ＥＳシェルＤＥＸにおけるＣトランスレータ

3D－1

片山　立　　梶谷雄治　　松本健志　　西田行輝

三洋電機㈱　情報通信システム研究所

１．はじめに

知識ベースシステムの実用化が進展するにつれて、知識ベース構築支援ツールで構築した知識ベースを既存のソフトウエア資産に組み込んだり、個々のアプリケーションに応じたカスタマイズへの要求が高まっている。我々は、診断・分類型シェルＤＥＸ[1]の知識ベースをＣ言語に変換するＣトランスレータおよび推論関数ライブラリを開発した。本稿ではその概要と推論処理の高速化について述べる。

２．Ｃトランスレータの概要

ＣトランスレータはＤＥＸで開発した知識ベースをＣソースファイル（Ｃ言語の構造体）に変換するツールである。ＣトランスレータではＣソースファイルだけではなく、変数名の対応や定数を記述したヘッダファイル、推論関数の標準的な実行手順が記述されたｍａｉｎ関数ファイル、およびｍａｋｅｆｉｌｅが生成される。Ｃトランスレータの処理フローと、変換後のＣソースファイル例を図１、図２に示す。

(1)知識ベースのカスタマイズ機能

一般に、ＥＳシェルで開発した知識ベースをユーザプログラムに組み込む段階では、推論データの入出力方法やユーザインタフェースに関するカスタマイズが要求されることが多い。そこでＣトランスレータでは推論モード、推論データ入力モード、推論関数実行中の画面表示制御モードについて対応するメニューウィンドウから簡単に指定できるようにした。変換時には指定された各モードに対応するパラメータを推論関数を実行する直前に設定し、推論関数の動作を規定するようにしている（図３）。

```
/* dex knowledge base c source file */
/* file : "net.c" */
#include "ieval.h"
#include "lowacces.h"
#include "window.h"
#include "net.h"

/* knowledge base information */
KBINF net_kbinf = {"1.00",
"net",
"ネットワーク監視診断システム",
0x6f,
0x5,
'f'};

/* integer question fact */
IQFACT net_iqf[4] = {
{0,"ｋｉｋｕのディスクの残り容量は何ＭＢですか．",
0xffff,0,0xffff,"ｋｉｋｕのディスク残り容量（ＭＢ）","",},
{0,"現在のＵＵＣＰ連続通信時間はいくらですか．",
0xffff,0,0xffff,"ＵＵＣＰ連続通信時間（分）","",},
{0,"現在の時刻（０時からの累計分）を入力してください．",
0xffff,0,0xffff,"時刻（０時からの累計分）","",},
{0,"ＵＵＣＰのポーリング失敗回数はいくらですか．",
0xffff,0,0xffff,"ＵＵＣＰのポーリング失敗回数．","",},
};
```

図２　変換後のＣソースファイル例

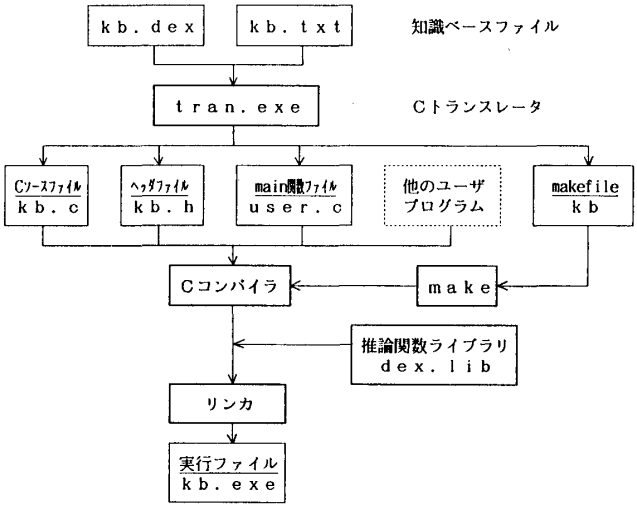

図１　Ｃトランスレータの処理フロー

(2)推論関数

推論関数の標準的な実行手順は、ｍａｉｎ関数ファイルに自動的に出力される。ユーザプログラムに組み込む場合には、これをもとにして適当な修正を加えるだけでよい。基本的な実行手順は

①知識ベースの使用宣言
②推論関数のための各種パラメータ設定
③外部ファイルから推論データを入力する場合には、推論データ入力関数の実行
④推論関数の実行

となる。推論関数の典型的な実行手順を図３に示す。

```
/* dex knowledge base main function source file "user.c" */

#include "ieval.h"
#include "lowacces.h"
#include "window.h"

extern KB net;

main()
{
UseKB(&net);                    /* 知識ベースの使用宣言関数 */

/* set parameter for inference */
/* 推論関数のためのパラメータ初期化 */
 fact_list = 0;                 /* ファクト通常処理 */
 input_flag = 2;                /* ファイル指定入力 */
 clear_display = 0;             /* 初期画面クリアしない */
 title_flag = 0;                /* タイトル表示しない */
 menu_flag = 0;                 /* 推論モードメニュー選択しない */
 result_flag = 2;               /* 推論結果表示しない */
 disp_order = 'd';              /* 確実レベル以上表示 */
 reset_display = 0;             /* 推論終了後画面クリアしない */

InputFactByFName("net.dat");    /* 推論データ入力関数 */

InferenceDEX();                 /* 推論関数 */
}
```

図３　推論関数の実行手順

C Translator for Expert Shell DEX
Ryu Katayama, Yuji Kajitani, Kenshi Matsumoto and Yukiteru Nishida
SANYO Electric Co., Ltd.

