EGG图文法语法分析算法的研究

朱 云 曾晓勤 朱 宁
(河海大学计算机与信息学院 南京 211100)

摘 要 EGG是一种基于边的上下文相关图文法形式化框架,其语法分析(归约操作)算法是该文法重要的组成部分。在简要介绍EGG的基础上,给出了EGG语法分析算法的设计,其中包括子图匹配算法、子图替换算法和算法计算复杂性的分析。为了展示如何用EGG来定义图语言,特别是如何用所设计的归约算法来分析图,文中以程序流程图为例,给出了相关的EGG形式定义以及对一个具体流程图的归约过程,并探讨了可能降低分析算法复杂性的一些途径。

关键词 图文法,语法分析,子图匹配,子图替换,推导,归约
中图法分类号 TP391 **文献标识码** A

Research on Parsing Algorithm of EGG Graph Grammar

ZHU Yun ZENG Xiao-qin ZHU Ning
(Computer and Information College, Hohai University, Nanjing 211100, China)

Abstract EGG is an edge-based context-sensitive graph grammar formalism, in which the parsing (reduction operation) algorithm is a very important part. On the basis of a brief introduction of EGG, this paper presented the design of a parsing algorithm, which includes subgraph matching algorithm, subgraph substitution algorithm, and their computational complexity analyses. In order to show how to apply EGG to define graph languages and especially parse graphs with the designed algorithms, the paper taking program flow diagram as an example, presented its formal definition of productions and its parsing procedures for a given flow diagram. As the future research directions, the possible ways to reduce the complexity of the algorithms were also discussed.

Keywords Graph grammar, Parsing, Subgraph matching, Subgraph substitution, Derivation, Reduction

1 引言

随着计算机科学和技术的迅速发展,特别是人机友好界面技术的发展,具有图操作功能的可视化图语言正在扮演越来越重要的角色,成为计算机科学中的一个非常重要的领域[1]。相对于传统的字符语言而言,可视化图语言具有表达直观、语义丰富、操作简便等诸多优点。正如字符语言的定义和实现需要形式化字符文法的支持一样,可视化图语言的定义和实现也需要相应的形式化文法的支持。毋容置疑,对形式化图文法的研究将为图语言的定义、图的构造及分析、图的转换和识别等奠定理论基础。

近些年来,图文法已经在许多领域有了不同程度的应用,主要是用于识别和分析各种图表。例如,Pfaltz[2]在所提出的NRNTST系统中用图文法分析了神经网络结构,Janssens等人[3]给出了用图文法对流程图和示意图进行分析的方法,王莉等人[4]使用了以有向边为属性的edNLC图文法及其产生式规则对网状结构目标进行了有效的识别。另外,图文法也被广泛应用到软件工程领域,例如对XML文档的分析和转换、并发系统的研究和软件体系结构描述等。不仅如此,有人还应用图文法来对音符[5]和数学公式进行识别[6],对带噪音的二维图像和手写体字符等进行识别[7]。

与字符文法类似,图文法也是由一组产生式规则组成,只是每条产生式的左部和右部都是一个图。基于产生式的推导和归约操作可以分别用来形式化定义图语言以及分析图的语法结构。然而,图文法中引出了字符文法中所没有的嵌入问题。嵌入问题是指,在利用产生式对图的一部分进行替换后,替换的部分如何嵌入到图中而不产生悬边。正是解决悬边的方法可有不同,才出现了多种不同的形式化图文法机制。

在图文法发展初期,研究较多的是相对简单的上下文无关图文法。但是,随着图文法的应用越来越广,在一些较为复杂的应用领域出现了一些难以用上下文无关图文法进行描述的问题,因此近些年来具有更强表达能力的上下文相关图文法有了迅速的发展[8]。自Rekers在1997年提出了一种上下文相关分层图文法LGG(Layered Graph Grammar)[9]之后,又出现了RGG[10]、SGG[11]、RGG+[12]、EGG[13]等上下文相关图文法。

到稿日期:2011-12-19 返修日期:2012-04-05 本文受国家自然基金(61170089)资助。
朱 云(1987—),女,硕士,主要研究方向为图文法,E-mail:zhuyunhhu@163.com;曾晓勤(1957—),男,博士,教授,博士生导师,CCF高级会员,主要研究方向为图文法、人工神经网络、机器学习;朱 宁(1986—),男,硕士,主要研究方向为图文法。

LGG 文法通过在规则中使用上下文并且引入通配符来解决嵌入问题。该语法形式较为直观、易于理解，但是这种将所有上下文结点和边都写进产生式中的做法，会增加产生式的长度和数量。虽然使用通配符可以减少产生式的数量，但在某些情况下通配符的数量会很多。RGG 继承了 LGG 的许多思想，但其将图中的结点定义成了双层结构，并引入了一种称为“标记”的机制来解决嵌入问题。但是 RGG 中图的双层结构和标记机制不够直观，不方便实际应用。EGG 形式化框架的出发点之一是让产生式能有效地表达抽象的上下文结构信息，而尽量少地涉及具体的上下文语义信息。EGG 承袭了 LGG 和 RGG 在产生式两端定义的一些图元素，并利用这些元素建立对应关系来解决嵌入问题的思想，但这种元素仅用边(结构信息)而略去了边的一端的上下文结点(语义信息)。

任何形式化文法都必须伴有相关的推导和归约操作。一般而言，通用的归约算法的计算复杂性是很高的，降低它的计算复杂性仍是个值得深入研究的问题。LGG 采用的一种分层机制和引入的通配符使其归约过程中图的匹配特别复杂，导致了 LGG 文法的归约算法复杂性非常之高，难以在实际问题中得到有效应用。在继承了 LGG 的分层方法的基础上，RGG 提出了 SFPA 算法[11]，较 LGG 的归约算法，该算法的复杂性大大降低，但是它要求文法产生式集合必须满足 selection-free 条件。这个条件不易判定，限制了 RGG 的应用范围。对 EGG 来说，除了根据自身的文法形式改写 selection-free 条件来采用 SFPA 算法或是采用复杂性很高的简单回溯的归约算法，目前还没有找到适合自身文法形式的高效的归约算法。

