1S-1

ジャクソン法によるプログラム生成の半自動化

中島　孝士，乾　隆夫，高松　忍，　西田　富士夫
（大阪府立大学　工学部）

1．まえがき

本稿はジャクソン法により与えた仕様からプログラムを自動的に生成する一つの方法について述べている。先ず、入出力データの構造と制限した日本語で書いたプログラム仕様を与え、これを詳細化してプログラム構造をつくる。このとき、ライブラリモジュールを参照して入出力データ間に構造の不整合や不一致があれば、中間的なデータノードなどを作成し、能率的なプログラム構造をつくる。プログラム構造からCやCOBOLで書いたプログラムを生成する。

2．ジャクソン法による仕様表現

ジャクソン法は構造をもつ大量のデータを処理する事務処理などの仕様によく用いられる。従来、　ジャクソン法では処理対象のデータ構造は入出力ともツリー状に与えられる。このツリー状のデータ構造を機械処理し易い形にするために、例1のように横書きのリスト構造とし、seq(連接)，sel(選択)，iter(繰返し)などのデータ構造の見出しを付けインデントを行う。また、処理すべき内容は、従来、システムを記述したり処理内容を指示するために、日本語文で表している。この研究ではこれらは機械処理に便利なように制限日本語文で記述することとする。例えば手続き表現では

”入力データ(d_i)*をデータ(d_r)*を参照して
(proc)*し、出力データd_oを作（る）れ。”
”出力データ(d_o)*を出力するために
入力データ(d_i)*を(proc)*する(せよ)。”

入出力関係表現では

”入力データ(d_i)*が与えられ、(属性をもつ)*
出力データd_oが作られる。”

のような形に制限して仕様を与える。ただし＊印は0回以上の繰り返しを表す。制限日本語で与えた仕様は構文解析の後、形式表現に変換される。

例1

入出庫ファイルから報告書ファイルを出力する仕様を以下に示す。

(a)入力データ構造

```
入出庫ファイル
iter:入出庫レコード
     seq: 品名
          項目
          iter:項目レコード
               seq:日付
                   入出庫コード
                   sel:入庫
                       出庫
                   入出庫量
```

(b)出力データ構造

```
報告書ファイル
iter:物品レコード
     seq:品名
         変動量
```

(c)処理内容

```
入出庫ファイルと報告書ファイルは品名でソートされている；
入出庫レコードを品名でグループ化する；                    ①
入出庫ファイルと報告書ファイルをオープンする；            ②
各入出庫レコードに対し以下を繰り返す；                    ③
    入出庫レコードを読み込む；                            ④
    物品レコードの品名
        :=入出庫レコードの品名；                          ⑤
    変動量を0とする；                                     ⑥
    各項目レコードに対して以下を繰り返す；                ⑦
        (入出庫コード＝入庫)なら
            変動量を入出庫量だけ増加させる；              ⑧
        (入出庫コード＝出庫)ならば
            変動量を入出庫量だけ減少させる；              ⑨
    物品レコードを書き出す；                              ⑩
入出庫ファイルと報告書ファイルをクローズする；            ⑪
```

3．仕様の整備と詳細化

出力データ構造の各データ名d_oを中心に仕様を一つにまとめる。すなわち、出力データファイルの各データに要求される属性や処理内容と必要な入力データを、処理仕様文と入力データ構造を参照して次のような形にまとめる。

Semi-automating of Program Generation by Jackson's Method

Takasi NAKAJIMA, Takao INUI, Shinobu TAKAMATSU and Fujio NISHIDA
University of Osaka Prefecture

```
do: (IN:GIVEN(di), PROP(di),
     OUT:GIVEN(do), PROP(do))      (1)
    PROC:手続き文;
```

(1)仕様文の中にdoをGOAL格に含む手続き文があれば、これにラベルPROCを前置してdoに付加する。

(2)仕様文の中にdoに関する性質関係文があれば、これをdoのOUT部にPROPerty文として付加する。

(3)(1)(2)で求めた仕様文の中に参照すべき入力データ名diがあれば、これをdoのIN部に付加する。なお、diに関する性質関係文があればこれをPROPerty文としてdiに付加する。ただし、doやdiの性質に関する記述がないときには、簡単のためにOUT部は省略し、入力部はdiに前置したGIVEN述語詞を省略することができるものとする。

