様　式　Ｃ－１９、Ｆ－１９－１、Ｚ－１９　（共通）

科学研究費助成事業　　研究成果報告書

令和　６ 年　 ６ 月 １７ 日現在

機関番号：３２６７５
研究種目：基盤研究(C)（一般）
研究期間：2018～2023
課題番号：１８Ｋ１１２４７

研究課題名（和文）言語処理系のプログラムの例示に基づく実装法の研究

研究課題名（英文）A Study on Realization of Language Processors by Providing Program Examples

研究代表者
佐々木　晃（Sasaki, Akira）

法政大学・情報科学部・教授

研究者番号：９０３９６８７０

交付決定額（研究期間全体）：（直接経費）　　3,300,000円

研究成果の概要（和文）：新規プログラミング言語の開発は、開発者の知識と職人的技術に頼って行われることが多い。本研究では、得たい言語の設計があいまいでも、行きつ戻りつしながら「探索的」に言語設計と実装を行える手法の開発を目標とする。言語開発において開発者は、(1) 目標とする言語によるプログラム例を構想する、(2)これに基づいた抽象化を行う、これらを繰り返すことで言語設計を得る。本研究ではこの暗黙のプロセスを明示的にする手法を明らかにし、言語開発の負担軽減を目指した。具体的には、言語設計者が複数のプログラム例を提示し、これらの例を解釈、実行可能な言語処理系の実装を効果的に導く手法を提案した。

研究成果の学術的意義や社会的意義
本研究では、例文を用いた段階的・対話的アプローチによるプログラム言語の構文構築手法を提案した。本手法では、開発者が新規言語の「例文」の一部をシステムに与え、システムは言語の構文を推論し開発者に提示する、この作業を繰り返す対話的なアプローチを採用した。対話的な手法によってシステムが言語設計の一部である構文を推論する点、また構文解析法や形式文法の違いにとらわれずに構文を構築することが可能となる点が新規である。また、応用研究として、例題を用いて汎用言語のサブセット言語を得るための研究を行った。本研究においては、言語サブセットを例題から生成する手法が新規的である。

研究成果の概要（英文）：In this research, we study a development method of programming languages through exploratory approaches. Generally, language development starts from defining a grammar of the language and its semantics as a specification of the language, which requires complicate tasks because of the nature of programming languages. The main aim of this research is to obtain language specification and its implementation in an efficient way. Our proposed approach takes interactive and exploratory style in the sense that the development proceeds with interleaving the two processes: (1) developer provide sample source code as examples to the system, and (2) the system infer the language specification and expose to the developer.

研究分野：プログラミング言語

キーワード：プログラミング言語　言語処理系　開発環境　言語推定

様　式　Ｃ－１９、Ｆ－１９－１、Ｚ－１９（共通）

１．研究開始当初の背景

プログラミング言語の実現法の研究は古くからなされ、生成系による自動生成のアプローチが確立されるなど、その成果は実用的な技術として利用されている。一方で、言語開発の探索的側面にはあまり目を向けられていない。統合環境やツールチェーンによって設計から実装を得るサイクルを早める手法は、一定の開発効率向上を実現したが、言語の設計から実装を得る流れをスムーズにした結果としての効率化であり、設計における探索的な側面を本質的に解決していない。そこで、本研究では「行きつ戻りつ」を繰り返す探索的なスタイルでの言語開発を可能とする効果的な手法を追求する。

２．研究の目的

新規プログラミング言語の開発は、開発者の知識と職人的技術に頼って行われることが多い。原因として、言語設計が決まらないまま処理系の実装が行われる場合が多いため、設計に十分な検討を重ねた後に実装を集中的に行う従来のソフトウェア構築法は適用しにくい点が挙げられる。そこで本研究では、言語設計があいまいでも、行きつ戻りつしながら「探索的」に言語設計と実装を行える手法の開発を目標とする。

