

JavaScriptにおけるマルチスレッド機能の実現とその応用 Building Multi-Thread Function based on JavaScript Programing

伊藤 正都[†] 大園 忠親[‡] 新谷 虎松[‡]

Masato Ito, Tadachika Ozono, Toramatsu Shintani

1 はじめに

現在、既存 Web ブラウザ上で JavaScript のみを用いて動作するマルチエージェントシステムの作成は困難である。JavaScript のみではマルチエージェントシステム等を作成する為に必要なプログラムのマルチスレッド化が難しい。また、リソース消費が大きい JavaScript プログラムを実行した際に、Web ブラウザがユーザからの操作要求を満たせない場合がある。その際にリソース消費量の大きいロジックをマルチスレッド化することで、リソースの消費量をコントロールすることが可能となる。既存の JavaScript マルチスレッド化手法では JavaScript 以外のプログラムを必要とし、プログラマに JavaScript 以外のプログラミング言語の習得が必要である。JavaScript 以外のプログラミング言語の習得および JavaScript とスレッド機能を提供する言語を結びつける手法の習得は、プログラマの多大な負担となる。

本稿では、JavaScript 以外を用いない JavaScript プログラムのマルチスレッド化手法を提案する。さらに本手法を用いることにより、マルチスレッド機能が必要な Web ブラウザ上で動作する JavaScript マルチエージェントシステムの作成を提案する。

本稿の第2節では、マルチスレッドの特徴および既存 JavaScript のマルチスレッド化プログラミングについて述べる。第3節では本研究で提案する JavaScript におけるマルチスレッド機能の実装方法について述べる。また、第4節では JavaScript マルチスレッドを用いた Web ブラウザ上で動作するマルチエージェントシステムについて述べる。最後に、第5節で本稿をまとめる。

2 既存 JavaScript のマルチスレッド化手法

マルチスレッド機能とは、1つのプログラム中の処理をスレッドと呼ばれる処理単位で複数作成し並行処理することである。複数のスレッドを短い時間で順番に処理することにより、同時に複数の処理をしているように見せかける技術である。マルチスレッドを用いることにより外見上は並行して処理が可能となる為、バックグラウンドでの別処理が可能となる利点がある。同時に共有資源へのアクセス等が行われない様に、共有資源の排他制御機構が必要となる [1]。

JavaScript においてマルチスレッドを実現する既存の手法として、Rhino[2]がある。Rhino は、Java で実装された JavaScript 処理系である。Rhino, XPConnect[3] および XPCOM[4] を用いることにより JavaScript と Java を相互に利用可能となる。Java のスレッド機能を用いマルチスレッドプログラミングが可能となる。Rhino を用いた既存手法では JavaScript のみでマルチスレッド機能を実現できない。そのため、組み込みシステム上で動作する Web ブラウザ等において JavaScript は利用できるが、Java アプレットは使用できない環境では Rhino を用いたマルチスレッド機能が利用できない。また、JavaScript プログラマが Java を新たに習得する必要があり、プログラマの多大な負担となる。

3 JavaScript におけるマルチスレッド化手法

3.1 マルチスレッドプログラミングにおける制限

JavaScript では標準でスレッド機能を利用できない。Cooperative Multi task thread を用いスレッド機能を実現する為に JavaScript プログラムのプログラミングに対して制限を行う必要が生じる。制限として具体的に以下の4点が挙げられる。

- スレッド内では別のスレッドを呼び出さない
- スレッドを切り替える時間内に終了しない処理は行わない
- スレッド間で使用する変数はグローバル変数と相互に変換可能である
- スレッド内で使用する変数は使用前にグローバル変数リストに格納する

3.2 スレッドの用意

JavaScript では標準でスレッド機能が無いため、JavaScript プログラマがスレッドの代用となる代替関数を作成しなければならない。作成する代替関数内において使用する変数はスレッド処理をする前にグローバル変数から取得し、代替関数が終了する際にプログラム全体で使えるグローバル変数に値を代入する。また、再度代替関数の実行が必要な場合には、代替関数の実行待ち状態を保持しているキューに再度実行する代替関数をプッシュする。

