# キュー実行方式に基づくキューJava 仮想マシンの実現

茂野 収[†] 繁田 聡一[‡] Ben A. Abderazek[‡] 吉永 努[‡] 曽和 将容[‡]
電気通信大学 情報工学科[†] 電気通信大学 情報システム学研究科[‡]

## 1．はじめに

Java は,実行環境がプラットフォームに依存しないという長所によって広く普及してきているが,実行速度が遅いという問題点がある.Java の実行速度を向上させる１つの手法として,バイトコードレベルの並列性を抽出する研究が行われている.我々は Java が採用するスタック計算モデルをキュー計算モデルに置き換えた QJava を提案している[1].本研究ではこの QJava を実行する実行環境[2],[4]として QJava 仮想マシン(QJVM)を実装した.

## 2．キュー計算モデル

キュー計算モデルとは,キューと呼ばれる FIFO(first in first out)のデータ構造を計算結果の格納場所として用いて計算を行う計算モデルである.Java で採用されている,スタックと呼ばれる LIFO(last in first out)のデータ構造を計算結果の格納場所として用いるスタック計算モデルと対照的な計算モデルである.

図１は( a + b )/( c ? d )という計算式の構文木である.各ノードが命令に対応し,ノード間のエッジはデータの依存関係を示している.演算子が書かれたノードは演算命令に対応し,変数が書かれたノードはデータをロードする命令に対応する.キュー計算モデルでは図１の破線で示されるように,構文木を幅優先探索して命令列を生成する.

図２は図１で生成された命令列を,キューを用いて計算する様子を示したものである.キュー計算モデルでは,データの取り出しはキューの先頭から,データの挿入はキューの末尾から行う.一番上のキューは初期状態の空のキューである.最初の load a 命令でキューの末尾から a というデータが挿入される.初期状態は空なので,このデータがキューの先頭にくる．以下,４番目の load

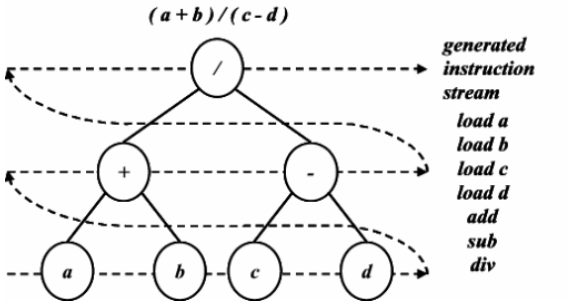


図 1 構文木の幅優先探索による命令列の生成

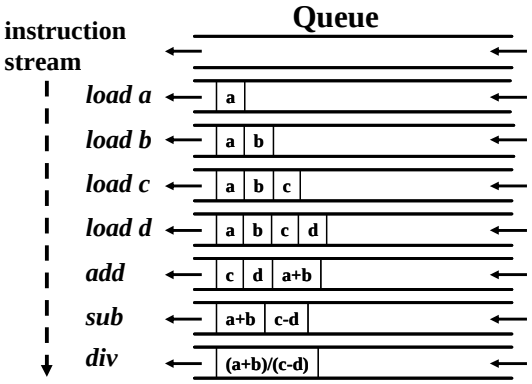


図 2 キュー計算モデルによる命令実行の様子

d 命令まで同様にキューの末尾からデータがキューに挿入される.続いて,５番目の add 命令ではキューの先頭からオペランドである２つのデータ a , b を取り出し,加算を行う.結果の( a + b )はキューの末尾から挿入される.以下同様に sub 命令と div 命令が実行されると,( a + b )/( c ? d )がキューに残り,計算式の結果が得られる.

## 3．QJava バイトコード

QJava バイトコードのプログラムは，プログラミング言語 Java で記述されたプログラムを，キューを用いて計算を行うようにコンパイルすることで生成する．このコンパイラには我々が開発中の QJava コンパイラ[3]を用いた.

### 3.1 命令の追加

QJava バイトコードは基本的に Java のバイトコードと同じであるが,キュー計算モデルを実現するためにバイトコードの定義を変更している.また,キューに挿入されたデータの順序を入れ換

Realization of a Queue Java Virtual Machine on the basis of a queue execution method
Osamu Shigeno[†] Souichi Shigeta[‡] Ben A. Abderazek[‡] Tsutomu Yoshinaga[‡]Masahiro Sowa[‡].
† Department of Computer Science, University of Electro-Communications
‡ Graduate School of Information Systems, University of Electro-Communications

えるためのバイトコードを新たに追加した.

