Web ページを開いておくだけの
ボランティアコンピューティングフレームワークの開発

1. 背景

World Community Grid では今現在も HIV の新しい候補薬の特定、ヒト・タンパク質構造予測の限界探査、炭素ベースの光発電効率を高める新素材の探索、新たなフィルター素材で効率的な水分子の流れの様相探索など、様々なボランティアコンピューティングのプロジェクトが運営されている。しかし、現在のボランティアコンピューティングは開発を行うのも、計算資源の提供を行うのにもハードルが高い。

2. 目的

我々人類が解決するべきより多くの問題への大王と、そのためのより多くの計算資源を得るために、これまでの BOINC フレームワークとは異なって、簡単なタスクファイルを記述することで自動的にブラウザをプラットフォームとして分散処理を行うためのボランティアコンピューティングフレームワークを開発し、ボランティアコンピューティングがより身近なものになることを目的とする。

3. 開発の内容

本プロジェクトでは、ブラウザをボランティアコンピューティングのプラットフォームとしているため、サーバのタスク管理の仕組みとブラウザでのタスク実行の仕組みをフレームワークとして実装した。サーバサイドは node.js で実装し、サーバ・ブラウザ間の通信に Socket.IO と呼ばれる非同期通信ライブラリを用いて、ブラウザ側はバックグラウンドでタスクの実行を行っている。それぞれについて説明する。

3. 1 タスクの管理

ブラウザから HTTP でアクセスするために、まず Web サーバの選定を行った。Apache と Node.js のベンチマークの結果、Node.js の方が多数のリクエストを上手くさばけ、また今回はタスクファイルの記述を簡単にしたかったため、サーバサイドとクライアントサイドを同じ言語で書けるという理由から Node.js を採用した。Node.js の Express というフレームワークに HTTP サーバとしての機能を任せ、それをラップする形でタスク管理の機能を追加した。

タスクがブラウザに配信されるまでの処理の流れを説明する。まずサーバ起動時に実行したい処理を記述するタスクファイルを書いておく。するとサーバ起動時にタスクファイルが読まれて多数の JSON-RPC のような形式に変換されサーバに保存される。そしてブラウザから要求があったときにタスクを読み出しブラウザに配信し、処理が終わったら結果を受け取る。すべての処理が終わったときにセットされたコールバック関数が呼ばれるので、そこでデータを整形して DB に格納した画面に出力したりする（図 1）。

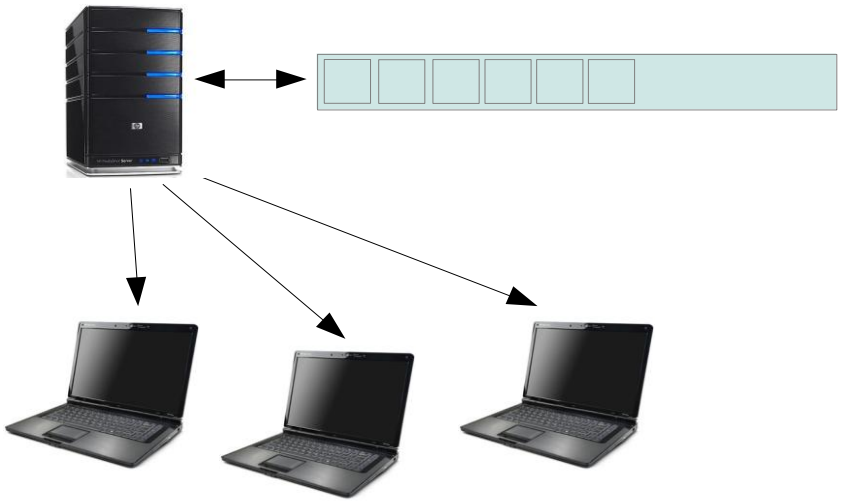

図 1 タスクファイルが配信される様子

これらの一連の流れをフレームワークとして実装し、プロジェクトの雛形は bin ファイルを実行することで自動生成されるようにした。

3. 2 タスクファイルの記述

タスクファイルは JavaScript で記述される。基本的に分散処理を意識せずに書いた JavaScript のファイルをほぼそのまま置くだけで、フレームワーク側でタスクを分割し、処理される。変更点は、ループの中身を関数オブジェクト化し、関数オブジェクトと引数を登録するところである。単純な処理であればそのファイルだけで完結するが、処理が長くなり分割したい場合は、外部ファイルを importScript で読み込むようにする。また、ブラウザでのバックグラウンド処理の前後に UI スレッドで処理を実行したいときがある。たとえば、外部サーバのリソースの読み込みや、DOM の操作をしたいときなどがそうである（バックグラウンドのスレッドではブラウザの一部の組み込みオブジェクトにアクセスできない）。その場合は別のクライアントの JavaScript ファイルに記述する必要がある（図 2）。

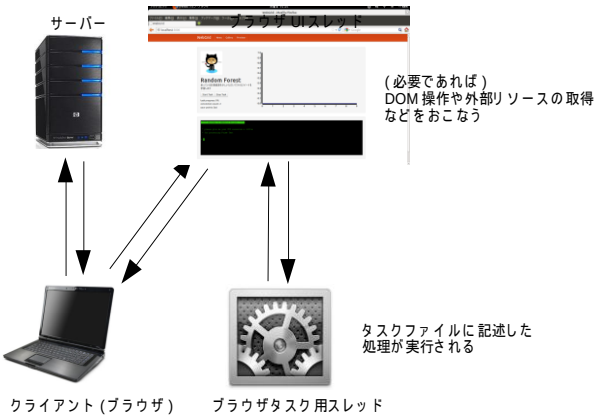

図 2 タスクファイルの記述とフレームワークの動作

4. 従来の技術（または機能）との相違

ボランティアコンピューティングを行うクライアントとしては BOINC と呼ばれるものがある。BOINC でボランティアコンピューティングの参加者を行うためにはまず、会員登録を行って専用ソフトウェアのインストールを行い、各種設定を行う必要がある。本プロジェクトで開発したフレームワークでは、ボランティアコンピューティングの参加者は URL にアクセスするだけなので、参加のハードルを下げることが可能である。

また BOINC でのプロジェクトの開発・運用は難しく、開発者に大きな負担を強いているが、本プロジェクトで開発したフレームワークなら、タスクファイルをダウンロードして JavaScript で書いてサーバを起動するだけなので、ボランティアコンピューティングプロジェクトの立ち上げのハードルを下げることが可能である。

5. 期待される効果

ブラウザ上でボランティアコンピューティングが行えるということで、もっとボランティアコンピューティングを身近に感じ、もっと人々に世界中で起こっている問題を認知してもらえるようになることが期待される。

6. 普及（または活用）の見通し

まず本フレームワークの Coffee Script への移植を行う予定なので、ダウンロード環境とドキュメントを整えて誰でも使えるようにしていきたい。また JavaScript はブラウザ上でのアプリケーションを開発するためのプログラミング言語として発展してきたため、機械学習や数値計算のためのライブラリが非常に少ない。それらのライブラリを整えるとともに本フレームワークからシームレスに使えるようにすることで、開発者を増やし、コミュニティを大きくしてフレームワークの改善を行い、プロジェクトの実績を増やしていきたいと考えている。

7. クリエータ名（所属）

滝口　健太郎（株式会社ミクシィ）