Vol.2018-SE-198 No.13
2018/3/9

ビッグデータにおける契約による設計を拡張した正当性検証手法の提案

岩本 侑也[1]　大森 洋一[1]　荒木 啓二郎[1]

概要：近年, ビッグデータ分析による意思決定や価値創出が注目されている. しかしながら, これまでのソフトウェア設計手法だけでは, 結果の正当性を保証するのが難しい. ビッグデータはその膨大なデータ量, データの速度などの理由により, 従来のデータベース処理と同様の分析をするのは時間や計算資源等の制約により不可能である. そのため, ビッグデータに対して統計分析などの処理を適用する際は, ビッグデータから一部を抽出したデータマートを対象とするのが一般的である. したがって, 適用された処理の出力結果は, データマートのデータに対しては妥当であっても, 抽出前のデータに対して妥当であるかを保証できないという問題がある. 本研究ではデータマートのデータに適用する統計分析処理の事前条件に統計的条件を含めることを提案する. すなわち, ビッグデータに対する統計分析処理の正当性を示すために必要な統計的性質を事前条件として明示する手法を示す. さらにデータソースの統計的性質が完全に明確にできないような場合に統計的に正当性を示す代替手法を提案する.

A Statistic Software Validation Method by Extending Design by Contract for Big Data

Yuya Iwamoto[1]　Yoichi Omori[1]　Keijiro Araki[1]

1. はじめに

本章では，本研究の背景・目的・方針と，本論文の構成について述べる.

1.1 研究の背景

近年, ビッグデータとその分析は小売, 広告, 金融, 医療, 教育など様々な分野において大きな注目を集めている [1]. また, 「機械学習」の技術などの発展によって, データを使ったシステム開発への期待は以前にも増して高まっている. ビッグデータは, 従来のリレーショナルデータベースなどのデータベース処理に対して, Volume(大きさ), Velocity (速さ), Variety (多様さ), Veracity (不正確さ) といった面で 4 つの V とも呼ばれる特性を持つ [2]. ビッグデータを分析する際は, その膨大なデータ量, データの増加速度などの特性から, 全てを分析するには不可能なほど計算資源や時間のコストがかかる場合がしばしばある. そのため, データソース全体から一部のデータを抽出したデータマートに対して統計的な分析を行うことがある. このとき, データマートに偏りがあったり, サンプル数が十分でなかったりすると, 分析結果を誤ることがある [1].

1.2 研究の目的

本研究では, ビッグデータから一部のデータを抽出したデータマートに行う処理の正当性が元のビッグデータにも成立することを示すために, データの統計的な性質を考慮したアルゴリズムの事前条件の記述法を明らかにし, さらに, それらを統計的に扱う方法を提案する.

1.3 研究の方針

ビッグデータ対する処理の正当性を契約による設計に則って示すことを考える. 契約による設計では, 関数やクラスの仕様を表明によって記述する. 表明には関数の事前条件事後条件やクラスに関する不変条件がある [4]. まず,

[1] 九州大学 Kyushu University

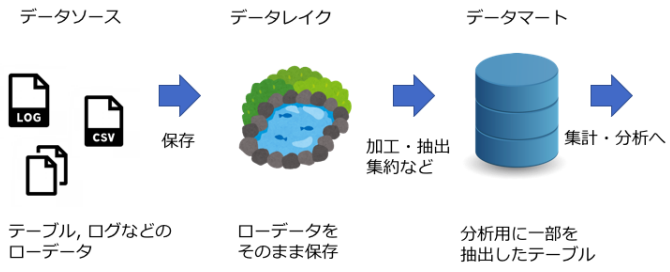

図 1　データ収集・処理プロセス
Fig. 1　Data collection and processing process

ビッグデータに対する処理でしばしば見られる統計分析処理に関する事前条件を一部データを抽出したデータマートの統計的性質として設計する方法を考察する．具体的には，データマートに対して，サンプル数が足りずデータに偏りが生じていないかをデータソース全体とカイ 2 乗検定を行って検証するというプロセスを関数の事前条件に含めることを提案する．しかし，一般的にデータソースの統計的性質が不明な場合も多い．この時，データソースを保存した同じデータレイクから得られる複数個のデータマートについて，統計的性質を検証し，確率的に正しさの保証を行う方法を検討する．