本文讨论了 EGG 归约算法的设计，主要贡献有：(1)给出了适合 EGG 文法形式的归约算法的设计，包括如何在主图中寻找图柄的子图匹配算法，如何在主图中删除图柄并且嵌入图柄所对应的产生式左图的子图替换算法，以及结合匹配算法和替换算法实现归约操作的归约算法，并给出了算法复杂性分析；(2)讨论了降低归约复杂性的可能途径，如对产生式排序以及判别其是否满足 select-free 条件等；(3)给出了一个应用 EGG 的归约算法来判定程序流程图语法正确性的具体实例，展示了如何应用 EGG 来定义图和分析识别图。

2 EGG 图文法形式化框架简介

EGG 在对图文法进行形式化定义时，利用集合来进行描述。EGG 使用边作为产生式两端的对应元素，指出上下文信息，在构建文法产生式时引入了悬边[12]的概念。下面简要地介绍 EGG 中的基本概念，并给出符号记号。

定义 1 $Q=(V,\hat{E},l,s,t,m)$称作标号集 L 和悬边标记集 M 上的一个悬边图，其中：$V=V_T\cup V_N$ 是由终结点集和非终结点集组成的结点集合；$\hat{E}=\bar{E}+E$ 是由悬边集 $\bar{E}$ 和非悬边集 E 组成的边集合；$\bar{E}=\vec{E}+\overleftarrow{E}$是由起点悬边集 $\vec{E}$ 和终点悬边集 $\overleftarrow{E}$ 组成的悬边集合，两者分别代表起点没有指定的边和终点没有指定的边；$l:(V\cup E)\to L$ 是结点和边的标号函数；$s:(E+\overleftarrow{E})\to V$ 和 $t:(E+\vec{E})\to V$ 是边的起点函数和终点函数；$m:\bar{E}\leftrightarrow M$ 是一个双射，称作悬边图的悬边标记函数。

定义 2 由两个悬边图 Q_L 和 Q_R 通过符号“:=”连接而得到形如 $Q_L:=Q_R$ 的结构是一个产生式，记为 $p=(Q_L,Q_R)$，其中 Q_L 和 Q_R 的悬边间必须存在一一对应关系并且对应悬边的标记也必须相同，特别地，当 Q_L 是文法的初始图时，Q_R 必须是不带悬边的图。

为了保证文法的归约的可判定性，EGG 对产生式两端图的结点数进行了限制，要求产生式右图的结点数不能小于左图的结点数，如果二者相等，则右图中的非终结符必须比左图的非终结符多。

为了方便接下来的定义和讨论，有必要先作以下几点约定：(1)$x.y$ 用来表示 y 隶属于 x，例如 $p.Q_L$ 和 $p.Q_R.v$ 分别表示产生式 p 的左图和 p 的右图的结点。(2)在悬边图 Q 中除掉悬边集之后得到的图称作它的核图，记为 $K(Q)$。(3)与悬边有关联的结点称作悬边图的关联结点，以关联结点 v 为起点的悬边的个数称作 v 的关联出度，记为 $d_{rout}(v)$，即 $d_{rout}(v)=|\{\bar{e}|\bar{e}\in\bar{E}\wedge s(\bar{e})=v\}|$；以 v 为终点的悬边的个数称作它的关联入度，记为 $d_{rin}(v)$，即 $d_{rin}(v)=|\{\bar{e}|\bar{e}\in\bar{E}\wedge t(\bar{e})=v\}|$。(4)两个图 G' 和 G'' 同构，是指在它们的结点和边之间存在一一对应关系，对应结点和边具有相同的标号，并且对应边的起点和终点恰好是对应的结点，记为 $G'\Leftrightarrow G''$。(5)对于图 G，假设 G_1 是 G 的子图，把在 G 图中删除 G_1 得到的图称为 G_1 的余图，记为 $G_1\backslash G$。(6)对于 G_1 中的结点 v，从 v 连接到 $G_1\backslash G$ 的结点的边数称为它的连通出度，记为 $d_{out}(v)$；从 $G_1\backslash G$ 的结点连接到 v 的边数称为它的连通入度，记为 $d_{in}(v)$。(7)连接子图 G_1 结点和余图 $G_1\backslash G$ 结点的边称为桥边，记为 $E(G_1\backslash G)$。(8)对于图中任意两个结点 $v1$ 和 $v2$，定义函数

$$\Phi(v1,v2)=\begin{cases}0, & v1\text{ 和 }v2\text{ 之间没有边}\\ \text{边的标号}, & v1\text{ 和 }v2\text{ 之间有边}\end{cases}$$，通常边的标号是字符串，当边没有标号时，字符串为空串 NULL。

定义 3 设 X 是图 G 的一个子图，Q 是一个悬边图，如果 X 与 $K(Q)$ 同构，并且对于同构函数，任意一组对应结点都满足 $d_{out}(v)=d_{rout}(v')$ 和 $d_{in}(v)=d_{rin}(v')$，其中 v 是 X 的结点，v' 是 Q 的结点，则把 X 称作悬边图 Q 在 G 中的一个图柄，记为 $redex(G,Q)$。

