

文章编号: 2096-1472(2016)-11-04-06

基于图元装接模式由程序流程图自动生成源代码

许秀林¹, 李蕴华²

(1.南通职业大学电子信息工程学院, 江苏 南通 226007;
2.南通大学电子信息学院, 江苏 南通 226007)

摘 要: 针对程序流程图到代码自动翻译过程中, 通常需要经过代码优化及二次编程, 翻译效率不高, 提出基于图元装接模式的流程图到C语言源程序转换的解决方案, 实现流程图到源程序的双向转换。由图元同步产生代码元, 通过装接自动生成程序代码。所生成的程序代码无需修改, 可直接编译运行。通过词法分析, 识别并产生代码元和对应图元, 可逆向生成流程图。其有效性在原型系统中得到了验证。

关键词: C语言; 流程图; 程序代码; 自动翻译

中图分类号: TP311.5 **文献标识码:** A

Automatic Translation from Flowchart to Program Code Base on Graphics Primitive Assembling

XU Xiulin¹, LI Yunhua²

(1. Department of Electronic Information Engineering, NanTong Vocation University, Nantong 226007, China;
2. Department of Electronic Information, NanTong University, Nantong 226007, China)

Abstract: The translation from flowchart into program code automatically usually requires code optimization and quadratic programming, and works inefficiency. The solution is proposed for translation from flow chart to source program of C language base on graphics primitive assembling. According the solution, the conversion is in both directions of flowchart and the source program. The code primitive is created with graphics primitive synchronously. Then, the program code is automatically generated while the code primitives are assembled together. It can be compiled and run directly. Instead, with the identification and generate of code primitive through lexical analysis, a flowchart can be produced reverse from the source program. Its effectiveness has been verified in the prototype system.

Keywords: C language; flow charts; program code; automatic translation

1 引言(Introduction)

对于程序设计语言初学者来说, 通过画流程图来学习程序编码, 将程序代码与直观的图形相结合, 可以起到事半功倍的学习效果。若用草图来绘制程序流程图, 并且实现流程图的自动识别和代码生成, 不仅符合人的思维习惯, 也将大大方便程序设计课程的教学, 同时也可应用于程序员的快速程序设计和编码。

目前, 国内外很多学者对手绘草图识别的研究已有较大进展, 可较好地识别程序流程图中的简单图元符号。但是对程序流程图控制结构的判别, 以及代码的自动翻译的研究较少。程序流程图到代码自动生成的流程, 通常分为草图识别和代码优化两个阶段, 当前的研究热点或技术难点是图元的识别和翻译, 文献[1,2]设计了图元节点存储结构, 在图元识别的基础上, 提出基于上下文的程序流程草图自动翻译算法, 实现了在线手绘程序流程草图到C语言代码的自动翻译。

但该系统所能判别的基本图元比较单一, 而且在判断程序流程图控制结构时, 采用栈技术, 若用户采用非习惯性思维画图, 可能会导致栈顶元素无法正确判别, 在一定程度上影响识别结果。文献[3-5]则是对程序流程图到代码的自动生成技术的研究, 然而所研究的程序流程图到代码的自动生成工具对流程图有诸多限制, 并不是基于标准的程序流程图。文献[6]针对标准的程序流程图, 实现程序流程图到代码的自动生成, 并且对于含有continue/break/return语义的程序流程图也能够很好地识别并生成相应的代码。文献[7]融合图形化和文本语言两种编程方式的优点, 提出流程图与类C语言双向实时转换方法: 一方面通过遍历模块树并参照功能模块封装的库函数和语句完成流程图到类C语言的转换; 另一方面采用XML作为中间代码完成类C语言到流程图的转换。文献[8]实现了程序流程图的自动生成和实时修改, 提高了程序流程图的设计效率, 但没有实现程序代码的同步生成。

本文结合上述文献的研究成果，提出手工草图→机绘流程图→C语言源程序转换的解决方案，目前已实现了机绘流程图到源程序的双向转换，即按照“图元+装接”模式绘制流程图，同步生成程序代码；或通过本系统生成的源代码，逆向生成流程图。

2 图元与代码元(The graphics primitive and the code primitive)

