FRI-2G.303-1-CCT1-02

BOTTOM-UP PARSING SIMULATOR FOR CONTEXT-FREE SIMPLE PRECEDENCE GRAMMARS [2]

Assist. Prof. Tzanko Golemanov, PhD
Department of Computer Systems and Technologies,
"Angel Kanchev" University of Ruse
Tel.: +359 82 888 681
E-mail: TGolemanov@uni-ruse.bg

Assoc. Prof. Emilia Golemanova, PhD
Department of Computer Systems and Technologies,
"Angel Kanchev" University of Ruse
Tel.: +359 82 888 681
E-mail: EGolemanova@uni-ruse.bg

Abstract: *Bottom-up parsing, a technique used in compiler design, enhances problem-solving skills by requiring a deep understanding of grammar and syntax rules to construct parse trees from the leaves up to the root. The paper presents the Synt_Up tool, designed for educational projects in compiler construction, aids in teaching language processors, and helps students design parsers for their own high-level programming languages. It facilitates a smooth transition from theoretical concepts to practical application by visualizing the step-by-step bottom-up construction of the syntax tree, complete with detailed documentation of each step of the algorithm's execution.*

Keywords: *Language processors teaching, Compiler construction tools, Simulators*

JEL Codes: C88

ВЪВЕДЕНИЕ

В основата на проверката за синтактична коректност на зададена последователност от символи (токъни) стои построяването на уникално синтактично дърво за тази последователност. Конструкцията на синтактичното дърво задава синтактичния разбор на анализирания низ и се базира на правилата (продукциите) на граматиката на езика. В случай, че синтактичния анализатор (parser) не успее да построи синтактично дърво, това е индикация за наличието на синтактична грешка в низа. Посоката, в която се строи синтактичното дърво (надолу или нагоре) определя и вида на синтактичния анализ: Низходящ (Top-down) и Възходящ (Bottom-up) (Aho, A. V., 2008; Aho, A. V. et al., 2007; Milne, A. C. and McAdam, E. V., 2011).

Докато низходящото строене на дървовидна структура (от корена към листата) не е нещо ново и непознато (такова е дървото на директориите на файловата система), то алгоритмите на възходящия синтактичен анализ често са трудни за усвояване от студентите. Затова тук бихме искали да споделим нашия опит в преподаването на този материал в Русенския университет, като акцентът е Алгоритъмът на Bottom-up парсера, особеностите при реализацията му по време на практическите упражнения, както и основната функционалност на авторския софтуерен инструмент за преподаването му Synt_Up.exe.

BOTTOM-UP СИНТАКТИЧЕН АНАЛИЗ

При низходящия анализ парсерът започва от стартовия символ (аксиома на граматиката) и строи синтактичното дървото надолу, като прилага продукции в пряка посока, докато не достигне крайните листа (токъни от анализирания низ). Bottom-up

[2] Докладът е представен на 25 октомври 2024 г. в секция „Комуникационна и компютърна техника" с оригинално заглавие на български език: СИМУЛАТОР НА ВЪЗХОДЯЩ СИНТАКТИЧЕН АНАЛИЗ ЗА КОНТЕКСТНО-СВОБОДНИ ГРАМАТИКИ С ПРОСТО ПРЕДШЕСТВАНЕ

парсерът започва с от низа с токъните (терминални символи) и изгражда дървото от листата нагоре. Общата идея е да се сканират токъните от входния поток и да се записват в синтактичен стек, в който се изгражда последователност, разпознавана като дясна част на някоя продукция. Когато се намери съвпадение, тази последователност се замества с нетерминалния символ от лявата част на правилото. Този процес изгражда синтактичното дърво нагоре и ако всичко е наред, анализът приключва при разпознаването на дясната част на правилото със стартовия символ.

Главната задача на парсера тук е да намери (***Основна фраза, ОСНОВА***, ***HANDLE***) от текущата сентенциална форма, който съвпада с дясната страна на някое от правилата на граматиката. След това намерената ***ОСНОВА*** (***HANDLE***) се заменя (редуцира) с нетерминалния символ от лявата страна на правилото.