３．研究の方法

言語開発は、言語仕様としての構文と意味を定義することから始まるが、この定義は簡単に得られるものではない。開発者は(1) 目標とする言語によるプログラム例を構想する(2)これに基づいた抽象化を行う、これらを繰り返すことで定義を得る。本研究ではこの暗黙のプロセスを明示的にする手法を明らかにし、言語開発の負担軽減を目指す。具体的には、(a) 言語設計者が複数のプログラム例を提示し、これらの例を解釈、実行可能な言語処理系の実装を効果的に導く手法を提案する。また、(b)探索的な設計を支援するためにライブプログラミングの手法を利用する。

４．研究成果

本研究の研究期間内で得られた研究結果のうちの主要な成果、「例文を用いた段階的・対話的アプローチによる構文構築手法」および「例題プログラムを用いた、汎用プログラミング言語に対する言語機能制限に関する研究」について詳述する。

[4.1 例文を用いた段階的・対話的アプローチによる構文構築手法]

本研究では構文構築を段階的・対話的に行える手法を提案する。構文の構築順を定め段階的に構文を構築させることで、手戻りの防止など構文構築の妨げとなる要因を排除する。また、1 つ 1 つの生成規則を作成する際、開発しようとする新規言語（以下、「新規言語」と呼ぶ）の入力例文から構文規則を提案し対話的に編集させることで、構文解析法や形式文法の違いにとらわれずに構文を構築することができる。

文脈自由文法 G をタプル G=(N、T、P、S)で表す。N は非終端記号の集合、T は終端記号の集合、S は開始非終端記号、P は生成規則の集合で $P \subseteq (N \times V^*)$で表現できる。V は $N \cup T$ で表現される文法記号列である。生成規則 $A \rightarrow \beta$ は$(A、\beta) \in P$ と等価である。以降、本研究での依存とは$(A、\beta) \in P$ の β に B が現れる時「A は B に依存する」と定義する。

非終端記号の依存関係や演算子の優先順位を考えると効率的な構文の構築順はある程度決まってくる。しかし、1 つの非終端記号が複数の非終端記号に依存している場合や依存している非終端記号がさらに他の非終端記号に依存している場合など依存関係は複雑になることが多い。これを意識しながら構文構築を進めることは容易ではなく、作業の妨げになってしまう。そこで非終端記号の分類から構文構築順を導出し、それに沿って作業を行うことで効率的な構文構築を支援できると考えた。例えば、簡単な演算と print 機能を持つ言語 MyLang の非終端記号の分類と依存は図 1 のようになる。ただし、PrimaryExpr は構築済みであるとする。MyLang の構文構築順は数字で示されている通りである。MyLang では PrintStmt が Expr に依存しているため、PrintStmt の前に Expr を作成する必要がある。このように被依存構文を優先して作成することで、段階的に構文構築を行うことができる。

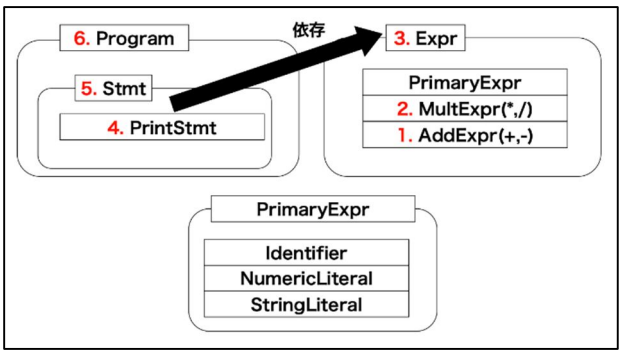

図 1 非終端記号の分類と構築順

次に、対話的な構文構築について述べる。段階的に構文を構築するという特徴によって構文構築中の不整合を抑えながらの開発が可能となるが、1 つ 1 つの構文規則の作成に関しては依然問題が残る。構文解析法や形式文法にはそれぞれルールや制約が存在する。構文規則を作成する際にはこれらのルールや制約を理解し実装を行わなければならない。構文規則の量が多くなればなるほど実装は複雑になり、ルールや制約違反によるバグやエラーが増えてしまう。新規言語の構文を構築する前には、ふつう入力例文が想定されているはずである。本手法では、この入力例文から構文規則例を提案し、対話的に編集を加えていくことで構文を完成させる。構文規則例の提案の際には、ルールや制約に則った形で提案されるため、それらを意識することなく構文構築を進めることができる。

