B-20　オブジェクトフロープログラム自動作成方法への一提案

One Proposal to The Object Flow Program Automatic Creation Method

平嶋史武†　池本悟‡　山口大輔‡　小田智大†　島田馨†　永井正武‡
Fumitake Hirashima Satoru Ikemoto Daisuke Yamaguchi Tomohiro Oda Kaoru Shimada Masatake Nagai

1．はじめに

近年,コンピュータの処理内容が複雑化,多様化および肥大化の一歩を辿っている.しかし,プログラミング手工業の時代が依然として続いている.そのため,ソフトウェアを構築する際にかなりの人手による労働力が必要となっている.そこで,本論文ではオブジェクトフローという概念を導入し[1],[2],容易かつソフトウェア及びハードウェアのセマンティックギャップを埋める，本格的なプログラム自動作成の一方法を提案する.

2．提案概念の基礎理論

本提案のオブジェクトフロープログラム自動作成法は以下の基礎理論により実現される．オブジェクトフローとは,データと機能で構成されたオブジェクト同士の関係を,処理フロー形式化することによって,ソフトウェアの生産性を向上させる技法である.

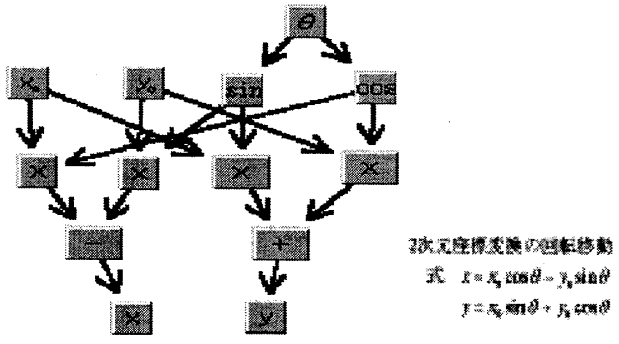

図1　オブジェクトフローの例

2.1 グラフ理論を導入した構文解析

いままでの構文解析では構文木または逆ポーランド記法の使用が一般的であるが，本提案では数理的処理を実現するために，グラフ理論を導入し図2のように展開する.

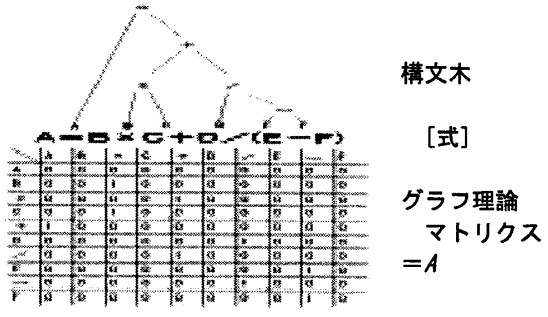

図2 従来の構文木とグラフ理論を用いた構文解析

†(株)ソフテック, SOFTECH CO.,LTD.
‡帝京大学理工学部, School of Science and Engineering, Teikyo University

2.2 ISM 階層化分析

データと機能で構成されたオブジェクト同士の関係を,処理フロー形式化することによって,ソフトウェアの生産性を向上させる技法である.図2より，導入されるグラフ理論の行列をAとおけば，次のように表せる.

$$A = \begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & a_{ij} & \vdots \\ a_{n1} & \cdots & a_{nn} \end{bmatrix} \quad (1)$$

ただし,

$a_{ij} = 1 \quad iRj \quad :i,j$間に関係がある.
$a_{ij} = 0 \quad \overline{iRj} \quad :i,j$間に関係がない.

Aに単位行列Iを足したものをBとする.

$$B = A + I \quad (2)$$

$$B \neq B^2 \neq \cdots \neq B^{n-1} = B^n \equiv T \quad (3)$$

このときのTを可達行列という.

3．システムの概要

予め登録してあるオブジェクトフロー型，制御オブジェクトを利用して視覚的に検索を行うことによるプログラム作成も可能だが，本提案法ではプログラム自動作成のためシステム自体が図3のようにそのオブジェクト同士を関連付けし,ISM 階層化分析で構造化することにより，オブジェクトフロープログラムの生成を行い，ソースプログラムおよび，ネットリストの出力を得ることができる.

