5G-2

名前の統括管理システム NameMasterの方式

・神田 陽治 ・ 森 厚 ・ 田中 英彦
東京大学 工学部

我々はプログラムの読みやすさの本質が、名前の記号性にあると考えた。ORAGA計画の中で、NameMaster は並列オブジェクト指向言語DinnerBellのコンパイラと協同してプログラムの記号性を検査する〔1〕。

センスあるプログラマはネーミングにも気をつかう。良い名前を考えるのは思いのほか大変である。一つにはプログラム用の語彙の不足がある。言葉を知らないから、勢い場当たり的になりがちである。この困難を積極的に助けてくれるプログラミングツールを作りたい。記号の生成を助けるツールは未だ模索中でアイデア固めの段階にあるが、記号の検査をするツールはその方針が見えつつある。その方式を説明する。

我々の方法は、記号性があることを、「名前からの連想」が、「プログラムの文脈」によって十分に絞れるかどうかで判断しようとする。人間がプログラムを読むときにやっている作業を、モジュールベース（コンパイルされたクラスの集積をこう呼ぶ）を基に機械的にシミュレートしようというのである。

1．記号性検査の方法

まず、プログラミング言語に新しい言語要素を導入する。大筋は以下のようである。A）プログラムの識別子は基本単語を並べて作る。B）複合化にも簡単なシンタックスを課す。C）クラスのデータ記述とメソッド記述に、名前記述を追加する。名前記述はクラスに許される（複数の）名前を決める。ある名前が、あるクラスの名前となりうるとき、このクラスは名前を「受理できる」という。名前記述は名前式として具体化される。D）メソッドヘッダに、メソッドの入出力仕様を明確に盛り込む。従来と違って、仮引数の名前を重視する。後述するように、名前式で名前の連想範囲をシミュレートし、メソッドヘッダでプログラムの文脈をシミュレートする。

記号性を検査するコンパイラの仕組みは、大略次のようである。コンパイラの語彙解析部は、a）識別子を基本単語に切り分け、b）簡単なシンタックスに基づいて識別子の構造を解析する。コンパイルの構文解析部は「記号性検査ルーチン」を呼出し、c）（モジュールベースに格納されたクラスの中で）その表現を受理できるクラスの集合を、識別子の連想範囲と定め、意味と定義し、d）識別子の意味候補（のクラス）が、メソッドヘッダの作る文脈に照らして矛盾していないか調べ、矛盾する意味候補を消去した後、プログラム全体で筋の通る解釈が一組だけ残ったとき、記号性あるプログラムと判断する。

2．名前の記号化の方法

A）識別子は基本単語から造る

識別子の名前に、基本単語のレベルを設け、識別子は基本単語を並べて作る。基本語彙は原則的に固定とする。新しい名前は、既にある基本単語を流用するか（例えば、Stack）、複合語として（例えば、QuickSort）作るのがふつうだからである。そこで、基本単語には番号をあらかじめ割り付けておける。これを「基本単語番号」と呼ぶ。

B）識別子に簡単な構造を課す

複合語は語彙の少なさを補う効果的な方法であるが、複合語化の方法自体明確にわかっていない〔2〕。ここでは便宜的に次に示す「ルールスキーム」を使う。自然言語の文法に倣ってはいるが、副詞や接続詞を扱わない点で制限がある。前置詞句を文とできるなどの点でより自由である。

```
文        ::=  名詞名  |  動詞名
名詞名    ::=  名詞句
名詞句    ::=  〔限定詞〕〔形容詞〕... 名詞 ...〔前置詞句〕
動詞名    ::=  動詞句  |  前置詞句  |  名詞句  動詞句
動詞句    ::=  動詞    |  動詞    前置詞句
前置詞句  ::=  前置詞  |  前置詞  名詞句
```

基本的には、動作に関する名前には「動詞名」の、物を指す名前には「名詞名」のルールスキームを当てはめる。オブジェクト指向言語に現れる名前は、キー名とスロット名である。メッセージに付けられたキーは、レシーバに引数を渡し充足されたメソッドを起動する。キーは動作を指定し物も指すので、キー名には「文」のルールスキームを与える。また、スロットはオブジェクトポインタが格納される箱であり、物を指すので、スロット名には「名詞名」のルールスキームを与える。

基本単語から文や名詞名を作るときは、空白を開けず、各基本単語の頭文字を大文字にして接合する。

C）クラスへ名前記述を加える

名前記述はクラス候補名を受理、生成するための記述である。ここで便宜的に使う名前記述を、名前式と呼ぶ。

```
名前式          ::=  クラス候補名〔|  クラス候補名〕...
クラス候補名    ::=  単語          〔|  単語        〕...
単語            ::=  基本単語      |  引用
引用            ::=  $  引用クラス名
引用クラス名    ::=  基本単語 ...
```

クラスの名前記述は複数のクラス候補名を、「または」の意味を持つ区切記号「|」で区切った並びであり、個々のクラス候補名は基本単語か他のクラスからの引用である。引用は、基本単語だけから作った引用クラス名に「$」を前置して示す。引用されるクラスは、引用クラス名を受理できるクラス（の一つ）である。（この指定だけでは複数のクラスが、引用されるクラスの候補として挙がる可能性があるが、プログラムの文脈などを利用して最終的には一つに絞られる。名前式も記号性検

査の対象であり、最終的に引用クラスの候補が一つに定まらなければ、このクラスのコンパイルは失敗する。）

引用の位置には、先方のクラスの（展開済の）候補名が置き換わると考える。クラス候補名が複数あれば一つ一つを当てはめ、それらを「｜」で並べる。結局、引用を展開すれば、クラスの名前式は基本単語列であるクラス候補名の並びとなる。引用形になっているので、引用先のクラスの名前式が変更された場合、自動的に引用している側の名前も変わることになる。

