社団法人　電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
TECHNICAL REPORT OF IEICE.

グラフ文法による図と表の処理の定式化

有田　友和†　土田　賢省††　杉田　公生†††　夜久　竹夫†

† 日本大学 文理学部 情報システム解析学科
†† 東洋大学 工学部 情報工学科
††† 東海大学 理学部 数学科

あらまし　機械的描画に関し，グラフ文法に基づいたプログラム図の構文的定義と処理を扱う．本稿では [14] を元にその後の結果について解説する．まず，構造プログラム図のような階層的図に対する属性 NCE グラフ文法を紹介する．さらに，モジュール型の表及び格子状の表に対して，それぞれ文脈自由と文脈依存の属性グラフ文法を提案する．属性は，それぞれの図表の機械的描画のために用いられる．また，我々は，これらの文法に基づいた統一的な図表処理方法を紹介する．これらの結果は，一般的な図表処理に応用できると考えられる．

キーワード　順位グラフ文法, 属性グラフ文法, 図表処理

Application of Graph Grammars to Program Diagrams and Tabular Form Processing

Tomokazu ARITA†, Kensei TSUCHIDA††, Kimio SUGITA†††, and Takeo YAKU†

† Dept. Comp. Sci. & Sys. Analy., Nihon University
†† Dept. Inf. & Comp. Sci., Toyo University
††† Dept. Math., Tokai University

Abstract　We deal with syntactic definitions and processing of program diagrams based on graph grammars with respect to the mechanical drawing. In this paper, we provide a survey of our recent results based on [14]. We introduce an attribute NCE graph grammar of hierarchical diagrams such as structured program diagrams. We also introduce attribute context-free and context-sensitive NCE graph grammars for nested and tessellation diagrams, respectively. Attribute rules are used for the mechanical drawing. Furthermore, we introduce an integrated diagram processing method based on NCE graph grammars. The results could be applied to general diagram processing.

Key words　precedence graph grammar, attribute graph grammar, graph parser

1. はじめに

我々は，機械的描画に関するグラフ文法に基づく構文的定義と処理を扱う．

グラフ文法の幾つかのモデルとその特性は，Franck [1], Vigna [2], Rozunberg [10] など多くの研究者により調査されている．近年，NCE グラフ文法は，人工的オブジェクトに対し特にデザインや解析等の合理的なモデルとして考えられてきた．

また，制約アルゴリズム等を使用したグラフエディタやグラフ描画システムは開発されてきている．

グラフ文法の定理の開発に伴いDiaGen [20] のような構文的グラフ処理システムも開発されている．それらの大きなプロジェクトとしてAPPLIGRAPH のような大規模なプロジェクトも開発されている．[3] においては，Nagl がIPSEN システムを提案している．

1978年に，夜久と二木はプログラム流れ図Hichart の書法を提案した．Hichart 図の記号や全体構造は，他のプログラム図言語のように使用される．グラフ処理に関する我々のプロジェクトはKEYAKI (see e.g. [8], [12], [15]) と名づけられている．1987 年に，我々は，フローチャート処理のための構文抜きの特徴を紹介した ([5])．1996 年に，我々は，D. Vigna の文脈自由グラフ文法 [2], [8] に基づいた属性グラフ文法により Hichart 図の編集方法を提案した．我々のグラフ文法の結果を使用したCAI システムは, [9] で紹介し，ルールベースのプログラムの可視化のためのシステムは1998 年に提案された [13].

本研究では，我々はプログラム仕様書内で使用する 3 種類の図に対して考える．それらは，(1) 構造型プログラム図のような階層構造図, (2) 表形式のプログラム仕様書のためのモジュール型の表，そして (3) 変数表のような，格子型の表である．本研究の目的は，3 種類の図表を生成するグラフ文法の種類を特徴付けることと，そのグラフ文法を使用した図の処理方法を提案することである．

我々は，モジュール型と格子状の表に対しそれぞれ属性文脈自由グラフ文法と属性文脈依存グラフ文法を紹介する．また，NCE グラフ文法に基づいた統一的な図表の処理方法について紹介をする．これらの結果は，一般的な図に応用可能であると考えられる．

本稿では [14] を元にその後の結果 [16]〜[18] について解説する． 2 章では，今回扱う図表について述べる． 3 章において，階層型プログラム図を定式化するグラフ文法について述べる． 4 章において， 2 種類の表を定式化するグラフ文法について述べる． 5 章では，これらのグラフ文法を応用した図表処理システムを紹介する．最後に，本研究の結果について述べる．

