演算子順位法に対する Web ベース学習支援システムの開発

久保俊貴[*1], 香川考司[*2]
*1 香川大学大学院工学研究科 *2 香川大学創造工学部

Development of a Web-based Learning Support System for Operator-Precedence Parsers.

Toshiki KUBO [*1], Koji KAGAWA[*2]
*1 Graduate School of Engineering, Kagawa University
*2 Faculty of Engineering and Design, Kagawa University

著者らの所属する学科で開講されている「コンパイラ」で学ぶ演算子順位法では，シフト，還元といった計算過程を理解することが重要である．シフト，還元は後に学習する LR 構文解析でも用いるため正しい理解が必要である．そこで本研究では，計算過程をスタックや JSAV を用いた構文木の変化と共に確認できる，演算子順位法に対する Web ベースの学習支援システムを開発した．既学者に対しては，可視化のシステムが学習の役に立つという評価が得られた．初学者に対しても同様の評価が得られるか，確認する必要がある．現在は計算部分を JavaScript で行っているが，将来は，ユーザが作成した C 言語の構文解析プログラムをサーバ側で JavaScript に変換し，それをブラウザで実行することで，ユーザの作成したプログラムによる計算と見比べられるシステムを開発する．

キーワード: コンパイラ，演算子順位法，JSAV，emscripten, Web ベース，学習支援システム

1. はじめに

演算子順位法とは，演算子同士に順位をつけ，順位の高いほうを先に計算するという構文解析法の１つである．比較的平易なアルゴリズムであって，人手で記述でき，その結果，ほとんどどんなプログラミング言語でも実装できる，構文解析の実行時に演算子を追加したり変更したりもできるため，ユーザ定義の結合性と優先順位を持つ中置演算子に対応できる，などの特徴を持つ．このため，LR 構文解析法や再帰下降構文解析法などの他の構文解析法と組み合わせることも行われる．これを学習する際に重要になることが，シフト，還元といった計算過程を理解することである．これは，後に学習するより実用的な構文解析法である LR 構文解析にも同様の計算を用いるからである．しかし，演算子順位法の学習では，演算子の構文規則から演算子順位行列を作り，構文解析プログラムを作成するため，理解するべき事柄が多く，また，誤って理解する場合もある．また，既存のコンパイラ関連の学習支援システムはいくつかあるものの，演算子順位法の学習をサポートするシステムがほとんどない．そこで，演算子順位法に対する学習支援システムが必要になる．また，学習支援システムを開発する際には学習者の利用しやすさを考え Web ベースで開発するべきだと考えられる．そのため本研究では，演算子順位法に対する Web ベースの学習支援環境の開発を行う．

本研究ではこれを，JavaScript を用いて，演算子順位表と還元時のアクションの編集をブラウザ側で行えるようにし，構文木とスタックの変化，計算過程の説明

を表示する機能を実装する．その後，学習者の作成した C 言語の構文解析プログラムをサーバ側で JavaScript ファイルに変換し，ブラウザ側で実行できるように改良する．計算過程と構文木の出力には，アルゴリズム可視化のためのライブラリである JSAV[1] を用いる．

本論文の構成は以下の通りである．2 章では本システムで利用した JSAV について，3 章ではシステム実装，4 章ではまとめ，5 章では今後の課題について述べる．

2. JSAV

JSAV とは，The JavaScript Algorithm Visualization Library の略称である．これは OpenDSA プロジェクトの一部であり，OpenDSA[2] とはデータ構造とアルゴリズム，形式言語，有限オートマトン，プログラミング言語など，様々なコンピュータサイエンス関連の授業をサポートするための教材と基盤である．OpenDSA では，プログラミングのチュートリアルから，ポインタ，再帰などを，以下の図 1 のような，可視化されたプログラムの流れや，アルゴリズムの過程を見ながら学習することができる．本研究では，これらの中の木を可視化するシステムを用いる．

