Modin で pandas* のワークフローをスケーリング

書き換え不要のスケーラブルなデータ・アナリティクス

Vasilij Litvinov インテル コーポレーション AI フレームワーク・エンジニア

[この記事は Anaconda.com（英語）に掲載された記事を許可を得て掲載したものです。]

AI やデータサイエンスが急速に進化することで、膨大な量のデータが利用できるようになり、日々複雑化し続けるアイデアや解決策を導き出すことができます。一方で、これらの進化は、価値抽出からシステム・エンジニアリングへと焦点が移っています。また、ハードウェアの性能は、人が適切な活用方法を学ぶよりも速いスピードで成長しています。

このような傾向から、「データエンジニア」という新しいポジションが登場したり、データ・サイエンティストがデータサイエンスの中核である調査ではなく、インフラストラクチャー関連の問題に対処する必要性が出ています。その主な理由の 1 つは、データ・サイエンティスト向けに最適化されたデータサイエンスとマシンラーニングのインフラストラクチャーが存在しないことです。すべてのデータ・サイエンティストがソフトウェア・エンジニアリングに精通しているわけではありません。この 2 つは、別々の、ときどき重複するスキルセットと考えることができます。

データ・サイエンティストは習慣の生き物です。彼らは、pandas*、scikit-learn*、NumPy*、PyTorch* など、Python* データスタックで慣れ親しんだツールを好みます。しかし、これらのツールは、通常、並列処理やテラバイトのデータには適していません。

Anaconda とインテルは、データ・サイエンティストにとって最も重要な、使い慣れたソフトウェア・スタックと API をいかにスケーラブルかつ高速にするかという問題を解決するため協力しています。この記事では、Anaconda の「defaults」チャネル (および conda-forge) から利用可能な、インテル® oneAPI AI アナリティクス・ツールキット（AI キット）（英語）に含まれるインテル® ディストリビューションの Modin（英語）を紹介します。

pandas* だけでは十分でない理由

pandas* は業界標準ですが、本質的に多くの場合シングルスレッドで動作し、大規模なデータセットでは遅くなります。メモリーに収まらないデータセットでは動作しない場合もあります。この問題を解決するほかの選択肢（例えば、Dask、pySpark、vaex.io など）もありますが、どれも pandas* と完全に互換性のあるインターフェイスを提供していないため、ユーザーはワークロードを変更する必要があります。

Modin はエンドユーザーに何を提供するのでしょうか？ Modin は、「ツールはデータ・サイエンティストの役に立つべきであって、その逆はない」という考えに忠実であろうとしています。「`import pandas as pd`」文を「`import modin.pandas as pd`」文に変更するだけで、多くのユースケースでスケーラビリティーを向上できます。

Modin を使用する理由

「pandas* ワークフローを別のフレームワークに書き換える」必要をなくすことで、データインサイトの開発サイクルの短縮が可能になります（**図 1**）。

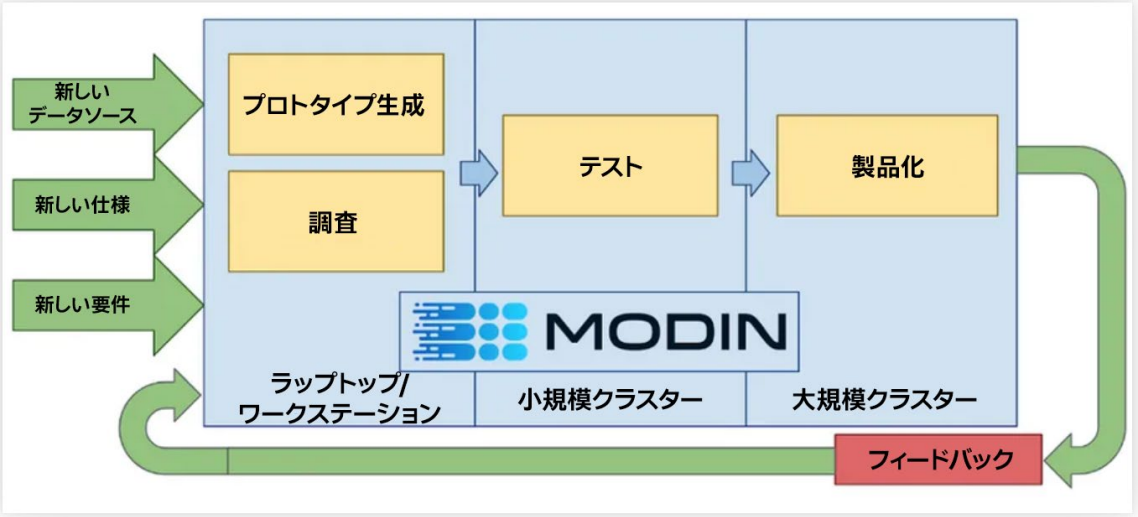

図 1. 継続的な開発サイクルでの Modin の使用

Modin は、DataFrame をグリッド分割することで、特定の操作をセル単位、列単位、行単位で並列分散して実行し、ハードウェアを効率良く活用します（**図 2**）。特定の操作では、OmniSci* エンジンとの実験的な統合を利用して、マルチコアのパワーをさらに活用することが可能です。