本研究では対話的な構文構築を実現するためプロトタイプシステムを実装した。対話的な操作と GUI 表現は相性がよく、効率的に構文構築を進められると考える。図 2 の①は新規言語の入力例文を入力する「エディター部」である。エディター部内の入力欄は任意個作成が可能になっている。図 2 の②は「ツールバー」である。ツールバーでは作成する構文の予約語、構文の分類、構文名を指定することができる。generate ボタンを押下することで構文規則例が提示される。右端の setting ボタンを押下することでカテゴライズウィンドウが現れる。カテゴライズウィンドウでは構文の分類に関する操作と構文に対する依存性の注入を行うことができる。図 2 の③は編集領域となっている。ここでは入力例文から提示された構文規則例に対して編集を行うことができる。

本システムで提示されるのは PEG 構文である。 PEG には曖昧さがなく、構文規則の右辺を「 / 」で区切る。これによって構文規則内に明確に優先順位が生まれるため段階的な構文構築に適していると考えた。また、PEG では字句解析器を別途作成する必要がないため実装が行いやすいと考えた。

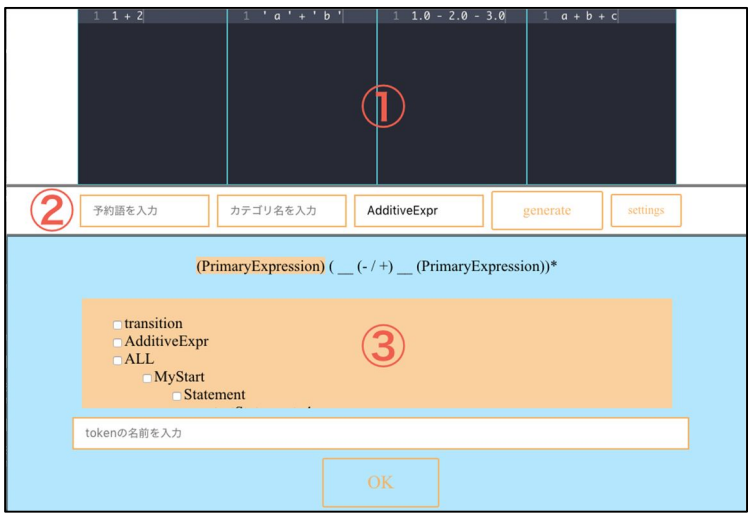

図 2 システム外観

本研究では、プロトタイプシステムを用いて構築した構文に関して(1)構築した構文に基づいて作成したパーサが正しく入力を解析できるか、(2)生成された構文規則が適切な形式であるかの 2 点について実験を行い、正確性について考察を行う。評価にあたって JSON、 Uniform Resource Locator(URL)、SQL サブセットの 3 つの言語の構文を構築した。また著者がこれらの構文を作成するためにかけた時間を計測し、考察を行った。表 1 に構文の規模（構文数)、構文の分類にかかった時間、構文構築にかかった時間を示す。

表 1 言語の規模と構築にかかる時間

	構文数	分類時間	構築時間
JSON	6	3 分	3 分
URL	11	4 分	22 分
SQL	33	6 分	65 分
Tiger	35	14 分	30 分

評価対象の言語は繰り返し構造と依存関係の両方を持つものから選んだ。JSON はデータ記述言語の 1 種である。URL はインターネット上のリソースの場所を特定するための書式である。SQL サブセットは SQL 構文の内、テーブル操作に関するもののみを持った部分的な SQL 言語である。Tiger はコンパイラの教科書で紹介されている小さめな手続き型言語である。(1)の評価では構築された PEG 構文から Packrat Parser を生成し、テスト入力を正しく解析できるかどうかを実験する。JSON、URL、 SQL サブセットの実験では、テスト入力は github 上の ANTLR の文法例リポジトリ内にあるテスト用例文を利用する。SQL の例文にはテーブル操作以外の構文も含まれているため、適宜削除する。Tiger のテスト入力は教科書内のコードをテスト入力とする。(2)の評価では構築された PEG 構文が不自然な形式になっていないかどうかを確認する。