Нека има извод**:** $Z \Rightarrow^{*} \alpha \mathbf{U} \beta \Rightarrow \alpha\gamma\beta$ където:

$\alpha\mathbf{U}\beta$, $\alpha\gamma\beta$ са две сентенциални форми, изведени от аксиомата Z,

а $\mathbf{U} \rightarrow \gamma$ е правило от граматиката

ОСНОВА **(*HANDLE*)** на сентенциалната форма $\boldsymbol{\alpha\gamma\beta}$ е поднизът $\boldsymbol{\gamma}$, който трябва да се замени с нетерминалния символ **U** за да се получи *предходната* сентенциална форма. Така, прилагайки продукциите в обратна посока парсерът се движи нагоре към стартовия символ.

Като цяло, алгоритмите за възходящ синтактичен анализ са по-мощни от низходящите, и не е изненадващо, че необходимите за реализацията им конструкции също са по-сложни. По принцип е трудно да се напише парсер за възходящ анализ ръчно, освен за съвсем тривиални граматики.

BOTTOM-UP ПАРСЕР ПРИ ГРАМАТИКИ С ПРОСТО ПРЕДШЕСТВАНЕ

Реализацията на Bottom-up анализатора при граматики с просто предшестване е подобна на общия модел на възходящия анализатор и дава добра представа за метода, но е опростена и подходяща за използване в учебния процес. Използва се стек за съхраняване на анализирана част на сентенциалната форма и матрица с отношения на предшестване ($\lessdot$, $\doteq$ и $\gtrdot$), които се използват за идентифициране на основната фраза и за управление на действията Изместване и Редуциране. Алгоритъмът има четири основни стъпки, изпълнявани в цикъл:

1) Откриване на **края** на основата - сканира се сентенциалната форма отляво надясно, като се търси отношение $\gtrdot$ между някои два съседни символа. Обработените символи се изместват в стека.
2) Откриване на **началото** на основата - започвайки от върха на стека (края на основата), сентенциалната форма се сканира отдясно наляво, като се търси отношение $\lessdot$ между някои два съседни символа в стека. Запомня се позицията на символа, стоящ в началото на основата. Основата се намира между двете запомнени позиции.
3) Търсене на правило в граматиката с идентична на основата дясна част.
4) Редуциране на основата в стека – замяна със символа от лявата част на правилото.

ИНТЕРФЕЙС И ФУНКЦИОНАЛНОСТ НА СИМУЛАТОРА SYNT_UP

Софтуерният симулатор Synt_up е авторски продукт, предназначен за обучение по темата „Възходящ синтактичен анализ“ в курса „Езикови процесори“ на Русенския университет. Стремежът при разработката му е да бъдат избягнати редица недостатъци и/или ограничения на налични подобни обучаващи системи:

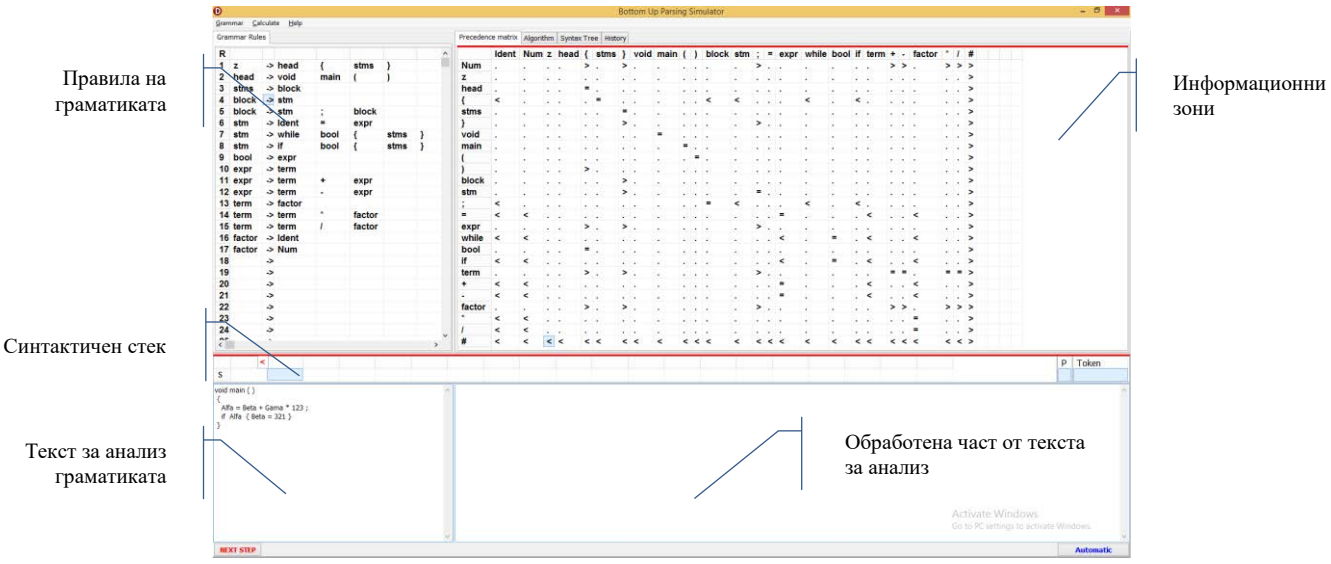
- Реализации като WEB приложения – по принцип с облекчен достъп, но и по-неудобен и ограничаващ потребителски интерфейс (Almeida-Martínez, F. J., Urquiza-Fuentes, J. and Velázquez-Iturbide, J. Á., 2009);