定义 4 一个 EGG 是一个四元组 (A,T,N,P)，A 为文法初始图，T 和 N 分别为终结标号和非终结标号集，即 $T+N=L$，P 是文法产生式集。

定义 5 假设有图 G 和产生式 p，在 G 中寻找 $p.Q_L$ 的图柄 $redex(G,p.Q_L)$ 并用 $p.Q_R$ 进行替换得到新图 G' 的过程称作图的推导，记为 $G\overset{p}{\vdash\!\!\to}G'$；反之，在 G 中寻找 $p.Q_R$ 的图柄 $redex(G,p.Q_R)$ 并用 $p.Q_L$ 进行替换得到新图 G'' 的过程称作图的归约，记为 $G\overset{p}{\to}G''$。若图 G 经过 $n(n\geqslant 0)$ 步推导得到 G'，则记为 $G\vdash\!\!\to^* G'$；类似地，若图 G 经过 $n(n\geqslant 0)$ 步归约得到 G''，则记为 $G\to^* G''$。

归约与推导是形式化文法的两个最基本的操作，图文法自然也不例外。在不严格区分推导与归约操作时，将其统称为图变换操作。

定义 6 若 (A,T,N,P) 是一个 EGG，则将 $\{G|A\vdash\!\!\to^* G\wedge$

$|G.V_N|=0\}$称为 EGG 定义的图语言。

图文法不仅可用来定义图的语言，还是对图进行语法分析的依据，可用来判定一个图是否属于给定图文法所定义的语言，也就是根据文法的产生式对所给图进行归约来判定。如果一个图可以归约到初始图，就说明这个图属于图文法定义的语言；反之则说明它不属于图文法定义的语言。EGG 语法分析是指归约过程，它是由归约算法指定的一个持续迭代的过程，其中每一步包括选择产生式、寻找图柄、选择图柄进行替换等等。归约结果不仅可指出被分析图的合法性，而且其路径可揭示出被分析图的语法结构。由于 EGG 语法分析过程中都是对二维图进行归约操作，因此操作一次需要的时间和空间分析复杂性都比一维字符文法大得多。为了提高 EGG 文法的分析效率，需要从多方面研究能降低 EGG 语法分析复杂性的算法，主要有子图的匹配算法、子图的替换算法和归约算法等等。

3 EGG 图文法分析算法的设计

图的分析过程是一个不断地寻找图柄并进行图替换的归约过程。寻找图柄是子图匹配过程，而图替换则是子图替换过程。每进行一步归约都要执行子图匹配和子图替换操作。设要被分析的主图为 H，其是一非悬边图且结点个数为 n；当前使用的产生式为 p，且 $p.Q_R$ 的结点个数为 m。

3.1 子图匹配算法

在图论中，子图匹配属于 NP 问题。由于图文法中图的特殊性，可以使用文法中一些额外附带的信息来降低子图匹配的复杂性，即充分利用图的结点和边都可能带有标号来缩小子图搜索时的搜索空间。EGG 归约算法中使用产生式右图的核图到主图中寻找匹配子图，本文将直接使用产生式的整个右图来寻找匹配子图，这样可充分利用 EGG 文法中指出的上下文结构信息。悬边的加入使得与其相连的结点的出入度信息会被利用，这样有利于缩小搜索空间。

在主图中寻找匹配子图，就是找主图中与子图对应的结点。首先，任何两个对应结点的标号必须相同，并且出入度相同(这里子图必须为产生式右端带悬边的图，否则与悬边相连的结点出入度会存在差异)。这样找出来的结点只能算是可能存在对应关系。然后在此基础上检查任意两组对应结点之间的边是否对应，只有其全部对应，才算是找到一个匹配子图，即图柄。

考虑到被分析图中的图柄可能有多个，以及尽可能地降低计算复杂性，下面给出一个通用的子图匹配算法执行步骤。

输入：主图 H 和产生式 p 的右部 $p.Q_R$。

1. 对图 H 中的所有结点按照标号的从小到大顺序进行排序；
2. 在排出的序列里，依次找出与 $p.Q_R$ 的某个结点的标号相同的所有结点，将其作为与该结点匹配的候选结点；
3. 比较 $p.Q_R$ 的每一结点与其匹配的候选结点的出入度是否相等，删除不相等的候选结点，设函数 Ψ 可指出图 H 结点是 $p.Q_R$ 哪个结点的候选结点，即 $\Psi(v_H)=v_{p.Q_R}$；
4. 给 $p.Q_R$ 的结点排序，对应候选结点个数少的结点排前面；
5. 按如下步骤建立一个 m 层的多叉树来存储已匹配结点：
   5.1. 根结点(0 层结点)为一虚设的结点；
   5.2. 树中除了根结点和叶结点外，其余每个结点的儿子结点的确定取决于图 H 和 $p.Q_R$ 中相应两结点间边的信息，具体地：
   5.2.1. 根结点的儿子结点是第 1 层上的所有结点；
   5.2.2. 循环，从树的第 1 层到第 m−1 层：循环，对层 l 上的每个结点 v^l(设其前辈结点依次为：$v^{l-1},\cdots,v^1$)：v^l 的儿子结点仅可能是 $p.Q_R$ 的结点序列中第 l+1 个结点的候选结点中满足如下条件的结点 v^{l+1}：

$$(v^{l+1}\neq v^l \wedge v^{l+1}\neq v^{l-1}\wedge\cdots\wedge v^{l+1}\neq v^1)\wedge$$
$$\Phi(\Psi(v^l),\Psi(v^{l+1}))=\Phi(v^l,v^{l+1})\wedge$$
$$\Phi(\Psi(v^{l-1}),\Psi(v^{l+1}))=\Phi(v^{l-1},v^{l+1})\wedge\cdots\wedge$$
$$\Phi(\Psi(v^1),\Psi(v^{l+1}))=\Phi(v^1,v^{l+1});$$

