ストリームデータ処理を実現する
アニーリングマシンアプリケーションの探索と実装
― フレームワーク「sawatabi」とデモアプリの開発 ―

1. 背景

ウェブサービスや、多量の製品や工場を保有するサプライチェーンを扱う産業等、大量のアクセスログデータや大量のセンサデータ（ビッグデータ）を取り扱う産業において、ストリームデータ処理の重要性が高まっている。近年は、IoT 推進によるセンサデータの増加や、モバイル端末・ウェアラブル端末の普及に伴い発生するウェブサーバのアクセスログや行動ログ等のデータ増加によるストリームデータ処理の需要がさらに増している。

ストリームデータ処理とは、継続的に発生し続けるデータを入力とし何らかのデータ処理をして結果を継続的に出力するようなデータ処理のことである。ストリームデータ処理の特徴として、

(1) 終わりの無い入力データが継続的に到着し、永続的に処理を実行する必要がある
(2) アプリケーションの用途に応じて低レイテンシが求められる
(3) 1 回（1 イベント）で処理するデータサイズは小さい（数 B (byte) から最大でも数 KB）

の 3 点が挙げられる。ストリームデータ処理は、ある時点で蓄積されたデータを入力とし処理してまとまった結果を返すようなバッチ処理と対照的な位置付けとされる。ストリームデータ処理アプリケーションとして多く利用されるものは即時性が求められるものであり、例えば、システムの異常検知、リアルタイムのアナリティクスやレコメンドが挙げられる。

ここ数年でストリームデータ処理のための様々なストリーム処理実行エンジンやプログラミングモデルがソフトウェアプロダクトとして OSS 公開され、開発が続けられている。これらの実行エンジンはいずれも古典コンピュータを対象としたものである。

一方、量子アニーリングや GPU・FPGA を利用したデジタルイジングマシンを用いてアプリケーションを実装し、古典コンピュータと比較して高速化・低消費電力化を実現した研究が量子コンピューティング・アニーリング分野で多く知られている。既存の大規模な組合せ最適化問題を解く用途では、イジングマシンをバッチ処理的に実行する方法が主流である。イジングマシンをストリーム処理エンジンとして利用する方法やその効率性を検証した取り組みは我々が知る限りこれまで実施されていない。

ストリームデータ処理は、小さいデータサイズの問題を絶え間なく実行する、用途によっては出力解の正確さがそれほど重要視されないというデータ処理の特徴がある。そこで、ストリームデータ処理での問題サイズと利用可能なイジングマシンのサイズがマッチしていること、イジングマシンの計算出力は近似解であることから、ストリームデータ処理がイジングマシンで実行するアプリケーションとして適していると着想するに至った。ところが、開発ユーザーがストリームデータに対してイジング計算するアプリケーションを開発する環境がこれまで整備されていないため、アプリケーション開発が簡単ではないという点が課題である。また、ストリーム処理にイジング計算をどの程度活用できるかという点が課題である。

2. 目的

以上の背景のもと、ストリームデータ処理アプリケーションがイジングマシンのキラーアプリケーションの 1 つになり得るのかを探索する目的で、本プロジェクトを実施した。

本プロジェクトの目的は以下の 2 つである。

(1) 開発ユーザーがストリームデータに対してイジング計算するアプリケーションを開発、またそのアプリケーションを実行する障壁を低くすること

(2) イジング計算をするストリームデータ処理アプリケーションを探索し、具体的なアプリケーションを実装し評価することで、イジングマシンをストリームデータ処理エンジンとしてどのように用いたら効率的に、またどの程度活用できるのかを明らかにすること。

3. ソフトウェア開発内容

ストリームデータに対するイジング計算を開発・実行する環境整備が追いついておらず障壁がある目的に対して、「sawatabi」と命名したアプリケーション開発フレームワークを開発した。sawatabi を用いることで、開発ユーザーはストリーム処理アプリケーション特有なイジングモデルの差分更新操作、入力ストリームデータに対するイジングモデル更新のアルゴリズムを記述できる。また、開発したアプリケーションはそのままクラウド上のストリーム処理エンジン（Google Cloud Dataflow）で実行できる。これにより、開発ユーザーはこれまでの方法を利用するよりもストリームデータに対するイジング計算を実行することが容易になる。sawatabi の high level architecture を図 1 に示す。

