DEIM Forum 2021 J24-4

エッジコンピューティング環境における低遅延かつ高可用な耐障害性保証

高尾　大樹† 　杉浦　健人† 　石川　佳治†

† 名古屋大学大学院情報学研究科　〒 464–8603 愛知県名古屋市千種区不老町
Email: {takao,sugiura}@db.is.i.nagoya-u.ac.jp, ishikawa@i.nagoya-u.ac.jp

あらまし　エッジコンピューティング環境を想定したデータストリーム処理においても耐障害性保証は重要である．しかし，待機系への処理移譲及び損失データの再処理に基づく従来の耐障害性保証にはエッジ単位での物理的な多重化が必要となり，システム規模が大きくなるほど各エッジに求められる多重度も増加するという課題がある．そこで本稿では，環境モニタリングのアプリケーションを想定し，エッジ単位での物理的多重化を必要としない低遅延かつ高可用な耐障害性保証アプローチを提案する．本アプローチではデバイス間に存在する相関関係に着目し，障害エッジの出力を他のエッジの出力から近似的に復元することで，障害エッジの復旧を待たずに処理継続を可能とする．また本稿では，各エッジでの処理結果に含まれる曖昧性を考慮した障害エッジの出力推定手法，及び推定精度を考慮したキーパーティショニングアルゴリズムについても提案する．

キーワード　データストリーム処理，エッジコンピューティング，耐障害性，高可用性，近似処理

1　背　　景

分散データストリーム処理システムにおいて耐障害性保証は重要な要件である．Flink [1] をはじめとする多くのシステムでは，複数タスクを介したパイプライン処理によって，動的に生成されたデータをリアルタイムに処理する．すなわち，各タスクでは前段のタスクから入力データを受け取るたび，その処理結果を後段のタスクへと配送する．これを多段接続された複数タスクで繰り返すことによって，システム全体として 1 つの問合せ処理を実現する．しかし，この処理形態ではあるタスクからの出力が途絶えると以降全てのタスクでの処理が停滞してしまうため，環境モニタリングをはじめとするリアルタイムな出力が常に求められるアプリケーションでは，耐障害性保証が重要になる．特に，分散システムにおいて各タスクを複数のサーバへ分散したとき，全てのサーバは単一障害点となってしまうため，各サーバに障害が生じた場合にも低遅延かつ高可用なデータ処理を提供するための仕組みが必要である．

既存システムでは一般的に待機系への処理移譲，及び損失データの再処理によって頑健な耐障害性を保証する [1–4]．各ノードの処理状況を監視し，障害ノードでの処理内容を待機ノードが引き継ぐことで，システム全体として処理を継続する．また，障害によって損失したデータを再処理することで誤差のない障害復帰も可能である．クラスタ環境に注目した場合，豊富なリソースが物理的に集中しており，またシステム前段のメッセージングキューに入力データの管理を一任できるため，この耐障害性保証アプローチは理に適っているといえる．

しかし，エッジコンピューティング環境においては，異なる耐障害性保証のアプローチが必要である．エッジコンピューティングとは，図 1 のように集約処理やフィルタリングなどの簡単なデータ処理をネットワークエッジで施すことで，データ通信量の削減や処理の負荷分散が可能な処理形態である [5]．例と

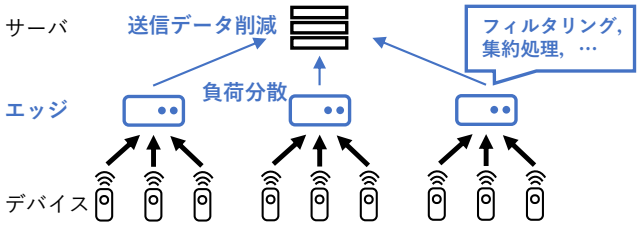

図 1: エッジコンピューティングの概要図

してスマートシティでの環境センシングを考える．各区画や建物内に広く設置された環境センサデバイスは計測データを無線通信によってそれぞれ近くのエッジへと送信する．各エッジは一定時間ごとに計測データを集約することでクラウドへ転送するデータ量を削減する．しかし，ある地点のエッジが故障したとき，距離の離れた別のエッジが代わりに計測データを受信できないため，処理を肩代わりできず入力データを完全に喪失してしまう．すなわち，待機系への処理移譲及び損失データ再処理のためにはエッジ単位での多重化が必要となるが，代替ノードをデータセンターなどで一括管理できるクラウドとは異なり，そのコストは無視できない．

そこで，本稿では環境センシングのアプリケーションを想定して，エッジ単位での物理的多重化を必要としない耐障害性保証アプローチを提案する．環境センシングなどでは計測データ間に相関が見られることから，センサネットワークの時代からセンサ間の相関を利用した効率化が提案されてきた [6]．また，著者らはこれまでデバイス間の相関関係を活用することで，デバイス故障や通信障害により入力が欠損する場合にも，誤差量を理論的に保証可能な近似集約処理手法を開発してきた [7]．本稿ではこの手法を拡張することで，障害エッジの復旧を待たずにシステム全体として処理継続を可能とする手法を提案する．

デバイス故障や通信障害などにより全ての入力データが揃わないとき，各エッジではデバイス間に存在する相関関係の情報

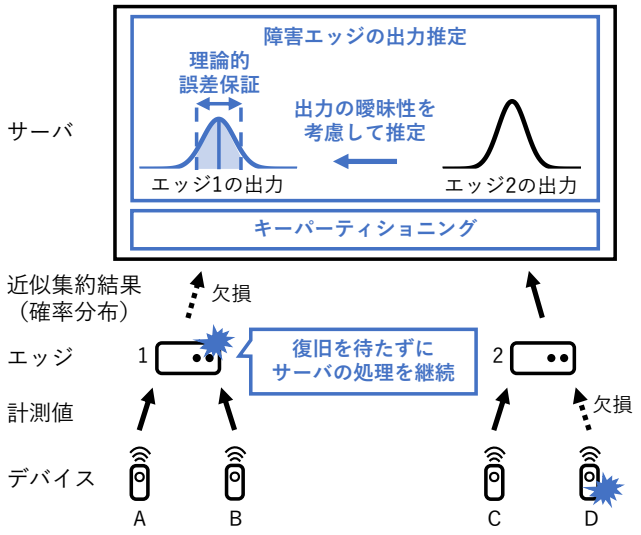

図 2: 提案アプローチの概要図

をもとに，得られたデータのみから各デバイスの集約結果を近似的に求める状況を想定する．例えば図 2 においてデバイス D が故障すると，エッジ 2 ではデバイス C の計測値からデバイス D の集約結果を近似的に求める．このように集約結果を近似的に求めることで，各エッジは障害デバイスの復旧を待たずに処理を継続できる．

本アプローチではこの考えをエッジ障害の問題にも拡張し，障害エッジの出力を他のエッジの出力から同様に推定することで，システム全体としての処理継続を目指す．すなわち，図 2 において障害が発生したエッジ 1 から得られるはずであったデバイス A，及び B の集約結果を，エッジ 2 から得られるデバイス C，及び D の集約結果から推定することで，エッジ 1 の復旧を待たずに以降のサーバでの処理を継続する．

しかし，本アプローチでは先行研究と異なり，各エッジでの処理結果に含まれる曖昧性を適切に考慮して障害エッジの出力を推定する必要がある．単に各エッジで近似的に求めた集約値から障害エッジの出力を推定する場合，集約値自体には曖昧性の情報が含まれないため，各エッジでの近似処理による曖昧性を過小評価することになり誤差保証が破綻するなど信頼性の問題がある．そこで本稿では，各エッジの近似処理による曖昧性を確率分布の形で表現し，この確率分布の情報を最大限活用して障害エッジの出力を推定する手法を提案する．