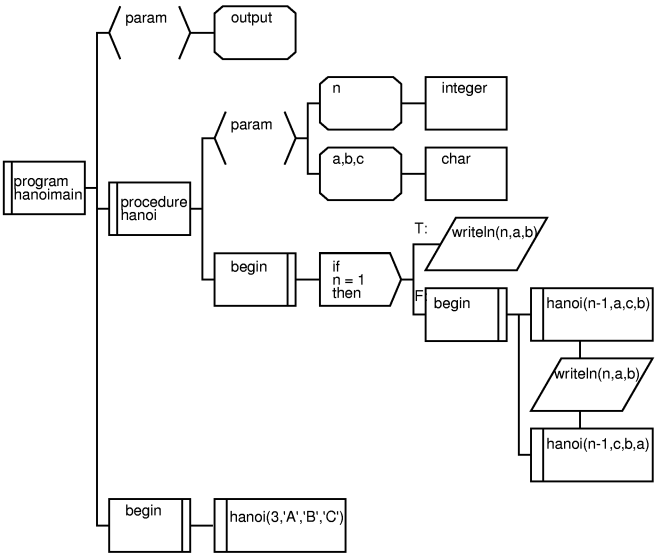


図 1　Hichart 流れ図の一例

2. プログラム流れ図と仕様書

この章では，我々はソフトウェアの可視化に必要な図について述べる．我々は，２種類の図表について考える．一つは，プログラム流れ図のための階層構造図，もう一つはプログラム仕様書のための表形式の図である．

2.1 プログラム流れ図のための階層的な図

プログラム流れ図記述言語 Hichart (Hierarchical flow CHART description language) [5]. を紹介する．Hichart は，木構造型プログラム流れ図である．図 1 は，ハノイの塔のための Hichart 流れ図の例である．

Hichart 流れ図は次の特徴をもつ．(1) 処理はそれぞれ一致するセルの中に記述する．(2) セルは階層的に配置する．(3) セルはお互いに線で接続される．(4) 線は水平，もしくは垂直方向に走り，互いに重ならない．(5) 図は平面に表示される．

階層構造図の形式的な定義は 3 章において言及する．

2.2 プログラム仕様書のための表形式の図

我々は，ISO6592 に基づいたプログラム文書化言語 Hiform を紹介する．

ISO は，プログラム文書に関してISO6592 でガイドラインを発行した．ISO6592 に付随する付属文書 A,B,C 中にはプログラムの文書化に関する項目の記述がされている．我々は，ISO6592 [4] の項目を考慮に入れ，付属文書内で定義された項目をすべて含む Hiform96 を提案した．

Project Code: hanoi_main	A 5
Program Name: procedure hanoi	**Program Specification-1** p
Library Code: cs-2000-01	Version: 2.1
Author: Tomokazu Arita	Original Release: 1999/12/25
Approver:	Current Release: 2000/1/31

Problem Description:

How to discribe the structure of this program.

Problem Supplementary Information
(Theoretical Principles, Methods and References):