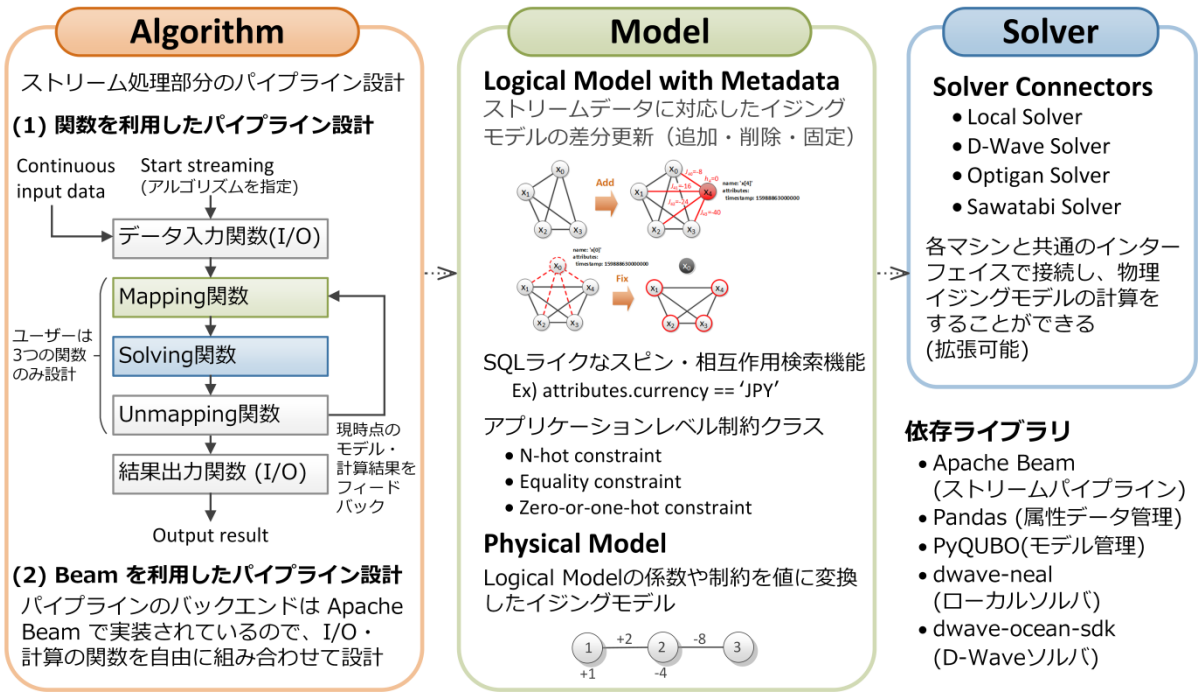

図 1: sawatabi の high level architecture。ストリームデータに対するイジング計算実行のデータパイプラインを提供するアルゴリズム層、モデルの管理とストリームデータに対するイジングモデルの操作インターフェイスを提供するモデル層、物理マシンとの接続を提供するソルバ層の 3 層から構成される。

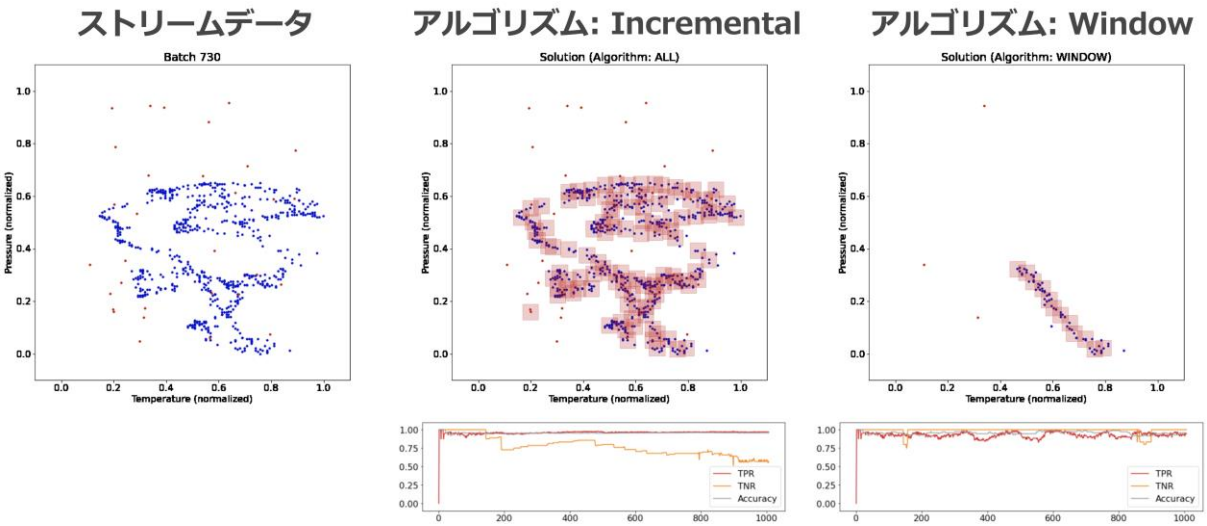

図 2: 気象データ(東京の 2020 年 10 月 1 週目の 10 分間隔の気温と気圧データ)を入力ストリートデータとしたときの、ある途中の時刻での 2 つのデータ更新アルゴリズムでのアニーリング計算結果を示した図。上の分布図の横軸は正規化された気温、縦軸は正規化された気圧を表す。データ点の青色は正常値、赤色は異常値を表す。ピンクの箱はクラスタを表す。下の図は各時刻での TPR、TNR、Accuracy を表す。横軸は時刻、縦軸は評価値を表す。