また本稿では，推定精度を考慮したキーパーティショニングについても言及する．多数のエッジからの出力に対応するため，サーバでは複数スレッドによる並列処理が必要になる．各スレッドでは割り当てられたデータのみを用いて障害エッジの出力を推定するため，このときのキーパーティショニングが障害エッジの出力推定精度に大きな影響を与えると考えられる．

本稿の構成は以下のとおりである．2 章では，提案アプローチの基礎となるエッジ環境での障害を考慮した近似的集約処理手法について述べる．3 章では提案アプローチによる障害復帰の流れ，及び処理結果の曖昧性を考慮した障害エッジの出力推定手法についてそれぞれ説明する．また，4 章では推定精度を考慮したキーパーティショニングについて検討する．5 章では評価実験について述べ，6 章ではストリーム処理システムにおける耐障害性保証に関する先行研究について概説する．最後に，本研究のまとめと今後の課題を 7 章で述べる．

2 準　　備

提案アプローチでは，各エッジから近似的な集約結果がサーバへ送信される状況を想定し，障害エッジの出力を他のエッジの出力から推定することで，低遅延かつ高可用なデータ処理を実現する．そこで本章では議論の準備として，各エッジでの障害を考慮した近似的集約処理手法について述べる．

全 n 台のデバイス $\boldsymbol{X} = \{X_1, X_2, \ldots, X_n\}$ はそれぞれが近い位置に存在するエッジに対して定期的かつ同期的に計測値を送信する．ここで，各時刻 $t \in \mathbb{N}^+$ におけるデバイスの真値を $\boldsymbol{x}^t = \langle x_1^t, x_2^t, \ldots, x_n^t \rangle$ と表す．しかし，デバイス故障や通信障害などにより各エッジは必ず全デバイスのデータを受け取るとは限らない．つまり，あるエッジ i に割り当てられたデバイス集合 $\boldsymbol{X}_i \subseteq \boldsymbol{X}$ に対して，各時刻 t ではその一部のデバイス $\boldsymbol{O}_i^t \subseteq \boldsymbol{X}_i$ の観測値 $\boldsymbol{o}_i^t$ しか得られない．ただし，各デバイスはいずれか 1 つのエッジに割り当てられ，また各エッジで観測された値は常に真値と等しいものとする．

各エッジでは得られた観測値系列を窓幅 w の時間窓により分割し，時間窓ごとに各デバイスの集約値を計算する．すなわち，各時間窓の始点 t' から窓幅 w の観測値系列 $\boldsymbol{o}_i^{[t',t'+w)} = \langle \boldsymbol{o}_i^{t'}, \boldsymbol{o}_i^{t'+1}, \ldots, \boldsymbol{o}_i^{t'+w-1} \rangle$ から，各デバイス $X \in \boldsymbol{X}_i$ の集約値 $\boldsymbol{Y}_X$ を求める．ただし，集約問合せとして平均値及び合計値問合せを想定する．このとき，本来欲しいのは $\boldsymbol{x}_i^{[t',t'+w)}$ に対する集約結果なので，観測値系列 $\boldsymbol{o}_i^{[t',t'+w)}$ を単に集約するだけでは欠損値が原因で誤差が生じ，その誤差量に対する理論的な保証もない．

そこで，デバイス間に存在する相関関係に着目し，各エッジでは各時刻の観測値から欠損値を推定する．デバイス $\boldsymbol{X}$ 間の相関関係を**多変量正規分布** $\mathcal{N}(\boldsymbol{x} \mid \boldsymbol{\mu}, \Sigma)$ を用いて表す．なお，$\boldsymbol{\mu}$ 及び Σ はそれぞれ $\boldsymbol{X}$ の平均ベクトルと分散共分散行列を表す．時刻 t において，デバイス $\boldsymbol{O}_i^t$ から得られた観測値 $\boldsymbol{o}_i^t$ に対して，デバイス $\boldsymbol{X}_i^t = \boldsymbol{X}_i \setminus \boldsymbol{O}_i^t$ の事後確率を表す正規分布 $\mathcal{N}(\boldsymbol{x} \mid \boldsymbol{\mu}_{\boldsymbol{X}_i^t|\boldsymbol{o}_i^t}, \Sigma_{\boldsymbol{X}_i^t|\boldsymbol{o}_i^t})$ は以下の式から求められる [8].

$$\boldsymbol{\mu}_{\boldsymbol{X}_i^t|\boldsymbol{o}_i^t} = \boldsymbol{\mu}_{\boldsymbol{X}_i^t} + \Sigma_{\boldsymbol{X}_i^t\boldsymbol{O}_i^t}\Sigma_{\boldsymbol{O}_i^t\boldsymbol{O}_i^t}^{-1}\left(\boldsymbol{o}_i^t - \boldsymbol{\mu}_{\boldsymbol{O}_i^t}\right) \quad (1)$$

$$\Sigma_{\boldsymbol{X}_i^t|\boldsymbol{o}_i^t} = \Sigma_{\boldsymbol{X}_i^t\boldsymbol{X}_i^t} - \Sigma_{\boldsymbol{X}_i^t\boldsymbol{O}_i^t}\Sigma_{\boldsymbol{O}_i^t\boldsymbol{O}_i^t}^{-1}\Sigma_{\boldsymbol{O}_i^t\boldsymbol{X}_i^t} \quad (2)$$

ここで，各記号の添字はベクトルないし行列から添字に対応する次元を抽出したことを示す．例えば，$\boldsymbol{\mu}_{\boldsymbol{X}_i^t}$ は $\boldsymbol{X}_i^t$ に対応する次元の値を平均ベクトル $\boldsymbol{\mu}$ から抽出したもの（つまり $\boldsymbol{X}_i^t$ の平均ベクトル）であり，$\Sigma_{\boldsymbol{X}_i^t\boldsymbol{O}_i^t}$ は分散共分散行列 Σ の $\boldsymbol{X}_i^t$ に対応する行から $\boldsymbol{O}_i^t$ に対応する列を抽出したもの（つまり $\boldsymbol{X}_i^t$ と $\boldsymbol{O}_i^t$ の共分散を表すブロック行列）である．

そして，各時刻の計測データに対する正規分布を集約することで，最終的な処理結果を得る，以下では平均値問合せを例に説明するが，合計値問合せに対しても同様の議論が可能であ

る．正規分布は再生性を持つため，デバイス $X \in \boldsymbol{X}_i$ の平均値 $Y_X = (\sum_{t \in [t', t'+w)} X^t)/w$ もまた正規分布に従う．特に時間的な相関を無視するとき，$\boldsymbol{X}_i$ の集約結果 $\boldsymbol{Y}_{\boldsymbol{X}_i}$ に対する正規分布 $\mathcal{N}(\boldsymbol{y} \mid \boldsymbol{\mu}_{\boldsymbol{Y}_{\boldsymbol{X}_i}}, \Sigma_{\boldsymbol{Y}_{\boldsymbol{X}_i}})$ は，各時刻の平均ベクトル $\boldsymbol{\mu}_{\boldsymbol{X}_i|\boldsymbol{o}_i^t}$ 及び分散共分散行列 $\Sigma_{\boldsymbol{X}_i|\boldsymbol{o}_i^t}$ からそれぞれ次のとおり求められる [7].

$$\boldsymbol{\mu}_{\boldsymbol{Y}_{\boldsymbol{X}_i}} = \frac{1}{w}\left(\sum_{t \in [t', t'+w)} \boldsymbol{\mu}_{\boldsymbol{X}_i|\boldsymbol{o}_i^t}\right) \tag{3}$$

$$\Sigma_{\boldsymbol{Y}_{\boldsymbol{X}_i}} = \frac{1}{w^2}\left(\sum_{t \in [t', t'+w)} \Sigma_{\boldsymbol{X}_i|\boldsymbol{o}_i^t}\right) \tag{4}$$

