6B-8

PROLOGとREDUCEの統合による問題解決システム

富森 博幸 高松 忍 西田富士夫
大阪府立大学工学部

1. まえがき

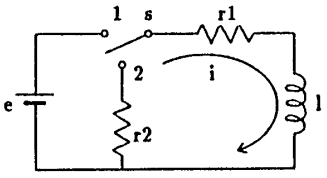
本研究では、回路等に関して与えられた方程式に対しPROLOGと数式処理システムREDUCEを用いて、解析解を自動的に求め、定性的な性質を明らかにするとともに、解析解をCコンパイラでコンパイルして数値解を能率的に求め、これをグラフなどに表示するシステムを試作した。

2. 属性表現

例えば次の問題を考える。
「始めにスイッチsが1に閉じ、1を通る電流iの初期値がi= e/r1であった。t=0でスイッチsを2に閉じるとき、微分方程式

l di/dt+(r1+r2) i=0 (1)

によりiの過渡電流を求めよ。」



この問題をPROLOGにより解く方法を考える。PROLOG等の機械的求解法では一般に反駁法を用いるが、問題の適切な表現法が重要である。いろいろの種類の問題を解く場合には、問題に含まれる関係や項などの種類や形に関する属性が解決の重要なキーになることが多い。このような点から、次のような属性表現を導入する。

述語名(obj:(ターム式,
属性名$_1$: 属性値$_1$, ・・・,
属性名$_n$: 属性値$_n$), ・・・) (2)

ここに、述語名は、変換関係などobjectとgoal間の関係などを表わし、各述語の属性名としては、タームの式が含む変数(var)、クラス(class)、形(form)などを設ける。クラスの属性は関係の種類などを表わし、線形微分方程式(li_diff)や整方程式(poly)などの属性値をとり、形(form)の属性は、注目する変数に関するターム式の形を表わし、一般形(gen)、解形(sol)などの属性値をとる。

例えば(1)に対しては

```
┐ equat_transient_solve(
    obj:(l*diff(i,t)+(r1+r2)*i=0,
         init(i,e/r1),class:li_diff,
         var:i,t,form:gen),
    go:(OUT,var:i,t,form:sol(i))) (1a)
```

のように表わすことができる。また、知識ベースには次のようなルールを設ける.

```
equat_transient_solve(
     obj:([IN,IN1],class:li_diff,
          var:Y,t,form:gen),
     go:(OUT,var:Y,t,form:sol(Y)))
:- define(IN1),lap_transform(
     obj:(IN,class:li_diff,var:Y,t,
          form:gen),
     go:(OUT1,class:deg1_alg,lap,
          var:X,s,form:gen)),
   alg_equat_solve(
     obj:(OUT1,class:deg1_alg,lap,
          var:X,s,form:gen),
     go:(OUT2,class:lap,var:X,s,
          form:sol(X))),
   inv_lap_transform(
     obj:(OUT2,class:lap,var:X,s,
          form:sol(X)),
     go:(OUT,var:Y,t,form:sol(Y))).(3)
```

A Problem Solving System by Integration of PROLOG with REDUCE
Hiroyuki TOMIMORI, Shinobu TAKAMATSU and Fujio NISHIDA
University of Osaka Prefecture

(3)の下位のルールを以下に示す。

```
lap_transform(
     obj:(IN,class:li_diff,var:Y,t,
          form:gen),
     go:(OUT,class:deg1_alg,lap,
          var:X,s,form:gen))
:- lap_trans(obj:IN,go:OUT).          (4a)

inv_lap_transform(
     obj:(IN,class:lap,var:X,s,
          form:sol(X)),
     go:(OUT,var:Y,t,form:sol(Y)))
:- lap_trans(obj:OUT,go:IN).          (4b)

alg_equat_solve(
     obj:(IN,class:deg1_alg,SUB_CLASS,
          var:X,SUB_X,form:gen),
     go:(OUT,class:SUB_CLASS,var:X,
          SUB_X,form:sol(X)))
:- solve_deg1_dimn(
     obj:IN,go:OUT,sol:X).            (4c)
```

(4a)は、Yに関する線形微分方程式INから、ラプラス変換によりYのラプラス形Xに関する1次整方程式の一般形OUTが得られることを表わす。同様に、(4b)は逆ラプラス変換のルールを表わす。(4c)は、Xの1次整方程式の一般形INからXの解形OUTを求めるルールを表わし、solve_deg1_dimnはその処理手順を表わす。

3. 統合システムの構築

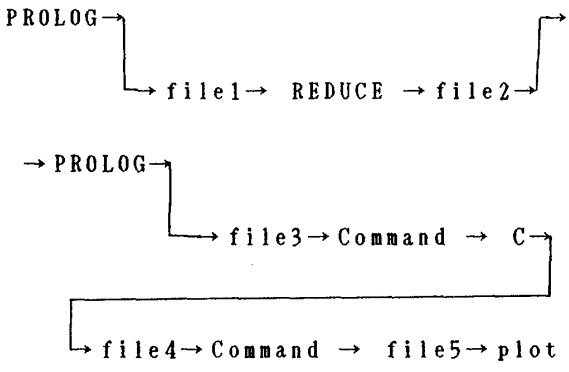
与えられた式や処理の途中で生成された式が、ある属性や特徴をもつかどうかは、これらに対応する構文解析により一般に同定することができる。

計算機処理の迅速性、確実性などの観点から、将来、各専門分野における解析や設計における問題解決の中、定型的なものは逐次数式処理ルーチンで置き換えられて行くものと思われる。現在においても、問題解決の中、適用可能なルールの探索や問題の要求に沿うルールのリンクはPROLOGで行なうことができる。また、高度で複雑であるが定型的な処理はREDUCEなどの数式処理システムで行なうことができる。それで両者を統合して数式処理など定型的で複雑な処理はPROLOGからREDUCEを呼んで問題の解析解を求め、解の性質の定性的吟味を可能にするとともに、さらに数値解をグラフに表示する。グラフ表示には過渡特性やボード線図などを対象に作図コマンドに何個かのシステムコマンドを作成した。

ここで、現在の問題解決システムの環境を以下に示す。

◇コンピュータ:　SONY NEWS NWS-821
◇オペレーティングシステム:
　NEWS-OS release 2.2 (UNIX)
◇使用プログラミング言語又はシステム
(1)C-Prolog version 1.5 and 1.5+
(2)REDUCE version 3.3
(3)C-コンパイラ

全体的な処理の流れを下図に示す。PROLOGは常時LOADしたまま、PROLOGからREDUCEやUNIXのシステムコマンドを呼び出す。PROLOGとシステム間では、ファイルの入出力を介してデータの受渡しを行う。



処理の流れ

(1a)から(3)を用いて数値解をグラフとして出力する場合、先ずPROLOGで(1a)のラプラス変換を求め、求める電流など未知量に関する1次方程式(一般に多元)を解くためにこの方程式の部分をfile1に出力する。次に方程式をREDUCEで解いてその解をfile2に出力してPROLOGに返しラプラス逆変換を行ない、解をfile3に出力する。さらに数値解を求めるため、パラメータ指定後Cによるコンパイル出力をfile4に送り、COMMANDにより実行した数値計算結果をfile5に出力しこれをplotによりグラフに変換する。

参考文献

藤田, 高松, 西田: 抽象化された変換規則関係表による問題解決法, 電子通信学会論文誌, Vol.J65-D, No.2, pp.250-257 (1982)