第2章 大きなデータの処理なら劇的に高速化できる

データ処理＆分析ライブラリ「pandas」の並列化

佐藤 聖

並列処理の効果を実験で確かめる

● Google Colabでプログラムの実行時間を測定する

今回は，読者が同じ環境でサンプル・プログラムを試せるようにGoogle Colaboratory（以下，Colab）を使います．サンプル・プログラムは，Colabで動作確認済みです．Colab環境で使われているサーバのCPUコア数は2，スレッド数は4でした．

Pythonはシングル・スレッドなので1つのCPUコアしか利用できないのですが，C/C++などで記述された，マルチスレッドやマルチプロセスの機能を提供するPythonライブラリを使用することでPythonのGILの制限を受けずに，サブスレッドやサブプロセスを生成して複数のCPUコアで処理できます．

ここではpandasライブラリによるデータ操作を複数CPUコアで処理して並列処理の効果が得られるかどうかを試してみます．

● 処理時間の計測方法

処理時間を計測するのにJupyter Notebookの%%timeコマンドを利用します．

Colabノートブックを使う場合でもセルの最初の行にコマンドを書いておけば，セルの処理終了後に測定結果が表示されます．使い方は，測定対象のセルで1番最初の行に記述します．1行だけ処理時間を測定するときは1つ前の行に%timeを書きます．

● 測定結果の見方

以降で紹介するプログラムを実行すると次のような画面出力が得られます．

```
CPU times: user 127 ms, sys: 63.2 ms, total: 190 ms
Wall time: 19.5 s
```

これは処理時間の計測結果です．CPU timesとWall timeの行があります．CPU timesはコンピュータ内部のCPU利用時間になり，**表1**のようなサブ項目で詳細が出力されます．

Wall timeはユーザが体感するプログラムの処理時間です．この値は合計CPU処理時間，スリープ処理時間などのCPU処理以外の経過時間や処理時間を加えたプログラム全体の処理時間です．

Pythonプログラムの目的やプログラムの用途によって，処理が速いか遅いかの判断基準が変わってきます．例えばスケジュールに基づいて自動で定期実行するときには，CPU timesが重要になりますし，ユーザが操作するプログラムではWall timeの方が重視されます．

本稿では，CPU timesを基準に比較します．

pandasを並列化するライブラリpandarallel

Pythonでデータの処理や分析によく使われるライブラリpandasのデータを並列処理で扱えるpandarallelというライブラリを使ってみます．

サンプル・プログラムを使って後述するデータ・フレームに対して処理を適用します．サンプル・プログラムの中では，pandasでの処理と，ライブラリを使って並列化した処理を比較しています．

ライブラリを使って並列化した処理では，複数のサブプロセスが生成されています．メイン・プロセスは各サブプロセスに担当させるデータを決め，それぞれに割り振って処理させます（**図1**）．

● 処理対象はpandasライブラリのデータ・フレーム

pandasとpandarallelでデータ操作の比較実験をします．

表1　並列化の有用性を見極める指標として使った項目

項　目	内　容
user	ユーザ空間でCPUが利用した時間（ユーザが実行したプログラムやアプリケーションのプロセスで使用された時間表示）
sys	プログラム（プロセス）がカーネル空間でCPUを使用した時間（システムの機能を呼び出した時間表示）
total	userとsysの合計CPU利用時間