若层 1 上所有结点都没有儿子结点，则图 H 中没有要找的句柄，结束；

输出：一棵多叉树。

上述算法的前几步主要是为降低计算复杂性而做的考虑，例如，第 1 步的排序可减少第 2 步查找候选结点的时间，第 3 步的减少候选结点个数可避免后面不必要的操作，第 4 步的目的在于很快就可以确定不存在匹配子图的情况，因为如果排在最前面的结点一个对应结点都没有，就可以直接得出图 H 中没有匹配的子图。算法的第 5 步详细地给出了在前几步基础上构造一棵多叉树的方法，树的第 m 层上有多少个叶结点，则图 H 中就有几个匹配子图，从根结点到任一叶结点路径上的所有结点构成的子图是一个图柄。其中条件判断语句中的 $v^{l+1}\neq v^l \wedge v^{l+1}\neq v^{l-1}\wedge\cdots\wedge v^{l+1}\neq v^1$ 是为了避免 $p.Q_R$ 中两个结点与 H 中的同一个结点对应。

通过分析可知，子图匹配算法的最坏时间复杂性是多项式级的。具体分析如下：步骤 1 中排序的最坏时间复杂性为 $O(n^2)$；步骤 2 和 3 的时间复杂性均为 $O(n)$；步骤 4 的最坏时间复杂性为 $O(m^2)$；步骤 5 的时间开销为 m 次循环，每一次处理结点数不超过 n，即有：$i\times j\times k\times\cdots(i\leqslant j\leqslant\cdots\leqslant n)$，共 m 个数相乘，因此最坏时间复杂性为 $O(n^m)$。这样，算法总的最坏时间复杂性为 $O(n^2)+2*O(n)+O(m^2)+O(n^m)$。当文法确定后其产生式就已经确定，那么 m 是一定数，且通常都有 $n>m$ 和 $m>2$，所以子图匹配算法最坏时间复杂性可归为 $O(n^m)$，是多项式级的。当 $m>2$ 时，最坏时间复杂性由步骤 5 产生，但是由于每判断一次边是否匹配时，都会让一些候选结点在当前情况下不可用，从而不再判断其后的结点所对应的候选结点，所以，算法的时间复杂性应该远小于最坏情况。

3.2 子图替换算法

经过子图匹配过程，假设 $p.Q_R$ 与 H 的子图 G_H 匹配，$p.Q_R$ 的结点序列为 $(v_1,v_2,\cdots,v_q)$，G_H 的结点序列为 $(v_1',v_2',\cdots,v_q')$，结点对应关系为 $(v_1,v_1'),(v_2,v_2'),\cdots,(v_q,v_q')$，$p.Q_L$ 的结点个数为 s，产生式左右图的悬边的标记分别为 $1,2,3,\cdots,t$，子图替换算法可描述如下。

输入：主图 H、产生式 p 以及与 $p.Q_R$ 匹配的 H 的子图 G_H。

1. 在 $E(G_H\backslash H)$ 的 $p.Q_R.\bar{E}$ 之间建立边的映射关系；
2. 根据此映射关系以及 $p.Q_R$ 和 $p.Q_L$ 之间的悬边标记，建立 $E(G_H\backslash H)$ 和 $p.Q_L.\bar{E}$ 之间的映射关系 $g:e\leftrightarrow\bar{e}$，其中 $e\in E(G_H\backslash H)$，$\bar{e}\in p.Q_L.\bar{E}$；
3. 在图 H 中删除子图 G_H；
4. 把 $p.Q_L$ 嵌入 H 中，其每一条悬边 $\bar{e}$ 连到与其对应的边 e 在 $G_H\backslash H$

中的端点。

输出:替换后的图。

子图替换算法最坏时间复杂性为常数级,具体分析如下:步骤1的最坏时间开销为 t;步骤2的最坏时间开销也为 t;步骤3的最坏时间开销为 m;步骤4的最坏时间开销性为 s。所以,算法总的最坏时间复杂性为 $O(2t+m+s)$。当产生式确定后,t、m、s 3个值通常不会很大且均可视为定数,故替换算法的最坏时间复杂性为常数级。

3.3 归约算法

归约是要判断一个给定的图是否属于某一 EGG 文法定义的语言,即检查从给定图的当前状态是否能归约到文法的初始状态。一个图的当前状态对于每一个产生式可能会有多个图柄,这里先介绍一种最一般的算法,即每归约一步,只随机选择一个图柄来进行替换。

找出一个图的所有图柄的方法为:依次从第一个产生式开始,找出每个产生式右图的所有图柄,一直到最后一个产生式,所有产生式对应的图柄即为当前图的所有图柄。最一般的情况下,归约过程是一个不断进行图转换(用产生式左端替换主图中产生式右端图柄)的过程,成功则继续归约,失败则回溯,直至归约到文法的初始图,或所有可能的回溯尝试都失败了。下面给出了归约算法的具体步骤,其中用了二个堆栈来实现回溯,堆栈 hostStack 用来存储归约过程中生成的中间主图,堆栈 redexStack 用来存储归约过程找到的图柄,另外在堆栈 redexStack 中用一个分界符来间隔不同主图的图柄。

输入:主图 Graph,产生式集合 P;

循环:当 Graph 不是文法初始图 A,执行以下几步:

1. 向 redexStack 中压入分界符;
2. 对于 P 中每个产生式调用子图匹配算法,得到 Graph 的所有图柄;
3. 把所有图柄压入 redexStack 中;
4. 从 redexStack 栈顶弹出一个元素赋值给 redex;
5. 循环:当 redex 为分界符

   则从 hostStack 中弹出一个元素,并从 redexStack 栈顶弹出一个元素赋值给 redex;

   如果 redex 为空,则输出被分析的图不是该文法的语言,算法结束。
6. 把 Graph 压入 hostStack 中;
7. 调用子图替换算法,对于 Graph 使用 redex 对应的产生式转换后,得到的新图重新赋值给 Graph;