2.1 图元

C语言流程图有传统流程图(FC)、盒图(N-S)、PAD等。其中传统流程图比较直观、形式化，易于理解。但由于对流程连接线的使用没有严格限制，使流程随意转来转去，缺少规律。特别是传统流程图绘制后，修改流程图需要大量移动流程连接线，非常烦琐。为此，我们根据C语言程序语句基本结构，将基本图形和连接线组合，定义了六种图元(图1)：if图元、for图元、while图元、do-while图元、switch图元、基本语句图元。图元中菱形为决策图标，矩形为基本语句图标，实线为普通连接线，虚线为图元装接线。

根据程序语句的基本功能，基本语句图元又分为文件包含、常量定义、变量定义、键盘输入、屏幕输出、文件打开、文件关闭、文件输入、文件输出、赋值语句，以及其他基本语句(通用语句)等十类。

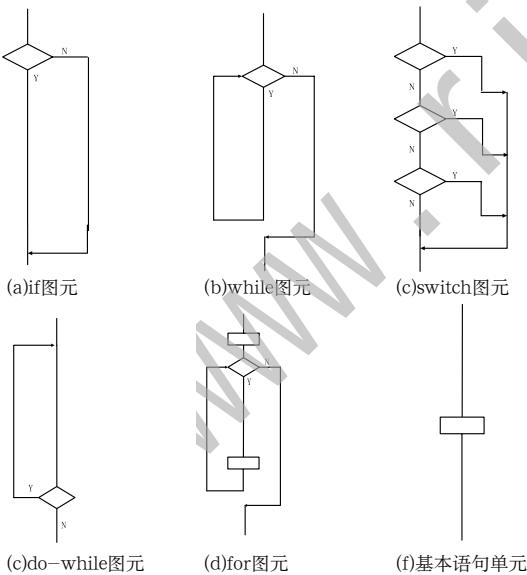


图1 流程图基本图元

Fig.1 The basic graphics primitive of flowcharts

2.2 代码元

代码元是与图元相对应的程序代码，分别有if代码元、for代码元、while代码元、do-while代码元、switch代码元、基本语句代码元(表1)。表中的代码元对应的程序代码只是其初始状态，呈现为程序语句的骨架，只有通过进一步处

理之后，才能成为一条完整的程序语句。

表1 基本代码元

Tab.1 The basic code primitive

名称	代码	基本语句代码元	
if代码元	if ()	文件包含代码元	#include
	{		
for代码元	for (=, //初值	常量定义代码元	#define
	=, //终值比较	变量定义代码元	int
	= //循环变量值	赋值语句代码元	=
	递增或递减	键盘输入代码元	scanf()
	{		
while代码元	while ()	屏幕输出代码元	printf()
	{	文件打开代码元	fopen();
do-while代码元	do	文件关闭代码元	fclose()
	{	文件输入代码元	fscanf()
switch代码元	while ()	文件输出代码元	fprintf()
	switch()	通用语句代码元	*
	case :		
	break;		
	case :		
	break;		
	default:		
	}		

3 图元处理与流程图绘制(Primitive processing and flowchart drawing)

3.1 图元生成

(1)定义图元基类

图元基类包含图元必备元素和基本操作。必备元素主要有图元的坐标位置、高度、宽度、图元文本(含量代码与注释)、友元信息、图元装接信息、图元编辑信息等。基本操作有位置移动、大小改变、选择、取消选择、插入图元、删除图元、绘制图元。

public class FC_Control //图元控件基类

{

private int x,y; //图元坐标

private int width,height; //图元宽度与高度

private int wLeft,wRight; //图元左宽与右宽

private bool selected; //图元选中标识

private int child_brother; //图元装接信息

//1 为左孩子结点，2为右孩子结点，0为兄弟结点，-1为初始值(无效)

```

private CODE_Control friendObj; //图元友元信息
public string[] txtCode=new string[11]; //图元文本
信息
public string[] txtComment=new string[11]; //图
元文本信息

virtual public void moveTo(int x,int y) //移动到
新坐标
{
    X=x;
    Y=y;
}

virtual public void reSize() { } //大小改变
操作

virtual public FC_Control select(int x,int y)
{ return null; }
//选择操作(已知坐标位置选择对象)
virtual public bool select(FC_Control obj) {
return false; }
//选择操作(已知对象信息, 选择对象位置)
virtual public void escape() {
bSelected=false; } //取消操作
virtual public void draw() { } //绘图操作
virtual public void delete(FC_Control obj) {
} //删除操作

virtual public FC_Control insert(int x,int
y,FC_Control obj) { return null; }
//插入操作(已知坐标位置, 插入特定对象)
virtual public void insert(FC_Control
insertObj,FC_Control indexObj,int index) { }
//插入操作(已知对象信息, 插入特定对象)
}

