ＡＩＰ－ＬＩＳＰ：（２）ＬＡＰ，シンボルマージャー

7P-2

小泉善裕　木村兼江
（株）東芝

0．はじめに

ＡＩＰ－Ｌｉｓｐは、ＲＩＳＣ風プロセッサＡＩＰをターゲットとしたＬｉｓｐコンパイラである。本報告では、ＡＩＰ－Ｌｉｓｐシステムの中核をなすＡＩＰ－Ｌｉｓｐコンパイラのうち、ＬＡＰ（List Assembly Program）、およびシンボル情報を統合するシンボルマージャーの実現方法の概要と高速化について述べる。

1．ＬＡＰ

1．1　ＬＡＰフェーズ

既に文献［1］で報告したように、ＡＩＰ－Ｌｉｓｐコンパイラは全体として３つのフェーズ（プリプロッセサ、コードジェネレータ、ＬＡＰ）からなり、ユーザソースプログラムを入力してアセンブラソースプログラムを出力する。ユーザソースプログラムはプリプロセッサ、コードジェネレータの各フェーズを通ってコンパイラ上でネイティブコードに展開される。ＬＡＰは、このネイティブコードを変換して、ＡＩＰのアセンブラが受け入れることが可能なアセンブラソースファイルを出力するフェーズであり、Ｃｏｍｍｏｎ　Ｌｉｓｐで記述されている。

1．2　処理の概要

現在ネイティブコードは、中間コード表現の１命令につき１つのストラクチャが対応し１つの関数はそれらのストラクチャを要素とするような、リスト形式で表現されている。ＬＡＰはこのネイティブコードを変換してコードセクションに出力する処理を行なう。この時、プログラムから参照されている定数データは定数テーブルからのオフセットを用いることにより、定数テーブルセクションに出力される。さらに即値型データ（character,fixnum,short-float）以外のデータはその実体をＡＩＰ－Ｌｉｓｐのデータ表現に従ってヒープセクションに生成する。ただしシンボル型のデータは次に述べるシンボルマージャーで処理するため、別のシンボルファイルに出力する。図１にＬＡＰによる処理の一例を示す。

上で述べたコード変換は現在 case マクロを用いてその処理が行われている。これを Hash Table を用いて実現することにより、ＬＡＰの処理の高速化が期待される。

2．シンボルマージャー

2．1　処理の概要

現在のＡＩＰ－Ｌｉｓｐ第１版システムはスタティックリンク方式を採用しているため、プログラムのリンクに先だって、各プログラムから参照されるすべてのシンボルのスロットの内容をあらかじめ統合しておかなければならない。シンボルマージャーは、リンクすべきプログラムファイルからＬＡＰ実行時に出力されるシンボルファイルを読み込み、シンボルファイルに出力されているシンボルのスロットの内容を調べ、ｅｑなシンボルについてはスロットのマージを行なう。

次に、参照されているが定義されていない関数シンボルに対してダミーの定義（stub）を与える処理を行う。これによりプログラムのリンク時における Undefined: エラーを回避することができコンパイル処理の効率が向上する。

さらに、出力されたすべてのシンボルをパッケージに登録するためのデータを生成する。以上の処理の後、これらをアセンブラファイルとして出力する。このシンボルマージャーもまたＣｏｍｍｏｎ　Ｌｉｓｐで記述されている。

```
((#S(IENT OP LABEL O1 NIL O2 NIL S1 AL::FOO S2 NIL S3 NIL S4 NIL)
  #S(IENT OP LPSH O1 NIL O2 NIL S1 R5 S2 R1 S3 NIL S4 NIL)
  #S(IENT OP SETDP O1 NIL O2 NIL S1 NIL S2 NIL S3 NIL S4 NIL)
  #S(IENT OP LDC O1 R1C O2 NIL S1 '(AL::BAR) S2 NIL S3 NIL S4 NIL)
  #S(IENT OP LPOP O1 R5 O2 NIL S1 R1 S2 NIL S3 NIL S4 NIL)
  #S(IENT OP BAI O1 NIL O2 NIL S1 R0 S2 '4 S3 NIL S4 NIL)))


#include        "alglobals.h"
.ecode
AL$IFOO:
        lpsh    r5,r1
        ldfx    r5,L21357
        ldui    r5,L21357
        ldo     0,r1c,r5,0
        lpop    r5,r1
        bai     r0,4
.sdata
L21357:
        .long   L21358+0xe0000000
        .globl  AL$IFOO
.heap
L21358:
        .long   AL$@BAR+0x90000000
        .long   nil
```

図１　ＬＡＰによる処理の一例

2．2　高速化

シンボルファイルに出力されるシンボル情報はストラクチャにより表現されていたが、これをストラクチャを用いないファイルフォーマットに変更することによりシンボルマージャーが高速化された。また、シンボルファイルに出力されるシンボルのうちかなりの部分はライブラリシンボルのようなあらかじめ予想されるシンボルであり、これら予想されるシンボルをレジストリファイルとして登録し、これを利用することによってＬＡＰおよびシンボルマージャーの高速化が行われた。

3．おわりに

以上、ＡＩＰ－Ｌｉｓｐコンパイラのうち、ＬＡＰとシンボルマージャーの概要と高速化について報告した。

参考文献

［1］河込　他：“ＡＩＰ－ＬＩＳＰの実現（Ｉ）”
情報処理学会第３６回全国大会（１９８８）

LAP and Symbol-merger of AIP-LISP
Yoshihiro KOIZUMI, Kanee KIMURA
TOSHIBA Corp.