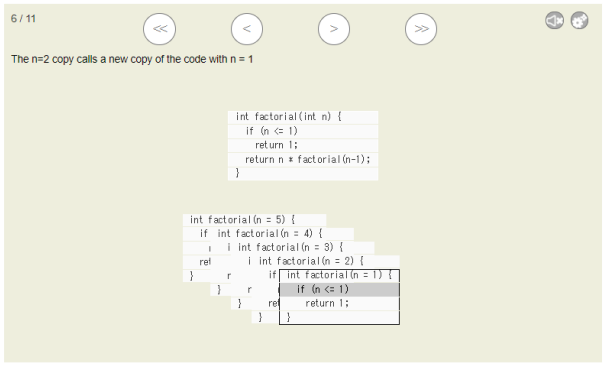


図 1 再帰の教材

3. システム実装

本章では実装したシステムの概要を説明したのち，各機能について詳しく説明する．

3.1 システム概要

本システムは，「コンパイラ」の演習授業で用いることを想定し，演習で行う 1 つ 1 つの問題に対応したページを実装する．本システムの概要を以下の図 2 に示す．

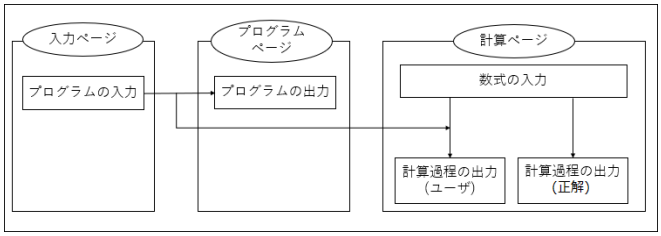


図 2 システムの概要

最初に，演算子順位行列と実装する演算子の還元時のアクションを入力する．そして，入力内容を JavaScript に保存する．入力内容はプログラムページで確認を行うことができる．次に計算ページでは，保存された内容をもとに計算を行う．この際に，ユーザが定義した計算ルールによる計算と，各問題の正解による計算を両方行い，比較できるようにする．

3.2 各ページの実装

3.2.1 入力ページ

入力ページでは，演算子順位行列を示す多次元配列である prec_table と演算子の計算内容を定義する関数である binary_op()の必要部分のみ編集を行う．これは，「コンパイラ」の演習では，演算子順位法による計算プログラムの本体は実装積みのものを利用し，学習者には必要な部分のみを編集させるためである．入力ページのスクリーンショットを以下の図 3 に示す．

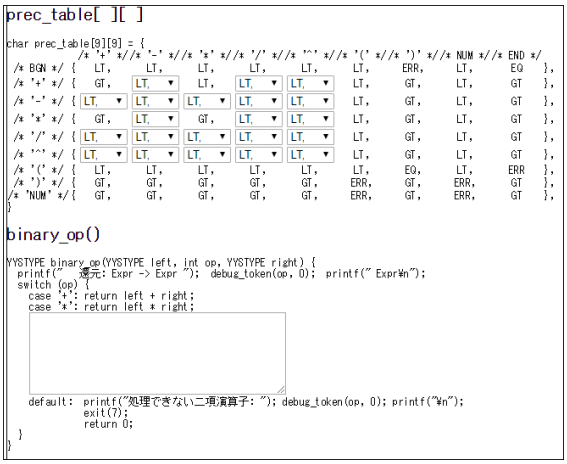


図 3 入力ページ

prec_table 内で定義されている LT, EQ, GT, ERR

は比較した左右の演算子の優先度の強弱を示し，それぞれ，「左を優先」，「優先度は同じ」，「右を優先」，「エラー」を示す．定義された内容は，JavaScript 内の変数に保存する．prec_table の内容は，JavaScript 内に実装済みの配列に新しい演算子の情報を追加していく．binary_op の内容は，関数の内容を文字列として保存し，計算ページでは eval 関数を用いて利用する．ただし，C 言語の場合，累乗の計算は pow 関数を用いればよいが，JavaScript の場合は Math.pow 関数を用いなければならないため，内容を一部書き換えなければならない．そのため保存する際に，「pow」を「Math.pow」に置換する．また，計算プログラムでは，計算結果を関数から受け取るのではなく，関数内で大域変数に結果を代入するため，「return」を 「answer＝ 」に置換する．answer が計算結果を保存する大域変数である．

3.2.2 プログラムページ

プログラムページは，入力した内容を出力する．このページは，作成したプログラムの内容を学習者自身の C 言語環境で実行したい際に，そのまま保存して利用できるように実装した．

3.2.3 計算ページ

計算ページでは，入力された計算式を，定義した計算ルールを元に計算をし，その計算過程を出力する．その際に，スタックの様子や演算子順位行列の比較箇所，構文木の生成過程も出力する．計算ページのスクリーンショットを図 4 に示す．

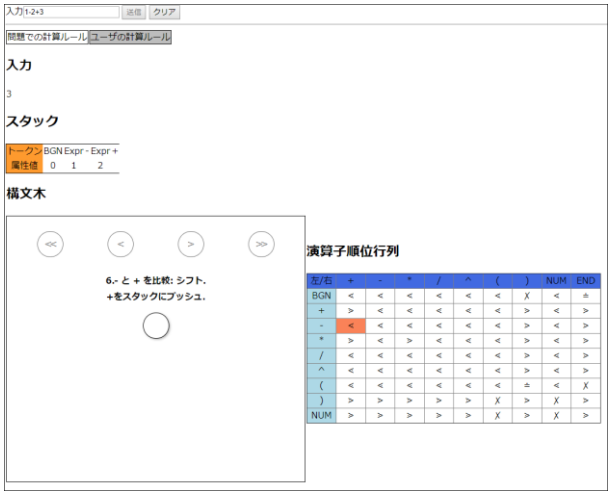


図 4 計算ページ

計算過程のステップは JSAV の構文木システム内のボタンで行う．このシステムは，一度すべてのステップを保存した後に，各ステップを確認できるものなので，一度計算をすべて行ってから出力する必要がある．また，このシステムとスタックの様子と演算子順位行列の比較箇所の過程は共有できていないので，各ステップのスタックの様子と演算子順位行列の比較箇所は配列に保存し，構文木システムのステップに合わせて配列を読み込み，その内容に合わせて変化させる．同様の処理を，正解での計算ルールでも行う．ユーザによる計算ルールによる計算過程と正解での計算ルールによる計算過程はページ上部のボタンで切り替えるようにする．なお，これは問題演習で用いるため，正解での計算ルールによる計算過程のうち，演算子順位行列は表示しないようにする．