输出:被分析的图是该文法的语言。

该算法具有一般性,但时间复杂性很高。设产生式集合有 w 个产生式,产生式右图最大结点个数为 m,则有 n 个结点的图对于一个产生式的图柄最多只可能有 A_n^m 个。算法的复杂性由外层循环的次数(设为 N)以及该循环体的复杂性决定。循环体的复杂性可由7部分相加组成,第1步、第4步、第6步和第7步的复杂性都为常数级,第2步的复杂性为 $O(wm^n)$,第3步最大复杂性为 $O(wA_n^m)$,小于第2步,第5步的复杂性一定小于第2步,所以,整个算法的最大时间复杂性为 $O(Nwm^n)$。

最坏情况为判定出该图不是文法的语言,且归约过程中的所有图柄都压入栈中,并且每个图柄都被用来进行转换,此时 N 达到最大值,N 的值为找到的所有图柄的个数。根据对文法归约可判定性的规定,即对产生式两端图的结点数的限制,一个 n 个结点的图至多归约 n 步其长度不减,以后每一步都至少使其长度减1。所以找到的图柄个数最多为

$$\begin{aligned}N&=(wA_n^m)^{n+1}(wA_{n-1}^m)\cdots(wA_m^m)(wA_{m-1}^{m-1})\cdots(wA_2^2)(wA_1^1)\\&=w^{2n}\left(\frac{n!}{(n-m)!}\right)^{n+1}\frac{(n-1)!\,(n-2)!\cdots 2!\,1!}{(n-1-m)!\,(n-2-m)!\cdots 2!1!}\\&=w^{2n}\left(\frac{n!}{(n-m)!}\right)^{n+1}(n-1)!\,(n-2)!\cdots(n-m)!\end{aligned}$$

则

$$\begin{aligned}O(Nwm^m)&=O\left(w^{2n}\left(\frac{n!}{(n-m)!}\right)^{n+1}(n-1)!\,(n-2)!\cdots(n-m)!\,wm^m\right)\\&=O\left(w^{2n+1}n^{m^2+2m}\prod_{i=1}^{m}(n-i)!\right)\end{aligned}$$

归约算法的回溯是导致算法的时间复杂性很大的原因。如果文法满足 seclect-free 条件[9],则每次只需要找一个图柄,不需要回溯,只要不断地找出一个图柄并替换,如果不能持续归约到文法初始图,则可以立刻判断该图不是该文法的语言。但是文献[9]中提到的 select_free 条件限制太强,并且对具体文法判定其是否满足 select_free 条件也是一个相当棘手的问题;如果判定后不满足,则仍须使用回溯的办法。为了提高 EGG 文法的分析效率,可以从多方面来考虑降低 EGG 语法分析的复杂性,包括产生式的选择、子图的匹配、图柄的确定、回溯的减少、子图的替换等等。

3.4 降低分析算法复杂性的考虑

这里提出一种可能降低回溯次数的办法,即对产生式规范化并赋优先级进行排序。产生式的优先级排序的最大好处就是使得寻找图柄时不再盲目地从所有产生式中寻找,而是有优先次序的。具体操作分为以下3个步骤:

1. 产生式规范化预处理

1)消除包含关系,即任意两个产生式的右图不允许存在任何包含关系。若产生式 pi 的右图是产生式 pj 的右图的子图,则对产生式 pj 的右图使用产生式 pi 进行归约。产生式 pj 的右图修改为归约后得到的图。

2)消除冗余性。不允许存在由其他产生式推导得到的产生式。若某产生式 pi 经过消除包含关系处理后得到的产生式与原来存在的产生式 pj 相同,则可以删除产生式 pi。

2. 产生式优先级排序

1)从左图为起始状态的产生式开始,把其优先级定为0。

2)假设其优先级为 k 的产生式 pi 右图中有非终结结点 vN,且初次出现,则左图中有非终结结点 vN 的产生式 pj 的优先级定为 $k+1$,并将 vN 结点标记为已出现。

3)经过第2)步的排序,在优先级同为 k 的产生式中,右图全为终结符的产生式的优先级略高,改为 $k+0.5$。

3. 产生式使用顺序

1)归约的第一步选用右图全为终结符的产生式。

2)其后每步归约选用产生式时都按优先级从高到低的顺序。

步骤1主要是为了减少图柄的产生,如果产生式 pi 的右图是产生式 pj 的右图的子图且 pj 在主图中有对应的图柄,那么 pi 在主图中必然也有对应的图柄,该图柄为 pj 在主图

中有对应的图柄的子图，这样导致找到多个图柄。步骤 2 和步骤 3 是给产生式排序，并按优先级顺序使用产生式。首先，主图的结点肯定全是终结符。其次，从产生式的结构可以看出，某些非终结符是由含有其他非终结符的图推导得来。若产生式 $p1$ 的右图非终结符 $vN1$，且产生式 $p2$ 的左图含有非终结符 $vN1$，右图含有非终结符 $vN2$，那么在归约时产生式 $p2$ 必然比产生式 $p1$ 优先使用。因为假使主图是指定文法的语言，那么归约过程中图的当前状态含有非终结符 $vN2$，那么必然是由产生式 $p2$ 产生的，所以可以优先使用产生式 $p2$ 进行归约。

4 EGG 图文法在程序流程图中的应用

4.1 程序流程图的 EGG 文法

1966 年，Bohra 的 Jacopini 提出 3 种基本程序流程结构，可用这 3 种结构作为表示一个良好算法的基本单元。它们分别为顺序结构、选择结构、循环结构[14]。本节将 EGG 文法应用到对程序流程图的语法分析中。对程序流程图进行语法分析可以判断此流程图是否合法，如果不合法则可以尽早发现并加以修改，避免了到程序编码阶段才发现问题，从而节约了程序设计时间。