```

(2)定义图元类

图元类是图元基类的子类, 它继承了基类的所有属性和方法, 同时还具有以下两个属性: 图元的孩子对象集和对象数。图元的孩子对象包括判断图标、判断分支、图元连接线和图元装接线。以while图元为例, 它由1个判断图标、2个判断分支(Y/N)、2条(5段)图元连接线和1条图元装接线, 共计9个对象。下面while图元类属性定义及其构造方法:

```

public class FcWhile : FC_Control //while图元类
{

```

```

public int numOfObject=9; //组成图元的对象数
public FC_Control[] N=new FC_Control[9]; //组
成图元的对象集

public image; //画布
public FcWhile(int x,int y,FC_image )
{
    X=x;
    Y=y;
    image=FC_image;
    bool arrow=true;
    int defaultWidth=50;
    int defaultHeight=50;
    N[0]=new FcJudge(X,Y,FC_image); //判断图标
    N[1]=new FcJudgeBranch(N[0].X,N[0].Y+N[0].
H,FC_image,true,LINEWAY.DOWN); //判断Y分支
    N[2]=new FcCodingLine(N[1].X,N[1].Y+N[1].
H,LINE_TYPE.CODINGLINE_IN_FUNCTION,FC_
image); //图元装接线
    N[3]=new FcLine(N[2].X,N[2].Y+N[2].
H,LINEWAY.LEFT,!arrow,LINE_TYPE.
NOTCODINGLINE,FC_image); // (左下端)向左连接线
    N[3].W=defaultWidth;
    N[4]=new FcLine(N[3].X-N[3].W,N[3].
Y,LINEWAY.UP,!arrow,LINE_TYPE.
NOTCODINGLINE,FC_image); // (左边)向上连接线
    N[4].H=N[1].H+N[2].H+N[0].H/2;
    N[5]=new FcLine(N[4].X,N[4].Y-N[4].
H,LINEWAY.RIGHT,arrow,LINE_TYPE.
NOTCODINGLINE,FC_image); // (左上端)向右带箭头连接线
    N[5].W=N[3].W-N[0].WL;
    N[6]=new FcJudgeBranch(N[0].X+N[0].WR,N[0].
Y+N[0].H/2,FC_image,false,LINEWAY.RIGHT); //判断
N分支
    N[7]=new FcLine(N[6].X+N[6].W,N[6].
Y,LINEWAY.DOWN,!arrow,LINE_TYPE.
NOTCODINGLINE,FC_image); // (右边)向下连接线
    N[7].H=N[4].H+defaultHeight;
    N[8]=new FcLine(N[7].X,N[7].Y+N[7].
H,LINEWAY.LEFT,arrow,LINE_TYPE.
NOTCODINGLINE,FC_image); // (右下端)带箭头连接线
    N[8].W=N[6].W+N[0].WR;

```

```

WR=N[8].W;
WL=N[3].W;
W=WL+WR;//计算图元宽度
H=N[0].H/2+N[7].H;//计算图元高度
txtCode[0]="";
txtComment[0]="";
}
.....
}

```

(3)生成图元对象

当用户通过菜单项或图标工具插入某个图元时，生成该图元对象。同样以while图元为例，在插入while图元方法中生成图元对象。

```

public void insert_While(int x,int y)
{
    int X,Y;
    X=x;
    Y=y;
    FcWhile w=new FcWhile(X,Y,FC_image);
    //生成while图元对象
    ... ..
}

```

3.2 图元装接

图元装接是将图元对象按指定的坐标位置插入到图元装接线上。图2是流程图的初始画面，从开始到结束之间，是由两条装接线组成。上端黄色装接线用于插入带#的图元对象，下端蓝色装接线用于插入其他图元对象。图元装接根据装接对象的位置关系分为顺序装接和嵌入装接两类：对于某个已装接的图元对象A，当待装接的图元对象B与图元对象处于同一条装接线，则为顺序装接如图3(a)所示；当图元对象B处于图元对象A内部的装接线时，则属于嵌入装接如图3(b)所示。