1.4　本論文の構成

以下，第 2 章では，ビッグデータの特性とその処理の例，契約による設計について説明する．また，それぞれに用いるツールを紹介する．第 3 章では，ビッグデータから抽出したデータを分析する関数について，その関数の仕様の事前条件の設計手法について述べる．第 4 章では，第 3 章で述べた手法を実際の事例について適用し，検証した結果について述べる．第 5 章で結果に対する考察と今後の課題を述べる．

2.　ビッグデータの正当性検証を行う枠組み

本章では，本研究に関連する用語について説明する．主にビッグデータと契約による設計について解説する．

2.1　ビッグデータの収集・処理プロセス

本研究で想定するデータの取得から分析のプロセスについてそれぞれ説明する．想定するシステムでは，分散システムを活用しながらデータを次々と加工していく (図 1)．まず，データの収集を行い，データを分散ストレージに蓄積し，分散データ処理によってデータを加工，集約，抽出し，抽出されたデータについて分析を行う．

2.1.1　データ収集

スマートフォンなどのモバイルアプリから集めるイベントデータや，組み込み機器から送られてくるセンサデータ，サーバーのログファイルなどを分散ストレージに転送する．データは様々な場所で生成され，それぞれが異なる形をしている [1]．

2.1.2　分散ストレージ

集めたデータは分散ストレージに格納される．ここでいう分散ストレージとは多数のコンピュータとディスクからなるストレージシステムの総称である．分散ストレージとして，オブジェクトストレージや NoSQL データベースなどが用いられる．今回使用する NoSQL データベースは，オンラインデータベースの一種である．インターネットの普及によって世界中からアクセスされるシステムが増えるにつれて，伝統的なリレーショナルデータベースでは扱えないほどの大量のデータが次々と集められるようになった．従来んのオンラインデータベースでは，原子性 (Atomic)，一貫性 (Consistency)，独立性 (Isolation)，永続性 (Durability) といった性質が保証されるが，これらのチェックにかかるコストは非常に大きく，性能の限界がある．このため NoSQL では，ストア時のデータ変換は行わない．NoSQL は例えば，一貫性の制約を取り除き，更新などの競合処理の代わりに追加で置き換えると行った工夫により，高速化を実現している．NoSQL は伝統的なリレーショナルデータベースの制約を取り除くことを目指したデータベースの総称である [7]．NoSQL データベースには様々な種類があり，代表的なものに，多数のキーと値を関連づけて保存するキーバリューストア，JSON のような複雑なデータ構造を保存するドキュメントストア，複数のキーを用いて高いスケーラビリティを実現するワイドカラムストアがある．

2.1.3　データレイク

ビッグデータでは，データの形式が多様で，テーブル設計を行ってデータを蓄積するプロセスが複雑になる．そのため，あらゆるデータをそのままの形で蓄えておいて，それを後から必要に応じて加工する仕組みが必要になる．ビッグデータの世界では，あちこちから流れ込んで来る「データを蓄えた湖」になぞらえて，データの貯蓄場所をデータレイクと呼ぶ [8]．具体的には，任意のデータを保存できる分散ストレージがデータレイクとして使用される．

2.1.4　データマート

データ分析のような目的には，データレイクから必要なデータだけを取り出してデータマートを構築する [8]．データレイクは単なるストレージであり，それだけでデータを加工できるわけではないので，分散データ処理を行なってデータ分析に必要な加工，集計を行い，必要な情報をデータマートとして取り出す．

2.1.5　MongoDB

MongoDB はオープンソースの NoSQL データベースで，分散型ドキュメントストア [7] である．JavaScript など各種のプログラミング言語を用いてデータを読み書きする．手軽さから，NoSQL データベースの中でも高い人気があ

る [6].

2.2 分散データ処理

分散ストレージに蓄えたデータを処理するには, 分散データ処理のフレームワークが用いられる. 分散データ処理のフレームワークには MapReduce や Spark がある. 分散データ処理によって, 分散ストレージに蓄積されたデータの加工や集計, 抽出を行うことができる.