- Неоправдано усложнена функционалност, изискваща продължителен период на обучение за използването на симулатора (García-Osorio, C., Gómez-Palacios, C. and García-Pedrajas, N., 2009; Karkare, A. and Agrawal, N., 2016);
- Предварително фиксирани граматики, с които се извършва експериментирането (Mernik, M. and Žumer, V., 2003);
- Липса на визуализация на възходящото строене на синтактичното дърво (Lonati, V., Mandrioli, D. and Pradella, M., 2011; Jabri, R., 2013).

При отчитане на горните съображения, основните характеристики на представяната разработка са:

- Desktop базирана софтуерна система, без необходимост от инсталационна процедура;
- Опростен потребителски многоезиков интерфейс, разширяващ се при преминаване към нова функционалност;
- Гъвкаво задаване на граматики и низове за анализ;
- Постъпков и автоматичен режим на проследяване на алгоритъма;
- Визуализация на възходящото строене на синтактичното дърво;
- Автоматично генериране на отношенията на предшестване с възможност за експортиране в Excel и последващо използване в студентски проекти;
- Пълно документиране на постъпковото изпълнение на алгоритъма.

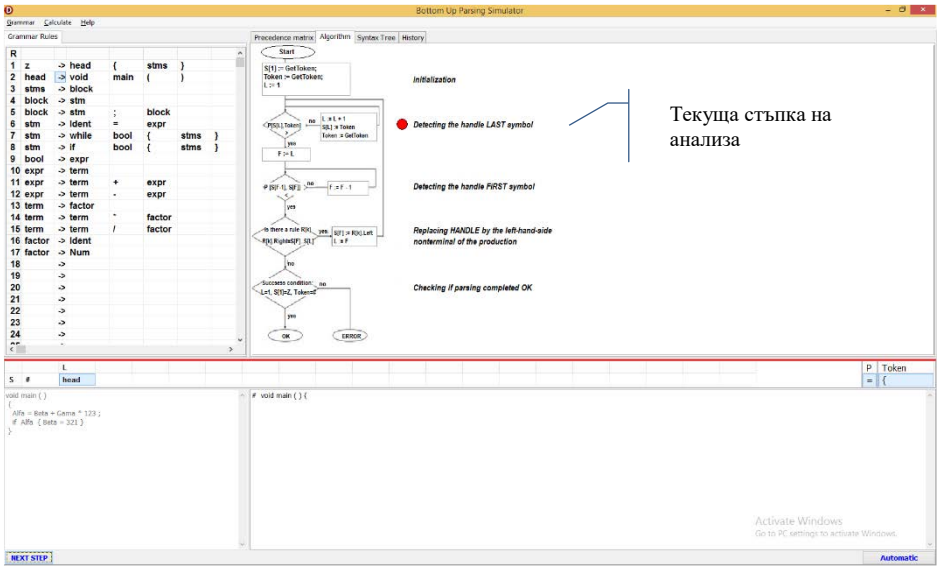
Интерфейсът на приложението включва няколко работни и информационни зони, които се откриват последователно за детайлна визуализация на процеса на анализ (Фиг.1).