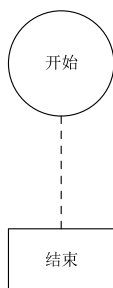
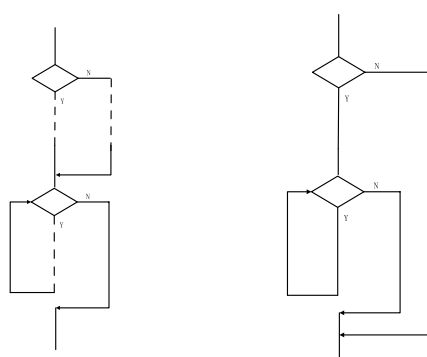


图2 流程图初始状态

Fig.2 The initial state of flowcharts



(a)if图元和while图元顺序装接 (b)if图元和while图元嵌套装接

图3 图元两类装接方式

Fig.3 Two ways of graphics primitive assembling

3.3 图元文本编辑

图元装接只是决定了流程图的流程走向，程序功能则需要通过图元文本编辑，确定每个图元的具体功能。现以while图元为例，说明图元文本编辑的过程。while图元的文本编辑，主要是while循环条件表达式的编辑。先选中while图元，然后打开该图元的属性页进行相关的文字编辑，如图4所示。编辑后的while图元在左上方添加了条件表达式，如图5所示。



图4 while图元属性页

Fig.4 The property page of while primitive

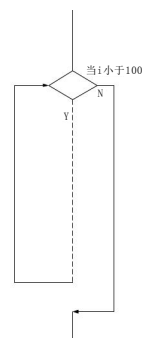


图5 while图元(编辑后)

Fig.5 The while primitive(edited)

3.4 示例

任务：输入两个正整数a、b，求两个数的最大公约数。

算法:

- ① 定义变量a、b、c为正整数
- ② 键盘输入a和b
- ③ if (a<b) c←a, a←b, b←c,
- ④ if b≠0 goto ④, 否则goto ⑧
- ⑤ c←a除b的余数;
- ⑥ a←b
- ⑦ b←c
- ⑧ goto ④
- ⑨ 输出公约数a
- ⑩ 结束

本任务主要使用一个if图元和while图元和若干基本语句图元。通过图元装接和图元编辑后, 所绘制的流程图, 如图6所示。

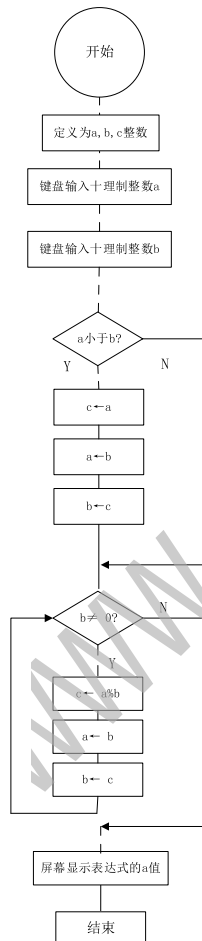


图6 求两个数最大公约数流程图

Fig.6 The flowchart of greatest common divisor

4 代码程序自动生成(The program code being automatically generated)

与图元处理相同, 代码元处理同样有生成、装接和编辑等操作。由于代码元与图元一一对应, 图元处理和代码元处

理同步完成。当流程图绘制完毕时, 对应的源程序也生成完毕, 处理过程如图7所示。

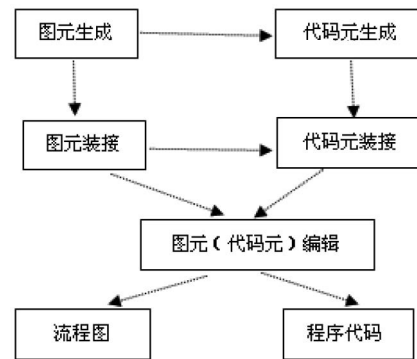


图7 图元和代码元同步处理过程

Fig.7 The synchronous processing of graphics primitive and code primitive

为了确保图元对象与代码元对象一一对应, 在图元基类和代码元基类中均设置了一个友元属性变量friendObj, 分别存储对方信息。在图元基类和代码元基类定义的相关代码为:

```
public class FC_Control //图元控件基类
{
    ... ..
    private CODE_Control friendObj; //图元友元信息
    ... ..
}

public class CODE_Control //图元控件基类
{
    ... ..
    private FC_Control friendObj; //代码元友元信息
    ... ..
}
```

