型代入を遅延する最適化型推論アルゴリズム

上野 雄大 (410015)

北陸先端科学技術大学院大学 情報科学研究科

2006 年 2 月 9 日

キーワード: 型推論アルゴリズム，最適化，効率化，関数型言語，ML.

型推論機構は，Standard ML や Haskell，ObjectiveCaml などの近代的な関数型言語の大きな特徴のひとつである．これは，プログラムが持つ最も一般的な型を自動的に推論する機構である．この機構によって，コンパイラは明示的に型付けされていないプログラムの型を自動的に推論する．この機構の中核をなすのが，多相型を持つ変数束縛を許す多相型 let 構文を含む拡張ラムダ式に対する型推論アルゴリズムである．

多相型型推論が実践的に非常に有用であることは広く知られており，様々な言語がこれを採用している．しかし一方で，型推論アルゴリズムは関数型言語のコンパイラフロントエンドの中でも最も複雑かつ時間のかかる処理のひとつであり，コンパイラの複雑化やコンパイル時間の増大を招いている．宣言的かつ効率的な型推論アルゴリズムの開発は，多相型型推論機構を装備した次世代プログラミング言語の設計と実装にとって重要な課題である．

ところが，多相型を持つ関数型言語の型推論問題は，多相型 let 構文が存在するために DEXPTIME 完全であることがすでに示されており，漸近的な動作を考えた場合，アルゴリズム論の意味において効率的な型推論アルゴリズムの構築は不可能である．このような理論的な限界が存在している一方，コンパイル時間の短縮のため，現実には実用上より効率的な型推論アルゴリズムが求められている．本稿では，高度な機能を含む最先端の関数型プログラミング言語のコンパイラの高信頼かつ効率的な実装の基礎として，宣言的記述が可能でかつ実用上より効率的な新しい型推論アルゴリズムを提案する．

現在の関数型言語のコンパイラの多くが採用している Milner の型推論アルゴリズム $\mathcal{W}$ には，以下のような効率上の問題点がある．

- 型環境などの一般に大きな型を含む構造に対して，頻繁に型代入を適用している．
- 現実のプログラムでは頻繁に出現する多相型 let 構文の型推論において，束縛変数の集合を計算するために型環境全体の走査を必要としている．

前者の問題は，型代入を型環境を評価する際に使用すべき明示的な環境として保存し，型代入の適用を遅延することで解決できると期待できる．また，後者の問題については，型

環境から到達可能な自由な型変数の集合を常に把握しておくことが可能ならば，let 構文の出現の度に型環境全体の走査を行うことを省くことができると考えられる．加えて，型代入の適用を遅延するという戦略を実現するためには，単一化アルゴリズムも型代入環境の下で型等式の単一化を計算するアルゴリズムへと洗練しなければならない．

本稿では，以上のような洞察に基づき，実用上より効率的な型推論アルゴリズム $\mathcal{DW}$ と，$\mathcal{DW}$ で用いる単一化アルゴリズム $\mathcal{DU}$ を構築し，$\mathcal{DU}$ についてはその健全性と完全性を，$\mathcal{DW}$ についてはその型健全性をそれぞれ示す．例えば，ラムダ抽象がネストする式などの型推論には，従来のアルゴリズム $\mathcal{W}$ では n^2 に比例する量に対する型代入の適用を要するのに対し，本稿の提案するアルゴリズム $\mathcal{DW}$ は n に比例する量に対する型代入の適用で推論を完了することができると考えられ，従来のアルゴリズムと比較して格段の高速化が見込まれる．

また，$\mathcal{DW}$ を用いて Standard ML 言語の Core Syntax 相当の型推論機構を SML$^\sharp$コンパイラ上に実装し，$\mathcal{DW}$ が実際の関数型言語のコンパイラ上で実現可能であることを示すとともに，$\mathcal{DW}$ が実用上より効率的であることを示すために，その実装と実際のプログラムを用いてアルゴリズムの比較評価を行った．評価の結果，$\mathcal{DW}$ は従来のアルゴリズム $\mathcal{W}$ と比較して非常に高速であり，さらに従来のアドホックに拡張されたアルゴリズムに匹敵する性能を有していることが示された．

最後に，$\mathcal{DW}$ の更なる高速化のための拡張に関して議論する．