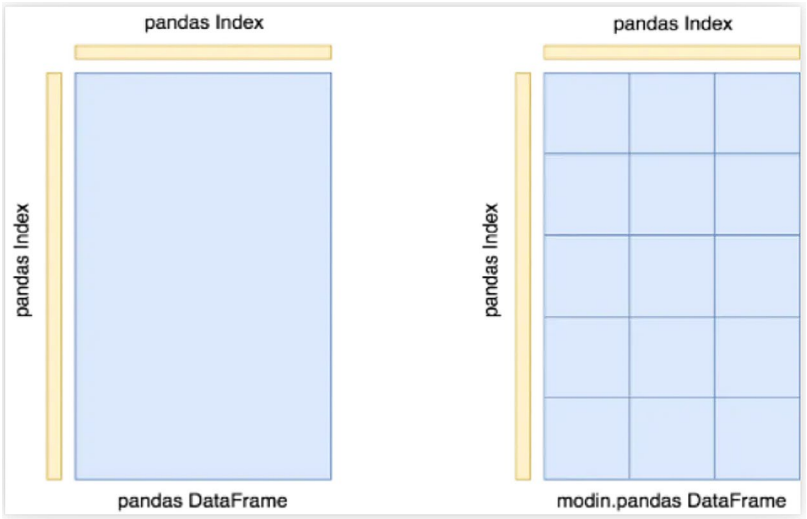

図 2. pandas* と Modin の DataFrame の比較

AI Kit や Anaconda defaults（または conda-forge）チャネルから Modin をインストールすることで、さらに高速な OmniSci* バックエンドも Modin で利用できるようになります。実験的な OmniSci* バックエンドは、以下の簡単なコード変更で有効になります。

```
import modin.config as cfg
cfg.Engine.put('native')
cfg.Backend.put('omnisci')
import modin.experimental.pandas as pd
```

実行結果

ベンチマークを見てみましょう。異なる Modin エンジンの詳細な比較は、http://modin.org/modin-bench (英語)にあるコミュニティーによって測定されたマイクロベンチマークを参照してください。Modin リポジトリーにコミットされたさまざまなデータサイエンス操作のパフォーマンスを追跡しています。

このアプローチのスケーラビリティーを実証するため、インテル® Xeon® Platinum 8368 プロセッサー・ベースのサーバー（下記のシステム構成を参照）上で、Modin を介して OmniSci* を使用して大規模なエンドツーエンドのワークロード（英語）を実行します（**図 3 ~ 5**）。