ただし，各時刻の平均ベクトル $\boldsymbol{\mu}_{\boldsymbol{X}_i|\boldsymbol{o}_i^t}$ 及び分散共分散行列 $\Sigma_{\boldsymbol{X}_i|\boldsymbol{o}_i^t}$ について，$X_j \in \boldsymbol{O}_i^t$ に対する期待値はその観測値 $o_j^t \in \boldsymbol{o}_i^t$ であり，分散及び他デバイスとの共分散はすべて 0 とする．

このとき，集約結果に対する誤差上限を理論的に保証できる．デバイス $X \in \boldsymbol{X}_i$ の平均値 Y_X が $\boldsymbol{\mu}_{Y_X}$ である確率はある誤差上限 e を用いて以下のように求められる．

$$\begin{aligned} &P\left(Y_X \in [\boldsymbol{\mu}_{Y_X} - e, \boldsymbol{\mu}_{Y_X} + e]\right) \\ &= \int_{\boldsymbol{\mu}_{Y_X}-e}^{\boldsymbol{\mu}_{Y_X}+e} \mathcal{N}\left(y \mid \boldsymbol{\mu}_{Y_X}, \Sigma_{Y_X}\right) dy \end{aligned} \tag{5}$$

すなわち，ユーザ要求として要求信頼度 δ が与えられたとき，$P(Y_X \in [\mu_{Y_X} - e', \mu_{Y_X} + e']) = \delta$ を解くことで，要求信頼度 δ を満たす最小の推定誤差 e' を計算できる．このとき，δ の確率で Y_X の真値が $\mu_{Y_X} \pm e'$ 以内に存在することを保証する．ただし，$\Sigma_{Y_X} = 0$ のとき時間窓中の全時刻において観測値が得られているため，常に $e' = 0$ であり集約結果に誤差は含まれない．

3　障害エッジの出力推定による耐障害性保証

本章では，エッジ単位での物理的多重化を必要としない低遅延かつ高可用な耐障害性保証アプローチを提案する．本手法は，障害エッジの出力を他のエッジの出力から推定することで，障害エッジの復旧を待たずにシステム全体として処理継続を可能とする．以下では，本アプローチによる障害復帰の流れ，及び処理結果の曖昧性を考慮した障害エッジの出力推定手法についてそれぞれ説明する．

3.1　本アプローチによる障害復帰の流れ

全 m 台のエッジから各デバイスの集約結果が定期的かつ同期的にサーバに対して送信される状況を考える．ただし，デバイス故障や通信障害によりデバイスの計測データに欠損が発生するため，各エッジで計算される集約結果は近似的なものであり，エッジ i の集約結果として $\mathcal{N}(\boldsymbol{y} \mid \boldsymbol{\mu}_{\boldsymbol{Y}_{\boldsymbol{X}_i}}, \Sigma_{\boldsymbol{Y}_{\boldsymbol{X}_i}})$ が得られるとする．例えば図 2 において，エッジ 2 ではデバイス C，及び D の集約結果を二変量正規分布 $\mathcal{N}(\boldsymbol{y} \mid \boldsymbol{\mu}_{\boldsymbol{Y}_{\boldsymbol{X}_2}}, \Sigma_{\boldsymbol{Y}_{\boldsymbol{X}_2}})$ として計算し，サーバへ出力する．サーバでは全てのエッジでの処理結果が揃い次第，アプリケーションに応じた処理を施すことでサービスを提供する．

エッジ単位で多重化しない場合，あるエッジに障害が発生するとその間の処理結果が失われるため，サーバでは以降の処理を続けられない．図 2 ではエッジ 1 の障害により，デバイス A, 及び B の集約結果 $\mathcal{N}(\boldsymbol{y} \mid \boldsymbol{\mu}_{\boldsymbol{Y}_{\boldsymbol{X}_1}}, \Sigma_{\boldsymbol{Y}_{\boldsymbol{X}_1}})$ がサーバに送られず，このままでは以降の処理を進められない．

そこで本アプローチでは，生存エッジの出力から障害エッジの出力を推定することで，システム全体として処理継続を目指す．障害エッジに割り当てられたデバイス集合（以下「復元対象デバイス」という）を $\boldsymbol{X}' \subseteq \boldsymbol{X}$，生存エッジに割り当てられたデバイス集合（以下「観測デバイス」という）を $\boldsymbol{O} = \boldsymbol{X} \setminus \boldsymbol{X}'$ とする．例えば図 2 の場合，復元対象デバイス $\boldsymbol{X}' = \boldsymbol{X}_1 = \{X_A, X_B\}$ であり，観測デバイス $\boldsymbol{O} = \boldsymbol{X}_2 = \{X_C, X_D\}$ である．このとき，$\boldsymbol{O}$ の各デバイスの集約結果 $\boldsymbol{Y}_{\boldsymbol{O}}$ の曖昧性を表す分布 $\mathcal{N}(\boldsymbol{y} \mid \boldsymbol{\mu}_{\boldsymbol{Y}_{\boldsymbol{O}}}, \Sigma_{\boldsymbol{Y}_{\boldsymbol{O}}})$ から，$\boldsymbol{X}$ の各デバイスの集約結果 $\boldsymbol{Y}_{\boldsymbol{X}'}$ に対する分布 $\mathcal{N}(\boldsymbol{y} \mid \boldsymbol{\mu}_{\boldsymbol{Y}_{\boldsymbol{X}'}}, \Sigma_{\boldsymbol{Y}_{\boldsymbol{X}'}})$ を推定することで，サーバでの以降の処理を継続できる．例えば，デバイス $X \in \boldsymbol{X}'$ の集約結果 Y_X が θ 以下となるものをフィルタリングしたい場合，Y_X に対する正規分布 $\mathcal{N}(y \mid \boldsymbol{\mu}_{Y_X}, \Sigma_{Y_X})$ から Y_X が θ 以下となる確率 $P(Y_X \leqq \theta)$ を求め，これが要求信頼度 δ 以上となるかを判定すればよい．

3.2　処理結果の曖昧性を考慮した障害エッジの出力推定手法

本アプローチでは，各エッジの処理結果に含まれる曖昧性を考慮した障害エッジの出力推定が重要になる．例えば，エッジ 2 の出力 $\mathcal{N}(\boldsymbol{y} \mid \boldsymbol{\mu}_{\boldsymbol{Y}_{\boldsymbol{X}_2}}, \Sigma_{\boldsymbol{Y}_{\boldsymbol{X}_2}})$ からエッジ 1 の出力 $\mathcal{N}(\boldsymbol{y} \mid \boldsymbol{\mu}_{\boldsymbol{Y}_{\boldsymbol{X}_1}}, \Sigma_{\boldsymbol{Y}_{\boldsymbol{X}_1}})$ を推定するとき，2 章で述べたとおり単に平均ベクトル $\boldsymbol{\mu}_{\boldsymbol{Y}_{\boldsymbol{X}_2}}$ に対する事後確率を用いる方法が考えられる．しかし，この方法ではエッジ 2 の出力の曖昧性を表す $\Sigma_{\boldsymbol{Y}_{\boldsymbol{X}_2}}$ の値を考慮できず推定誤差を過小評価することになり，その後のサーバでの処理結果も不適切なものになる恐れがある．そこで本節では，各エッジの出力に含まれる曖昧性を考慮した障害エッジの出力推定手法について説明する．