データを可視化, 分析するソフトウェアは多種多様でそれぞれが異なる特徴を持つ. 本研究では JupyterNotebook を使用した. JupyterNotebook は, 対話型のデータ分析の実行環境であり, Python や Ruby, R 言語などのスクリプト言語を実行することができる [9].

2.2.1 Spark

Apache Spark は, オープンソースの分散クエリおよびデータ処理エンジンで, Matei Zaharia が UC バークレー在籍中に博士号論文の一部として開発した. 最初のバージョンは 2012 年にリリースされた [5]. Spark は大量のメモリを活用して高速化を実現する.

2.3 契約による設計

本研究では, ソフトウェアの意味論に基づいてビッグデータの処理プロセスが正しいことを示す. ソフトウェアの正当性は部分正当性と停止性により定義される. 部分正当性はソフトウェアの評価値が正しいことを保証する. 停止性はソフトウェアが必ず停止し, 評価値を必ず返すことを保証する. これらを合わせて完全正当性と呼ぶ. 一般に停止性を示すためには, 数学的な証明が必要であり, 一般化は困難である. このため, 本研究では部分正当性を対象とする. 契約による設計は, 部分正当性を示す手法であり, モジュールやルーチンの意味的な性質を表明する. 表明には主に 4 つの使い道がある [4].

- 正しいソフトウェアを書くのを助けるため.
- 文書化支援のため.
- テスト, デバッグ, 品質保証をサポートするため.
- ソフトウェアの耐障害性 (fault tolerance) をサポートするため.

ただし, 最後の 2 つは, 実行時に表明を監視できることが前提である. モジュールが表明を備えているならば, そのモジュールの部分正当性を形式的に定義することが可能である. 表明には, 事前条件, 事後条件の記述が含まれる.

2.3.1 事前条件・事後条件

事前条件とは, あるルーチンを利用する直前に満たしていることを求められる条件である. 事後条件は, 事前条件を満たすルーチンの呼び出しに対して, ルーチンの終了時に保証される特性である. あるルーチンの事前条件と事後条件は, そのルーチンのクラスと顧客との間で交わされた契約と考えることができる. つまり, ルーチンのコールが事前条件に従っている限り, 契約はルーチンに対して拘束力をもち, ルーチンは, お返しに, 事後条件を保証しなければならなず, 期待する正しさの基準となる [4]. 逆に事前条件を満たさないルーチンの呼び出しについては事後条件はもちろん, その結果に対してなんの保証もされない.

3. 提案手法

本章では, ビッグデータに適用する統計処理の課題と, その課題を解決するための手法を提案する.

3.1 ビッグデータに適用する統計処理の課題

ビッグデータの処理において, データレイクから一部データを分析用に取り出してデータマートを構築し, データマートのデータに対して関数を適用する場合を想定する. このとき, 適用されたアルゴリズムの出力が, 抽出後のデータに対しては妥当である. すなわち, 入力が事前条件を満たし, 出力が事後条件を満たしているとしても同じアルゴリズムが抽出前のデータレイクに対して妥当であるか分からない. 一般的に事前条件の確認は静的に行われる. しかし, ビッグデータ処理においては, データレイクの状況が刻々とオンラインで変わっていく. このため, 実行時に事前条件, またはデータに関する不変条件を満たしていたとしても, 実行時に変わる可能性がある.

3.2 データ抽出過程を考慮した事前条件の設計手法

契約による設計では, データ処理の仕様を事前条件, 事後条件を用いて記述する. ここでビッグデータの性質を考慮すると, 対象の処理は統計的な性質を扱うものが多いことが分かる. したがって事前条件にも統計的な性質に関する条件を次のように設計する.

(1) データレイクの変数について相対度数分布を作成する. ここで対象となるデータソースは時間的に離散値で, かつ十分な数があるものとする.
(2) データマートの変数について度数分布を作成する.
(3) 有意水準 α を決定する.
(4) 全体データの相対度数分布とサンプリング後のデータの度数分布について, 偏りがないことを帰無仮説として有意水準 α でカイ 2 乗検定の適合度検定を行う.

