発表概要

C++テンプレートを分割コンパイルするためのアプローチ

増山 隆[†] 住井 英二郎[†] 米澤 明憲[†]

既存の C++テンプレートのコンパイルモデルは，インライン化によるものである．このモデルでは，各オブジェクトコード，実行可能コードのサイズが大きくなりがちであり，またテンプレートで書かれた実装が隠蔽できないという問題がある．本発表では，C++のテンプレートを分割コンパイルする方法を示す．変換後のコードは実行時に，(1) テンプレートの parametric 多相性を扱うために，型多相なデータに対し，(多くの ML コンパイラが行うように) boxing と unboxing の操作をする．(2) C++の ad hoc 多相性を扱うために（Glasgow Haskell コンパイラが行うように）オーバロードされた関数を渡す．また，型多相なオブジェクトに対するメソッド起動を扱うために，C++の多重継承と抽象インタフェースの機能を用いる．

An Approach to Separate Compilation of C++ Template

Takashi Masuyama,[†] Eijiro Sumii[†] and Akinori Yonezawa[†]

The existing compilation model of C++ templates—namely inlining —suffers from a problem that each object code and the resulting executable tend to become huge, and also, that implementations of templates cannot be hidden. In this presentation, we propose a method of separately compiling C++ templates. This method is based on source-to-source transformation of C++ code. At runtime, the transformed code (1) performs boxing and unboxing on polymorphic data (as many ML compilers do) in order to deal with the parametric polymorphism of templates and (2) passes around overloaded functions (as the Glasgow Haskell Compiler does) in order to treat the ad hoc polymorphism of C++. Our method also exploits the multiple inheritance and abstract interface mechanisms of C++ to handle method invocations on polymorphic objects.

（平成 14 年 6 月 17 日発表）

† 東京大学情報理工学系研究所コンピュータ科学専攻
Department of Computer Science, Graduate School of Information Science and Technology, The University of Tokyo