

7Q-6 拡張可能な C++ ソースコード・ブラウザ — プログラム・データベース —

中村 宏明[†] 安田 和[†] 三ッ井 欽一[†] Shahram Javey[‡]

[†]日本アイ・ビー・エム (株) 東京基礎研究所 [‡]IBM Canada, Toronto Laboratory

1 はじめに

われわれは C++ プログラムの理解、デバッグ、保守を効果的に支援することを目的として C++ ソースコードブラウザ [4] を開発している。C++ ソースコード・ブラウザに要求される機能は、目的・ユーザ・対象とするアプリケーションによって異なるため、設計に際して拡張性をもっとも重視している。

ブラウザは次の2種類の部品から構成されている：

- プログラム情報を保持し、それに対する問い合わせを処理するプログラム・データベース・サーバ
- データベースのフロント・エンドとなって、さまざまなビューを提供するグラフィカル・ユーザ・インタフェース。

本稿では、プログラム・データベースのモデルと実現、データベース・サーバの実装と評価について述べる。グラフィカル・ユーザ・インタフェース部に関しては [6] で述べている。

2 データ・モデルとその実現

われわれはプログラム・データベースの設計にあたって、C++ の言語仕様 [1] にそったプログラムを表現するために、E-R モデルに一貫性制約を付加して C++ プログラムの概念モデルを構築した [2]。このモデルでは、たとえば、「変数」はエンティティに、「記憶クラス」は属性に、「変数とそれを定義しているもの（クラスなど）の関係」はリレーションシップに、「プログラム中で同一の変数が2回以上定義されないこと」は一貫性制約に、それぞれ割り当てられる。

データベースの実現には Prolog を用いた。Prolog ではデータとプログラムを同一視できるため、プログラムを実行時に変更したり追加したりが容易に行なえる。また Prolog をデータベース言語として見ると、関係型の問い合わせの他に、再帰型の問い合わせや、手続き

を含んだ問い合わせなどを行なうことができる。このため Prolog は、拡張可能なブラウザを実現するために適している。

概念モデルの要素は Prolog によるデータベースでは、次のように実現される。

概念モデルの要素	実現
エンティティとその属性	ファクト
リレーションシップ	ファクトとルール
一貫性制約	ルールとトランスレーション

ここでトランスレーションとは、C++ プログラムに関する情報を対応する Prolog のファクトに変換する過程のことである。C++ コンパイラと、コンパイラの出力から Prolog のファクトへのトランスレータで実装される。

3 データベース・サーバの実装

データベース・サーバは、Prolog インタプリタ、プログラム情報から Prolog のファクトへのトランスレータ、通信コントローラの3つのモジュールからなっている。

3.1 Prolog インタプリタ

データベース・サーバが扱うデータは、数MB から数10MB の大きさを持つ。最近ではこの程度の大きさのデータはメイン・メモリ上にすべて置くことが可能なので、Prolog インタプリタそのものをデータベース・サーバのコアとして用いることにした。

ところが従来の Prolog 処理系はわれわれの目的のためにはいくつかの欠点がある。Prolog を他言語で書かれた構成要素と結合して使うとき、例えば複数のゴールを同時に生成・実行できないなど、制御方法に関する制限がある。また、クローズの格納法やインデキシングの方法などの観点から、大量データを扱うのに向い

Extensible C++ Source Code Browser — Program Database

Hiroaki Nakamura[†], Kazu Yasuda[†], Kin'ichi Mitsui[†], and Shahram Javey[‡]

[†]IBM Research, Tokyo Research Laboratory

[‡]IBM Canada, Toronto Laboratory

ていない。そこでわれわれは、他言語と自由な組合せが可能で、大量データの処理に向けた Prolog 処理系 [5] を開発し、データベース・サーバに用いることにした。

この Prolog 処理系は、組み込み述語などの点で DEC-10 Prolog とほぼ同等なものであるが、プログラム・データベース用に次のような拡張を行なっている。

- SQL が提供しているような、結果のグループ化機能・集計機能
- プログラム・テキストを扱うための、アトムを文字列とみなして行なう編集機能
- ファイル・システムに対する種々の操作