4.1.1 EGG 产生式的设计

根据程序流程图的结构特点，EGG 产生式的设计如图 1 所示。

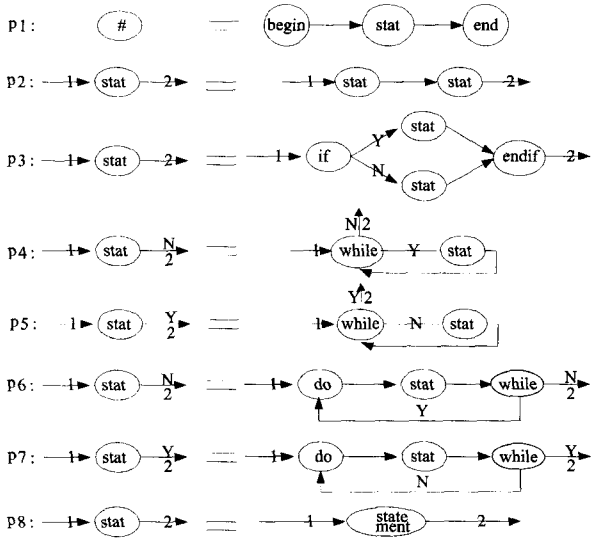

图 1 程序流程图的产生式组

产生式组中含有两个非终结点，即标号为 # 和 stat 的结点。# 代表文法的起始状态，stat 结点代表一个语句块。产生式 $p1$ 是从起始状态推出的第一个状态；产生式 $p2$ 代表顺序结构；$p3$ 代表选择结构；$p4$、$p5$ 代表 while 循环；$p6$、$p7$ 代表 do_while 循环；$p8$ 代表任意一个语句都是一个语句块。

对于程序流程图，根据 3.4 节的方法给该文法的产生式组做一个优先级排序。产生式 $p1$ 的优先级为 0，产生式 $p2-p7$ 优先级为 1，产生式 $p8$ 优先级为 2。产生式 $p8$ 的优先级最高，其次是产生式 $p2-p7$，最后是产生式 $p1$。$p2-p7$ 优先级相同，归约时，先使用产生式 $p8$ 去找图柄，找到立刻去替换，直至没有产生式 $p8$ 对应的图柄，再使用产生式 $p2-p7$，在产生式 $p2-p7$ 均没有对应图柄时，使用产生式 $p1$ 去匹配。

4.1.2 程序流程图转化为 EGG 形式的图

在应用 EGG 来分析一个给定的流程图时，首先需要把该流程图转换成为 EGG 形式的主图，主要有以下几个步骤：

1. 流程图的开始和结束分别抽象为标号为 begin 和 end 的结点。

2. 流程图中的 If 判断语句抽象为标号为 if 的结点；在 if 循环的两个分支的汇合处添加一个结点，标号为 endif。

3. While 循环中的 while 判断语句抽象为标号为 while 的结点。

4. Do_while 循环中的进入循环处添加标号为 do 的结点，while 判断语句抽象为标号为 while 的结点。

5. 除上述类型的语句外，其他语句全部抽象为标号为 statement 的结点。

4.2 用 EGG 归约程序流程图的一个实例

这里举一个大家熟识的流程图实例，即判断一个数是否为素数[14]。判断一个数是否为素数的程序流程图如图 2 的左边的图，右边的图为左边图转换后的 EGG 主图。

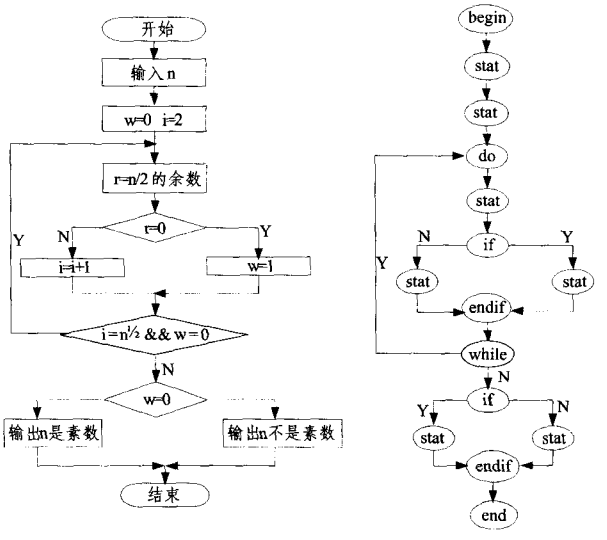

图 2 程序流程图原图及其 EGG 形式化后的图

在转换后的主图上展示一下模拟 EGG 归约算法的执行过程。由于图中含有产生式 $p8$ 对应的 7 个图柄，$p8$ 的优先级最高，可以全部都替换，这 7 步过程就不具体演示，图 3 演示了这 7 步以后的全部过程，表 1 列出了归约过程的每步状态下的所有图柄以及选择进行替换的图柄。图 3 中用红色框标出了每个状态的所有图柄。经过若干步的转换，最终得到状态 9，状态 9 为起始状态 #。最后可以判定，该图是该文法的语言，此流程图的结构是正确的。

表 1 归约过程中的状态表

状态	(图柄，对应产生式)	选择使用的图柄
状态 1	(图柄 1，p2)，(图柄 2，p3)，(图柄 3，p3)	图柄 1
状态 2	(图柄 1，p3)，(图柄 2，p3)	图柄 1
状态 3	(图柄 1，p2)，(图柄 2，p3)	图柄 1
状态 4	(图柄 1，p6)，(图柄 2，p3)	图柄 1
状态 5	(图柄 1，p2)，(图柄 2，P3)	图柄 1
状态 6	(图柄 1，p3)	图柄 1
状态 7	(图柄 1，p2)	图柄 1
状态 8	(图柄 1，p1)	图柄 1
状态 9		