(1)の実験では全てのテスト入力を正しく解析することができた。(2)の実験で不適切な形式で表現されていたのは、URL 言語の「url 構文」であった。url 構文のシンボル列には Optional なシンボル「Token(“:”)、Token(portNum)、Token(“/”)、Token(Path)」が連続して現れる。ユーザの意図を汲むなら、これらのシンボルは「(“:” portNum)? (“/” Path)?」という構文規則に変換されるのが適切である。しかし、実際には「”:”? portNum? “/”? Path?」という構文規則に変換される。システムがどこでシンボルを区切れば良いか判断できないからである。この点は今後の課題として改善が必要である。

[4.2 例題プログラムを用いた、汎用プログラミング言語に対する言語機能制限に関する研究]

プログラミング初学者がプログラム言語を学習する際、その汎用言語の処理系を用いて学習を進める。学習者が誤った記述をした際、処理系は未学習の内容に関連した警告を表示する場合があり、学習者はこれを理解することが難しい場合がある。また学習者が新たな機能に関する例題を解く際、問題の意図から外れた機能を使用してしまい、本来学ぶことが期待されていた機能に触れる機会を逃してしまうことがある。そのため、教育者が学習内容に合わせて汎用言語に含まれる言語機能のうち、学習者が利用可能な機能を制限することにより、学習者が教育者の意図に沿った学習が行えることが有効だと考える。本研究では例題から言語サブセットを自動生成するアルゴリズムを開発し、Python を用いた支援ツールを開発した。

本研究では 2 種類の方法で制限を指定する。言語機能の制限が、指定した構文の使用を許可しそれ以外を禁止する「許可」、指定した構文の使用を禁止・義務化する「禁止・義務化」の 3 つを用いてそれぞれの言語機能の制限を行う。これらの制限情報を用いて学習者が使用する言語サブセットを作成する。

言語サブセットの指定方法のひとつ目は、例題プログラムから得た情報から制限を抽出・指定する方法である。例題プログラムからの制限抽出では、抽象構文木を利用して「許可」の制限の抽出を行う。

図 3 ：ノードの追加例

図 3 は代入式と For 文からなるプログラムの抽象構文木の例である。For 文使用時には左のような木を取得できる。さらに「For 文のイテレータ部分にリスト型を利用すること」を許可する制限を指定する際に、型情報が必要なため、右の図のように情報を取得するノードを木に追加し、実行することで制限の指定・取得を行う。

構文サブセットの指定方法の 2 つ目は、教材提供者が、明示的に制限を指定する方法である。これは、制限を記述した制限ファイルを編集し主に「禁止・義務化」の制限情報を追加し、それらの制限情報から言語機能の使用範囲を決定する。これにより教育者の意図を学習者に伝える支援を行うことができる。

本研究では上述の手法を実現するためのシステム作成を行った。教育者は例題プログラムから制限を抽出し、その後任意の制限を追加する（図４）。学習者は、教育者が上記の方法で作成した制限ファイルを使用して作成した言語サブセットを利用して学習を行うことができる。例題からの制限の抽出や言語サブセット作成、サブセット下での実行のプログラムは python を用いた。図５はシステムの実行結果の例である。この言語サブセットではリスト内包表記の使用を義務化しており正答の内包表記は「data2=[i * 10 for i in data]」となるが、問題を解くため For 文を使用している。そのため内包表記の未使用について警告されている。