3.2 トランスレータ

トランスレータは 2 つの役割を持っている。ひとつは、コンパイラによって生成されたプログラムの構文・意味の解析データを受けとり、これを Prolog のファクトに変換することである。もうひとつの役割は、データベースの一貫性制約を保証することである。

コンパイラによって生成されるデータは、コンパイル単位内での一貫性は保証されているが、コンパイル単位間での一貫性はない。これは一つのヘッダ・ファイルが複数のコンパイル単位に共有されることや、ライブラリがその利用者モジュールとは独立に開発されることなどによって発生する。したがって、複数のソース・ファイルの情報を同時にデータベースに読み込むときに、重複した情報を取り除く、別ファイルからの名前をリンクするなどの操作が必要になる。

また、ソース・ファイルの書き変えの結果としてプログラム・データベースの内容を更新するときにも一貫性制御が必要である。このときのトランスレータの動作はデータベースの内容に依存するため、必要に応じて Prolog のルールを起動して、データベースの一貫性制約を満たすようにファクトを生成する。

3.3 通信コントローラ

通信コントローラは、クライアントからの要求を解釈し、プログラム・データベースへの問い合わせを行ない、結果を並び変え、加工してクライアントに返す。結果の返し方は要求ごとに変更可能で、プログラム・データベースのさまざまな利用のされかたに対応している。

また、クライアントとプログラム・データベースの結合は、各マシンごとに配置された C++ ソースコード・ブラウザ用のネーム・サーバによって解決される。

4 評価

グラフィカル・ユーザ・インタフェースから発行できる約 300 種類の問い合わせに対応する処理を、約 2,200

行の Prolog プログラムの形で記述した。これらの問い合わせには、「インヘリタンス・グラフ」、「クラス A の外から使えるベースクラスのメンバー」、「クラス B を変更した時に影響を受ける関数」、などを求めるものが含まれる。この結果、問い合わせ言語として Prolog を利用することは、複雑な処理を簡潔に記述するのに有益で、また仕様の変更に迅速に対応するのに役立つことを確認した。一方で、ルールの書きかたによって探索時間が大きく変化するので、効率のよいルールを書くには変数の入出力やゴールの実行の順序などに対する注意が通常の Prolog プログラムよりも必要である。

プログラム・データベースの効率を、IBM RS/6000 model 550 上で InterViews ツールキット [3] のソース・コードをデータとして用いて調べた。InterViews は 908 個のファイルからなり、約 11 万行の C++ プログラムである。InterViews のプログラム・データベースを生成するのに 412 秒かかった。このうちの 77% は一貫性制御のために使われている。ファクトは全部で約 587,000 個生成され、メモリ上で 30.8MB を占める。すべてのデータをメモリ上に載せることはできたが、このような C++ ライブラリは一般に大きいので、複数ユーザで共有することが望ましい。また、たいいていの問い合わせに対しては数秒以内で結果が返される。一方で、「関数 A から到達可能な関数」を求める問い合わせなど、十秒以上必要とするものもある。これは対話的な応用では許容されないが、このような時間のかかる問い合わせは予測可能なので、われわれはユーザ・インタフェースの側で「関数呼び出しの深さ」を指定させるようにしてこの問題を回避している。

参考文献

- [1] M.A. Ellis and B. Stroustrup: The Annotated C++ Reference Manual, Addison-Wesley, 1990.
- [2] S. Javey and K. Yasuda: The Conceptual Model for the C++ Program Database, IBM Technical Report, TR-74.093, 1992.
- [3] M.A. Linton, J.M. Vlissides, and P.R. Calder: Composing User Interface with InterViews, *IEEE COMPUTER*, Vol. 22, No. 2, 1989, pp 8-22.
- [4] 三井他: 拡張可能な C++ ソースコード・ブラウザ - 基本設計, 第 45 回情処全大, 7Q-05, 1992.
- [5] H. Nakamura: Embeddable PROLOG Interpreter for Data-Intensive Applications, IBM Technical Report, TR-74.091, 1992.
- [6] 大平他: 拡張可能な C++ ソースコード・ブラウザ - ユーザインタフェース, 第 45 回情処全大, 7Q-07, 1992.