Theoretical Principles: Hichart Diagram

Problem Solution:
1.Conventions and Terminology 2.Principles and Algorithms

1. Convention : Hichart Diagram

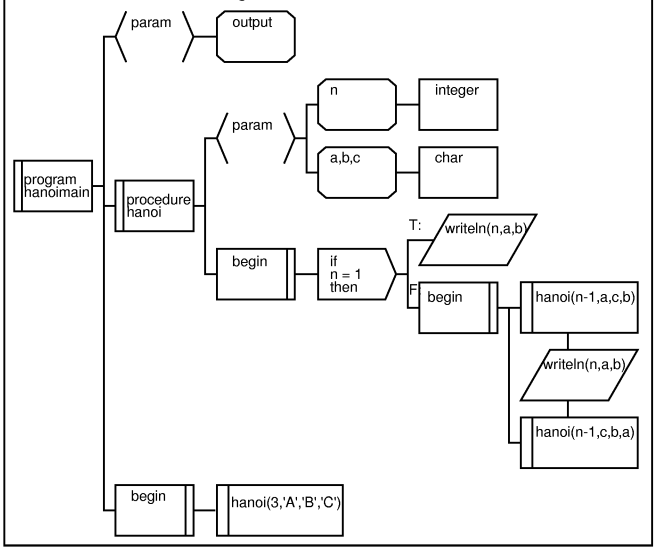


図 2　Hiform 仕様書の一例

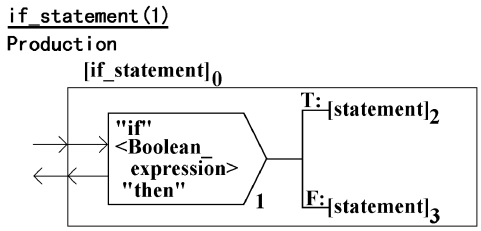
Hiform は，本来は教育の場でのソフトウェア開発を容易にする目的で開発された．Hiform 文書は表形式の仕様書の集まりである．表形式を採用する目的は，“何が得られている情報なのか”，“どの情報が欠けているのか”，“プロジェクトの進行はどのようになっているのか”，“ソフトウェアの開発管理の仕方”等を一目で理解するためである．その他に文書や図のような様々な記述形式を表形式ならば含めることができるという特徴を持つ．

Hiform [12] は，17 種類の様式により定義される．様式間の作成順序は文脈自由な文字列文法により定義されている [9]．表内部のセルの配置や描画構造は 4 章で言及する．

図 2 において Hiform のプログラム様式の例を示す．

3. 階層構造図に対する属性グラフ文法

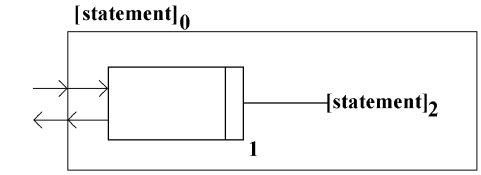
この章では，Hichart 流れ図で使用する階層構造図に対する属性グラフ文法 HCGG について述べる．HCGG は，グラフ文法的定義と属性規則を提供する．グラフ文法部分は文脈自由 NCE グラフ文法 [16] に



Semantic Rules

top(2)=top(0)	cl(2)="T:"
top(3)=bottom(2)+GapY	cl(3)="F:"
x(1)=x(0)	id(1)=id(0)
x(2)=x(0)+w(1)+GapX	id(2)=id(1)+1
x(3)=x(0)+w(1)+GapX	id(3)=id(2)+nc(2)
y(0)=(y(2)+y(3))/2	nc(0)=1+nc(2)+nc(3)
bottom(0)=max(bottom(1),bottom(3))	

w(1)=MinW
h(1)=get_height(["if",<Boolean_expression>,"then"])
cell(1)="exclusive_selection"
string(1)=get_str(["if",<Boolean_expression>,"then"])
lines(1)=get_line(1,[2,3])



Semantic Rules

top(2)=top(0)	id(1)=id(0)
x(1)=x(0)	id(2)=id(1)+1
x(2)=x(0)+w(1)+GapX	nc(0)=1+nc(2)
y(0)=y(1)	
y(1)=y(2)	
bottom(0)=max(bottom(1),bottom(2))	
bottom(1)=y(1)+h(1)	

w(1)=MinW
h(1)=MinH
cell(1)="continuous_iteration"
string(1)=get_str([" "])
lines(1)=get_line_begin(1,[2])

