特集2

第1章 機械学習やデータ分析の大量な演算の高速化に効く

Pythonの並列処理…特徴と主なライブラリ

佐藤 聖

大量のデータを扱うなら処理速度も重要

● Pythonのプログラムは処理速度が遅い？

読者の中にもPythonを使ってデータ分析やデータ・マイニング，機械学習を学ばれる方がいるのではないでしょうか．実際にPythonはビッグ・データ処理，データ分析，機械学習などの大量の演算が必要な分野でプログラム開発によく利用されています．当然大量のデータを扱うので，処理速度も重要な要件になります．

一方で，従来のプログラミング言語よりもPythonで書かれたプログラムは処理速度が遅いとも言われます．両者は一見すると矛盾を感じるかもしれません．

● 並列に処理を実行できるライブラリを使えば高速化できる

実は，Pythonでは処理の安全性[注1]を高めるために，基本的には同時に複数の処理を実行できないという制約があります．

Pythonで書くプログラムでは，基本的には複数の処理を同時に実行できませんが，Pythonのライブラリの中には並列に処理を実行できるものもあります．

Pythonには32万を超えるライブラリ群があり，それらの中にビッグ・データ処理やデータ分析，機械学習など，大量データの演算に最適なライブラリもあります．

これらは並行処理や並列処理も自在に実行できるので，うまく使えば他のプログラミング言語に負けない高速な処理ができます．

今回は，Pythonの並列処理に注目しつつ，参考として並行処理も取り上げます．

注1：ここで言う安全性とは，変数へのアクセス競合などにより，格納されているデータを壊さないようにしたり，処理が正しい順序で行われることを保証したりすることです．

マルチタスクやマルチスレッドで単一処理の課題を解決する

● 今どきのOSやPCは既に並列処理に対応している

Windows，macOS，Linuxなどの現代の主要なOSはマルチタスク機能をサポートしています．単一処理しかできないと，実行中の処理が終わるまでユーザの操作を受け付けられませんが，マルチタスク機能によって，何らかのプログラムを実行中でも，他のプログラムやユーザによる操作を受け付けられます．

最近では，PCやサーバ用のCPUは，内部に複数のCPUコアが搭載されています．複数コアがあるとユーザの操作に対する応答性が良くなります．マルチタスク機能やマルチスレッド機能も快適に動作するようになります．

Pythonで書いたプログラムでも，並列処理ができれば同時に複数のCPUコアで処理を分担し，高速な処理を実現できます．

● 並列処理すると効率のよい例

例えば，Pythonでウェブ・サイトからデータを取得するウェブ・スクレイピングを行う際に，ウェブ・サーバの応答待ちやネットワークの輻輳（ふくそう）によって，レスポンスが悪くデータをなかなか受信できないことがあります．単一処理ではデータ受信が終わるまで，次のウェブ・サイトを巡回する関数などを実行できません．

プログラム全体でも処理完了までに時間がかかります．このケースは外部要因により時間がかかっています．ハードウェアを高性能なものに変えても時間短縮にならない可能性があります．

ここでプログラムを並列に処理するようにすれば，処理を待っているCPUコアとは別のCPUコアを使って別のウェブ・サイトのスクレイピングを開始できます．

応答待ちの間に他の処理を進めれば，プログラム全体としての処理時間短縮になります．