ここで, 事前条件として データレイクとデータマートで度数分布がある有意水準で適合していることとすると, この度数分布を用いたアルゴリズムについては事前条件が成立し, 事後条件も成立する. つまり, アルゴリズムの部分正当性を示すことができる. 逆に, 検定によって帰無仮説が棄却されれば事前条件違反となり, 結果は保証されない. すなわち, 全体データの変数の相対度数分布を reahist[ ], サンプリング後のデータの変数の度数分布を hist_sample[ ], 有意水準を a, 相対度数分布と度数分布と有意水準を引数としてカイ 2 乗検定を行う関数を chi2test とすると, chi2test(realist,

hist, a) が False となるとき, 事前条件違反となり, 事後条件の成立は保証されない.

ここで重要なのは, 事前条件に記述された以外のデータレイクの性質が変わっても, 事後条件の成立する正当性には影響しないという点である. 静的な表明検査と異なり, 特定の性質にのみ着目することで, 統計処理に影響しない変化を無視できる. いわば, 特定の統計処理に関してのみ, データの不変条件が成立していることになる. 当然, この事前条件は適用する処理によって異なる.

3.3 データの抽出過程を考慮した事前条件の設計手法 (データソースの統計的性質が不明な場合)

データの統計的性質を考慮した事前条件の設計手法を提案したが, この手法では, データレイクについての相対度数分布を知る必要がある. しかし, 一般的にデータレイクはビッグデータの特性である 4Vs を満たしており, データの相対度数分布は不明な場合も多い. その場合, データレイクとデータマートで偏りがないかを事前条件とするのではなく, データマートに複数個のサンプルを抽出し, サンプル同士についてお互いにカイ 2 乗検定を行う. 具体的には, 事前条件を次のように設計する.

(1) データレイクから複数個のサンプルを抽出し, それぞれの度数分布を作成する
(2) 有意水準 α を決定する.
(3) それぞれのサンプルの度数分布について, 偏りがないことを帰無仮説として有意水準 α でカイ 2 乗検定を行う

それぞれのサンプルについて, 検定によって帰無仮説が棄却されなければ, この度数分布を用いたアルゴリズムについては事前条件が成立し, 有意水準 α に対して事後条件も成立する. 逆に検定によって帰無仮説が棄却されれば事前条件違反となり, 結果は保証されない. 本提案手法により, 従来の契約による設計は確率的な意味に拡張される. このとき, 事前条件および事後条件が成り立つ場合, この処理の部分正当性は (1-α) の確率で成立する.

3.4 想定する開発プロセス

本手法で想定するプロセスは次の通りである.

まず第 2 章の 2.1 で説明したように, 想定するシステムでは, センサやサーバーのログやモバイルアプリケーションのイベントデータなどをデータレイクに蓄積する. 次に分散データ処理によってデータをクエリの発行が行えるように加工し, 抽出を行なってデータマートを構築する. このデータマートのデータを処理する際に, その演算について上記の提案手法によって事前条件および事後条件を設計する.

4. 本手法の適用事例

第 3 章で提案した事前条件の設計手法を Twitter から得られたデータに適用して, 有用性を検証した.

4.1 データの説明

Twitter に流れるツイートを分析する. Twitter とは, Twitter, Inc が運営するソーシャルネットワーキングサービスである. ユーザーはツイートと呼ばれる 140 文字以内のメッセージを投稿することができる. Twitter に投稿された 140 文字以下の日本語のツイートの分析を行う. データの収集方法としては, Twitter の StreamingAPIs[10] を用いてツイートデータをリアルタイムに受け取る. 各ツイートは, 図 2 のような JSON データとして受け取る. 投稿したメッセージ, 時間, ユーザー ID などの取得ができる. Twitter StreamingAPIs によって 2 日間で約 500 万ツイートを取得し, そのうち, 日本語の約 70 万ツイートに足して演算処理を行った.