ある時間窓中のデバイス $\boldsymbol{X}$ の相関関係を多変量正規分布 $\mathcal{N}(\boldsymbol{x} \mid \boldsymbol{\mu}, \Sigma)$ によって表せるとき，デバイス $\boldsymbol{X}$ の集約結果 $\boldsymbol{Y}_{\boldsymbol{X}}$ にも同様に相関関係が存在する．特に 2 章で述べたような窓幅 w の時間窓に対する平均値問合せを考える場合，正規分布の再生性から，$\boldsymbol{Y}_{\boldsymbol{X}}$ 間の相関関係は次式に示す正規分布 $\mathcal{N}(\boldsymbol{y} \mid \boldsymbol{\mu}', \Sigma')$ により表現できる．

$$\boldsymbol{\mu}' = \frac{1}{w}\left(\sum_{t \in [t', t'+w)} \boldsymbol{\mu}^t\right) = \boldsymbol{\mu} \tag{6}$$

$$\Sigma' = \frac{1}{w^2}\left(\sum_{t \in [t', t'+w)} \Sigma^t\right) = \frac{1}{w}\Sigma \tag{7}$$

このとき，観測デバイス $\boldsymbol{O}$ の集約結果 $\boldsymbol{Y}_{\boldsymbol{O}}$ がある $\boldsymbol{y}$ に定まれば，次に示す事後確率分布 $\mathcal{N}(\boldsymbol{y}' \mid \boldsymbol{\mu}_{\boldsymbol{Y}_{\boldsymbol{X}'}|\boldsymbol{y}}, \Sigma_{\boldsymbol{Y}_{\boldsymbol{X}'}|\boldsymbol{y}})$ によって，復元対象デバイス $\boldsymbol{X}'$ の集約結果 $\boldsymbol{Y}_{\boldsymbol{X}'}$ の値 $\boldsymbol{y}'$ を推定できる．

$$\boldsymbol{\mu}_{\boldsymbol{Y}_{\boldsymbol{X}'}|\boldsymbol{y}} = \boldsymbol{\mu}'_{\boldsymbol{X}'} + \Sigma'_{\boldsymbol{X}'\boldsymbol{O}}\left(\Sigma'_{\boldsymbol{O}\boldsymbol{O}}\right)^{-1}\left(\boldsymbol{y} - \boldsymbol{\mu}'_{\boldsymbol{O}}\right) \tag{8}$$

$$\Sigma_{\boldsymbol{Y}_{\boldsymbol{X}'}|\boldsymbol{y}} = \Sigma'_{\boldsymbol{X}'\boldsymbol{X}'} - \Sigma'_{\boldsymbol{X}'\boldsymbol{O}}\left(\Sigma'_{\boldsymbol{O}\boldsymbol{O}}\right)^{-1}\Sigma'_{\boldsymbol{O}\boldsymbol{X}'} \tag{9}$$

ただし，各記号の添字はベクトルないし行列から添字に対応する次元を抽出したことを示す．

実際には観測デバイスの集約結果 $\boldsymbol{Y_O}$ には曖昧性があり一意に定まらないので，$\boldsymbol{Y_O}$ の曖昧性を表した分布 $\mathcal{N}(\boldsymbol{y} \mid \boldsymbol{\mu}_{\boldsymbol{Y_O}}, \Sigma_{\boldsymbol{Y_O}})$ をもとに復元対象デバイスの集約結果を $\boldsymbol{Y_{X'}}$ 推定する．$\mathcal{N}(\boldsymbol{y} \mid \boldsymbol{\mu}_{\boldsymbol{Y_O}}, \Sigma_{\boldsymbol{Y_O}})$ によって $\boldsymbol{Y_O}$ が $\boldsymbol{y}$ となる確率が分かるので，次式のとおり $\boldsymbol{Y_{X'}}$ の分布を求めればよい．

$$P(\boldsymbol{Y_{X'}}) = \int P(\boldsymbol{Y_O} = \boldsymbol{y})\, P(\boldsymbol{Y_{X'}} \mid \boldsymbol{Y_O} = \boldsymbol{y})\, d\boldsymbol{y} \tag{10}$$

$$= \int \mathcal{N}(\boldsymbol{y} \mid \boldsymbol{\mu}_{\boldsymbol{Y_O}}, \Sigma_{\boldsymbol{Y_O}})\, \mathcal{N}\left(\boldsymbol{y'} \mid \boldsymbol{\mu}_{\boldsymbol{Y_{X'}}|\boldsymbol{y}}, \Sigma_{\boldsymbol{Y_{X'}}|\boldsymbol{y}}\right) d\boldsymbol{y} \tag{11}$$

ただし，実際には複数の生存エッジが存在するため，$\boldsymbol{Y_O}$ に対する分布 $\mathcal{N}(\boldsymbol{y} \mid \boldsymbol{\mu}_{\boldsymbol{Y_O}}, \Sigma_{\boldsymbol{Y_O}})$ はそれらエッジの出力を合わせたものになる．ある生存エッジ i に割り当てられたデバイス $\boldsymbol{X}_i$ の集約結果の推定に別のエッジ j に割り当てられているデバイス $\boldsymbol{X}_j$ の観測値を用いられないため，本稿では $\Sigma_{\boldsymbol{Y_O}}$ 中の $\boldsymbol{Y}_{\boldsymbol{X}_i}$，$\boldsymbol{Y}_{\boldsymbol{X}_j}$ 間の共分散を表す要素は全て 0 として扱う．

式 (11) を計算することで，多変量正規分布 $\mathcal{N}\left(\boldsymbol{y'} \mid \boldsymbol{\mu}'_{\boldsymbol{Y}_{\boldsymbol{X'}}}, \Sigma'_{\boldsymbol{Y}_{\boldsymbol{X'}}}\right)$ が得られ，この平均ベクトル及び分散共分散行列はそれぞれ次式のとおり求められる．

$$\boldsymbol{\mu}'_{\boldsymbol{Y}_{\boldsymbol{X'}}} = \boldsymbol{\mu}'_{\boldsymbol{X'}} + \Sigma'_{\boldsymbol{X'O}} \left(\Sigma'_{\boldsymbol{OO}}\right)^{-1} \left(\boldsymbol{\mu}_{\boldsymbol{Y_O}} - \boldsymbol{\mu}'_{\boldsymbol{O}}\right) \tag{12}$$

$$\Sigma'_{\boldsymbol{Y}_{\boldsymbol{X'}}} = \Sigma'_{\boldsymbol{X'X'}} - \Sigma'_{\boldsymbol{X'O}} \left(\Sigma'_{\boldsymbol{OO}}\right)^{-1} \Sigma'_{\boldsymbol{OX'}} + \Sigma'_{\boldsymbol{X'O}} \left(\Sigma'_{\boldsymbol{OO}}\right)^{-1} \Sigma_{\boldsymbol{Y_O}} \left(\Sigma'_{\boldsymbol{OO}}\right)^{-1} \Sigma'_{\boldsymbol{OX'}} \tag{13}$$

式 (12) 及び式 (13) の第 1，2 項は $\boldsymbol{Y_O}$ の期待値 $\boldsymbol{\mu}_{\boldsymbol{Y_O}}$ に対する事後確率分布の期待値ベクトルと分散共分散行列（$\boldsymbol{y} = \boldsymbol{\mu}_{\boldsymbol{Y_O}}$ としたときの式 (8) 及び式 (9)）にそれぞれ一致する．そして，式 (13) の第 3 項が $\boldsymbol{Y_O}$ の曖昧性を考慮した分散の補正項にあたる．すなわち，単に $\boldsymbol{Y_O}$ の期待値 $\boldsymbol{\mu}_{\boldsymbol{Y_O}}$ から $\boldsymbol{Y_{X'}}$ を推定する場合，$\boldsymbol{Y_O}$ の曖昧性を考慮できずに $\Sigma'_{\boldsymbol{Y}_{\boldsymbol{X'}}}$ が小さくなるため，誤差を過小評価してしまうことを示している．対して，提案手法では $\boldsymbol{Y_O}$ の曖昧性を考慮して $\Sigma'_{\boldsymbol{Y}_{\boldsymbol{X'}}}$ を補正することで，誤差量を適切に評価できるといえる．

