3T－2

グラフ生成システムFGSによる帰納学習システムについて

宮原 哲浩† 久保山哲二‡ 内田 智之† 山本 竜也†† 高橋 健一† 上田 祐彰†

†広島市立大学情報科学部 ‡東京大学国際・産学共同研究センター ††広島市立大学情報科学研究科

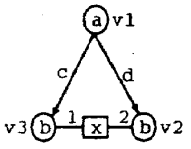
1 はじめに

グラフは、計算機科学における様々なデータを表現する手段として、重要な役割を果たしている。グラフ生成システム FGS (Formal Graph System, [4]) は、グラフを項として直接操作できる論理プログラミングシステムであり、グラフで表現できる様々なオブジェクト間の関係を論理的に表現するのに適している。つまり、FGSは、一階項(first order term)の代わりに、グラフを表す項(項グラフ,term graph)を対象とする論理プログラミングシステムである。また、FGSは、文字列を対象とする論理プログラミングシステムである基本形式体系 EFS(Elementary Formal System, [1]) を拡張したものである。一般に変数を含む2つの項グラフの最汎単一化 (mgu) は存在しないので、FGSインタプリタとは、変数を含まないゴールがFGSプログラムから導出できるかどうかを計算するものである。このインタプリタは、例としてグラフを受け取り、それらを説明するFGSプログラムを仮説として構成する帰納学習システムを実現する際の、仮説のチェックに必要である。

FGSインタプリタでは、導出の計算は、最汎単一化を計算するのではなく、変数を含まない項グラフと変数を含む項グラフとの単一化を計算することによって行う。我々は、FGSを知識表現言語として、グラフで表されるデータから知識を発見するための帰納学習システムについて研究している。単一化を枚挙するプログラム [2] をもとに、帰納学習システムの基礎となるFGSインタプリタを作成したので、本稿で報告する。

2 グラフ生成システムFGSの概要

彩色グラフ (V,E) とは、頂点集合 V, 辺集合 E, 頂点のラベル付け、及び辺のラベル付けから成るグラフである。**変数**とは、V の超辺(頂点の有限集合のこと)に、ラベル(**変数ラベル**という)を付けて、頂点に順序をつけた有限リストである。**項グラフ** (V,E,H) とは、彩色グラフ (V,E) 及び変数の有限集合 H から成るものである(図1参照)。変数を含まない項グラフを**基礎項グラフ**という。




```
(d-termgraph
   (vertex (a v1) (b v2) (b v3))
   (edge (d v1 v2) (c v1 v3))
   (variable (?x v3 v2)))
```

図1: 有向項グラフの図表示とリスト表現

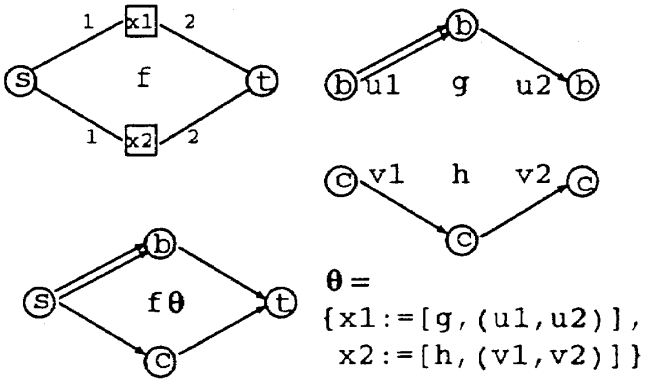


図2: 項グラフ f の代入 θ による代入例 $f\theta$

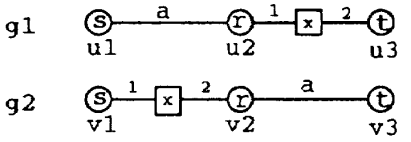


図3: 最汎単一化の存在しない項グラフ $g1, g2$

f を項グラフとする。x を変数ラベル、$(v_1,\ldots,v_m)$ を頂点のリストとするような、f に出現する変数を考える。g を頂点数 m 以上の項グラフとし、$(u_1,\ldots,u_m)$ を g の相異なる頂点のリストとするとき、$x := [g,(u_1,\ldots,u_m)]$ という形の表現を**束縛**という。項グラフ f に対してこの束縛を適用してできるグラフとは、f に出現する変数のうち、変数ラベル x を持つもの全てに対して、次の操作を行ってできるグラフである。変数ラベル x を持つ変数を f から削除し、g の頂点 $u_1,\ldots,u_m$ を、それぞれ f の頂点 $v_1,\ldots,v_m$ とみなしたグラフを f に追加して、追加する前に f にあった頂点のラベルは f でのラベルとする。