在图元和代码元生成时, 相互记录对方的信息, 仍以图元while为例:

```
FcWhile w1=new FcWhile(X,Y,FC_image);
CodeWhile w2=new CodeWhile(0,0,CODE_image);

w1.friend=w2; //friend 是设置或获取属性
//变量friendObj值的方法
w2.friend=w1;
```

当在流程图中插入图元对象w1时, 也在程序代码中插入代码元对象w2, 同步完成装接。由流程图(图6)同步生成的程序代码如图8所示。

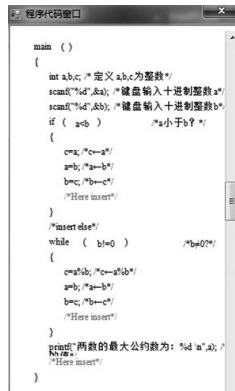


图8 求两个数最大公约数的程序代码

Fig.8 The code of greatest common divisor

5 代码元识别与流程图逆生成(The identification of code primitive and the flowchart being produced reverse from the source program)

对于生成的C语言源程序文件,通过词法分析器对关键词的识别,生成相应的代码元对象,并同步生成图元对象。再经过图元装接,逆向生成流程图。由于流程图文件是图片格式,一旦形成图片文件,再转换成图元对象,比文本识别要困难得多。通过程序代码文本文件阅读和识别,生成代码元对象,再通过代码元生成图元对象,然后接装为流程图,方便流程图的保存和修改。流程图逆向生成过程如图9所示。

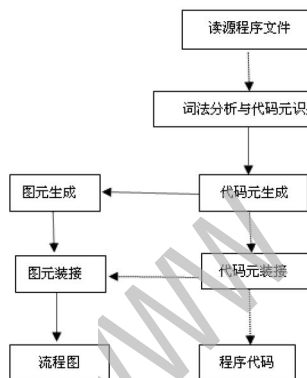


图9 流程图逆生成过程

Fig.9 Flowchart being produced reverse from the source program

代码元识别算法如下:

```

void codeRead(string s)
{
    if (s== "if" )
    {
        FcIf o1=new FcIf(X,Y,FC_image);
        CodeIf o2=new CodeIf(0,0,CODE_image);
        o1.friend=o2;

```

```

        o2.friend=o1;
        FC_p.insert(o1);
        CODE_p.insert(o2);
    }

    else
        if(s== "for" )
        {
            FcFor o1=new FcFor(X,Y,FC_image);
            CodeFor o2=new CodeFor(0,0,CODE_image);
            o1.friend=o2;
            o2.friend=o1;
            FC_p.insert(o1);
            CODE_p.insert(o2);
        }

    else
        if(s== "while" )
        {
            FcWhile o1=new FcWhile (X,Y,FC_image);
            CodeWhile o2=new CodeWhile(0,0,CODE_image);
            o1.friend=o2;
            o2.friend=o1;
            FC_p.insert(o1);
            CODE_p.insert(o2);
        }

    else
        if(s== "do" )
        {
            FcDowhile o1=new Dowhile (X,Y,FC_image);
            CodeDowhile o2=new Dowhile(0,0,CODE_image);
            o1.friend=o2;
            o2.friend=o1;
            FC_p.insert(o1);
            CODE_p.insert(o2);
        }

    else
        if(s== "switch" )
        {
            FcSwitch o1=new FcSwitch (X,Y,FC_image);
            CodeSwitch o2=new CodeSwitch(0,0,CODE_

```

```
image);  
  
        o1.friend=o2;  
        o2.friend=o1;  
        FC_p.insert(o1);  
        CODE_p.insert(o2);  
    }  
    else  
        ...  
}
```

5 结论(Conclusion)

本文提出了一个基于图元装接的流程图绘制原型系统，用户可以通过鼠标点击图元控件，在绘图区域绘制C程序传统流程图，并同步生成C语言源程序。所绘制的流程图以JPG文件格式、源程序以文本格式分别保存到磁盘。同时还具有源程序到流程图逆生成功能，通过源代码可以进行流程图二次编辑和修改。今后，我们将完成流程图手绘草图的识别和转换研究，将手工流程图的图元，转换为本系统的图元，装接后生成机绘流程图，再同步生成源程序，提高流程图的识别效率。

参考文献(References)

[1] 何骅,诸彬,徐从富.程序流程草图的存储表示及自动翻译算法[J].计算机系统应用,2009,18(4):57-60.

