DOI: 10.18957/rr.5.2.288 SPring-8 利用研究成果集 Section C

2013A1896 BL14B2

BL14B2 における遠隔 XAFS プログラムの開発
Development of Remote-XAFS Program at BL14B2

高垣 昌史[a], 井上 大輔[a], 古川 行人[b], 本間 徹生[a]
Masafumi Takagaki, Daisuke Inoue, Yukito Furukawa, Tetsuo Honma

[a](公財)高輝度光科学研究センター・産業利用推進室･産業利用支援グループ、[b]同・制御･情報部門・機器制御グループ
[a]Industrial User Support Group, Industrial Application Division, JASRI,
[b]Equipment Control Group, Controls and Computing Division, JASRI

BL14B2 において開発を進めている遠隔 XAFS 環境の基盤技術である Quick XAFS 測定プログラム「QXAFS」が完成した。初版は透過配置モードのみに対応する。従来のローカル版 Quick XAFS 測定プログラムに比べ、処理速度の向上と高い安定性を確保することができた。

キーワード： 遠隔実験、XAFS

背景と目的：

産業利用推進室では、制御・情報部門との協力体制のもと、BL14B2 の XAFS 自動化技術を基盤として、インターネット経由で XAFS 測定を可能にする「遠隔 XAFS システム」の開発を進めている。産業利用分野においては、人的、資金的、時間的資源上の制約から、ユーザー実験は少数の熟達した測定担当者が行い、実験結果を真に求めている試料提供者が実験に参加できず、その意見が実験進行にフィードバックされづらいケースが少なくない。遠隔 XAFS システムが完成すれば、ネット接続が可能な環境にいる限りどこからでも実験に参加することが可能となるため、試料提供者の意見のリアルタイムなフィードバックが可能となり、より商品開発に密着した高品質の実験結果の創出が期待される。

本課題の目的は、課題番号 2013A1827 において開発した遠隔 Quick XAFS 測定プログラム「QXAFS」のプロトタイプをもとに、処理速度の向上と、各関連サーバプロセスとの連携の最適化を図り、初版を完成させることである。遠隔 XAFS システムでは、すべてのプログラムはサーバプロセスであり、これらサーバの並列動作の最適化が、処理速度向上の大きな鍵となる。QXAFS の動作は 7 つのサーバの連携によるものであり、本課題では、これらの連携に無駄が生じないよう注意深い調整を行った。

方法と結果：

QXAFS は、「測定エンジン」と「測定パラメータ入力」の他、「カウンタ」「モノクロ」「モノクロエンコーダ」「データ変換」「アップローダ」の、合計 7 つのサーバの連携によって動作する。このなかで特に処理速度に影響するのが、測定エンジン、データ変換、アップローダの 3 つのサーバの連携である。従来のローカル版 Quick XAFS 測定プログラムにおいても、特にデータ変換処理に時間を要することが明らかになっており、これを高速化するため測定部プログラムとデータ変換部プログラムとの並列化がなされてきた。しかしローカル版測定部プログラムは、データ変換部プログラムの動作状況をモニタできないため、処理投入のタイミングがずれると、測定部、データ変換部双方が異常停止する危険性があった。QXAFS ではこの点を改善し、データ変換のサーバ化によって変換の処理状態を確認できるようにしたうえで、最適のタイミングで次の処理を投入する構成とした(図 1 上図および中図)。また QXAFS は、変換後のデータはデータリポジトリにアップロードされるが、この連携動作はデータ変換との並列化が不可能であるため、そのままでは測定エンジンの待機時間が長くなり、測定時間を圧迫する。そこでアップロード処理の投入をデータ変換サーバに担当させることによって、測定エンジンはデータ変換以降の処理を気にすることなく次の測定に移行できるようにした。これらの仕組みによって、サーバ間の並列性と連携上の安全性が大幅に向上し、特に連続測定においては、次の測定に移る時間が改良前は数十秒であったところが数十ミリ秒にまで短縮できた(図 1 下図)。

一方で、上記とは別に測定エンジンが担当する処理「関連機器からのデータの収集と整形」に時

間を要し、測定時間を圧迫していることが判明した。これは、1 測定あたり数千点と、測定点数が多いことが原因であり、測定エンジンの実装系であるスクリプト言語では、整形だけでも 5〜10 秒程度を要してしまうことが明らかとなった。本課題は全 5 シフトを 3 シフト+1 シフト+1 シフトと構成しており、前半 3 シフト終了後、データ収集/整形ルーチンを C 言語で実装しなおし、後半 2 シフトで動作検証を行った。その結果、処理時間を数十ミリ秒程度まで短縮することに成功した。

今後の予定：

本課題によって、QXAFS(透過配置モード)の初版が完成した。今後の運用テストを経て安定性の確認を進める予定である。また、ユーザーインターフェースであるウェブクライアントは、ユーザーの入力ミスを誘発しないよう、注意深くデザインする必要がある。図 2 に QXAFS ウェブクライアントのプロトタイプを示す。今後、ユーザーからの意見を取り入れつつ極力簡素化を図り、さらなる安全性の向上を目指す予定である。

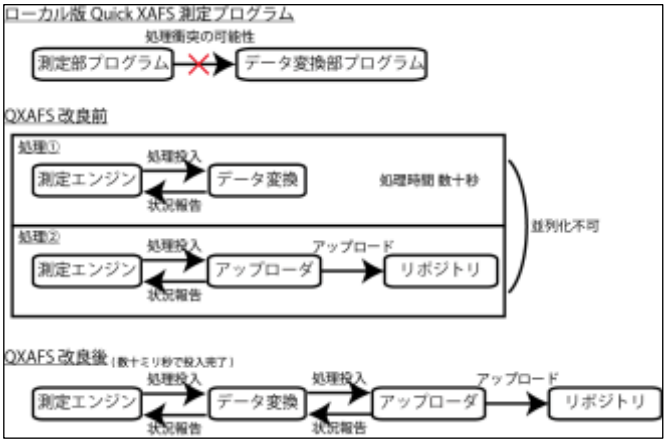

図 1. ローカル版 Quick XAFS 測定プログラムと QXAFS の構成(QXAFS の各部はすべてサーバとして実装)

図 2. QXAFS ウェブクライアントのプロトタイプ

(Received: September 7, 2016; Early edition: Febryary 24, 2017;
Accepted: July 18, 2017; Published: August 17, 2017)