図 3　HCGG の生成規則の一部

より定義されている．また，属性規則はそれぞれのセルのサイズや座標といった図のレイアウトのための属性を計算するために用いられる．

文法 3.1　HCGG は，Hichart 対応階層構造図に対する文脈自由属性 NCE グラフ文法である．

図 3 では HCGG の生成規則の一部を紹介する．

命題 3.2　HCGG の属性は線形時間で評価される．

HCGG による Hichart 図の導出プロセスは図 4 で例を示す．

4. 表形式の図のための属性グラフ文法

この章では，表形式の図 Hiform の構文的形式化を扱う．Hiform 文書の形式的定義は 2 種類の文法によ

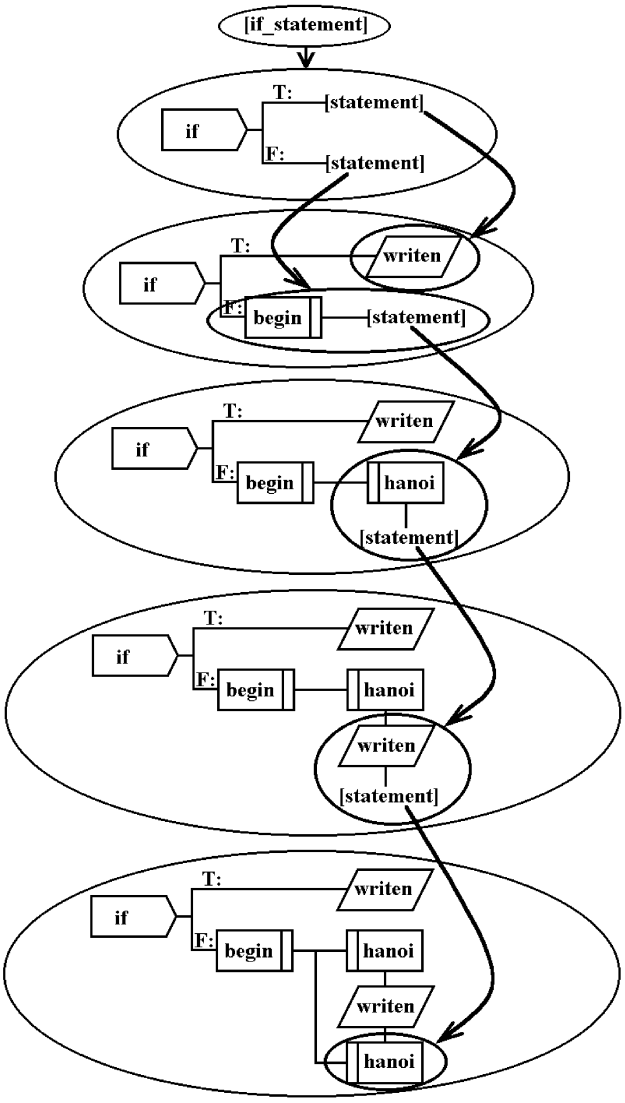


図 4　HCGG による Hichart 図の導出過程

り構成される．一方は各様式間の順序を提供し，他方は，各様式内の項目間の配置を提供する．前者は，文脈自由文法 [9] で，後者は属性グラフ文法で定義される．

ここでは，Hiform のグラフ的構文定義と描画条件用属性規則を定式化する属性グラフ文法について述べる．Hiform のプログラム仕様書は，そのグラフ文法に基づいた位置情報をもつ完全グラフにより表される．

我々は，図 5 でプログラム仕様書に対する複合図とマーク付きグラフの例を示す．図 5 において，マーク付きグラフの辺のラベルは辺の両端の頂点間の関係を表す．その関係とは，ラベル”in”が”within”を表し，ラベル”ov”が”over”を，ラベル”lf”が”left of”をそれぞれ表す．

文法 4.1　HNGG は Hiform のモジュール型図に対

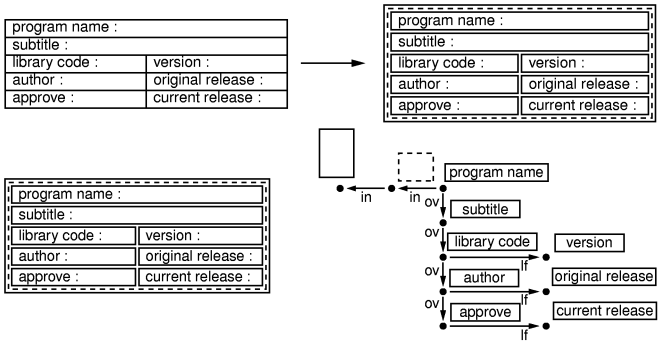


図 5　仕様書とそれに対応するマーク付きグラフ: (1) 仕様書 (左上) とそれに対応する項目の配置を表す複合図 (右上), (2) 複合図 (左下) とそれに対応するマーク付きグラフ (右下)