例2

例1について(a)(b)(c)の形式表現を(b)の出力データ構造を中心に一つにまとめるとつぎのようになる。

```
報告書ファイル
(IN:GIVEN(OBJ:入出庫ファイル),
    SORTED(OBJ:入出庫ファイル,INSTR:品名),
 OUT:SORTED(OBJ:報告書ファイル,INSTR:品名))
   PROC:GROUP_RECORD(OBJ:入出庫レコード,
                     INSTR:品名);                    ①
       OPEN_FILE(OBJ:入出庫ファイル,
                 報告書ファイル);                    ②
   iter:物品レコード  (IN:入出庫レコード)
     PROC:REPEAT(REF:入出庫レコード,
                 UNTIL:EOF-OF-入出庫ファイル,
                 OBJ:body);                          ③
       body:
       seq:PROC: READ_RECORD(OBJ:入出庫レコード);    ④
           品名          (IN:品名)
           PROC: :=(品名(OBJ:物品レコード,ATTR:*),
                    品名(OBJ:入出庫レコード,
                         ATTR:*));                   ⑤
           変動量        (IN:項目)
           PROC: :=(変動量,0);                       ⑥
                 REPEAT(REF:項目レコード,
                        UNTIL:EOF-OF-入出庫レコード,
                        OBJ:body);                   ⑦
                  body;
                    IF (COND: =(入出庫コード,入庫),
                        OBJ: ADD(OBJ:入出庫量,
                                 GOAL:変動量));      ⑧
                    IF (COND: =(入出庫コード,出庫),
                        OBJ: SUBTRACT(OBJ:入出庫量,
                                      SOURCE:変動量)); ⑨
           PROC: WRITE_RECORD(OBJ:物品レコード);     ⑩
   PROC:CLOSE_FILE(OBJ:入出庫ファイル,
                   報告書ファイル);                  ⑪
```

つぎに手続き名、入出力関係文、データ型名などを用いて、ライブラリモジュールを検索し、求める処理を行なうモジュールの存在や、入力条件の充足性などをチェックし、欠落部があれば補填し、仕様作成者に問い合わせる。例2では、②, ④, ⑩, ⑪はファイル処理に関するモジュールでチェックしたり、自動的に生成できる。また③, ⑦は繰返しデータ構造に関するモジュールから生成できる。

入出力ファイルがシーケンシャルファイルであるとき、出力データ構造が入力データ構造と一致しない場合の問題がある。データ構造の不一致は順序不一致、境界不一致、脈絡不一致などに分類されることが知られている。このうち、順序不一致や脈絡不一致は同じ属性に関するソートモジュールなどにより、境界不一致は境界整合化モジュールによりデータ構造を一致させることができる。

例3

例1において入出庫ファイルが品名でソートされていないとき、例のような仕様から、整備やリンクを行って詳細化した結果を以下に示す。ただし①②③④や⑩などのPROC文は仕様には省略しているものとする。

```
SORT_FILE(OBJ:NYUSHUKKO_FILE,INSTR:HINMEI,
          MODE:SHOJUN,GOAL:NYUSHUKKO_FILE_1);
GROUP_RECORD(SO:NYUSHUKKO_FILE_1,
             OBJ:NYUSHUKKO_RECORD,INSTR:HINMEI,
             GOAL:NYUSHUKKO_FILE_2);
OPEN_FILE(OBJ:NYUSHUKKO_FILE_2,MODE:READ);
OPEN_FILE(OBJ:HOKOKUSHO_FILE,MODE:WRITE);
REPEAT(UNTIL:EOF(OBJ:NYUSHUKKO_FILE_2),
       OBJ:READ_RECORD(OBJ:NYUSHUKKO_RECORD,
                       SOURCE:NYUSHUKKO_FILE_2);
           :=(HINMEI(OBJ:BUPPIN_RECORD,
                     ATTR:*),
              HINMEI(OBJ:NYUSHUKKO_RECORD,
                     ATTR:*));
           :=(HENDORYO(OBJ:BUPPIN_RECORD,
                       ATTR:*),
              0);
           REPEAT(UNTIL:EOF(OBJ:NYUSHUKKO_RECORD),
              OBJ:
                IF(COND:=(NYUSHUKKOKODO(OBJ:
                            KOMOKU_RECORD,ATTR:*),
                          "NYUKO"),
                   OBJ:ADD(OBJ:NYUSHUKKORYO(OBJ:
                             KOMOKU_RECORD,ATTR:*),
                           GOAL:HENDORYO(OBJ:
                             BUPPIN_RECORD,ATTR:*))
                  );
                IF(.................));
           WRITE_RECORD(OBJ:BUPPIN_RECORD,
                        GOAL:HOKOKUSHO_FILE)
       );
CLOSE_FILE(OBJ:NYUSHUKKO_FILE_2);
CLOSE_FILE(OBJ:HOKOKUSHO_FILE);
```

参考文献

（1）西田、藤田、高松：ライブラリモジュールのリンクの手法による仕様の詳細化と誤りの検出、情報処理学会論文誌、Vol.28,No.5,pp.489-498 (1987)

（2）西田、高松、谷：要求仕様における日本語表現と形式表現間の相互変換、情報処理学会論文誌、Vol.29,No.4,pp.368-377 (1988)