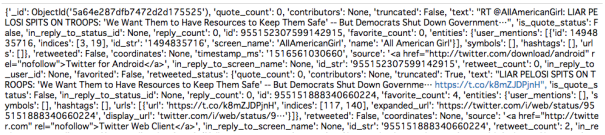
{'_id': ObjectId('5a64e287dfb7472d2d175525'), 'quote_count': 0, 'contributors': None, 'truncated': False, 'text': "RT @AllAmericanGirl: LIAR PELOSI SPITS ON TROOPS: 'We Want Them to Have Resources to Keep Them Safe' -- But Democrats Shut Down Government···", 'is_quote_status': False, 'in_reply_to_status_id': None, 'reply_count': 0, 'id': 955152307599142915, 'favorite_count': 0, 'entities': {'user_mentions': [{'id': 1494835716, 'indices': [3, 19], 'id_str': '1494835716', 'screen_name': 'AllAmericanGirl', 'name': 'All American Girl'}], 'symbols': [], 'hashtags': [], 'urls': []}, 'retweeted': False, 'coordinates': None, 'timestamp_ms': '1516561030660', 'source': 'Twitter for Android', 'in_reply_to_screen_name': None, 'id_str': '955152307599142915', 'retweet_count': 0, 'in_reply_to_user_id': None, 'favorited': False, 'retweeted_status': {'quote_count': 0, 'contributors': None, 'truncated': True, 'text': "LIAR PELOSI SPITS ON TROOPS: 'We Want Them to Have Resources to Keep Them Safe' -- But Democrats Shut Down Governme··· https://t.co/k8mZJDPjnH", 'is_quote_status': False, 'in_reply_to_status_id': None, 'reply_count': 0, 'id': 955151888340660224, 'favorite_count': 4, 'entities': {'user_mentions': []}, 'symbols': [], 'hashtags': [], 'urls': [{'url': 'https://t.co/k8mZJDPjnH', 'indices': [117, 140], 'expanded_url': 'https://twitter.com/i/web/status/955151888340660224', 'display_url': 'twitter.com/i/web/status/9···'}]}, 'retweeted': False, 'coordinates': None, 'source': 'Twitter Web Client', 'in_reply_to_screen_name': None, 'id_str': '955151888340660224', 'retweet_count': 2, 'in_re

図 2 Twitter Streaming API から受け取るデータの内容

4.2 処理手順

この事例で適用するデータの分析プロセスでは, データソースを TwitterStreamingAPI, データレイクを MongoDB, 分散データ処理のフレームワークを Spark, データマートを CSV ファイル, 分析ツールを JupyterNotebook とする (図 3). Twitter の StreamingAPI から受け取った JSON データを MongoDB に格納し, それを Spark で加工, 抽出し CSV ファイルにして JupyterNotebook で可視化, 分析する.

図 3 データ収集・処理プロセス
Fig. 3 Tweet data collection and processing process

4.3 適用する演算

抽出後のデータに適用する処理として統計処理の例であるツイートの文字数の平均の計算を行う. 文字数の平均は横軸に文字数, 縦軸に出現回数をとった度数分布から求めることができる. Twitter の文字数上限は日本語で 140 文

字であり, 横軸は高々 0 140 文字という有限離散値である. したがって, 平均を求めるアルゴリズムの事後条件となる演算結果の部分正当性が成立するためにはデータマートの度数分布とデータソースの相対度数分布に対して, 同じ分布であるという事前条件が必要となる.

4.4 適用する演算の事前条件の設計

第 3 章で提案した手順に沿って事前条件を設計した.

(1) 全体データの文字数について相対度数分布 reahist[ ] を作成した.
(2) サンプリング後のデータの変数について度数分布 hist_sample[ ] を作成した.
(3) 有意水準 α =0.05 とした.
(4) reahist[ ] と hist_sample[ ] で偏りがないことを帰無仮説として有意水準 α =0.05 でカイ 2 乗検定の適合度検定 chi2test を行った.

カイ 2 乗検定の適合度検定は Python の数値演算ライブラリ numpy, 科学技術計算ライブラリ spicy を用いて実装した.

4.5 適用結果

全体データは約 70 万ツイートで, 抽出データとしてその 1%の約 7000 ツイートを抽出した. 図 4 は全体データのヒストグラム, 図 9 は抽出後のデータのヒストグラムである.

