アプリケーションに特化した合成命令を持つ JavaScript 仮想機械

1190355　野中 智矢　【 プログラミング言語研究室 】

1　はじめに

JavaScript ではメモリ管理を気にせずプログラムを記述できるなど，簡単にプログラミングができる．そのため，組込みシステムでもプログラムを簡単に開発できる．しかし，組込みシステムは一般に，搭載しているメモリが少なく CPU も遅い．そのため，組込みシステム用の JavaScript 仮想機械（VM）は大きくなるのを避けながら高速化する必要がある．

本研究では対象とするアプリケーションで頻繁に実行される命令の組み合わせだけを合成した合成命令を VM に導入することで高速化する．さらに，合成命令を元となる命令のコードの一部と共有することにより，VM の大きさの増加を抑える．また，合成命令は特定のデータ型に特化した処理を行うため，実行が高速になる．本研究では VM の大きさの増加を抑えつつ VM 命令の組み合わせを合成した命令を VM に追加するシステムを開発した．

2　eJS 処理系

本研究では，少ないメモリ上でも動作する組込みシステム向け JavaScript 処理系である eJS[1] を対象とした．JavaScript のプログラムを専用のコンパイラ（eJSC）でコンパイルし，出力された命令列を VM（eJSVM）の上で実行する構成になっている．eJSVM は eJSTK によってアプリケーションに合わせて生成される．eJSVM は対象のアプリーションに特化することで，高速で小さな VM になっている．本研究も eJSTK に VM を特化する方法の一つとして組み込む．

3　提案手法

本研究ではアプリケーション毎に特化した合成命令を生成する．生成する合成命令は定数ロード命令と演算命令を合成した命令にする．

■システム全体像　開発したシステムを図 1 に示す．JavaScript のプログラムをプロファイラに入力し，頻繁に実行される合成命令を特定し，合成命令仕様に出力する．それを元に eJSC では合成命令を使用した命令列を出力し，eJSTK は合成命令を持つ VM を生成する．

■eJSVM　演算命令は，ソフトウェアで実現したレジスタである配列からオペランドを読み出し，そのデータ型に基づいて多段階にディスパッチして，各データ型に対応する処理をする．合成命令では，定数のオペランドを命令から読み出した後，演算命令の処理にジャンプする．そのとき，定数の型が決まっていることを利用して，ディスパッチの途中にジャンプする．

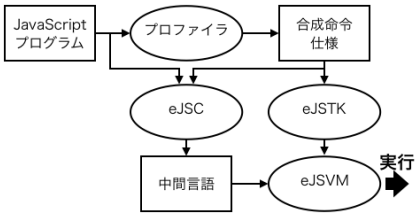

図 1　提案するシステムの全体図

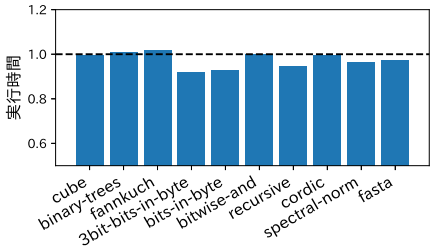

図 2　実行時間

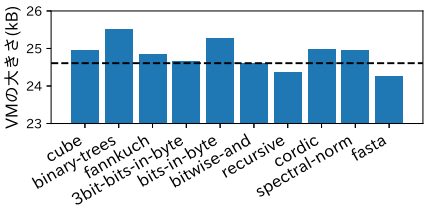

図 3　VM の大きさ

■eJSC　データフローグラフを作成し，各演算命令の到達定義を求める．その演算命令のオペランドの定義元が定数ロード命令だった場合，その演算命令を合成命令に置き換える．合成された定数ロード命令は他で使われていなければ，後の最適化フェーズで除去される．

4　評価

生成した合成命令を対象に，VM の大きさと実行時間の変化を調べるための実験を行った．これらを調べるために SunSpider ベンチマークスイートからいくつかのプログラムを使用した．既存の VM を基準とした実行時間の割合と VM の大きさを図 2，図 3 に示す．実験の結果，合成命令の追加によって実行の高速化が見られ，VM の大きさは 1kB 以下の増加で抑えられた．

5　まとめ

本研究では，VM に合成命令を追加するシステムを開発した．評価の結果，VM の大きさの増加を抑えつつ実行の高速化が実現できた．

参考文献

[1] T.Kataoka, T.Ugawa, and H.Iwasaki. “A Framework for Constructing Javascript Virtual Machines with Customized Datatype Representations.” SAC ’18, pp. 1238–1247, 2018.