$x_1,\ldots,x_n$ を互いに異なる変数ラベルで、$g_1,\ldots,g_n$ には出現しないものとするとき、**代入** θ とは、束縛の有限集合 $\{x_1 := [g_1,\sigma_1],\ldots,x_n := [g_n,\sigma_n]\}$ である。項グラフ f の代入 θ による**代入例** $f\theta$ とは、f に対して θ 中の束縛を同時に適用してできる項グラフである。代入を説明する図2では、代入に直接関係のない頂点、及び辺ラベルは、省略している。また、σ_i が g_i の頂点集合だけから成る

An Inductive Learning System Using Formal Graph System, Tetsuhiro Miyahara†, Tetsuji Kuboyama‡, Tomoyuki Uchida†, Tatsuya Yamamoto††, Kenichi Takahashi†, Hiroaki Ueda†, †Faculty of Information Sciences, Hiroshima City University, ‡Center for Collaborative Research, University of Tokyo, ††Graduate School of Information Sciences, Hiroshima City University

束縛 $x_i := [g_i, \sigma_i]$ を、消去束縛という。消去束縛を適用すると、見かけ上は変数ラベル x_i を持つ超辺を削除しただけのグラフができる。消去束縛を含む代入を、消去代入 (erasing substitution) という。

p を n 引数の述語記号, $g_1, \ldots, g_n$ を項グラフとするとき、$p(g_1, \ldots, g_n)$ という形をした表現をアトムという。$A, B_1, \ldots, B_n$ をアトムとするとき、$A \leftarrow B_1, \ldots, B_n$ という形をした節を、**グラフ書き換え規則** (graph rewriting rule) という。**FGS プログラム**は、グラフ書き換え規則の有限集合である。

3 グラフ生成システム FGS のインタプリタ

FGS では、一般に変数を含む 2 つの項グラフの最汎単一化 (mgu) が存在しないので (図 3 参照)、導出の計算には、最汎単一化の代わりに単一化を用いる。f, g を項グラフとし、θ を代入とする。代入例 $f\theta$, $g\theta$ が、彩色グラフとして同型であり、かつ、この同型によって 2 つの代入例の変数のラベルと変数の頂点が対応するとき、θ は f と g の**単一化**であるという。基礎項グラフ (変数を含まない項グラフのこと) g と変数を含む項グラフ t との単一化は複数あるので、バックトラックをして導出する。g と t の単一化の枚挙を行う効率のよい方法はないので、g の部分グラフから、候補の代入 θ を枚挙し、g と $t\theta$ が彩色グラフとして同型なるような θ だけを単一化として、枚挙する。

プログラム中の全ての節で、本体に現れる変数が全て頭部にも現れている FGS プログラムを、**変数限定 FGS プログラム**という。**FGS インタプリタ**とは、基礎項グラフだけを含むゴール G と、変数限定 FGS プログラム P が与えられて、G が P から導出できるかどうかを計算するものである。