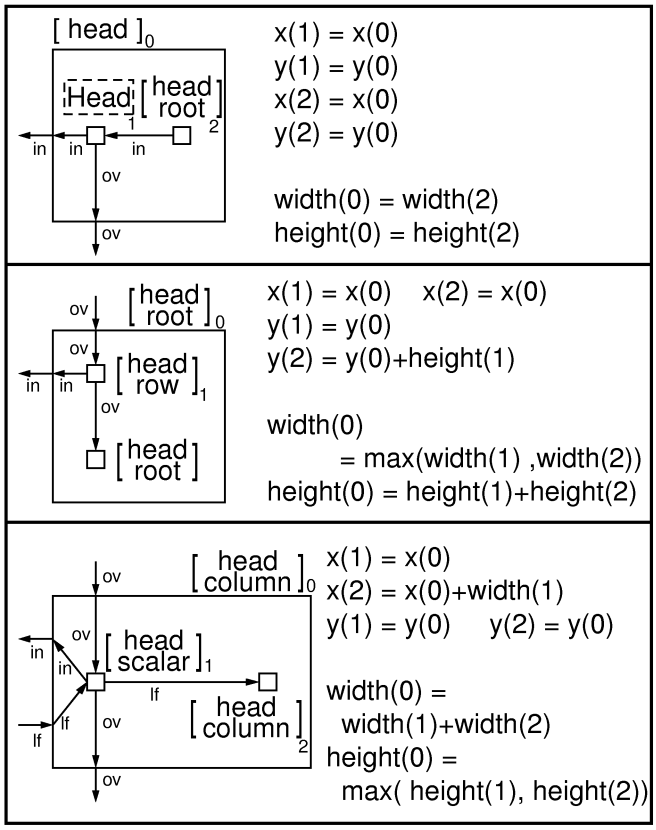


図 6　HNGG の生成規則の一部

する属性グラフ文法である．

図 6 に HNGG の生成規則の一部を示す．HNGG は 280 個の生成規則と 1248 個の属性規則で構成される．

HNGG の構文は，次の特性を持つ．

命題 4.2　文法 HNGG は順位グラフ文法である．

すなわち，Hiform の構文解析アルゴリズムは部分的に Franck の線形時間構文解析アルゴリズム [1] により与えられる．

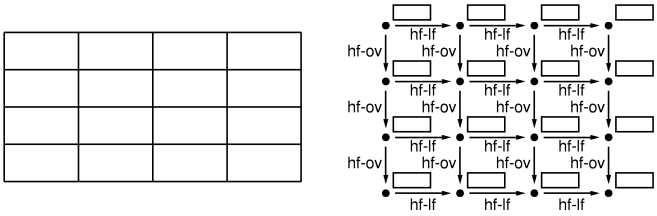


図 7　格子状の表とそれに対応するグラフ

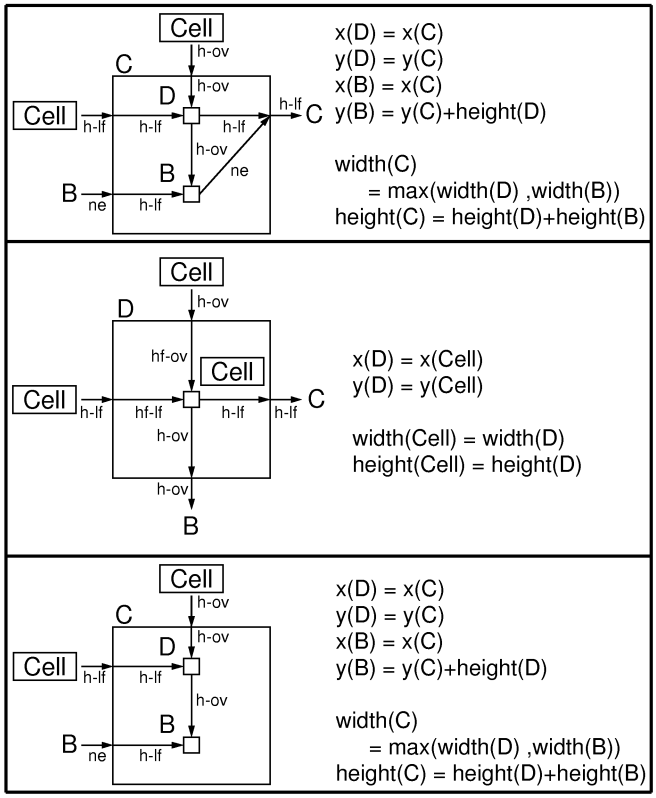


図 8　HTGG の生成規則の一部

次に，Hiform 仕様書内で使用する変数表のような格子型の表について考える．次の図 7 は格子状の表の例である．我々は，この格子状の表に対して文脈依存の属性 NCE グラフ文法を提案する．

文法 4.3　HTGG は Hiform 仕様書内の格子状の表に対する文脈依存属性 NCE グラフ文法である．

HTGG は 56 個の構文規則と 256 個の属性規則により構成される．次の図 8 は，HTGG の生成規則の一部である．HTGG により生成される格子状の表は，行依存もしくは列依存の表である．

5.　図表処理システム

この章では，KEYAKI-CASE2000 と呼ばれる図表処理システムについて述べる．KEYAKI–CASE2000 は次のようなコンポーネントで構成される．(1) Hichart プログラム図編集コンポーネント (HichartED), (2) Hichart プログラム図フィルタリングコンポーネント (HiTS), (3) プログラム変数解析コンポーネント (LIVE), (4) Hiform 図処理コンポーネント (HiformED).