3.3 スレッドの実行、実行順序

図1にスレッド処理実行時の様子を示す。代替関数の実行待ち状態を保持するキューを配列で用意し、待ち状態の代替関数をスケジューラを用いて実行する。ス

[†]名古屋工業大学 知能情報システム学科

[‡]名古屋工業大学 大学院 情報工学専攻

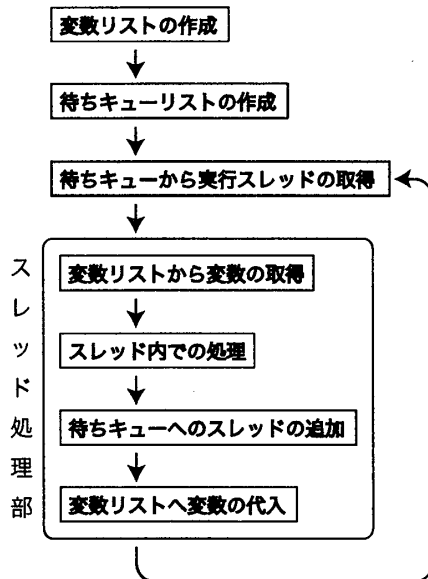


図1: マルチスレッド実行イメージ

スケジューラの例としてラウンドロビンが挙げられる。マルチスレッドで処理を行う部分では、代替関数の実行待ちキューをシフトすることで先頭にある待ち状態の代替関数を実行する。代替関数を実行する際に、インターバルを置くことでリソースの使用を制限することができる。各代替関数に優先順位を付与することで、複数のマルチスレッド処理部の処理順序を操作することが可能となる。

4 JavaScript マルチエージェントシステム

本研究の応用として、JavaScript を用いたマルチエージェントシステムがあげられる。

マルチエージェントシステムとは、多数の自律的に行動するエージェントから構成されたシステムである。エージェントは個々の環境を自覚し個々の問題を解決するように行動をする為、エージェントを集中的に管理する仕組みは存在しない。システム全体の振る舞いはエージェント同士の相互作用によって決定され、各エージェントの問題に影響を与える [5]。

特に本研究では、以下のエージェントシステムへ適用が考えられる。

- Web ページの先読みシステム

図2に本システムの概要を示す。ユーザが閲覧中の Web ページにエージェントを配置する。現在表示している Web ページからリンクされている Web ページをエージェントがバックグラウンドで読み込み、Web ブラウザのキャッシュに格納する。Web ブラウザに閲覧中のページからリンクされているページをキャッシュするにより、閲覧ページを移動する際に発生する読み込みの待ち時間を短くすることができる。

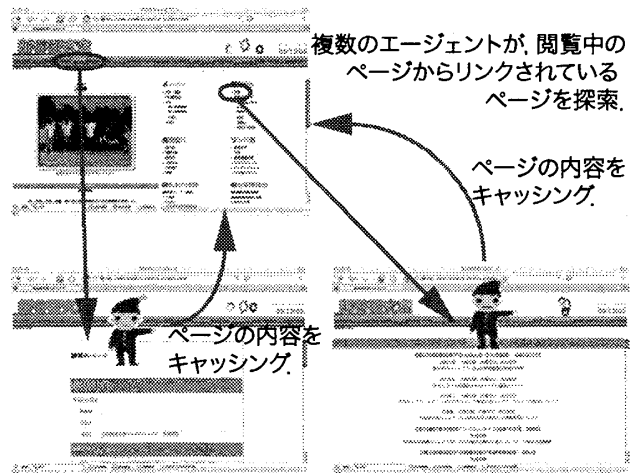


図2: Web ページの先読みシステム

- オークション入札支援システム

オークションサイトを閲覧中のユーザが、検索条件で出品物を絞り込む。エージェントを用いて絞り込まれた出品物のうち、検索条件からユーザの希望に一番適した出品物をハイライト表示する。ハイライト表示されることでユーザは、より直感的な商品選択が可能となり商品比較をする際の時間を節約できる。

5 おわりに

本稿では、Web ブラウザ上で動作する JavaScript プログラムのマルチスレッド化、および応用としてマルチスレッド化を用いたマルチエージェントシステムの提案について述べた。JavaScript プログラムをマルチスレッド化することで以下のような利点が挙げられる。(i) Web ブラウザ上で動作するマルチスレッド機能が必要な JavaScript アプリケーションが作成できる。(ii) JavaScript プログラムのマルチスレッド化によるパフォーマンスのチューニングができる。例として、多倍長演算等の高負担時のリソース消費をマルチスレッド化することによってコントロールすることが可能となる。

参考文献

- [1] 結城浩: Java 言語で学ぶデザインパターン入門 マルチスレッド編, ソフトバンクパブリッシング, 2002
- [2] Rhino
<http://www.mozilla-japan.org/rhino/>
- [3] XPCConnect
<http://www.mozilla-japan.org/scriptable/>
- [4] XPCOM
<http://www.mozilla-japan.org/projects/xpcom/>
- [5] 伊藤孝行, 新谷虎松: マルチエージェントシステムのための実装技術とその応用, 人工知能学会誌 Vol.16 No.4