P.Norvig の Common Lisp による Prolog インタプリタ ([3], 11 章) を変更して、FGS インタプリタを実現した。述語記号、頂点ラベル、辺ラベルの有効範囲はプログラム全体である。変数ラベルの有効範囲はグラフ書き換え規則であり、頂点名の有効範囲は述語の引数である。導出の際に、ゴールのリストから次に導出するゴールを選ぶ計算規則として、**最左アトム**を選ぶ方法以外に、**サイズ最小のアトム**を選ぶ方法も指定できる。ここで、基礎項グラフのサイズとは頂点数と辺数のペアであり、基礎アトムのサイズとは、引数の項グラムのサイズを、頂点数と辺数ごとに合計したペアである。例えば、図 5 の基礎アトムのサイズは、(4,5) である。2 つの基礎アトム A, B は、A の頂点数が B の頂点数より小さいか、頂点数が等しくて辺数が小さい時、A は B よりサイズが小さいという。また、導出で使う代入に、消去代入を許さないように指定することもできる。インタプリタは、ゴールが導出できる時には、T を返す。最初に見つかった導出に使われた代入を返すようにも指定できる。導出できないことがわかった時には、NIL を返す。

図 6 に、FGS インタプリタの実行例を示す。応答の一部は省略している。グラフ文法の分野で知られている TTSP グラフについて、すべての TTSP グラフを受理する FGS プログラム (図 4 に図表示) を、読み込んだ後、一つの TTSP グラフを引数に持つゴール (図 5 に図表示) を導出した。

4 おわりに

グラフ生成システム FGS による帰納学習システム作成するために必要な、FGS インタプリタを Common Lisp で実現した。現在、モデル生成の観点からボトムアップ型の証明手法に基づいた FGS インタプリタも実装中である。本研究の一部は広島市立大学特定研究費 (9763,9870) の助成による。

参考文献

[1] S. Arikawa, T. Shinohara, and A. Yamamoto. Learning elementary formal systems. *Theoretical Computer Science*, 95:97–113, 1992.

[2] 宮原 哲浩, 山本 竜也, 内田 智之, 高橋 健一, 上田 祐彰. 帰納学習のためのグラフ生成システム FGS のインタプリタについて. *1998* 年度人工知能学会全国大会論文集, pages 130–131, 1998.

[3] P. Norvig. *Paradigms of Artificial Intelligence Programming: Case Studies in Common Lisp*. Morgan Kaufmann, 1992.

[4] T. Uchida, T. Shoudai, and S. Miyano. Parallel algorithm for refutation tree problem on formal graph systems. *IEICE Transactions on Information and Systems*, E78-D(2):99–112, 1995.

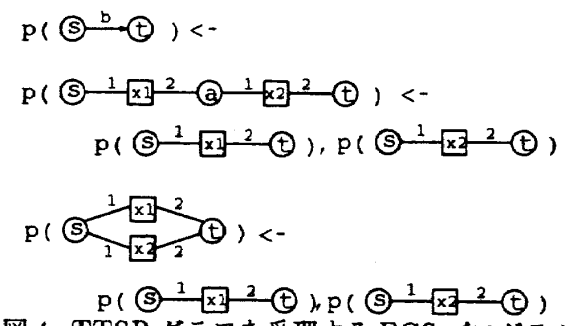


図 4: TTSP グラフを受理する FGS プログラム

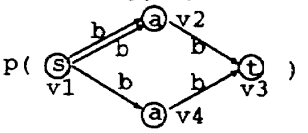


図 5: TTSP グラフを引数に持つゴール

```
>(<-(p (d-termgraph (vertex (s u) (t v))
           (edge (b u v))
           (variable))))
P
>(<-(p (d-termgraph (vertex (s u) (a w) (t v))
           (edge)
           (variable (?x1 u w) (?x2 w v))))
    (p (d-termgraph (vertex (s u) (t v))
           (edge)
           (variable (?x1 u v))))
    (p (d-termgraph (vertex (s u) (t v))
           (edge)
           (variable (?x2 u v)))))
P
>(<-(p (d-termgraph (vertex (s u) (t v))
           (edge)
           (variable (?x1 u v) (?x2 u v))))
    (p (d-termgraph (vertex (s u) (t v))
           (edge)
           (variable (?x1 u v))))
    (p (d-termgraph (vertex (s u) (t v))
           (edge)
           (variable (?x2 u v)))))
P
>(?- (p (d-termgraph (vertex (s v1) (a v2)
                     (t v3) (a v4))
           (edge (b v1 v2) (b v1 v2)
                 (b v2 v3) (b v1 v4) (b v4 v3)))))
T
```

図 6: FGS インタプリタの実行例