[2] 诸彬,何骅,徐从富.基于草图的程序流程图自动识别翻译系统[J].计算机应用与软件,2010,27(5):55-56.
[3] Martin C C,et al.RAPTOR:Introducing Programming to Non-Majors with Flowcharts[J].Journal of Computing Science in Colleges,2004,19(4):52-60.
[4] Kains C Somkiat W.Visual Programming Using Flowchart[C].Communications and information Technologies. Bangkok:ISCIT'06 2006:1062-1065.
[5] Drazen L,Ivan F.A.Visual Programming Language for Drawing and Executing Flowcharts[C].Proc of the 34th international Convention.Opatija:IEEE,2011:1679-1684.
[6] 王黎明,等.程序流程图到代码的自动生成算法[J].西安电子科技大学学报(自然科学版),2012,39(6):70-77.
[7] 钟志超,等.流程图与类C语言实时转换方法[J].东南大学学报(自然科学版),2009,39(3):503-507.
[8] 才忠胜,等.流程图的自动生成系统[J].计算机系统应用,2013,22(12):43-46.

作者简介:

许秀林(1965-),男,硕士,教授.研究领域:软件工程,智能算法.
李蕴华(1965-),女,硕士,副教授.研究领域:图像处理,信号系统.

(上接第35页)



图4 资源库

Fig.4 Resource library

5 结论(Conclusion)

随着网络的迅猛发展，如何让零散而大量的课程资源有效利用起来，设计构建实用性强的网络化课程教学支撑平台已成为重中之重。进入移动智能时代，信息的获取手段已经变得更加便捷，基于移动网络和智能终端设备的课程教学辅助平台已成为迫切需要^[10]。目前，关于移动学习的基础理论研究较多，但具体的移动学习类软件相对较少，因此，开发符合所在高校实际需求，实用性好的移动APP教学软件，辅助教学，方便学生学习，将具有很好的现实意义。

参考文献(References)

[1] Kang H,et al.Real-Time Motion Control on Android Platform[J].Journal of Supercomputing,2016,72(1):196-213.
[2] Cho H,et al.Anti-Debugging Scheme for Protecting Mobile Apps on Android Platform[J].Journal of Supercomputing, 2016,72(1):232-246.
[3] Yueyu Xu.Developing a Comprehensive Teaching Evaluation

System for Foundation Courses with Enhanced Validity and Reliability[J].Educational Technology Research and Development,2012,605:55-58.
[4] 张毅,张成霞.基于Android手机的课堂互动应用的设计和实现[J].现代电信科技,2015(1):35-39.
[5] 吴君胜,曾海.基于Android的网络课程移动学习辅助教学系统设计与应用[J].微型电脑应用,2014,30(3):18-20.
[6] 颜芳.基于Android的移动课堂学习系统的设计与实现[D].华中师范大学,2015.
[7] 夏帮贵.基于Android平台的移动学习系统[J].西华大学学报(自然科学版),2011(9):81-84.
[8] 施连敏,盖之华.基于Android的校园信息服务应用平台的设计与实现[J].软件导刊,2015,10(2):57-58.
[9] 张薇.基于WIFI移动开放课堂的构建[J].山西广播电视大学学报,2012,7(2):14-16.
[10] 柏忠贤.移动学习支持系统开发与应用研究[D].浙江师范大学,2012.

作者简介:

李 建(1979-),男,硕士,讲师.研究领域:计算机网络与信息安全,语义WEB,信息化教学.
葛竹春(1968-),男,硕士,教授.研究领域:软件工程,信息安全.
傅 伟(1982-),男,硕士,讲师.研究领域:计算机网络,物联网.