これらについて平均を求める関数に設計した事前条件を適用した. サンプリングされたデータは設計した事前条件を満たした. 全体データの平均は 60.829, 抽出後のデータの平均は 60.965 になった. 有意水準 α =0.05 であるから、今回の事例では 95%の確率で事前条件が成立する. 同様に事後条件が成立するのも 95%の確率である. つまり, この事例では 95%の確率で結果は正しいといえる.

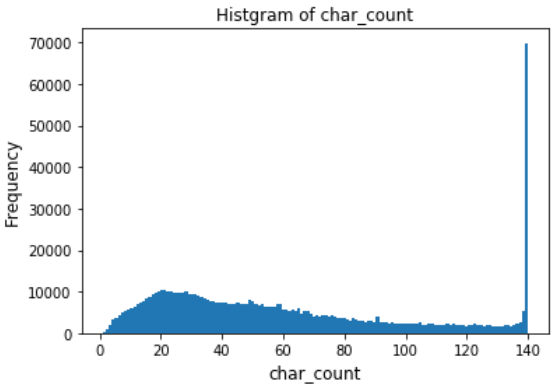

図 4 全体データ (約 70 万ツイート)
Fig. 4 All data(About 700 thousand tweets)

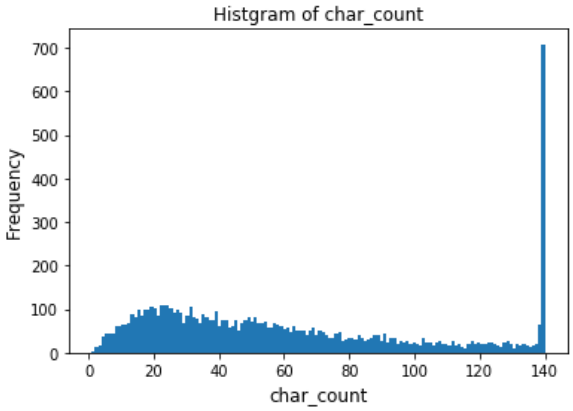

図 5 サンプルデータ (約 7000 ツイート)
Fig. 5 Sampling data(About 7 thousand tweets)

4.6 データソースの統計的性質が不明なとき

4.6.1 ランダムサンプリングによる検証

上記の例では全体データの文字数についての相対度数分布 reahist[] を作成したが, 一般的に全体データの相対度数分布は不明な場合も多い. ここでは, 3 章で示した全体データの相対度数分布が不明なときの事前条件設計手法に沿って事前条件を設計した.

(1) データマート A, B, C について度数分布 A_hist[ ], B_hist[ ], C_hist[ ] をランダムサンプリングにより作成した.
(2) 有意水準 α =0.05 とした.
(3) A_hist[ ], B_hist[ ], C_hist[ ] で偏りがないことを帰無仮説として, 有意水準 α =0.05 でカイ 2 乗検定 chi2test を行った.

ここで, ランダムサンプリングによって作成したデータマート A, データマート B, データマート C, の間で偏りがなければこのデータレイクにおいて, 対象とする統計処理について分布の一様性があることが分かる. したがって, ランダムサンプリングによるデータマート抽出によって統計的性質が保存されており, このデータマートに対する統計処理がデータレイクへの適用についても正当であることを示すことができる.

4.6.2 データソースの統計的性質が不明な場合の適用結果

全体データ約 70 万ツイートからランダムサンプリングを 3 回行い A, B, C とした. 表 1 のように A, B, C のサンプル数が約 35 万ツイート程度であれば, 事前条件を満たし, それ以下であると事前条件を満たさないことが分かった. 図 6 は約 22 万ツイート, 図 7 は約 35 ツイートのデータマートの度数分布である. このようにデータレイクの統計的性質が分からない場合においてもサンプリング数を増やすことによって, 代替となる事前条件・事後条件を確率的に求めることができた.

