4K-07

FAにおけるエッジアプリケーションフレームワークとその前処理言語

境　将輝†　　中島　毅†
芝浦工業大学†

高橋　和也‡
Edgecross コンソーシアム‡

1　研究背景

FA(Factory Automation)の IoT 化が急速に進み，製造機器に近いエッジ領域においてリアルタイム性の高い分散処理を行うエッジアプリケーション(以下 EAP)技術が注目されている．従来，生産ラインから取得したデータはオフライン上で分析され，その分析結果を生産ラインのオペレーションに反映させることで，生産性向上のために利用されてきた．この改善サイクルの効率を上げるために，オフライン分析の結果を直接EAP に反映することで，リアルタイムに機器を監視・制御することが求められている．

しかし，オフライン分析には多くの手法があること，製造現場のデータを取得する方法が製造機器ごとに異なることから，それらの差異によって発生する作業負担を軽減させるような EAP の開発支援が必要である[1]．分析結果を反映するためには，推論モデルの利用が必要となり，それにはモデルの作成だけでなく，機器から取得したデータを分析に活用するためのデータ加工処理(前処理)に関する負担軽減が求められている．

2　研究背景

本研究の目的は様々なEAP の開発を包括的に支援するためのフレームワークの構築とその評価である．本論文では特に実装コストが大きくなる前処理に重点を置き，EAP における前処理の要求事項の洗い出し，それを満たした記述言語の実装・評価を行う．

3　提案フレームワークと EAP

3.1　EAP フレームワーク構想

EAP フレームワークはオフライン分析での結果を用いることで，リアルタイムで取得したデータに対して推論を行い，推論結果に応じて機器の制御を行う．オフライン分析での結果をリアルタイムでの推論に反映させるには以下の 3 つのモデルを使用する．

① データモデル:統合データ処理層から出力されたデータ項目の定義．主な項目としてデータの型やデータ名などがある．オフラインとオンラインのどちらでも同じデータ名でアクセスできることを前提とする．

② 前処理記述モデル:収集したデータを推論モデルが必要とする説明変数に変換するための前処理方法の定義．

③ データマイニングモデル:データ分析結果を表現する推論モデル．

作成されたモデルに基づき，EAP は①製造機器からデータの取得，②取得したデータに対して前処理の実行，③前処理によって作成されたデータを用いた推論の実行の 3 つの動作を制御する．

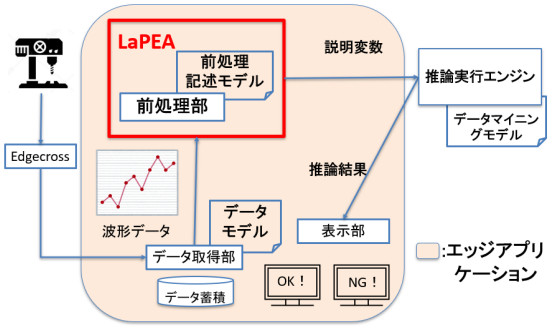

図 1　EAP 構成図

3.2　フレームワークを用いた EAP

図 1 はフレームワークを用いた EAP の構成図である．本 EAP は統合データ処理層にデータ収集プラットフォームの Edgecross[2]を用いており，Edgecross とデータの送受信を行うことを前提としている．各部の機能を以下に示す．

① データ取得部:Edgecross から MQTT 通信で複数のセンサデータを取得する．Publish/Subscribe 方式の Subscriber の役割を持つ．

② 前処理部:データ取得部が受け取ったデータに対して前処理を実行し，データ推論に必要なデータ(説明変数)を作成する．その後，推論実行エンジンに説明変数を送る．

③ 表示部:3 種類のデータの可視化を行う．過去に取得したセンサデータ，リアルタイムで取得しているセンサデータ，推論結果を表示する．

4　前処理記述言語

4.1　EAP におけるデータ前処理の要求事項

EAP における前処理には以下の 3 つの要求事項が存在する．

① データの切取り:センサからの時系列データに周期性があるとき，その一部分のパターンを解析する場合がある．そのため，周期データから特定部分を切取る処理を実行できる必要がある．

② 四則演算:データ同士の割合を求める場合などデータ同士の演算を実行できる必要がある．

③ 処理の組合せ:加工処理，統計処理及び四則演算を組み合わせて説明変数を作成する場合に対応できる必要がある．

4.2　要求事項を満たす前処理記述言語(LaPEA)

4.1 の要求事項に対応するために本研究では EAP で実行する前処理モデルを設定する前処理記述言語 LaPEA(Language for Preprocessing of Edge Application)とその実行エンジ

A edge application framework and its preprocessing language in FA
† Masaki Sakai・Shibaura Institute of Technology
† Tsuyoshi Nakajima・Shibaura Institute of Technology
‡ Kazuya Takahashi・Edgecross consortium

ンを開発した. LaPEA では要求事項に対応して以下の 3 つの設計を行った.