Фиг. 1. Интерфейсни зони на симулатора

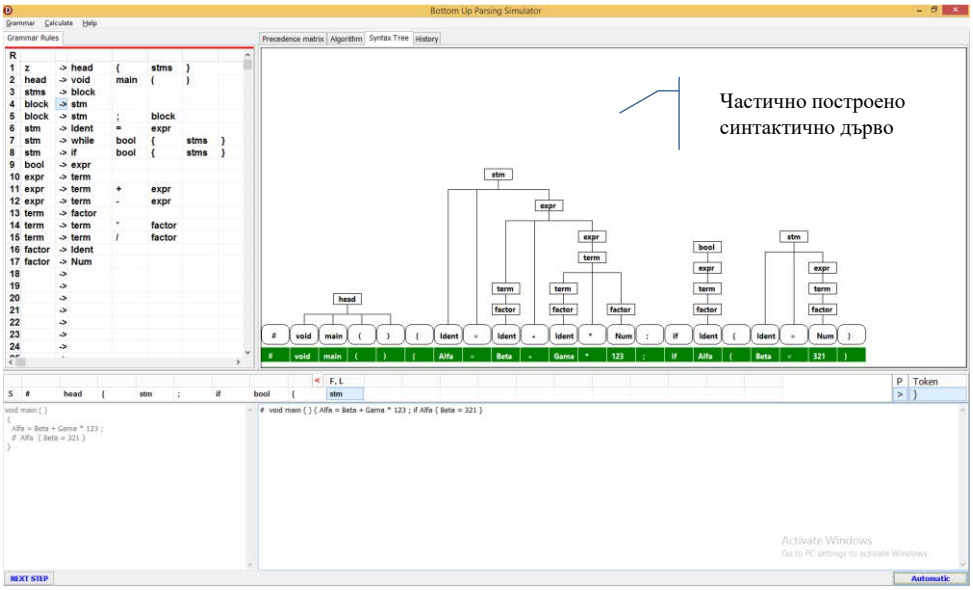
След въвеждане на правилата на граматиката се извършва изчисление и визуализиране на матрицата с отношенията на предшестване (Precedence matrix на фиг.1). Началото на възходящия анализ се дава след попълване на текста за анализ и избор на постъпково изпълнение (бутон NEXT STEP) или автоматичен режим (бутон Automatic). И в двата режима студентите могат паралелно да наблюдават в информационните зони:

- Постъпковото изпълнение на алгоритъма по зададената блок-схема (таб Algorithm на фиг.2) – червената точка маркира текущо изпълняваните операции. Показани са обработките, които се извършат на всяка стъпка и условията за преминаване към следваща;



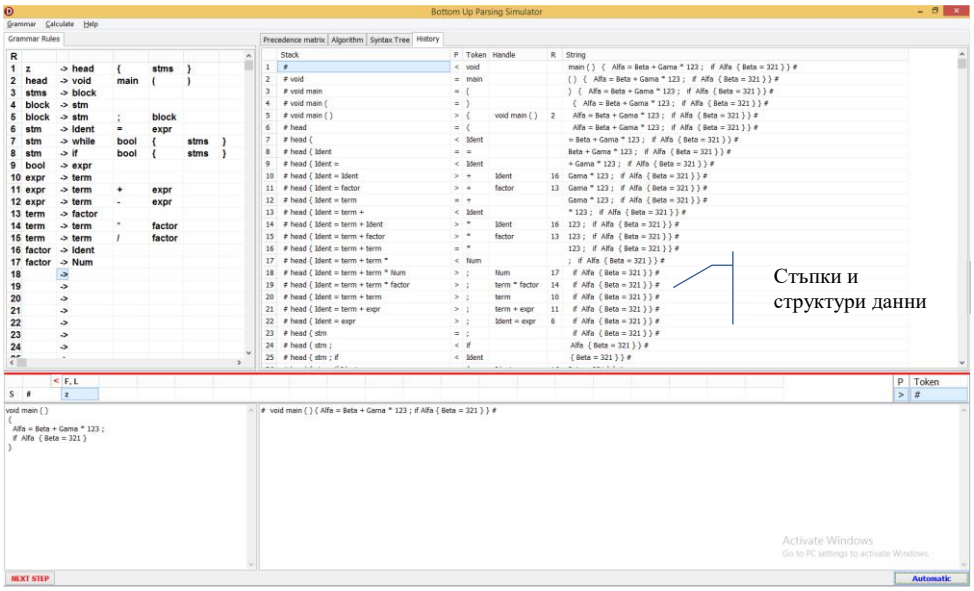
Фиг. 2. Постъпково проследяване на изпълнението на алгоритъма