4　推定精度を考慮したキーパーティショニング

本章では，推定精度を向上するためのパーティショニングについて検討する．提案手法による障害エッジの出力推定には行列演算が必要であり，システム規模が大きくなるほどこの計算コストは無視できなくなる．計算コストを抑制するアプローチとして，入力データのパーティショニングが考えられるが，障害エッジの出力推定精度への影響が懸念される．以降ではパーティショニングによる出力推定精度の影響，及び推定精度を考慮したキーパーティショニングについて説明する．

入力データをパーティショニングするとき，全 N 個のスレッドはそれぞれに割り当てられたパーティション内のデータのみを用いて，復元対象デバイスの集約結果を推定する．スレッド j へ割り当てられたデバイス（以下「スレッド j の割当」という）を $\boldsymbol{Z}_j \subseteq \boldsymbol{X}$ と表す．このときスレッド j では，スレッド j 中の観測デバイス $\boldsymbol{O} \cap \boldsymbol{Z}_j$ の集約結果からスレッド j 中の復元対象デバイス $\boldsymbol{X'} \cap \boldsymbol{Z}_j$ の集約結果を 3.2 節で述べたとおり推定する．例えば図 3 においてエッジ 1 に障害が発生したとき，スレッド 1 の割当は $\boldsymbol{Z}_1 = \{X_A, X_C, X_E\}$ であり，スレッド 1 中の復元対象デバイスは $\boldsymbol{X'} = \{X_A, X_B\}$ である．つまり，スレッド 1 では $\boldsymbol{O} \cap \boldsymbol{Z}_1 = \{X_C, X_E\}$ の集約結果から $\boldsymbol{X'} \cap \boldsymbol{Z}_1 = \{X_A\}$ の集約結果を推定する．

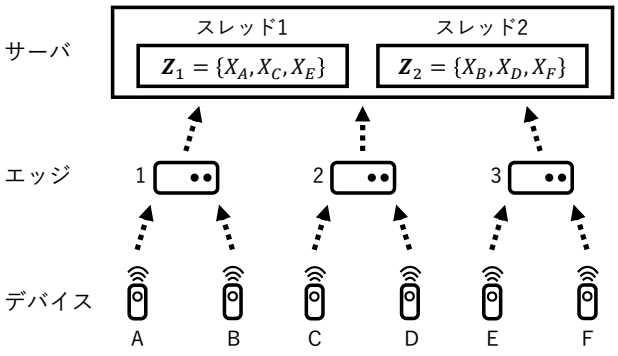

図 3: 入力データのパーティショニングの例

このとき，キーパーティショニングは障害エッジの出力推定精度に影響を与える．式 (5) より分散，すなわち式 (13) の対角要素が小さくなるほど推定誤差 e' も小さくなると分かる．式 (13) の第 2 項は観測デバイスを推定に用いることによる分散の減少を表しており，復元対象デバイスと相関の強い観測デバイスほどその減少具合は大きくなる．ただし，同じエッジに割り当てられているデバイスは必ず観測デバイスとならない．例えば図 3 においてエッジ 1 に障害が発生したとき，デバイス A の集約結果の推定にデバイス C や E の集約結果を使用できるので，相関関係が強いほど分散が減少し，高精度な推定が可能である．一方，同じスレッドにデバイス A，B を割り当てても，デバイス A の集約結果の推定にデバイス B の集約結果を使用できないので，推定精度は低くなる．すなわち，相関が強く別のエッジに割り当てられているデバイスを同じスレッドに割り当てるほど，式 (13) の対角要素が小さくなり推定精度が向上すると考えられる．

以上を踏まえて推定精度の観点からのキーパーティショニングの評価について考える．ただし，任意の障害パターンに対する推定精度を扱うのは計算コストの観点から現実的ではないため，本稿では単一エッジの障害を想定した推定精度を対象とする．つまり，スレッド j においてエッジ i に割り当てられたデバイス $\boldsymbol{X}_{ij} = \boldsymbol{X}_i \cap \boldsymbol{Z}_j$ の集約結果の推定に $\boldsymbol{O}_{ij} = \boldsymbol{Z}_j \setminus \boldsymbol{X}_{ij}$ の集約結果を用いる状況での推定精度を考える．上述のとおり，式 (9) の第 2 項が大きくなるほど推定精度も向上する傾向にあるので，各エッジに対するデバイスの割当 $\mathcal{X} = \{\boldsymbol{X}_1, \ldots, \boldsymbol{X}_m\}$ 及び事前確率分布の分散共分散行列 Σ' に対して，スレッド j の割当 $\boldsymbol{Z}_j$ を次式に示す評価関数 $f(\boldsymbol{Z}_j \mid \mathcal{X}, \Sigma')$ で評価する．

$$f\left(\boldsymbol{Z}_j \mid \mathcal{X}, \Sigma'\right) = \frac{1}{|\boldsymbol{Z}_j|} \sum_{i=1}^{|\mathcal{X}|} \sum_{X \in \boldsymbol{X}_{ij}} \Sigma'_{X\boldsymbol{O}_{ij}} \left(\Sigma'_{\boldsymbol{O}_{ij}\boldsymbol{O}_{ij}}\right)^{-1} \Sigma'_{\boldsymbol{O}_{ij}X} \tag{14}$$

ただし，$\boldsymbol{O}_{ij} = \phi$ のとき，$\Sigma'_{X\boldsymbol{O}_{ij}} (\Sigma'_{\boldsymbol{O}_{ij}\boldsymbol{O}_{ij}})^{-1} \Sigma'_{\boldsymbol{O}_{ij}X} = 0$ とする．この評価関数は，単一エッジ障害時の各デバイスの分散の減少

Input: デバイス集合 $\boldsymbol{X}$，各エッジへのデバイス割当 $\mathcal{X}$，
各デバイスの集約結果に対する分散共分散行列 Σ'
Output: 各スレッドへのデバイス割当 $\mathcal{Z} = \{\boldsymbol{Z}_1, \ldots, \boldsymbol{Z}_N\}$
/* 最小の分散のデバイスをあるスレッドへ割り当て */
1 $X_{min} \leftarrow \underset{X_i \in \boldsymbol{X}}{\arg\min} \Sigma'_{X_i X_i}$
2 $\boldsymbol{Z}_1 \leftarrow \{X_{min}\}$
3 $\boldsymbol{X} \leftarrow \boldsymbol{X} \setminus \{X_{min}\}$
/* 評価値が低くなるデバイスを別スレッドへ割り当て */
4 **for** $i = 2$ *to* N **do**
5 $X_{min} \leftarrow \underset{X_j \in \boldsymbol{X}}{\arg\min} \sum_{k=1}^{i-1} f(\boldsymbol{Z}_k \cup \{X_j\} \mid \mathcal{X}, \Sigma')$
6 $\boldsymbol{Z}_i \leftarrow \{X_{min}\}$
7 $\boldsymbol{X} \leftarrow \boldsymbol{X} \setminus \{X_{min}\}$
/* 評価関数に基づき残りのデバイスを各スレッドへ割り当て */
8 **while** $\boldsymbol{X} \neq \phi$ **do**
9 **for** $i = 1$ *to* N **do**
10 $X_{max} \leftarrow \underset{X_j \in \boldsymbol{X}}{\arg\max} f(\boldsymbol{Z}_i \cup \{X_j\} \mid \mathcal{X}, \Sigma')$
11 $\boldsymbol{Z}_i \leftarrow \boldsymbol{Z}_i \cup \{X_{max}\}$
12 $\boldsymbol{X} \leftarrow \boldsymbol{X} \setminus \{X_{max}\}$