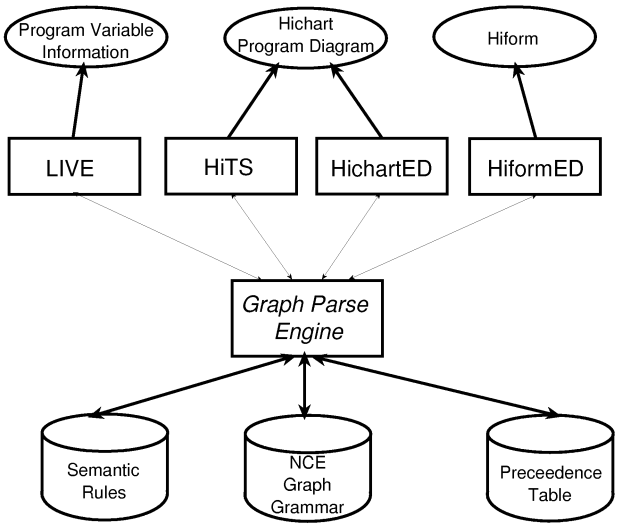


図 9　システムの構造

これらのコンポーネントは 3 章と 4 章の理論に基づき開発される．図 9 は，KEYAKI–CASE2000 のシステム構造である．

ここで，各コンポーネントの詳細について述べる．

5.1　HichartED [16]

Hicahrt プログラム図編集コンポーネントは，HichartED と言う．HicahrtED は，属性グラフ文法 [10] に基づいた構文指向エディタである．Hicahrt の編集操作は，Hichart プログラム図を形式的に定義する属性グラフ文法の生成規則により定義されている．この事は，エディタにより生成される図は全て文法的に正しい図であることを保証するだけでなく，エディタによるプログラムの生成と編集操作において構文的エラーが起こらないことを保証する．

5.2　HiTS

Hichart プログラム図フィルタリングシステムは，HiTS (Hichart Translation Service) [6], [7] と呼ばれる．HiTS の理論的研究は，3 章に記述されている．HiTS は，木の美的描画の理論研究の結果を取り入れた木構造型プログラム流れ図 (Hichart) の自動生成をするプログラム流れ図処理システムである．このシステムは，プログラム流れ図を自動生成し，プログラムのデータ構造やコントロールフローを可視化する際にユーザーを補助する．さらに，HiTS を使用した流れ図はプログラムソース (C,Pascal) と同様にプログラム仕様も作成する．