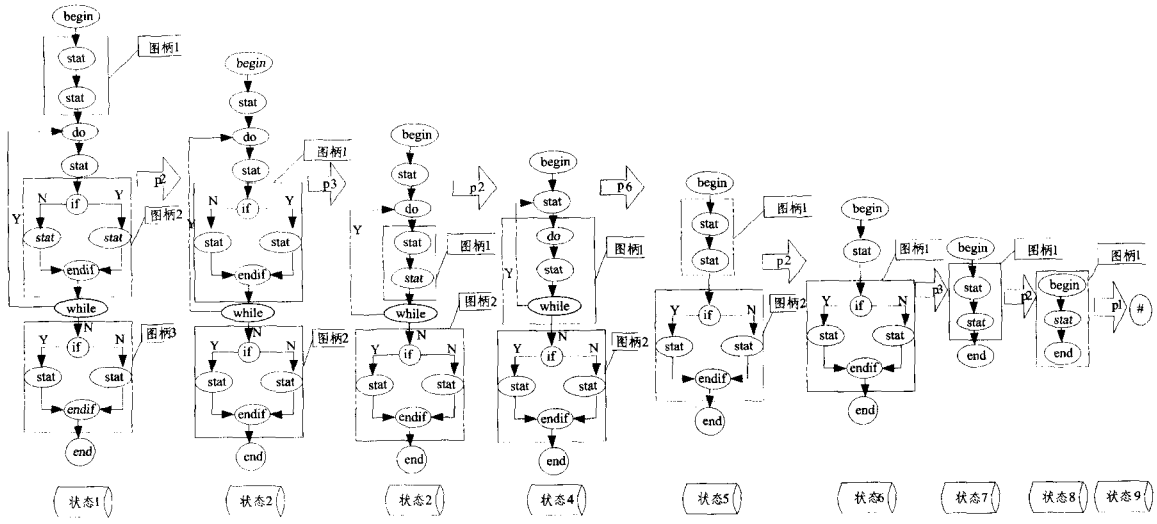

图 3　一个程序流程图的归约过程

子图匹配算法是本文的一个重点。故以状态 2 为例演示子图匹配算法如何找产生式 $p3$ 所对应的图柄。输入为状态 2 时的图以及产生式 $p3$ 的右图。给每个图的图结点一个唯一的序号，见图 4。根据 3.1 节的算法，5 个步骤依次如下。

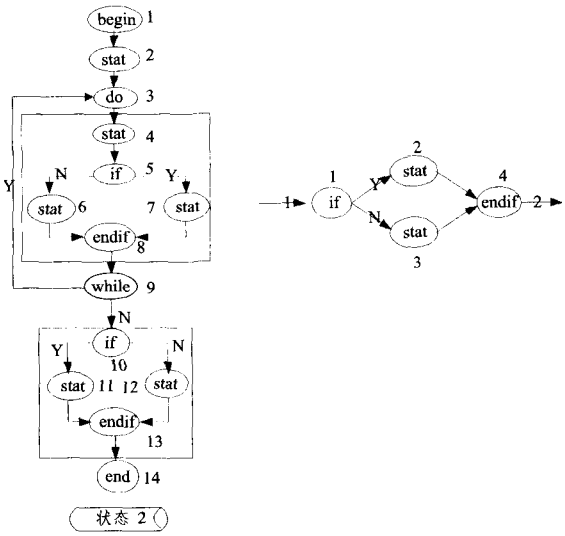

图 4　状态 2 时主图和 $p3$ 的右图序号

1. 主图 14 个结点按标号排序，见表 2。

表 2　主图结点标号排序

begin	do	end	endif			if
1	3	14	8	13	5	10
			stat			while
2	4	6	7	11	12	9

2. 依次找出候选结点，见表 3。

表 3　对应候选结点表

p3 右图的结点	1	2	3	4
p3 右图的结点的标号	if	stat	stat	endif
主图的结点	5,10	2,4,6,7,11,12	2,4,6,7,11,12	8,13

3. 删除出入度不等的结点：本例中无需删除结点。

4. 对应候选结点少的结点排在前面，见表 4。

表 4　重新排序后结点表

p3 右图的结点	1	4	2	3
p3 右图的结点的标号	if	endif	stat	stat
主图的结点	5,10	8,13	2,4,6,7,11,12	2,4,6,7,11,12

5. 建立多叉树，见图 5。

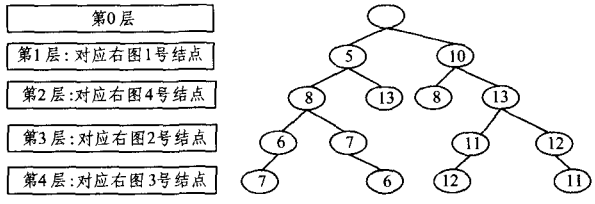

图 5　多叉树

从多叉树中可得到 4 个图柄，主图中结点序号为 5-8-6-7、5-8-7-6、10-13-11-12、10-13-12-11 分别与产生式右图序号为 1-4-2-3 的结点对应。由于 5-8-6-7 和 5-8-7-6、10-13-11-12 和 10-13-12-11 构成的子图完全相同，将前两个子图和后两个子图分别看成一个图柄，故状态 2 共有两个图柄。

结束语　本文给出了 EGG 图文法的归约算法，分析了算法的时间复杂性；然后探讨了降低归约算法复杂性的可行性方法；最后，给出了一个 EGG 实际应用的具体例子，即判定程序流程图的合法性。

在算法的设计上，子图匹配算法属于子图同构的问题。本文利用 EGG 图的特点，使得复杂性有所降低；子图替换是根据 EGG 的形式化结构，用其悬边标记来实现替换；整体的归约过程给出了最一般的算法，每一步归约都随机任选一个图柄进行，不能继续归约则回溯；并讨论了可能降低归约算法复杂性的可行性方法。