図 4: 推定精度を考慮したパーティショニングアルゴリズム

量を表しており，相関が強く別のエッジに割り当てられているデバイス同士を全スレッドでバランスよく割り当てるほどこの値は大きくなる．

また上述の評価関数を用いた，推定精度向上のためのキーパーティショニングアルゴリズムを図 4 に示す．本アルゴリズムは大きく 2 つのステップにより，同じエッジのデバイスを別のスレッドに分散しながら，相関の強いデバイス同士を同じスレッドへと割り当てる．これにより，エッジ障害による影響の偏りを抑制しつつ，スレッド全体的な推定精度の向上を可能とする．

最初のステップでは，各スレッドに対して起点となるデバイスを 1 つずつ割り当てる．まず，分散が一番小さいデバイスをあるスレッドへと割り当てる（1–3 行目）．図 3 においてデバイス A の分散が一番小さいとすると，まずデバイス A をスレッド 1 に割り当てる．その後，割当済みのスレッドにおいて評価値が一番低くなるデバイスを順に別のスレッドへと割り当てる（4–7 行目）．このとき，同じエッジのデバイス同士を同じスレッドへ割り当てるときに評価値が一番低くなるため，割り当て済みデバイスと同じエッジのデバイスを優先的に別スレッドへと割り当てることになる．図 3 ではデバイス B をスレッド 1 に割り当てるとき評価値が最低となるので，デバイス B をスレッド 2 に割り当てる．

次のステップでは，全デバイスの割当が決定するまで各スレッドへ順々にデバイスを割り当てる（8–12 行目）．全ての割当パターンを検討するのは計算コストの観点から非現実的なので，各デバイスの割当自体は評価関数に基づいて貪欲的に決定する．その一方で，全体的には各スレッドで順番に 1 つずつデバイスを割り当てることで，スレッド間のバランスを取るよう工夫している．図 3 では 1 周目の while ループでデバイス C をスレッド 1 に，デバイス D をスレッド 2 にそれぞれ割り当てる．

また，本アルゴリズムでは並列処理による高速化も可能である．アルゴリズム中，実行回数が多く行列計算も必要となる 9–12 行目の for ループがボトルネックになると考えられる．そこで，N 個のスレッドそれぞれで評価値 Top–N のデバイスを求め，その情報を元に各スレッドへのデバイス割当を一括して決めることで，9–12 行目の for ループを並列化できる．

5 評価実験

本章では，提案アプローチにおける障害エッジの出力推定手法，及びキーパーティショニングアルゴリズムの性能を実験により評価する．

5.1 実験設定

本実験で使用するデータセット，及び想定する欠損シナリオについて説明する．

5.1.1 使用データセット

実験データ作成のため，24 台の環境センサ 2JCIE-BL [9] によって，研究室内の気温を 1 分間隔で 24 日間計測した．得られたデータの内，16 日分をモデル学習用，残りの 8 日分をテスト用に使用する．しかし，R 言語の `mvnormtest` パッケージ [10] により多変量正規性の検定を行ったが，得られたデータの分布は多変量正規分布に従うとはいえないと判定された．提案手法では入力データが多変量正規分布に従うと仮定して誤差を保証するため，今回作成した実データでは提案手法における理論的誤差保証の妥当性の検証ができない．

そこで，16 日分の学習用データから求めた期待値ベクトル及び分散共分散行列に従うデータを，R 言語の `rmvnorm` 関数 [11] により 8 日分相当の 11520 タプル生成した．`rmvnorm` 関数は入力として与えられた n 次元正規分布（平均ベクトル及び分散共分散行列）に従う n 次元の乱数ベクトルを生成する．各時刻のタプルを `rmvnorm` 関数によって独立に生成するため，各タプルは時間的な相関を持たない．

5.1.2 欠損シナリオ

本実験では単純な欠損シナリオとしてデバイスとエッジ間の一時的な通信障害，及びエッジ障害を想定し，以下の手順でテストデータを作成する．ただし，各エッジに対するデバイス割当は実環境に準拠し，各エッジには 4 ないし 6 台のデバイスが割り当てられているものとする．

まず，デバイスとエッジ間の一時的な通信障害によるランダムなデータの欠損を表現するため，8 日分の人工データ及び実データをそれぞれ欠損率 r でランダムに削除する．次に，この削除済みデータを窓幅 w の時間窓によって分割し，各デバイスの平均値に対する多変量正規分布を 2 章で述べたとおり計算する．このとき，各エッジでの処理を模倣するため，デバイス X の欠損値の推定には同じエッジに割り当てられたデバイスのデータのみを使用する．最後にエッジ障害による欠損を表現するため，m' 個の障害エッジをランダムに選択し，当該エッジに

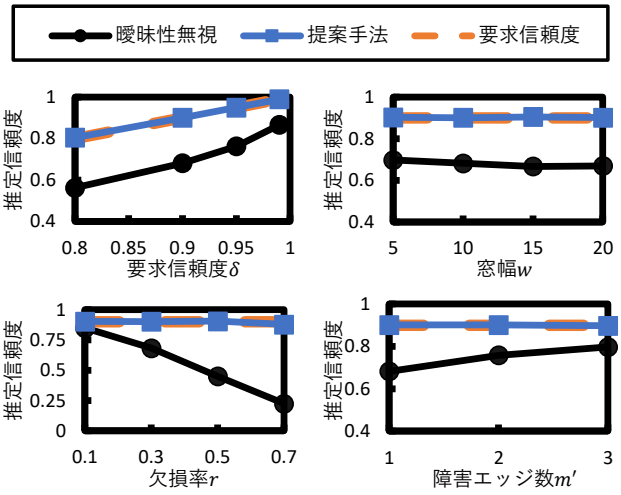

図 5: 人工データに対する推定信頼度

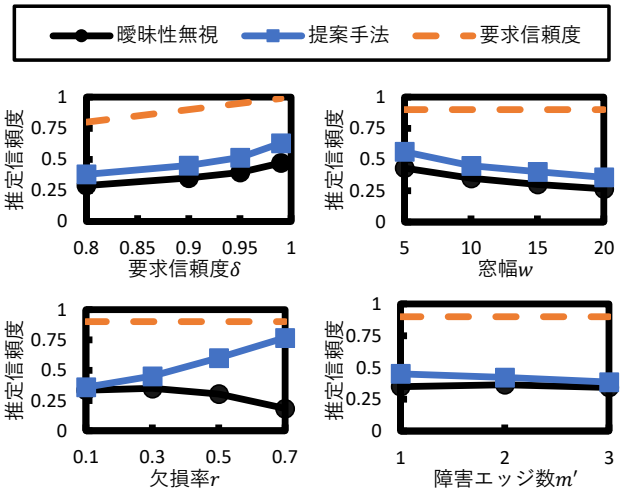

図 6: 実データに対する推定信頼度

割り当てたデバイスの集約結果を削除する．

5.2 曖昧性を考慮した障害エッジの出力推定

本節では，曖昧性を考慮した障害エッジの出力推定手法に対する評価実験について述べる．

5.2.1 評 価 方 法

提案アプローチにおいて特に重要である，理論的な誤差保証の妥当性の観点から評価を行う．本手法では，障害エッジに割り当てられたデバイス X の集約結果 Y_X を正規分布 $\mathcal{N}(y \mid \mu_{Y_X}, \Sigma_{Y_X})$ の形で推定し，ユーザ指定の要求信頼度 δ，及び式 (5) より得られる推定誤差 e' に対して，δ の確率で Y_X の真値が $\mu_{Y_X} \pm e'$ 以内に存在することを保証する．よって，欠損のないデータセットにおける集約結果を真値とし，これが $\mu_{Y_X} \pm e'$ 以内に存在する確率を提案アプローチにおける推定信頼度とする．この推定信頼度が δ を満たすとき，提案アプローチにおける誤差保証は妥当であるといえる．

