付録 A： 型付けと翻訳の規則

この付録では、形式的な型付けと翻訳の規則を示す。1 セットの規則が、型付けと翻訳の両方を実現する。これらの規則は、Damas と Milner によるもの [DM82] の拡張となっている。

A.1 言語

多重定義に対する型付けと翻訳の規則を示すにあたって、問題の本質をとらえた、少し単純化された言語を用いることが有用である。ここで用いる言語は、通常の構成要素（識別子、関数適用、ラムダ抽象、および、**let** 式）に加えて、2 つの新たな要素として **over** および **inst** 式を持つ。これらは、それぞれ `class` および `instance` 宣言に対応する。この言語の式と型の文法を図 5 に示す。

over 式

$$\textbf{over}\ x\ ::\ \sigma\ \textbf{in}\ e$$

は、x を多重定義された識別子として宣言する。この宣言スコープの中には、いくつかの **inst** 式がある

$$\textbf{inst}\ x\ ::\ \sigma'\ =\ e_0\ \textbf{in}\ e_1$$

ここで、型 σ' は型 σ （記法については後に詳しく述べる）のインスタンスである。ラムダや **let** 式と異なり、**over** や **inst** 式で束縛された変数は、より小さなスコープで再宣言されることはない。もうひとつ、ラムダや **let** 式と異なる点としては、**over** や **inst** 式における型は明示される必要がある。他の式においては、型は、ここで与えられる規則によって推論される。

例として、図 3 にある等値の定義の一部を、図 6 に示す。この例、および、本付録の以降の部分においては、Eq τ を $\tau \to \tau \to$ Bool の略として用いる。

もうひとつの例として、図 1 で与えた算術演算子の定義の一部を図 7 に示す。この図では、Num τ は次に示すような型の省略形である。

$$(\tau \to \tau \to \tau, \tau \to \tau \to \tau, \tau \to \tau)$$

形式言語への翻訳では、3 つの演算子をひとつの「辞書」にまとめた。これは素直なやり方であり、また、一番大事な問題である多重定義の解決方法とは独立である。