- Постъпковото възходящо строене на синтактичното дърво (таб Syntax Tree на фиг.3) – за възходящото създаване на дървовидната конструкция се използва нов специализиран, разработен от авторите, софтуерен модул. При всяка редукция на текущата сентенциална форма се изгражда съответстващия участък на синтактичното дърво;



Фиг. 3. Визуализация на постъпковото възходящо строене на синтактичното дърво

- Пълно документиране на стъпките на анализа и текущото съдържание на използваните структури данни (таб History на фиг.4) – помага на студентите да си представят анализа във времето с поетапните модификации на синтактичния стек - намиране на основната фраза на сентенциалната форма, търсене на правило в граматиката и редукцията с лявостоящия нетерминален символ.



Фиг. 4. Пълно документиране на стъпките на анализа
и структурите данни

ЗАКЛЮЧЕНИЕ

- Представени са възможностите на авторския инструмент Synt_Up, специално проектиран да се използва в дисциплината Езикови процесори
- Демонстрира се прилагането на Synt_Up като ефективно средство при изучаването техниките на възходящ синтактичен анализ
- Възможността за интерактивно задаване на граматики и стрингови потоци за анализ, както и наблюдението на постъпковото възходящо строене на синтактичното дърво позволява на студентите да се ориентират по-добре в преподавания теоретичен материал
- Synt_Up се използва от няколко години в катедра „Компютърни системи и технологии“ на Русенския университет и отзивите на студентите са положителни
- Липсата на каквато и да е инсталационна процедура, малкият размер и copyleft-лицензът позволяват Synt_Up да бъде използван без ограничения в произволна MS-Windows среда

БЛАГОДАРНОСТИ

Този доклад се публикува с подкрепата на проект 24-ФЕЕА-01 „Методи и средства за автоматизиран анализ, обработка, съхранение и управление на мултимедийни данни и документи“, финансиран от фонд „Научни изследвания“ на Русенски университет „Ангел Кънчев“.

REFERENCES

Aho, A. V. (2008) ‘Teaching the compilers course’, ACM SIGCSE Bulletin. doi: 10.1145/1473195.1473196.

Aho, A. V et al. (2007) Compilers: Principles, Techniques and Tools, Reading MA. Pearson Education.

Almeida-Martínez, F. J., Urquiza-Fuentes, J. and Velázquez-Iturbide, J. Á. (2009) 'Visualization of syntax trees for language processing courses', Journal of Universal Computer Science, 15(7), pp. 1546–1561. doi: 10.3217/jucs-015-07-1546.

García-Osorio, C., Gómez-Palacios, C. and García-Pedrajas, N. (2009) 'A tool for teaching LL and LR parsing algorithms', ACM SIGCSE Bulletin. doi: 10.1145/1597849.1384360.

Jabri, R. (2013) 'A Generic Tool for Teaching Compilers', Computer and Information Science, 6(2), pp. 134–150. doi: 10.5539/cis.v6n2p134.

Karkare, A. and Agrawal, N. (2016) 'ParseIT: A Tool for Teaching Parsing Techniques', in Proceedings of the 47th ACM Technical Symposium on Computing Science Education. doi: 10.1145/2839509.2850513.

Lonati, V., Mandrioli, D. and Pradella, M. (2011) 'Precedence automata and languages', in Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), pp. 291–304. doi: 10.1007/978-3-642-20712-9_23.

Mernik, M. and Žumer, V. (2003) 'An educational tool for teaching compiler construction', IEEE Transactions on Education. doi: 10.1109/TE.2002.808277.

Milne, A. C. and McAdam, E. V. (2011) 'Compilers, the forgotten subject?', ITALICS Innovations in Teaching and Learning in Information and Computer Sciences. doi: 10.11120/ital.2011.10020032.