本実験では，パラメータとして要求信頼度 δ，窓幅 w，欠損率 r，障害エッジ数 m' をそれぞれ変化させたときの誤差保証の妥当性について評価する．ただし，規定値はそれぞれ $\delta = 0.9$, $w = 10$，$r = 0.3$，$m' = 1$ とする．また，$\boldsymbol{y} = \boldsymbol{\mu}_{\boldsymbol{Y_O}}$ としたときの式 (8) 及び式 (9) より求められる処理結果の曖昧性を考慮しない場合の推定信頼度と比較することで，提案手法における曖昧性考慮の効果についても評価する．

5.2.2 実 験 結 果

人工データに対する実験結果を図 5 に示す．実験結果について，曖昧性を考慮しない場合の推定信頼度は常に要求信頼度 δ を下回っており，誤差の過小評価の影響が大きく表れている．対して，提案手法では様々なパラメータ設定において常に要求信頼度 δ を満たしており，理論的誤差保証の妥当性及び曖昧性を考慮した推定手法の有効性が確認できた．

各パラメータ変化について詳しく見ると，欠損率 r が大きくなるほど各エッジでの処理結果に含まれる曖昧性が大きくなるため，曖昧性考慮の効果がより顕著になり推定信頼度の差が大きくなっている．また，障害エッジ数 m' が小さくなるほど推定信頼度の差が大きくなっているが，これは損失した処理結果の推定に用いられるデータ数が増えることで，曖昧性考慮の効果がより顕著になったためと考えられる．

しかし，実データに対しては図 6 に示すとおり，提案手法の推定信頼度は曖昧性を考慮しない場合よりは高いものの，要求信頼度 δ に満たない結果となった．これは，時間経過による温度変化，及び日中の窓際は特に暖かくなりやすいなどの時間帯ごとの相関関係の変化を考慮できていない点が原因であると思われる．そのため，常に同じモデルを用いて推定するのではなく，例えばカルマンフィルタの適用による時間的な相関関係も考慮したモデル更新 [6] などが必要になると考えられる．このような動的なモデル更新のためには，エッジとサーバでのモデルの一貫性など検討すべき点も多く，今後の課題である．

5.3 キーパーティショニング

本節ではキーパーティショニングの効果に関する評価実験について述べる．

5.3.1 評 価 方 法

本実験では，キーパーティショニングにより出力推定精度の改善が可能か，また誤差保証に破綻が生じないかの二点に着目して評価する．出力推定精度の評価指標として，推定した期待値と正解データの mean absolute error（MAE）を採用する．また誤差保証の妥当性については，5.2 節と同様に推定信頼度が要求信頼度 δ を満たすか否かによって評価する．パラメータとして，要求信頼度 δ，スレッド数 N，欠損率 r，障害エッジ数 m' を用いる．ただし，規定値はそれぞれ $\delta = 0.9$，$N = 2$，$r = 0.3$，$m' = 1$ とする．

以上の条件のもと，以下の 2 種類のパーティショニングでの処理結果と比較することで，図 4 の提案アルゴリズムによるパーティショニングの有効性を確認する．

a） ラ ン ダ ム

スレッド数 N に対してランダムにデバイスを割り当てる．ただし，スレッド間でデバイス数が均等になるよう割り振るものとする．

b） 相 関 の み

各エッジへのデバイス割当を考慮せず，単に相関の強さのみ

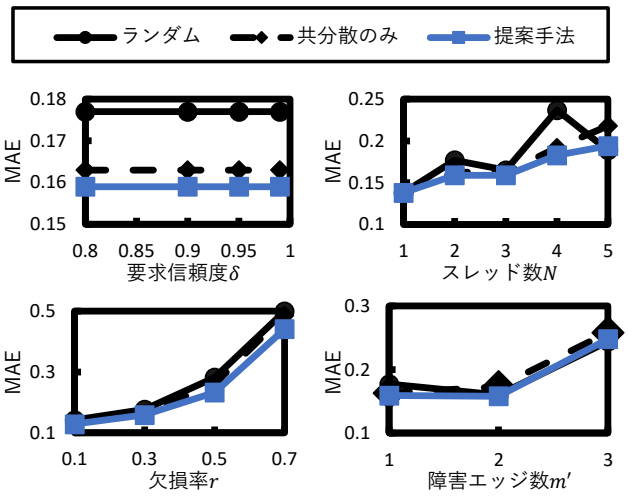

図 7: 各パーティショニングでの出力推定精度

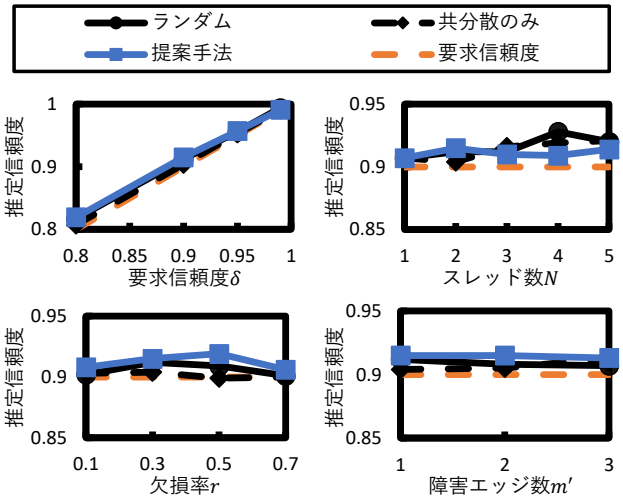

図 8: 各パーティショニングでの推定信頼度

に基づいてデバイスを割り当てる．具体的には図 4 中の評価関数として，式 (14) の代わりに次式を用いてデバイス割当を決定する．

$$f\left(\boldsymbol{Z}_j \mid \Sigma'\right) = \frac{1}{|\boldsymbol{Z}_j|} \sum_{X_i \in \boldsymbol{Z}_j} \sum_{X_k \in \boldsymbol{Z}_j, k>i} |\Sigma'_{X_i X_k}| \qquad (15)$$

式 (15) はスレッド j に割り当てられたデバイス間の共分散について絶対値の平均を取ったものである．この評価関数は各エッジに対するデバイス割当を考慮せず，単に相関の強いデバイスが同じスレッドに割り当てられるほど評価値が高くなる傾向にある．

5.3.2 実験結果

各パーティショニングでの出力推定精度を図 7 に示す．提案アルゴリズムは他の比較手法に比べ常に最小の誤差で障害エッジの出力を推定できているが，その差は大きく開いていない．今回使用したデータセットは全てのデバイスが物理的に近い位置に設置されている小規模なシステムのものであり，全体的に相関が非常に強いため，パーティショニングの効果が出にくかった可能性がある．特に，ランダムなパーティショニングであっても推定精度に大きな差はなく，本実験のような小規模なデータセットにおいてはパーティショニングのメリットは小さい．

一方で，相関のみを考慮したパーティショニングの結果との違いを詳しく見ると，状況によってはパーティショニングが推定精度に大きな影響を与える可能性があると分かる．例えば，スレッド数 N の増加に対して誤差量の差が大きくなっているが，これはスレッド数が増加するほど各スレッドに割り当てられるデバイス数が減少するため，パーティショニングの違いがより顕著に表れているものと考えられる．特に，相関のみを考慮したパーティショニングでは同じエッジのデバイスを同じスレッドに割り当てやすくなるため，エッジ障害による欠損が特定のスレッドに集中し，推定精度が大きく低下する恐れがある．実際，スレッド数 N をエッジ数と同数の $N = 5$ としたとき，誤差量の差が特に大きくなっている．また，欠損率 r が増加するほど各エッジの処理結果に含まれる曖昧性が増加するため，同様にパーティショニングの違いがより顕著になり，誤差量の差が大きくなっていると考えられる．なお，要求信頼度 δ の変化に対して常に一定の誤差量となっているが，これは式 (12) に示すとおり要求信頼度は期待値の推定に一切影響を与えないためである．