为了用 EGG 分析程序流程图，首先设计了合适的 EGG 产生式规则，接着给出了如何把一般图转换为 EGG 主图的步骤，最后展示了归约算法的执行过程。

接下来，将研究进一步降低子图匹配算法、子图替换算法和归约算法本身的复杂性，尽量减少归约过程中回溯的次数，从而更好地提升 EGG 分析算法的效率；我们还将深入研究文

（下转第 293 页）

结束语 针对视频处理中数据量大的特点，本文结合DWT变换和背景重构提出一种有效的运动目标分割方法。该算法无需对场景中的背景和目标建立模型，在像素级上结合改进的聚类算法并利用双阈值与类合并，使得类间相似度较小且类内相似度较大，从而实现背景重构。我们通过引用图像匹配的参数评价重构的背景，验证了本文方法的有效性，该方法能完成对运动目标的完整分割。但是，当视频序列不满足本文算法的假设时，会出现混合现象，如何消除该现象是我们下一步的工作。

参 考 文 献

[1] Silva L S, Scharcanski J. Video segmentation based on motion coherence of particles in a video sequence[J]. Image Processing, IEEE Transactions, 2010, 19(4): 1036-1049

[2] Chien S-Y, Ma S-Y, Chen L-G. Fast video segmentation algorithm with shadow cancellation, global motion compensation, and adaptive threshold techniques[J]. Multimedia, IEEE Transactions, 2004, 6(5): 732-748

[3] Venkateswaran S, Desai U B. DWT based hierarchical video segmentation[C]// Circuits and Systems, 2002, IEEE International Symposium. 2002: 815-818

[4] Cevher V, Sankaranarayanan A, Duarte M, et al. Compressive sensing for background subtraction[C]// Computer Vision-ECCV 2008. 2008: 155-168

[5] Mandellos N A, Keramitsoglou I, Kiranoudis C T. A background subtraction algorithm for detecting and tracking vehicles[J]. Expert Systems with Applications, 2011, 38(3): 1619-1631

[6] 郑锦，李波. 面向室外视频监控的背景重构算法[J]. 电子学报，2009，37(8)：1854-1859

[7] Lu Hong, Li Hong-sheng, Liu Lan-ying, et al. Background Reconstruction Using DWT and Grayscale Classification[C]// Artificial Intelligence and Computational Intelligence (AICI), 2010 International Conference. 2010: 57-61

[8] Elgammal A, Harwood D, Davis L. Non-parametric model for background subtraction [J]. Computer Vision—ECCV 2000, 2000: 751-767

[9] Stauffer C, Grimson W E L. Adaptive background mixture models for real-time tracking[C]// Computer Vision and Pattern Recognition, 1999, IEEE Computer Society Conference. 1999, 2: 246-252

[10] 林庆，徐柱，王士同，等. HSV自适应混合高斯模型的运动目标检测[J]. 计算机科学，2010，37(10)：254-256

[11] 侯志强，韩崇昭. 基于像素灰度归类的背景重构算法[J]. 软件学报，2005，16(9)：1568-1576

[12] 肖梅，韩崇昭. 基于在线聚类的背景减法[J]. 模式识别与人工智能，2007，20(1)：35-41

[13] 包红强，张兆扬. 一种基于区域Gibbs势能函数的视频运动对象分割算法[J]. 通信学报，2005，6(6)：57-61

(上接第277页)

法的自动生成，即产生式的自动生成，以及图文法在模式识别领域的更多应用。

参 考 文 献

[1] 许红霞，张莉. 可视化语言文法形式化描述综述[J]. 计算机科学，2005，32(4)：201-204

[2] Pfaltz J. Web Grammars and Picture Description[J]. Computer Graphics and Image Processing, 1972, 1(2): 193-220

[3] Bunke H. Attributed Programmed Graph Grammars and Their Application to Schematic Diagram Interpretation[J]. IEEE Pattern Analysis and Machine Intelligence, 1982, 4(6): 574-582

[4] 王莉，钟春香. 基于规则的网状结构目标识别研究[J]. 华中理工大学报，1996，24(2)：15-18

[5] Fahmy H, Blostein D. A Graph-Grammar Programming Style for Recognition of Music Notation[J]. Machine Vision and Applications, 1993, 6(2/3): 83-88

[6] Grbavec A, Blostein D. Mathematics Recognition Using Graph Rewriting[C]// Third Int. Conf. on Document Analysis and Recognition. Montreal, Canada, Aug 1995: 417-421

[7] SchiirrA B D. Visual Modeling and Programming with Graph Transformations[C]// Tutorial at 14th IEEE Symp on Visual Languages. Halifax, NS, Canada, 1998

[8] 冉平，石兵，马晓星，等. 上下文相关图文法分析及其应用初探[J]. 计算机科学，2006，33(3)：255-260

[9] Rekers J, Schürr A. Defining and Parsing Visual Languages with Layered Graph Grammars[J]. Journal of Visual Languages and Computing, 1997, 8(1): 27-55

[10] Zhang Da-qian, Zhang Kang, Cao Jian-nong, et al. A Context_sensitive Graph Grammar Formalism for the Specification of Visual Languages[J]. The Computer Journal, 2001, 44(3): 187-200

[11] Kong Jun, Zhang Kang. On a Spatial Graph Grammar Formalism [C]// Proc of the 2004 IEEE Symposium on Visual Languages and Human Centric Computing. IEEE, 2004: 102-104

[12] Zeng Xiao-qin, Zhang Kang, Kong Jun, et al. RGG+: An Enhancement to the Reserved Graph Grammar Formalism[C]// Proc of the 2005 IEEE Symposium on Visual Languages and Human-Centric Computing. IEEE, 2005: 272-274

[13] 曾晓勤，韩秀清，邹阳. 一种基于边的上下文相关图文法形式化框架[J]. 软件学报，2008，19(8)：1892-1901

[14] 谭浩强. C语言程序设计(第二版)[M]. 北京：清华大学出版社，1999：21-28