３．アクセス関数と推論関数の高速化

Ｃトランスレータにおいては以下の方法でアクセス関数および推論関数の高速化をはかっている。

(1)型を明示した変数名への変換

ＤＥＸにおける変数（ファクト、中間仮説、結論仮説）の管理は、データの登録、削除が容易で、かつデータのアクセス効率のばらつきが少ないＢ－Ｔｒｅｅ[2]を用いている。知識ベース作成時には、変数の新規登録、修正、削除、変数名の変更などが頻繁におこること、また推論時に変数名（文字列）から変数の実体を高速にアクセスする必要があるので変数名の管理にＢ－Ｔｒｅｅ を、また変数（ファクト、中間仮説、結論仮説）の実体は配列で登録している。この場合の変数のアクセスは以下のようにして行われる。

①変数名をＢ－Ｔｒｅｅ内でサーチし、タイプ（ファクト（ｆ）、中間仮説（ｉ）、結論仮説（ｃ））とインデックスｎを取得する。

②タイプに対応した配列のｎ番目のレコードから属性値（値、取得方法、説明文、…）を取得する。

ところで、知識ベースエディタによって構築が終了した知識ベースをＣ言語に変換した後では、知識ベースの変更は生じないので、変数の新規登録や削除などの動的な管理をするために導入したＢ－Ｔｒｅｅは必ずしも必要ではない。

そこで、Ｃトランスレータではもともとの変数名を

『ｔ．ｎ』

ただし　ｔ：タイプ、ｎ：インデックス

の形式に変換することにした。これにより、従来のＢ－Ｔｒｅｅ内における変数名のサーチ処理が不要になり、新しい変数名ｔ．ｎのタイプｔから直接、タイプ別の配列にアクセスできるようになり、アクセス関数の処理効率が向上している。

(2)ファクトタイプの拡張

ＤＥＸにおいてはファクトのタイプは１種類であり演算型ファクトや中間仮説などにあらわれる論理式の評価関数においては、ファクトの型（ブール、整数、演算）と取得方法（質問、演算、選択）の判定を行って、対応する評価関数に分岐していた。

Ｃトランスレータでは型と取得方法に応じてファクトタイプを１種類から７種類に拡張しこれを『ｔ．ｎ』のタイプとして追加することにより、論理式評価関数における型と取得方法の判定処理を不要にした。また、ファクトに関するアクセス関数においても、従来型と取得方法に応じてディスパッチしていた処理が不要となった。これによりアクセス関数、論理式評価関数、および推論関数が高速化されている。

ファクトアクセス関数の処理時間の改善を図４に示す。変換前では、Ｂ－Ｔｒｅｅ内のサーチがアクセス処理全体に占める割合は約９２％とその大半を占めていることがわかる。変換後ではタイプ判定とインデックスの取得処理がＢ－Ｔｒｅｅ内のサーチ処理の約１２％になっており、アクセス関数全体でも変換前の約１６％と大幅に改善されている。

また結論仮説１個あたりの評価処理時間を図５に示す。変換後の処理時間は変換前の約６０％に改善されており、アクセス関数の高速化が推論関数の処理効率の改善に寄与していることが確認できた。

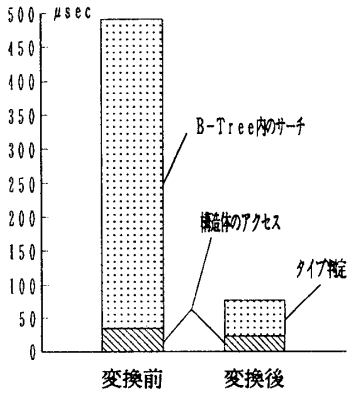

図４　ファクトアクセス関数の処理時間

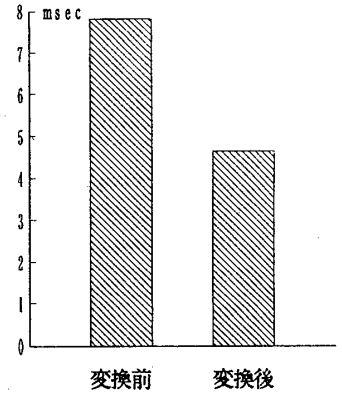

図５　結論仮説の評価処理時間

４．おわりに

ＤＥＸにおけるＣトランスレータと推論処理の高速化について述べた。知識ベースを手続き型言語に変換するこのようなツールは、既存ソフトウエア資産との融合や実行。運用システムの処理効率の点で有効であると考えられる。今後は、組み込み型やオンライン。リアルタイム型のエキスパートシステムなどに適用していく予定である。

［文献］

[1] 松本他：診断型シェルＤＥＸの開発，情報処理学会第３７回全国大会,7H-9,1988

[2] 溝口：”B-Treeによるデータ管理”，情報処理，vol.21,no.7,pp.769-776,1980