本実験では小規模システムのデータセットを用いたが，パーティショニングが特に必要となるのは大規模システムであると考えられる．システム規模が大きくなるほどデバイス間の距離の差も大きくなり，ごく一部のデバイス間にのみ強い相関が存在する状況が考えられる．このようなデータセットにおいてはパーティショニングの効果が顕著に表れると考えられるため，より大きなシステム規模での追加実験が必要である．

次に，各パーティショニングでの推定信頼度を図 8 に示す．実験結果より，どのパーティショニングにおいても常に要求信頼度を満たした誤差保証ができており，提案手法ではパーティショニングによって推定誤差に違いが生じるものの，その誤差量を正確に保証できるといえる．

6 関連研究

従来のデータストリーム処理システムでは耐障害性保証のために，クラウド環境を想定した様々なアプローチを採用している．待機系への処理移譲やチェックポインティング，損失データの再処理などの技術を組み合わせることで，誤差のない頑健な障害復帰も可能である．本章ではこれらデータストリーム処理システムでの耐障害性保証に関する先行研究について概説する．

多くの処理システムでは耐障害性保証セマンティクスとして，at-most-once セマンティクスや at-least-once セマンティクスを保証する [3, 12]．at-most-once セマンティクスを保証するシステムでは，内部状態や処理データのバックアップは取らず，障害発生時は待機ノードへの処理の移譲のみを行うため，データの損失の恐れがある．一方，at-least-once セマンティクスを保証するシステムでは，障害発生時に入力データの再送はできるが，同一データを複数回処理する可能性がある．これらセマンティクスを保証するシステムで集約処理を実行するとき，障害により生じる誤差量を保証できないため結果の信頼性に問題が

ある．

また，Flink [1] や Spark Streaming [2] などのストリーム処理システムでは，誤差のない頑健な耐障害性保証として exactly-once セマンティクスを保証できる．これらシステムでは，チェックポインティングによる内部状態のバックアップ，及び入力データの再処理に基づく復旧により，障害の発生によらず入力されたデータを過不足なく 1 回処理した結果の出力を保証できる．クラスタ環境でのストリーム処理に注目した場合，豊富なリソースが物理的に集中しており，またシステム前段のメッセージングキューに入力データの管理を一任できるため，この耐障害性保証アプローチは理に適っているといえる．

しかし，従来アプローチをエッジコンピューティング環境に適用する場合，物理的多重化コストの観点から様々な問題が生じる．具体的な障害復帰方法に違いはあるものの，従来アプローチは待機系への処理移譲によりシステム全体として処理継続を目指すことに変わりはない．しかし，エッジコンピューティング環境ではその物理的な制約から，デバイスやエッジ単位での物理的な多重化が必要になるが，システム規模が大きくなるほどこの多重化コストは無視できないものになる．また，障害復帰のために損失データの再処理が必要な場合，障害復帰処理中にも次々と新しい入力が生成されるため，データ生成時刻から実際の処理完了時刻までの遅延が増大する問題もある．クラスタ環境においてはこの遅延を抑制するため障害復帰処理を並列化するアプローチ [13] などが提案されているが，物理的多重化コストが無視できないエッジ環境においては異なるアプローチが求められる．

耐障害性保証によるコストの問題に対して，Huang らは結果の信頼性を保証した近似的耐障害性手法を提案した [14]．この手法は，未処理データ数と内部状態の変量に対する閾値を超した場合にのみバックアップを取ることで，バックアップするデータ量及び頻度を削減し，スループットを向上する．しかしこの手法では，適切な閾値の設定のためには実際のノード構成などを考慮した綿密な事前分析が必要であり，実運用において多くの困難が伴う．また，この手法は計算コストの問題のみに着目したものであり，待機系への処理移譲は必要であるため，物理的多重化コストの問題を解決できない．

7 ま と め

本稿では，環境センシングのアプリケーションを想定して，低遅延かつ高可用な耐障害性保証アプローチを提案した．本アプローチでは，デバイス間に存在する相関関係に基づき，障害エッジの出力を他のエッジの出力から近似的に推定することで，エッジ環境において問題となる物理的多重化コストを削減する．障害エッジの出力をより正確に推定するため，各エッジでの近似処理結果に含まれる曖昧性を考慮した出力推定手法を提案した．また，システム規模が拡大した際にも高スループットかつ高精度な出力推定を実現するためのキーパーティショニングアルゴリズムも提案した．評価実験により，提案アプローチの誤差保証の妥当性が確認できた．またキーパーティショニングにより，小規模システムでは効果が小さいものの，出力推定精度を改善できると分かった．しかし，実データに対しては誤差を正しく保証できなかったため，改善が求められる．

今回の実験結果を受け，現在は動的なモデル更新による時間的な相関関係の考慮に取り組んでいる．モデル更新による誤差保証の破綻を防ぐため，特にエッジとサーバ間でのモデルの一貫性の担保が課題である．その他，大規模システムでのデータセットを用いたパーティショニングの評価実験も必要である．

謝　　辞

本研究は，JSPS 科研費（16H01722，19K21530，20K19804）の助成，国立研究開発法人新エネルギー・産業技術総合開発機構（NEDO）の委託業務（JPNP16007），及び文部科学省委託事業「地球環境情報プラットフォーム構築推進プログラム（DIAS）」による．

文　　献

[1] “Apache Flink: Stateful Computations over Data Streams.” https://flink.apache.org/ (Accessed: Feb. 4, 2020).

[2] “Spark Streaming | Apache Spark.” http://spark.apache.org/streaming/ (Accessed: Feb. 4, 2020).

[3] “Apache Storm.” https://storm.apache.org/ (Accessed: Feb. 4, 2020).

[4] J.-H. Hwang, M. Balazinska, A. Rasin, U. Çetintemel, M. Stonebraker, and S. Zdonik, “High-availability algorithms for distributed stream processing,” in *Proc. ICDE*, pp. 779–790, April 2005.

[5] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, “Edge computing: Vision and challenges,” *IEEE Internet of Things Journal*, vol. 3, pp. 637–646, October 2016.

[6] A. Deshpande, C. Guestrin, S. R. Madden, J. M. Hellerstein, and W. Hong, “Model-based approximate querying in sensor networks,” *The VLDB Journal*, vol. 14, pp. 417–443, October 2005.

[7] 高尾大樹, 杉浦健人, 石川佳治, “エッジコンピューティングにおける低遅延かつ高信頼度なデータストリームの近似的集約処理（in press）,” **電子情報通信学会論文誌** *D*.

[8] C. M. Bishop, *Pattern recognition and machine learning*. Springer, 2006.

[9] “形 2JCIE-BL　環境センサ | OMRON-Japan.” https://www.omron.co.jp/ecb/product-detail?partNumber=2JCIE-BL (Accessed: Dec. 16, 2020).

[10] “mvnormtest package | R Documentation.” https://www.rdocumentation.org/packages/mvnormtest/versions/0.1-9 (Accessed: Feb. 4, 2021).

[11] “Mvnorm function | R Documentation.” https://www.rdocumentation.org/packages/mvtnorm/versions/1.0-11/topics/Mvnorm (Accessed: Feb. 4, 2020).

[12] L. Neumeyer, B. Robbins, A. Nair, and A. Kesari, “S4: Distributed stream computing platform,” in *2010 IEEE Int. Conf. on Data Mining Workshops*, pp. 170–177, January 2010.

[13] M. Zaharia, T. Das, H. Li, T. Hunter, S. Shenker, and I. Stoica, “Discretized streams: A fault-tolerant model for scalable stream processing,” tech. rep., CALIFORNIA UNIV BERKELEY DEPT OF ELECTRICAL ENGINEERING AND COMPUTER SCIENCE, 2012.

[14] Q. Huang and P. P. C. Lee, “Toward high-performance distributed stream processing via approximate fault tolerance,” *Proc. VLDB*, vol. 10, pp. 73–84, November 2016.