クラスが名前を受理できるのは、名前が、展開された名前式のうちのクラス候補名のどれかと「一致」するときである。与えられた名前もクラス候補名も基本単語列であるが、与えられた名前がクラス候補名の、後ろを接する部分列（例えば、YZはXYZ に後ろを接する部分列であるが、XYは後ろを接していない）であるときに「一致」すると定義する。この定義の根拠は、複合語では最も右側の語が主要語であることが多く、複合語を省略するときは先頭から落とすのが自然であることに拠る。

名前式に引用されたクラスは、名前式を定義するクラスのスーパークラスとなる。すなわち、名前式はスーパークラスの宣言を兼ねている。これはスーパークラスを相対的に指定することを意味する。受理のときに後ろを接する部分列で一致をとることと、スーパークラスを名前を受理できるクラスの中から選ぶことは、スーパーチェインで結ばれるクラス達が互いに関連した名前を持つことを強要する。チェインの開始に近いクラスほど短い名前であり、末端に近付くにつれて修飾語が前に付いた長い名前となる。後ろを接した部分が共通になっている。

D）メソッドヘッダはクラスの入出力仕様である

自然言語において、語を文法に従って並べても、必ずしも意味の通る文になるわけではない。ある単語の出現は他の単語の出現を制限したり促したりする。これが文脈である。辞書はそのような実例を集めたものと言える。辞書に用例があれば正しい結合と判断できる。イディオムはその典型である。

プログラムの場合にも、プログラムの中のスロット名とキー名は、メッセージの送受信の記述によって互いに関係付けられている。メッセージの送受信の記述がプログラムでの文脈に他ならない。スロット名やキー名が文脈に照らして意味の通る出現であるかどうかは、レシーバとなるスロット名が、実行時に送られて来るだろうメッセージのキーを受けることができるかどうかで、引数となったスロット名の場合、実行時に代入されるだろう仮引数の名前と適合するかどうかで決まる。キー名の場合は、二つの場合の裏返しである。この時、自然言語の辞書に当たるものはモジュールベースである。辞書のイディオムに相当するものは、コンパイルされたクラスに定義されたメソッドヘッダである。ヘッダはメソッドの入出力仕様を決める。

これまでの言語では、仮引数の名前はなんら重要視されていなかった。プログラムの文脈を定めるために、入力と結果用の仮引数の名前を明示したメソッドヘッダを導入する。

3．記号性検査のコンパイラへの組み込み

コンパイラの語彙解析部がａ）とｂ）を、構文解析部に呼ばれる記号性検査ルーチンがｃ）とｄ）を受け持つ。記号性検査ルーチンは、意味付けルーチンの一つとみなせる。

ａ）名前の切り分け

名前は単語を空白を開けずに並べて作られる。各単語の頭文字を大文字にして連接するので、容易に単語の列に切り分けられる。それぞれの単語は基本単語辞書を引いて基本単語番号に変換され、名前は基本単語列に変換される。

ｂ）名前の検査

基本単語番号の列を、簡単な文法に照らして正しいか検査する。検査後、記号表へ格納され「記号番号」に変換される。記号番号は一連のクラスに対して一意である。

ｃ）名前の意味付け

名前の意味（連想範囲）を、それを受理できるクラスの集合でシミュレートする。スロット名の場合、名詞句の修飾詞を除いた部分を「機能を表す部分」として、その意味（受理クラス集合）をスロット名の意味とみなす。（注意：記号番号で記号表を引けば、元の基本単語番号列を復元できる。）キー名の場合は、前置詞や接続詞のように、使う際に正確に綴るべき「予約語」と考え、意味は問わない。

ｄ）文脈に基づく記号性検査

プログラムの文脈は、メッセージ送受信の記述である。ここでは便宜的に次の簡略化されたメッセージ送受信を扱う。しかし、扱えるメッセージの流れは十分一般的である。

送受信式　::=　送信式〔→　変換式〕...　〔⇒　受信式〕
送信式　::=　メッセージ単位 ...
変換式　::=　スロット名　...
受信式　::=　メッセージ単位 ...
メッセージ単位　::=　キー名　スロット名
メソッドヘッダの入力仕様　::=　受信式
メソッドヘッダの出力仕様　::=　送信式

送信式ではキーとスロットの内容からメッセージが作られ、変換式のスロットが指すオブジェクトに渡される。オブジェクトは、依頼メッセージを受け結果メッセージを返す能力を持つ。最後に受信式が、同じキーを持つメッセージを受け、内容をスロットに代入する。オブジェクトはいくつかのメソッドの集まりであり、メソッドはメソッドヘッダで特徴付けられる。

基本的な方策は、送信式から出たメッセージが無事に最後の受信式に辿り着けるような、スロット名の意味付けができるかどうかを見る。まず、すべてのスロット名からクラス候補名を列挙する。次に考えられるすべてのクラス候補名の組み合わせに対して、メッセージが流れうるかどうかをみる。送信式が流しうるメッセージを、スロット名の意味候補の一つにキー名を付けたメッセージとする。受信式が受取りうるメッセージは、キー名が一致しメッセージの内容と仮引数のスロットの意味候補が一致するメッセージとする。メソッドヘッダは受信式と送信式に還元できる。最後の受信式に至るメッセージの流れがあれば、考慮中のクラス候補の組み合わせは意味が通る。

一つのクラスのすべてのメッセージ送受信式を記号性検査ルーチンで調べた後、全体でただ一つの意味の通るスロット名のクラス候補の組み合わせが残ったとき、このクラスは記号性があるものと判断し、コンパイルを成功とする。

〔1〕３１回情報処理学会全国大会　1E-7.
〔2〕Carroll,J.M.: What's in a Name? , Freeman(1985).