$N = 256$, North Virginia リージョンでは$N = 65536$の場合が最も転送時間が短いことがわかった． また，North Virginia リージョンを除く 3 リージョンでは$N = 65536$の場合の転送時間が$N = 256$の場合より長くなることがわかった．この結果より，Nには最適な個数が存在することが確かめられた．また，最適なNの個数は受信レプリカによって異なるが，この実験では$N = 256$が最適であることが多かった．

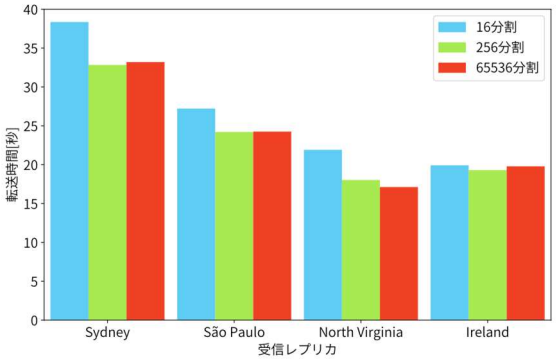


図 9 提案手法のチャンク数ごとの転送時間

5. まとめ

本論文では，広域 SMR に適した状態転送手法として，各レプリカ間の通信帯域に合わせて状態の分割サイズを調整して状態を転送することで，状態転送時間を短縮する手法を提案した．パブリッククラウドを用いた評価実験により，提案手法は状態サイズが大きく，全体のスループットが低速であるほど効果的であること，またチャンクの分割数は 256 分割が良いと思われることを確認した．

今後の課題として，レプリカ間の通信レイテンシが状態転送に与える影響の検証，状態サイズが小さい場合および通信帯域が広い場合に，実際の通信帯域が計測値と乖離する問題の検証および対処がある．

謝辞

本研究は JSPS 科研費 18K18029 の助成を受けて実施された．

参考文献

[1] F. B. Schneider, “Implementing Fault-Tolerant Services Using the State Machine Approach: A Tutorial,” *ACM Comput. Surv.*, vol. 22, no. 4, pp. 299–319, 1990.
[2] A. Bessani, M. Santos, N. Neves, and M. Correia, “On the Efficiency of Durable State Machine Replication,” *Usenix’13*, pp. 169–180, 2013.
[3] Azure network round-trip latency statistics, https://docs.microsoft.com/en-us/azure/networking/azure-network-latency, (参照 2019-12-23)
[4] X. Défago, A. Schiper, and P. Urbán, “Total order broadcast and multicast algorithms,” *ACM Comput. Surv.*, vol. 36, no. 4, pp. 372–421, Dec. 2004.
[5] T. D. Chandra and S. Toueg, “Unreliable failure detectors for reliable distributed systems,” *J. ACM*, vol. 43, no. 2, pp. 225–267, Mar. 1996.
[6] M. J. Fischer, N. A. Lynch, and M. S. Paterson, “Impossibility of distributed consensus with one faulty process,” *J. ACM*, vol. 32, no. 2, pp. 374–382, Apr. 1985.
[7] L. Lamport, “The part-time parliament,” *ACM Trans. Comput. Syst.*, vol. 16, no. 2, pp. 133–169, May 1998.
[8] D. Ongaro and J. Ousterhout, “In Search of an Understandable Consensus Algorithm,” *Usenix’13*, 2014.
[9] F. P. Junqueira, B. C. Reed, and M. Serafini, “Zab: High-performance broadcast for primary-backup systems,” *Proc. Int. Conf. Dependable Syst. Networks*, pp. 245–256, 2011.
[10] L. Lamport, R. Shostak, and M. Pease, “The byzantine generals problem,” *Dr. Dobb’s J.*, vol. 33, no. 4, pp. 30–36, 2008.
[11] M. Castro and B. Liskov, “Practical Byzantine Fault Tolerance,” *Proc. Symp. Oper. Syst. Des. Implement.*, no. February, pp. 1–14, 1999.
[12] Ramakrishna Kotla, Lorenzo Alvisi, Mike Dahlin, Allen Clement, and Edmund Wong, “Zyzzyva: Speculative Byzantine fault tolerance,” *ACM Trans. Comput. Syst.* 27, 4, Article 7 (December 2009), 39 pages.
[13] J. Sousa and A. Bessani, “From Byzantine consensus to BFT state machine replication: A latency-optimal transformation,” *Proc. - 9th Eur. Dependable Comput. Conf. EDCC 2012*, pp. 37–48, 2012.
[14] Y. Mao, F. P. Junqueira, and K. Marzullo, “Mencius: Building Efficient Replicated State Machines for WANs,” *Proc. Symp. Oper. Syst. Des. Implement.*, pp. 369–384, 2008.
[15] H. Zhao, Q. Zhang, Z. Yang, M. Wu, and Y. Dai, “SDPaxos: Building efficient semi-decentralized geo-replicated state machines,” *SoCC 2018 - Proc. 2018 ACM Symp. Cloud Comput.*, pp. 68–81, 2018.
[16] Amazon EC2 T3 Instances – Amazon Web Services (AWS), https://aws.amazon.com/ec2/instance-types/t3/, （参照 2019-12-23）