4. まとめ

本論文では，「コンパイラ」学習者のための，演算子順位法に関する Web ベースの学習支援システムの開発を行った．ユーザが入力した C 言語プログラムの内容をもとに計算を行い，その計算に伴い構文木，スタックの変化，計算過程を表示するプログラムの作成を行った．また，3 名の学生に試用の上，自由記述による評価をしてもらった．可視化のシステムが学習の役に立つという評価が得られたが，人数が少なく，また既学習者であるため，今後，初学者に対しての評価を得る必要がある．

5. 今後の課題

5.1 C プログラムの実行

計算部分を JavaScript で実装しているため，C 言語プログラムを実際に実行できるようにする．サーバ側で C 言語ファイルを JavaScript ファイルに変換する際には Emscripten[3]を用いる．ただし，ただコンパイルして実行するのではなく，計算過程を出力しなければいけないため，ユーザが書く構文解析プログラムには，一部の内容を指定する必要がある．

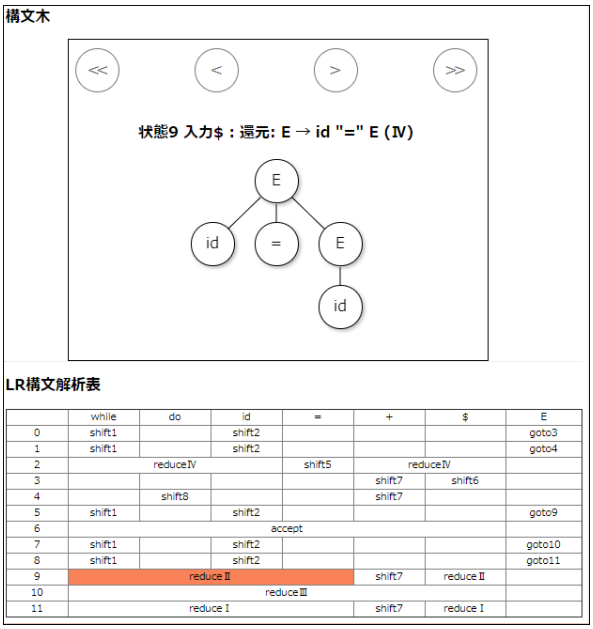
具体的には，C 言語プログラム中の printf 関数が呼び

出されたタイミングで，本論文で実装した JavaScript 側の関数を呼び出すことにより，実装できると考えられる．

5.2 LR 構文解析への対応

C 言語プログラムを実行できるようになれば，本システムを応用して，ユーザの作った Bison と Flex のファイルをサーバ側でコンパイルし，それをブラウザで実行することで，LR 構文解析に対しても同様の学習支援システムを開発する．本研究で開発したシステムのうち，計算過程，スタックの変化を出力する機能は同様に，構文解析過程，スタックの変化として利用できる．そして，演算子順位行列の部分を LR 構文解析表にすれば，LR 構文解析についても同様のシステムを開発できると考えらえる．LR 構文解析表は，受け取った Bison のプログラムをコンパイルする際に，-v オプションをつけることで生成し，可視化する．完成イメージを以下の図 5， 6 に示す．

図 5 LR 構文解析ページ 1



	while	do	id	=	+	$	E
0	shift1		shift2				goto3
1	shift1		shift2				goto4
2	reduceⅣ			shift5	reduceⅣ		
3					shift7	shift6	
4		shift8			shift7		
5	shift1		shift2				goto9
6	accept						
7	shift1		shift2				goto10
8	shift1		shift2				goto11
9	reduceⅡ				shift7	reduceⅡ	
10	reduceⅢ						
11	reduce I				shift7	reduce I	

図 6 LR 構文解析ページ 2

謝辞

本研究は JSPS 科研費 15K01075 の助成を受けたものです．

参 考 文 献

(1) Karavirta, V. and Shaffer, C. A.:“JSAV: the JavaScript algorithm visualization library”, Proceedings of the 18th ACM conference on Innovation and technology in computer science education, pp.159-164 (2013)

(2) Fouh, E., Karavirta V., Breakiron, D. A. et al.:“Design and architecture of an interactive eTextbook the OpenDSA system”, Science of Computer Programming 88, pp.22-40 (2014)

(3) Zakai, A.:“Emscripten: an LLVM-to-JavaScript compiler”, Proceedings of the ACM international conference companion on Object oriented programming systems languages and applications companion, pp.301-312 (2011)