4.6.3 時系列のサンプリングによる検証

上記では, サンプリングをランダムサンプリングによっ

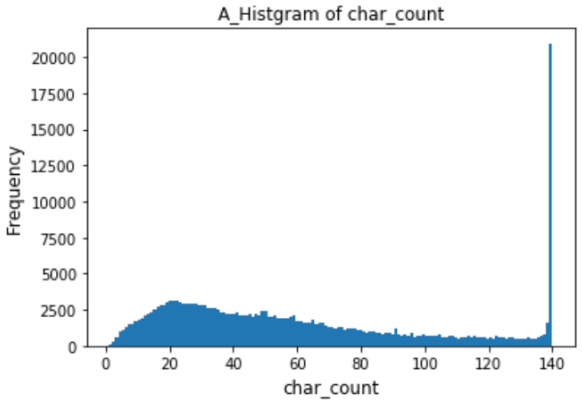

図 6　サンプル A(約 220000 ツイート)
Fig. 6　Sampling data A(About 220000 tweets)

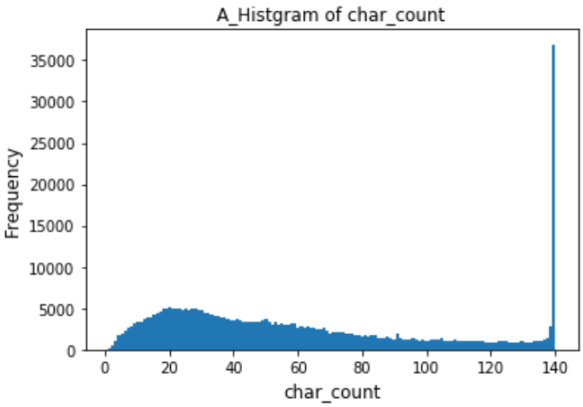

図 7　サンプル A(約 370000 ツイート)
Fig. 7　Sampling data A(About 370000 tweets)

表 1　サンプル数と統計的性質が一致するか (ランダムサンプリング)
Table 1　The number of samples and whether preconditions are satisfied.

サンプル数	統計的性質が一致するか
73865	×
221596	×
369326	○

て行った. 時間ごとに区切って次のように事前条件を設計した.

(1) データマート A, B, C について度数分布 A_hist[ ], B_hist[ ], C_hist[ ] を時間ごとに区切って, サンプリングにより作成した.
(2) 有意水準 α=0.05 とした.
(3) A_hist[ ], B_hist[ ], C_hist[ ] で偏りがないことを帰無仮説として, 有意水準 α=0.05 でカイ 2 乗検定 chi2test を行った.

4.6.4　時系列のサンプリングによる検証の結果

Twitter のデータは時期により大きく変化する. Twitter Trends はそのような性質を利用して, 流行の用語を表示するものである. このような Twitter の特性をふまえて, 時間による文字数の変動があるかどうかを確認した. 複数の短い時間によるサンプリングが一致すれば文字数の時間による変動は小さいこととなり, 逆に多くのサンプルが必要であれば, 時間による平均文字数の変化が大きいことになる.

全体データ約 70 万ツイートを時間ごとに 3 分割してサンプリングを 3 回行い A, B, C とした. 表 2 のように A, B, C のサンプル数が約 35 万を超えても事前条件は満たされず, 時間ごとに分割したサンプル同士は統計的に異なった性質を持つことが分かった. これは Twitter のツイートの文字数が時間よって変動することを示すと考えられる. 時間ごとに区切ったサンプル同士を比較することによって, ある時間の少ないサンプルをデータレイクと同じ性質のものであると処理してはならないことが分かった. 実際, サンプル数が約 24 万のとき, それぞれの文字数の平均は A が 62.36, B が 59.93 C が 60.19 となり, 誤差は大きくなった.