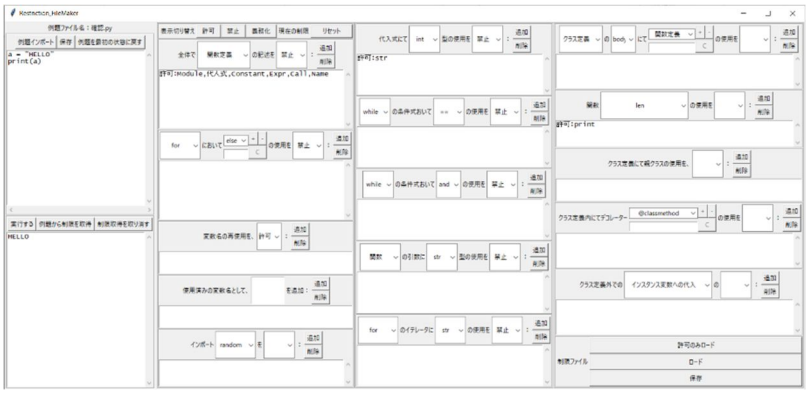

図４：教育者向け制限ファイル作成プログラムの編集画面

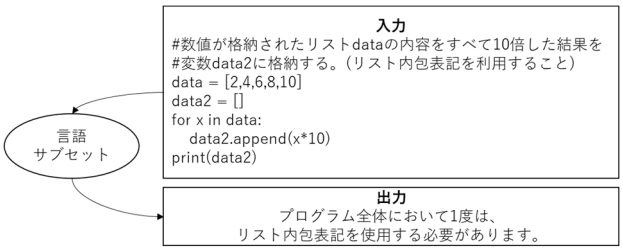

図５：システム実行の例

。

５．主な発表論文等

〔雑誌論文〕　計0件

〔学会発表〕　計28件（うち招待講演　0件／うち国際学会　1件）

１．発表者名 上野康介,佐々木晃
２．発表標題 ブロック言語×VRゲーム開発：プログラミング教育を目的としたVR開発環境の提案
３．学会等名 情報処理学会 第85回 全国大会2ZL-06
４．発表年 2023年

１．発表者名 西崎駿，佐々木晃
２．発表標題 エージェントシミュレーションによるイベント会場における混雑情報の提供手法の研究
３．学会等名 計測自動制御学会システム・情報部門　第27回社会システム部会研究会
４．発表年 2022年

１．発表者名 長谷健汰，佐々木晃
２．発表標題 プログラミング教育支援のための汎用プログラミング言語サブセットの作成手法の検討
３．学会等名 2022年度電子情報通信学会総合大会
４．発表年 2022年

１．発表者名 橋場悠人，佐々木晃
２．発表標題 GPGPUを用いるエージェントシミュレーション開発用フレームワークに関する実用化のための拡張
３．学会等名 計測自動制御学会システム・情報部門　第24回社会システム部会研究会
４．発表年 2021年

１．発表者名
福島和希,佐々木晃

２．発表標題
ライブプログラミングを用いたプログラミング学習支援ツールの試作と提案

３．学会等名
情報処理学会 第83回 全国大会

４．発表年
2021年

１．発表者名
佐藤琢斗、廣津登志夫

２．発表標題
エッジコンピューティングのための軽量なワークフローエンジンの開発

３．学会等名
情報処理学会 第82回 全国大会

４．発表年
2020年

１．発表者名
吉野貴大 , 佐々木晃

２．発表標題
汎用言語サブセットを用いるプログラミング学習システムの提案

３．学会等名
日本教育工学会研究会19-1

４．発表年
2019年

１．発表者名
澤入 圭佑 , 佐々木晃

２．発表標題
段階的・対話的プロセスによる例文に基づいた構文設計支援手法の提案

３．学会等名
情報処理学会 第123回プログラミング研究会

４．発表年
2019年

１．発表者名 Daiki Tanaka, Katunobu Itou
２．発表標題 Automatic Electronic Organ Reduction System Based on Melody Clustering Considering Melodic and Instrumental Characteristics
３．学会等名 ISM 2018（国際学会）
４．発表年 2018年

〔図書〕　計0件

〔産業財産権〕

〔その他〕

―

６．研究組織

	氏名 （ローマ字氏名） （研究者番号）	所属研究機関・部局・職 （機関番号）	備考
研究分担者	廣津　登志夫 (Hirotsu Toshio) (10378268)	法政大学・情報科学部・教授 (32675)	
研究分担者	伊藤　克亘 (Itou Katunobu) (30356472)	法政大学・情報科学部・教授 (32675)	

７．科研費を使用して開催した国際研究集会

〔国際研究集会〕　計0件

８．本研究に関連して実施した国際共同研究の実施状況

共同研究相手国	相手方研究機関