図 10 において，Dikstra アルゴリズムを使用した

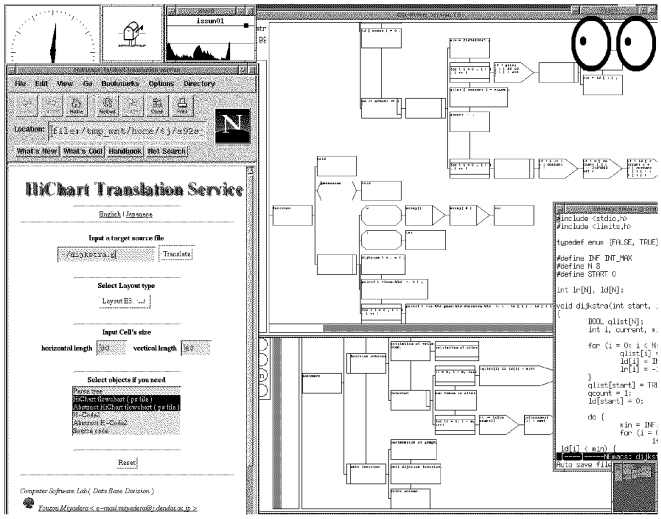


図 10　HiTS の実行画面

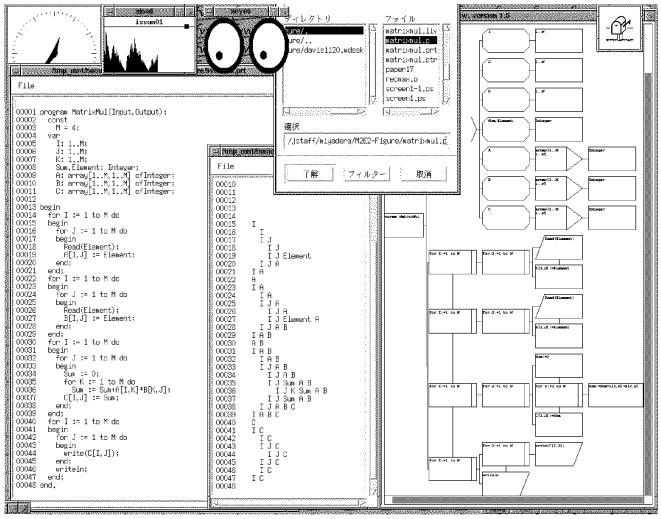


図 11　LIVE の実行画面

最短パスを決める C プログラムに対するプログラム図 (右上のウィンドウ) とプログラム概要図（右下のウィンドウ）である．そして，図 11 は，HiTS により生成された Pascal プログラムに対するプログラム図 (一番右のウィンドウ) である．このツールの Pascal バージョンは 26,500 ステップで実装され，C バージョンは，約 30,000 ステップで実装されている．

5.3　LIVE

我々は LIVE と呼ばれるプログラム変数解析コンポーネントも扱う．LIVE は，プログラムのモジュール化を支援するために開発された [7]．LIVE は，データフロー解析理論の研究に基づき開発される．HiTS は，同時に LIVE の情報を参照することにより適切なプログラムモジュールが作成可能かどうか判定する際にも使用することができる．

LIVE を適用した結果を図 11 に示す．スクリーンの左側のウィンドウは，行列の積を実装するプログラムソースコードを表示した例である．中心のウィンドウはアクティブな変数を示し，LIVE により導出される．そして，右のウィンドウはその Hichart 図である．このコードが，30 行目と 40 行目でモジュールに分割することが可能であることが LIVE の結果から明らかである．

このツールは，Pascal で記述され，約 15,000 ステップで実装されている．

5.4　HiformED

HiformED は，プログラム仕様書 Hiform 編集支援コンポーネントである．HiformED は，4 章の HNGG に基づいた構文解析システム上に構築される．現在，Hiform 仕様書の構造を解析する構文解析方法と，表描画に必要な属性評価方法は [18] が提案されており文

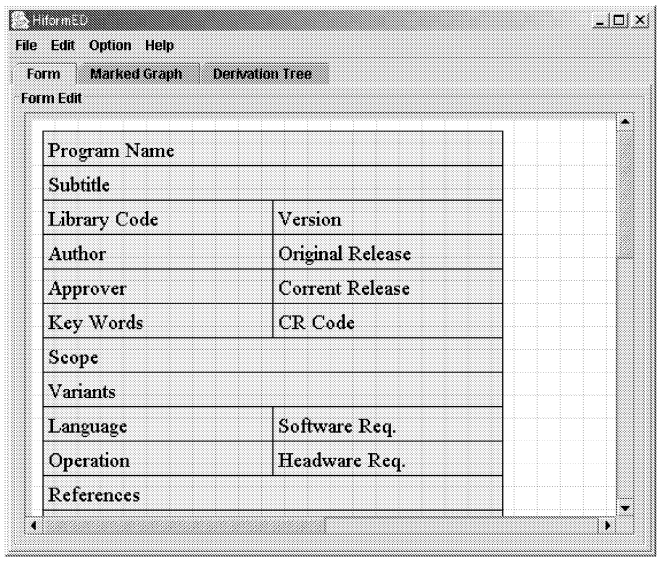


図 12　HiformED Parser の実行画面

法に基づく仕様書の編集操作方法についても [17], [19] において提案されている．現在，構文解析システムは実装されており，仕様書に対する XML ソースを出力することが可能である [21]．この XML ファイルは，属性評価に基づいて生成され，Hiform 仕様書用 XSL ファイルを用いて，IE 等の XML ブラウザで表示が可能である．編集操作の実装については，現在開発中である．

図 12 は，構文解析システムの実行画面である．

6.　結　　論

我々は，属性 NCE グラフ文法による図表処理の統一的なモデルを提案した．我々は，Hichart プログラム図に対して 67 個の生成規則と 723 個の属性規則を持つ属性 NCE グラフ文法を提案した．また，ISO6592 に基づく 137 項目を含むプログラム仕様書

に対して280個の生成規則と1248個の属性規則を持つ属性文脈自由NCEグラフ文法を提案した．さらに，我々は，格子状の図形に対して属性文脈依存グラフ文法を提案した．この文法は，辺依存のNCEグラフ文法である．また，それに基づいた図表の処理方法を考慮に入れた．

文　献

[1] Reinhold Franck, A Class of Linearly Parsable Graph Grammars, *Acta Infomatica* 10, 175-201, (1978).