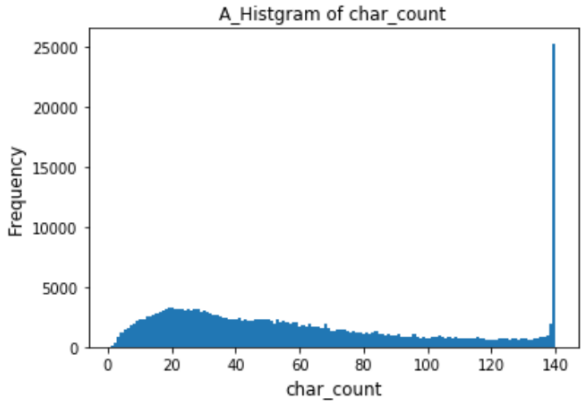

図 8　時系列サンプル A(約 240000 ツイート)
Fig. 8　Time series sampling data A(About 240000 tweets)

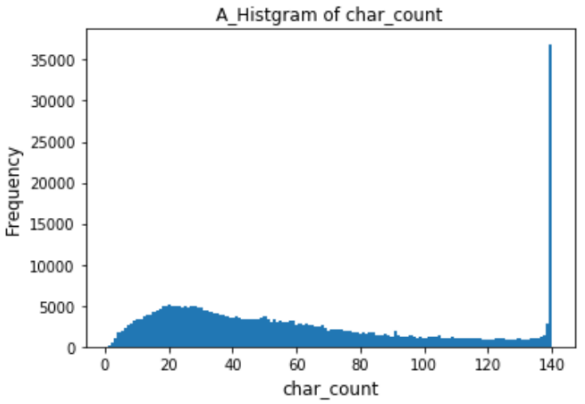

図 9　時系列サンプル A(約 370000 ツイート)
Fig. 9　Time series sampling data A(About 370000 tweets)

5.　考察

5.1　考察

第 4 章では, 第 3 章での提案手法に則って統計処理の事前条件を設計することができた. 設計した事前条件は, データ全体と抽出後のデータの関係をカイ 2 乗検定によって評

表 2　サンプル数と統計的性質が一致するか (時系列サンプリング)
Table 2　The number of samples and whether preconditions are satisfied.

サンプル数	統計的性質が一致するか
73865	×
246217	×
369326	×

価することができた. これにより, 抽出後のデータに対して統計処理を行うとき, 抽出後のデータが全体のデータと比べて偏りがあったり, 抽出データのサンプル数が十分かどうかの確認が確率的にできるようになった. 本研究により, ビッグデータに対する統計処理の正当性を契約による設計を用いて確率的に表現できることが分かった.

5.2　今後の課題

本研究では, 事前条件を設計する際, 使用するデータの度数分布を作成する必要があるが, 度数分布は離散値で作成している. この為, 連続値でも評価ができるように事前条件を設計する手法が必要であると考える.

謝辞　本研究は, 富士通九州ネットワークテクノロジーズよりの「自然言語解析による設計再利用技術の研究」の補助を受けている.

参考文献

[1] Chunli Xie, Jerry Gao Chuanqi Tao: Big Data Validation Case Study,IEEE Third International Conference on Big Data Computing Service and Applications,PP281-286 (2017).
[2] Ibtehal Noorwali, Darlan Arruda, Nazim H. Madhavji : Understanding Quality Requirements in the Context of Big Data Systems, BIGDSE '16 Proceedings of the 2nd International Workshop on BIG Data Software Engineering,PP.76-79, (2016).
[3] Dan Woods: Big Data Requires a Big, New Architecture, https://www.forbes.com/sites/ciocentral/2011/07/21/big-data-requires-a-big-new-architecture/, (最終確認日 2018 年 2 月 8 日).
[4] Bertrand Meyer: オブジェクト指向入門 第 2 版, 酒匂寛訳, (2007).
[5] The Apache Software Foundation: Apache Spark Lightning-fast cluster computing, https://spark.apache.org/, (最終確認日 2018 年 2 月 8 日).
[6] solid IT gmbh:DB-Engines Ranking, https://db-engines.com/en/ranking,(最終確認日 2018 年 2 月 8 日)
[7] MongoDB, Inc.: MongoDB for GIANT IDEAS, https://www.mongodb.com/, (最終確認日 2018 年 2 月 8 日).
[8] 西田圭介:ビッグデータを支える技術, 技術評論社, (2017).
[9] Project Jupyter:http://jupyter.org/, (最終確認日 2018 年 2 月 8 日).
[10] Twitter, Inc. :https://developer.twitter.com/, (最終確認日 2018 年 2 月 8 日)