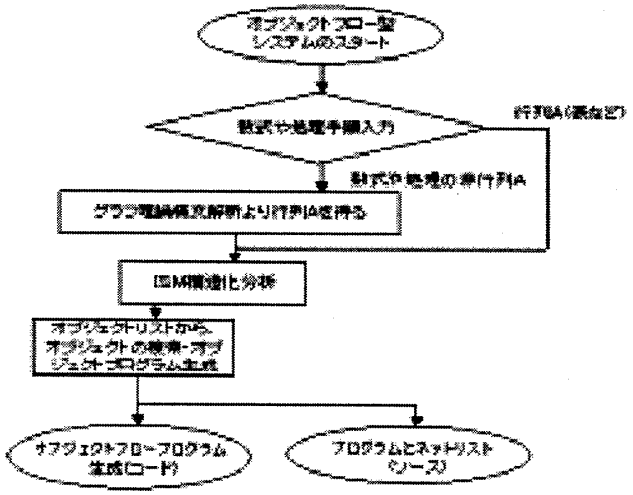

図3 オブジェクトフロープログラム生成処理フローの概念

4．実現例

本システムのメイン画面を図4に示す．本論文で提案しているシステムを使って実際にプログラムの自動作成を行ってみる．例として根の公式(2次多項式)の場合を示す．

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a} \qquad (4)$$

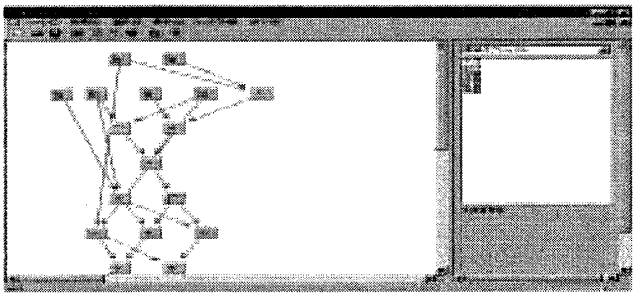

図4　本システムのメイン画面

実現例の流れは以下の通りである．

(1)関係行列A

式を×，＋，－，√などの要素がどのような関係になっているか，図5のような行列Aを得る．

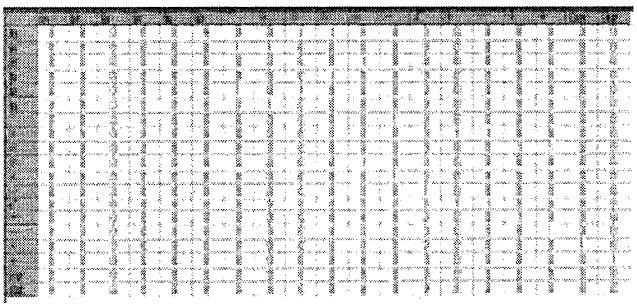

図5　作成された関係行列

(2)可達行列T

フェーズ(1)で作成した行列Aに式(3)を適用し，可達行列Tを得る．

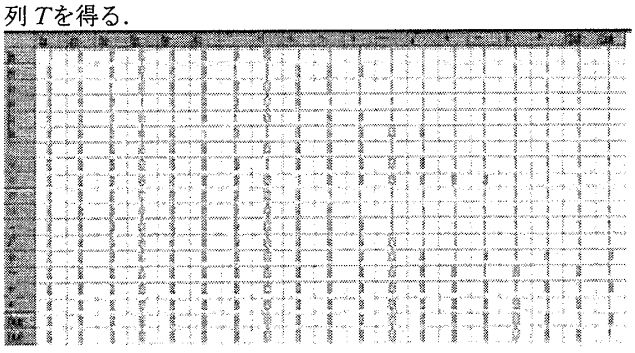

図6　作成された可達行列

(3)有向グラフ

可達行列より構造を表す有向グラフを図7のように得ることができる．ただし，これは可視化目的に使用されるものである．

(4)プログラムの作成

図8に示すように，ＩＳＭ階層化分析で得られたデータからプログラムを作成する．

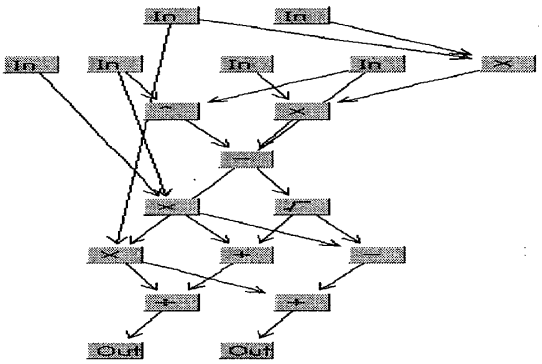

図7　本システムで作成された有向グラフ

```
#include <iostream.h>
#include <math.h>
void main()
{
    double obj1;
    double obj2;
    double obj3;
    double obj4;
    double obj5;
    double obj6;
    double obj7;
    double obj8;
    double obj9;
    double obj10;
    double obj11;
    double obj12;
    double obj13;
    double obj14;
    double obj15;
    double obj16;
    double obj17;
    double obj18;
    double obj19;
    cin >> obj1;
    cin >> obj4;
    cin >> obj2;
    cin >> obj3;
    cin >> obj5;
    obj10 = obj1 * obj4;
    obj7 = pow(obj2, obj5);
    obj11 = obj3 * obj10;
    cin >> obj6;
    obj12 = obj7 - obj11;
    obj8 = obj2 * obj6;
    obj13 = sqrt(obj12);
    obj9 = obj1 * obj5;
    obj14 = obj8 + obj13;
    obj15 = obj8 - obj13;
    obj16 = obj9 / obj14;
    obj17 = obj9 / obj15;
    obj18 = obj16;
    cout << obj18 << endl;
    obj19 = obj17;
    cout << obj19 << endl;
}
```

図8　得られたソースプログラム

5．評価

オブジェクトフロー型開発・デバッグシステムに数値構造化技法を導入することにより[3]，全体のオブジェクトフローの関係を階層化し，ソースプログラムおよびネットリストの自動生成を容易に得ることができた．

6．むすび

今回の試みではグラフ理論を構文解析に導入することにより，数理処理サブシステムを実現し，ソースプログラムの自動作成をすることが可能となった．このシステムを実装・稼動することにより，ソフトウェア生産性の向上を図ることができる。また，ソフトウェア製作に必要なＣソース等とハードウェア製作に必要な論理式(ネットリスト)をも出力することによって，ソフトウェアとハードウェアのセマンティックギャップの短縮化が実現可能となった。

7．参考文献

[1] 平嶋史武, 島田馨, 伊藤貴雄, 永井正武:オブジェクトフロー型ソフトウェア開発方法への一提案, 情報処理学会第62回全国大会, 8Q-7, 2001

[2] 伊藤貴雄, 平嶋史武, 黒坂功, 永井正武:オブジェクトフロー型概念によるソフトウェア開発技法への一提案, 情報処理学会　第62回全国大会, 5Z-4, 2001

[3] 平嶋史武, 山口大輔, 池本悟, 島田馨, 永井正武:オブジェクトフロー型ソフトウェア開発方法への一提案(その２)