[2] Pierluigi Della Vigna and Carlo Ghezzi, Context Free Graph Grammars, *Inform. Contr.* 37, 207-233, (1978).

[3] G. Engels, R. Call, M. Nagl, et al., Software specification using graph grammars, *Computing* 31, 317-346, (1983).

[4] ISO6592-1985, Guidelines for the documentation of computer-based application systems, (1985)

[5] T. Yaku, K. Futatsugi, A. Adachi and E. Moriya, HICHART-A Hierarchical Flowchart Description Language-, *Proc. IEEE COMPSAC* 11 (1987), 157-163.

[6] Y. Miyadera, K. Tsuchida, and T. Yaku, A Tidy Drawing Problem on the Minimum Area for Tree-Structured Diagrams and Its Application to Program Diagrams, *IFIP Transac.* A-51, 282-287, (1994).

[7] Miyadera, Y., Tsuchiya, A., Yaku, T. and Konya, H., Network-Based Programming Language Education Environment Based on a Modular Program Diagram, *Proc. IEEE International Conference on Multi Media in Education*, 425-434, (1996).

[8] Y. Adachi, K. Anzai, K. Tsuchida and T. Yaku, Hierarchical program diagram editor based on attribute graph grammar, *Proc. IEEE COMPSAC* 21, 205-213, (1996)

[9] K. Sugita, Y. Adachi, Y. Miyadera, K. Tsuchida and T. Yaku,
Proc. of Advanced Software Mechanisms for Computer-Aided Instruction information Literacy (APEC-CIL'97), (1997)

[10] Grsegorz Rozenberg (Ed.), Handbook of Graph Grammar and Computing by Graph Transformation,World Scientific Publishing(1997).

[11] Y. Miyadera, K. Anzai, H. Unno and T. Yaku, Depth-First Layout Algorithm for Trees, *Information Processing Letters* 66, 187-194, (1998).

[12] K. Sugita, A. Adachi, Y. Miyadera, K. Tsuchida and T. Yaku, A visual programming environment based on graph grammars and tidy graph drawing, *Proc. Internat. Conf. Software Engin.* (ICSE '98) 20-II, 74-79, (1998).

[13] A. Adachi, T. Tsuchida and T. Yaku, Program visualization using attribute graph grammars, *CD-ROM Proc. IFIP World Computer Congress* 98 (1998).

[14] T. Arita, K. Tomiyama, T. Yaku, Y. Miyadera, K. Sugita, K. Tsuchida, Syntactic Processing of Diagrams by Graph Grammars, *Proc. IFIP WCC ICS 2000*,145-151 (2000).

[15] Kimio Sugita et al, Integrated Visualization Environment for Computer Science Education, Proc. IFIP WCC ICEUT2000, (2000).

[16] 宮崎征宏, 類瀬健二, 土田賢省, 夜久竹夫, プログラム図に対する描画を考慮したNCE属性グラフ文法, 電子情報通信学会技術報告書 Vol.100 No.52, 1-8, (2000)

[17] 冨山聖宣, 有田友和, 夜久竹夫, 土田賢省, 属性edNCEグラフ文法による表の構文的編集, 電子情報通信学会技術研究報告, vol.100 No.471, 1-7, (2000)

[18] T. Arita, K. Sugita, K. Tsuchida and T. Yaku, Syntactic Tabular Form Processing By Precedence Attribute Graph Grammar, *Proc. IASTED AI 2001,pp.637-642 (2001)*

[19] T. Arita, K. Tomiyama, K. Tsuchida and T. Yaku, Application of Attribute NCE Graph Grammars to Syntactic Editing of Tabular Forms, Electric Notes in Theoretical Computer Science, Vol. 50, 3, (2001).

[20] O. Koeth and M. Minas, Abstraction in Graph–Transformation Based Diagram Editors, Electric Notes in Theoretical Computer Science vol.50, 3, 2001.

[21] O. Inoue, K. Tsuchida, S. Nakagawa, T. Arita, T. Yaku, An XML Viewer for Tabular Forms for use with mechanical Documentation, *IASTED AI'02*, 2002 (to appear).