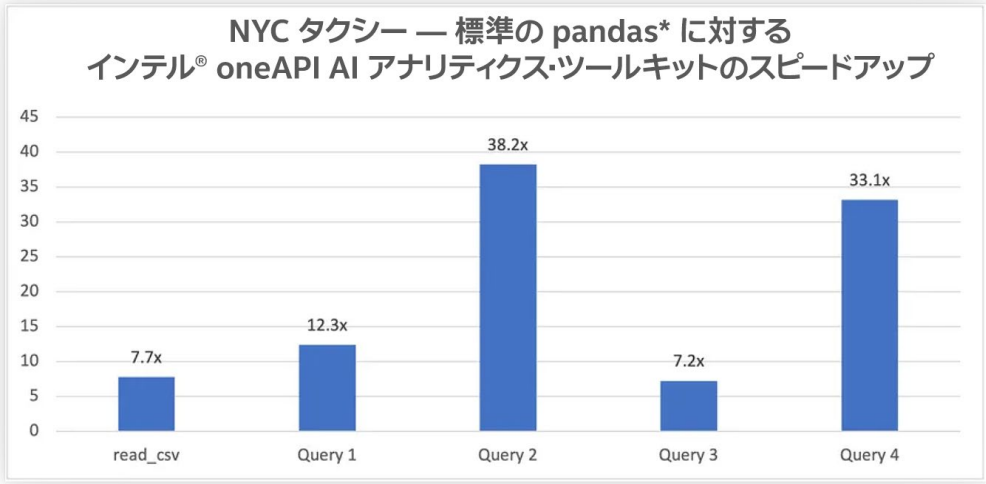

図 3. NYC タクシー・ワークロードの実行：レコード 2 億件、入力データセット 79.2GB

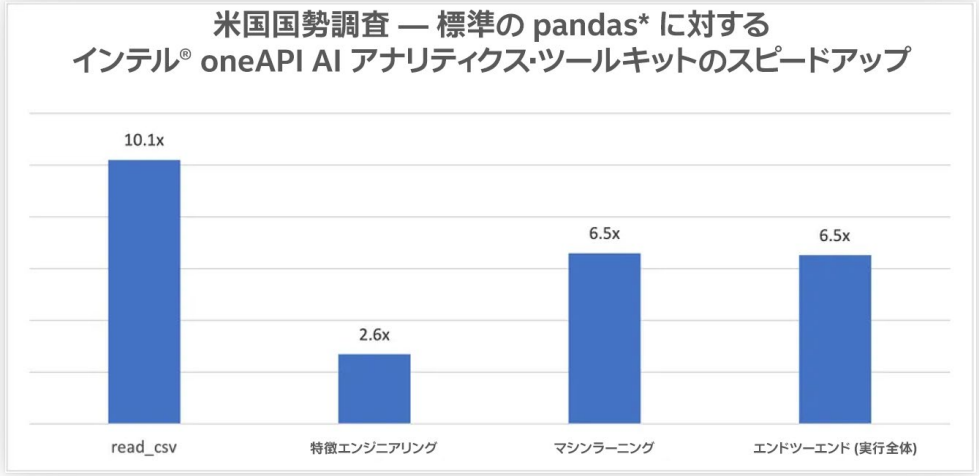

図 4. 米国国勢調査（英語）ワークロードの実行：レコード 2100 万件、入力データセット 2.1GB

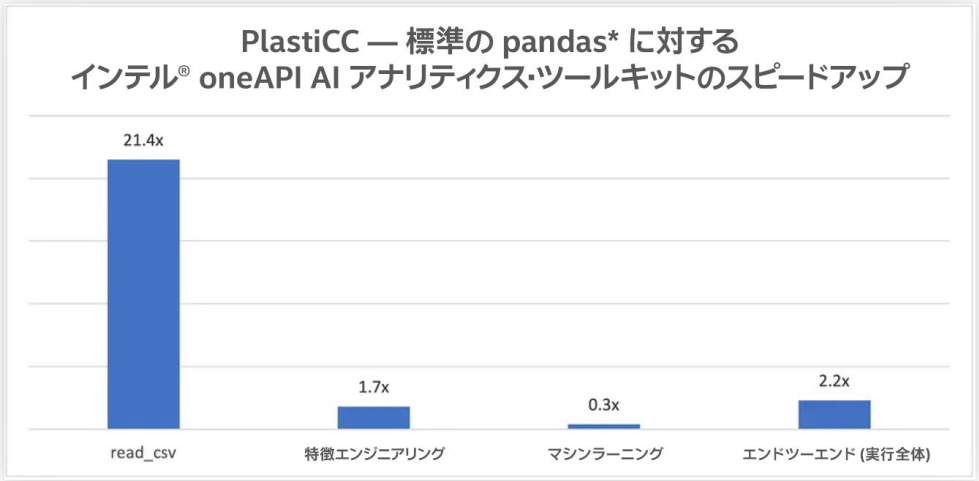

図 5. PlastiCC（英語）ワークロードの実行：レコード 4.6 億件、入力データセット 20GB

システム構成：ハードウェア：インテル® C620 ボード上の 2 基の第 3 世代インテル® Xeon® Platinum 8368 プロセッサー、合計 DDR4 メモリー 512GB（16 スロット /32GB/3200）、マイクロコード 0xd0002a0、インテル® ハイパースレッディング・テクノロジー有効、インテル® ターボ・ブースト・テクノロジー有効、インテル® SSD 960GB x1（OS ドライブ）、インテル® SSD 1.90TB x3（データドライブ）。ソフトウェア:CentOS* 7.9.2009 カーネル 3.10.0-1160.36.2.el7.x86_64、Python* 3.8.10、pandas* 1.3.2、Modin 0.10.2、OmniSci* DB 5.7.0、Docker* 20.10.8。2021 年 10 月 5 日現在のインテル社内での測定値。

追加情報

単一ノードでの実行では十分でない場合、Modin は Ray クラスター（英語）や Dask クラスター（英語）での分散実行をサポートしています。また、実験的な XGBoost との統合を使用することで、特別な設定なしに自動的に Ray ベースのクラスターを利用することも可能です。

関連情報

- インテル® oneAPI AI アナリティクス・ツールキットのインストール：
  `conda install intel-aikit -c intel`
- Modin のドキュメント（英語）
- Modin のソースと問題トラッカー（英語）
- Modin から OmniSci* バックエンドへの切り替え（英語）