① データの切取り:センサからの時系列データを切取り,リスト型の変数に代入するデータ抽出関数を提供する. 変数にはデータ抽出結果だけではなく,関数の処理結果を代入することができる.

② 四則演算:演算子を用いたデータの演算を可能とする. 演算の対象は単数だけではなく,抽出したリスト型のデータ同士でも行うことができる.

③ 処理の組合せ:関数適用を可能にすることにより,関数を別の関数の引数とすることで関数の組合せを実現する.

図2 は LaPEA の記述例である. XML 形式で,それぞれのタグの中に①,②,③を実行する命令文を記述する. 例では, EAP は 1ms 毎に 4 つのデータ(data1～data4)を受け取り合計 6000ms 分のデータを取得することを想定しており, 図の記述は data1 の 1ms から 1000ms までの 1000 個分のデータを L1 という変数に代入した後,その L1 の平均値に 10 を足したデータを”data1_1_1000_ave”という名前の説明変数として作成している.

```
<?xml version="1.0" encoding="utf-8"?>
<preprocessing>
    <dataDef>
        <!-- データの切取り -->
        <data>L1 = window("data1", 1, 1000)</data>
    </dataDef>
    <variables>
        <!-- 関数適用, 四則演算 -->
        <variable name="data1_1_1000_ave">ave(L1) + 10</variable>
    </variables>
</preprocessing>
```

図 2 LaPEA 記述例

LaPEA は, 変数名を用いて 1 つの命令文を簡略化するためのデータ定義部, どの命令文の処理結果が説明変数として扱われるかを明確にするための説明変数作成部の 2 つの構文要素を持つ.

(1) データ定義部(dataDef):データ抽出関数の処理結果を変数に代入する記述部分. 1 つのタグ(data)で 1 つの代入を行う.

(2) 説明変数作成部(variables):宣言した変数を用いて説明変数を作成する記述部分. 1 つのタグ(variable)で 1 つの説明変数を作成する. 説明変数名を name 属性で設定し,処理結果と名前を対応付けることによって説明変数を作成する.

5 前処理記述の評価

LaPEA の前処理記述の可読性と保守性に関する評価のために,純粋な XML ベースの前処理記述言語[3]を従来技術として,処理行数とネストの深さについて比較評価する.

5.1 従来の前処理記述

図3 は従来前処理記述言語の記述例である. 図2 の LaPEA の例と同じ処理内容を表現している. 従来の記述も要求事項①～③を満たすが,式で前処理を表現するのではなく,1 つのパラメータをタグで囲む方式を取っている.

```
<?xml version="1.0" encoding="UTF-8"?>
<Preprocessing>
    <variable name="data1_1_1000_ave">
        <function name="+">
            <Arg>
                <arg>
                    <function name="ave">
                        <Arg>
                            <arg>
                                <function name="window">
                                    <Arg>
                                        <arg>data1</arg>
                                        <arg>1</arg>
                                        <arg>1000</arg>
                                    </Arg>
                                </function>
                            </arg>
                        </Arg>
                    </function>
                </arg>
                <arg>10</arg>
            </Arg>
        </function>
    </variable>
</Preprocessing>
```

図 3 従来前処理記述言語の記述例

5.2 従来との比較

表1 は 図2 の LaPEA 記述と図3 の従来記述の行数とネストの深さを比較したものである. 従来記述は LaPEA と比べて,行数では約3 倍,ネストの深さでは5 倍要することが分かる. 従来記述は演算の項や処理の組合せが増えるほど行数, ネストの深さが増大し, 記述が複雑化してしまうのに対して, LaPEA はネストの深さは変化せず行数も 1 処理の増加に対して高々1 行の増加に抑えることができる. LaPEA の方が可読性と保守性を高く記述できることがわかった.

表 1. 行数とネストの深さの比較

	LaPEA	従来記述
行数	9	25
ネストの深さ	2	10

6 まとめと今後の課題

本研究では製造現場における EAP 開発における負担軽減のためのフレームワークの提案とそのフレームワーク上で前処理を表現する記述言語の実装・評価を行った.

現在は基本的な要約統計量を求める処理や四則演算などのみの実装となっているため, 二値化や離散化などの種々の処理をビルトイン関数として実装する必要がある. しかし前処理は無数に存在し, ビルトイン関数だけでは網羅できない処理もあるため, 簡単に新しい前処理を LaPEA で実装できる仕組みも必要である.

参考文献

[1] 製造業における IoT の活用例と将来像:e-F@ctory を例として (特集 IoT がもたらす製造業の新展開) オペーションズ・リサーチ=Communications of the Operations Research Society of Japan:経営の科学 ,2018(柳生 理子 高橋 雄一 大谷 治之) 著

[2] Edgecross コンソーシアム https://www.edgecross.org/ja/

[3] Sakai, Masaki, Tsuyoshi Nakajima, and Kazuya Takahashi. "A Proposal for Edge Application Framework for Smart Factory." 2020 IEEE 44th Annual Computers, Software, and Applications Conference. IEEE, 2020.