- push 命令 : Java バイトコードではスタックトップへの挿入であるが,QJava バイトコードではキューの末尾への挿入とする.
- dup 命令 : Java バイトコードではスタックトップの値をスタックトップに複製している.QJava バイトコードではキューの先頭からオペランドを１つ取り出して同じ値をキューの末尾に２つ挿入する.
- rotate 命令 : QJava バイトコードにはキューの先頭にあるオペランドをそのままキューの末尾に移動する命令を新たに追加する.この命令はキューに挿入されたデータの順序の入換えが必要な場合に用いる.

### ３.２ クラスファイル

QJava コンパイラが生成するクラスファイルの構造は QJava バイトコードを含んでいる点を除いて, Java のクラスファイルと同様である.QJava バイトコードの実行に必要なメソッドやキューの最大長などの情報が含まれている.

## ４. QJava 仮想マシン

実装した QJava 仮想マシンはオペランドスタックではなくオペランドキューを用い,QJava バイトコードを実行するという点で Java 仮想マシンと異なるが, その他の構成は Java 仮想マシンと変わらない. ガーベジコレクションとマルチスレッドは実装していない. 図３に実装した QJava 仮想マシンの構成を示す.

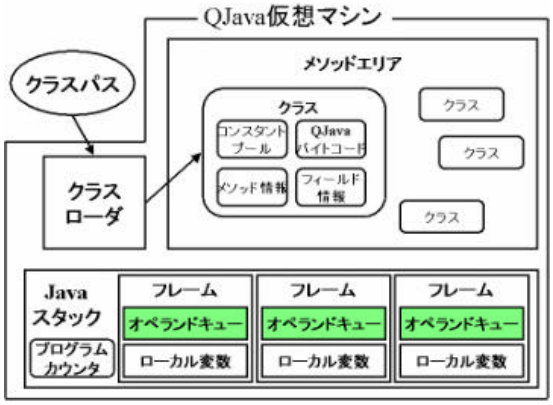


**図 3 QJava 仮想マシンの構成**

QJava では Java と取り出すオペランドの順序逆の順番になる. 演算のオペランドだけでなく, メソッド呼び出しの引数や配列要素の取り出しについてもこの変更を適用する必要がある.

- メソッド呼び出し

Java バイトコードではオブジェクトの参照, 第 1 引数‥第 n 引数の順でオペランドスタックに積み,第 n 引数‥第 1 引数, オブジェクトの参照の順で取り出す.QJava バイトコードではオペランドキューに積む順は同様だがオブジェクトの参照, 第 1 引数‥第 n 引数の順に取り出す.

- 配列要素の取り出し

Java バイトコードでは配列のインデックス, 配列の参照の順にオペランドスタックから取り出す. QJava バイトコードでは配列の参照, 配列のインデックスの順に取り出す.

## ５. 動作確認

QJava コンパイラが生成した QJava バイトコードを実行できることを以下の項目について確認した.

- 四則演算
- 配列を用いた演算
- メソッド呼び出し
- if 分岐, switch 分岐,ループ

実装した QJava 仮想マシンでは以上の演算,命令において正常に動作することを確認した.

## ６. おわりに

QJava バイトコードの実行環境として QJava 仮想マシンの実装を行った.実装した仮想マシンは基本的な動作を行うことができることを確認した.これは QJava バイトコードを実行する環境を実現するという目的を果たせたといえる.

今後の課題として,ガーベジコレクションやマルチスレッドの実装がある.

## 参考文献

[1] 繁田聡一, 王立強, B. A. Abderazek, 吉永努, 曽和将容: “バイトコードレベルの高い並列性を持つ QJava の提案”, 先進的計算基盤システムシンポジウム SACSIS2003, pp.77-80.

[2] 蛯澤明廣, 吉永努, 曽和将容 : “QJava バイトコードの実行環境の構築”, 信学会 情報・システムソサイエティ大会講演論文集 p.42(2001).

[3] L.Wang, B.A.Abderazek, S.Shigeta, T.Yoshinaga, and M.Sowa, “QJAVAC: Queue-Java Compiler Design for High Parallelism Queue Java Bytecode", Proceedings of International Technical Conference on Circuits/Systems, Computers and Communications, Vol.2, pp.900-903 (2003).

[4] 中村禎宏 : “QJava バイトコード仮想マシンに関する研究”電気通信大学電子工学科 平成 12 年度卒業論文(2001).