イジングマシンをストリーム処理エンジンとしてどのように用いたら効率的に活用できるのかを明らかにするという目的に対して、4 つのデモアプリケーションを開発して実験評価した。4 つのアプリケーションは、異常検知、ジョブショップスケジューリング問題の最適化、最適裁定取引機会の検出、巡回セールスパーソン問題の最適化である。デモアプリケーションを通して、常に発生し続けるストリームデータの入力に対しても継続的にイジングマシンでのアニーリング計算を実行できること、sawatabi のモデル更新操作の関数を利用することで特にすでに構築済みのモデルに対して相互作用を追加する操作を短いコーディング記述で実現できること、開発したアプリケーションを既存のストリーム処理エンジン(例: Google Cloud Dataflow)上でメッセージキュー(例: Google Cloud Pub/Sub)を入出力として動作できること、イジングモデルの更新アルゴリズムを適切に設定することで求めたいアプリケーションの結果の精度を向上させられることを確認した。ジョブショップスケジューリングを題材として、D-Wave をイジング計算のストリーム処理エンジンとして用いるときの最適実行オプションを探索した。探索には最良解を得られる確率をモデル化し、実験計画として Definitive Screening Design を用いて、応答曲面法により最適パラメータを算出した。最適裁定取引機会の検出問題では、過去のストリームデータに対する解を初期状態として利用しリバースアニーリングを利用することでより良い解を得られる場合があることを実験的に確認した。

sawatabi は Python パッケージであり、GitHub 上で Apache License 2.0 で OSS として公開しているため、誰でも利用・修正・拡張することができる。開発したアプリケーションはストリーム処理エンジンですぐに実行できることから、より現実的なアプリケーションとして応用可能だと考えられる。これらの点から利用障壁が低くなることで、ストリームデータに対してイジング計算するアプリケーション構築の事例が加速されることが期待され、価値がある。

4. 新規性・優位性

本プロジェクトの新規性・優位性は以下の 2 点にまとめられる。

i. 継続的なストリームデータに対するアニーリング計算を初めて試み、その開発フレームワークや具体的なデモアプリケーションを実装した。開発フレームワークを通じて作成したアプリケーションは既存のストリーム処理エンジン上で動作させることができる。

ii. デモアプリケーションを通し、データ更新のアルゴリズムの選択や過去データの結果をアニーリング計算の初期状態として利用することの工夫を採用することで、アプリケーションの結果が良くなるケースがあることを示した。ストリームデータに対してアニーリング計算を利用することの優位性を示した。

5. 期待されるユーザー価値と社会へのインパクト

本プロジェクトで開発したソフトウェアは以下の観点でユーザー価値と社会へのインパクトがある。

技術的な視点では、アプリケーション開発の工数削減と計算資源コスト削減の課題を解決できる可能性がある。sawatabi を用いることでストリーム処理アプリケーションの開発に必要となる共通処理をライブラリとして利用できることから、開発工数の削減に寄与することができる。また、ビッグデータを扱うような企業では、通常、ストリーム処理を実行するための古典コンピュータのサーバーを調達する必要がある。本プロジェクトでストリームデータに対するイジング計算アプリケーションの開発と実行が可能になったことから、従来の処理をイジングマシンにオフロードすることができ、性能の高速化または低消費電力化に寄与することができる可能性がある。また、ユーザーが目的の実現手段として、古典コンピュータに加えてイジングマシンという手段を選ぶことができる選択肢を与えることができるため、企業はユースケースや目的に応じたデータ処理の手段の選択ができる。

社会的な視点では、ストリーム処理システムを用いたことに起因する社会的課題を解決できる可能性がある。開発した異常検知アプリケーションを応用することで、例えば異常検知アプリケーションに与える入力データとしてセンサデータやログデータを用いてそのシステムの不正ログインやシステムの異常を検知することができ、システムのセキュリティ向上に寄与することができる。

6. 氏名

寺田 晃太朗
古山 慎悟
臼井 純哉
小野 和輝

（参考）関連 URL

sawatabi OSS 公開リポジトリ: https://github.com/kotarot/sawatabi