情報処理学会論文誌　プログラミング　Vol.13　No.1　22 (Jan. 2020)

発表概要

Ordered Choiceのかわりに Committed Choiceを用いる Guarded PEGの提案

小坂 俊介[1,a)]　前田 敦司[2]

2019年7月26日発表

Fordが提唱した分析的形式文法であるParsing Expression Grammar（PEG）では，選択肢のある規則において解釈に優先順位を設けることで解釈の曖昧さを取り除いている．このPEGを用いた構文解析アルゴリズムの1つにpackrat parsingがある．Packrat parsingではバックトラックに備えて入力文字列やメモ化表をすべて保持しておく必要がため，LL(1)などの他のアルゴリズムよりメモリ空間が必要になるうえ，解析に時間がかかる．しかし，もし解析の途中でバックトラックの可能性がなくなった場合，そのときの解析位置から巻き戻す必要がなくなり，すでに解析した位置に関連付けられたメモ化表は削除することができる．本発表では，PEGの順序付き選択をより制限のある形に置き換えた記法であるGuarded PEG（G-PEG）を提案する．この制限の考え方はcommitted choice言語とcut演算子に基づく．G-PEGでは順序付き選択の代わりに選択のネストを制限する「ガード付き選択」を用いる．ガード付き選択は選択のネストを許さないガード式を持ち，それがマッチした選択肢を解析して他の選択肢を捨てる．これによりメモ化表を引くことなく，バックトラックの情報がたかだか1つで構文解析が可能となる．

Presentation Abstract

Guarded PEG: Parsing Expreesion Grammar with Committed Choice in Place of Ordered Choice

Shunsuke Osaka[1,a)]　Atusi Maeda[2]

Presented: July 26, 2019

Parsing expression grammar (PEG) is an analytic grammar introduced by Ford. PEG eliminates ambiguity from grammars by ordered choices. Packrat parsing is a class of recursive-descent parsing algorithm with PEG. The feature of this algorithm is that it memoizes all intermediate results, and will backtrack when parsing fails. Therefore, this algorithm requires holding all input strings and memoization table for backtracking, so it uses more memory space and takes more time in comparison with other algorithms like LL(1). However, when there is no further alternative to backtrack, the parser can delete memoization table entries associated with previous input positions. In this presentation, we propose Guarded PEG (G-PEG), a notation that replaces PEG's ordered choice to more limited form. The idea of the limitation comes from committed choice language and cut operator. G-PEG has 'guarded choice' instead of ordered choice. Guarded choice has guard expressions. If a guard expression matches, the parser will choose the corresponding alternative and discard the other alternatives. We further put a condition for guarded choices that their guard expression must be simple (that is, which does not contain guarded choice). This condition guarantees that backtracking never occurs while evaluating a guard expression. So maximum backtrack depth is at most 1, and after evaluating a guard expression, we can discard all backtrack information.

[1] 筑波大学大学院システム情報工学研究科コンピュータサイエンス専攻
Department of Computer Science, University of Tsukuba, Tsukuba, Ibaraki 305–8573, Japan

[2] 筑波大学システム情報系情報工学域
Department of Information Engineering, Faculty of Engineering, Information and Systems, University of Tsukuba, Tsukuba, Ibaraki 305–8573, Japan

[a)] osaka@ialab.cs.tsukuba.ac.jp