计算机科学2002Vol. 29№. 8

基于通道的螺旋型布图算法研究与实现*)

Research and Implementation on Channel Based Volute Layout Algorithm

周 俊 孙昌爱 金茂忠

(北京航空航天大学软件工程研究所 北京100083)

Abstract In this paper, a channel based volute layout algorithm is presented according to the complex topology of the association and aggregation in the class diagram. The key points of this algorithm are discussed in details. This algorithm has been applied to draw class diagram in Safepro/Java, a kind of testing tools for Java program. It is proved that graph implemented by this algorithm is symmetric and of high cohesion. It is the great improvement for inheritance based hierarchical layout algorithm.

Keywords Class diagram, Association, Aggregation, Visualization technology, Directed graph

1 引言

类图是反映面向对象程序结构的重要视图,它由类及类与类之间的关系(包括继承、关联和聚集)构成。在面向对象程序理解与测试中,通过对源程序进行静态分析可以获取类以及类之间的关系。而将类与类的关系可视化的过程就必须研究类图的布图算法[1,2]。通常用树状层次化结构可以清楚地表示类间的继承关系,而聚集和关联关系是一种网状结构,并且随着软件规模的增加,结构就愈加复杂。采用基于继承关系的层次型布图算法,不能很好地满足聚集与关联关系布图要求。

目前,关于有向图的布图算法研究较多,但对类图中的聚集和关联关系的布图表示还不存在较好的解决方案。北航软件工程研究所开发的 SafePro 系列测试工具中,采用一种以方法连接度(扇入扇出系数和)为特征的广义张量平衡算法绘制出方法间调用关系图[3],采用了以类间继承关系为主序的层次型的布图算法,绘制出了符合 UML 标准的类关系图[4]。由于聚集和关联关系的网状特征使得类关系连线交叉较多、而且相对松散,布图效果不佳。

针对聚集和关联关系的网状结构特点,本文给出了一种基于通道的螺旋型布图算法,很好地解决了面向对象软件中类间复杂的聚集关系和关联关系布图问题。

2 基于通道的螺旋型布图算法

2.1 基本术语

下面,就算法中所要涉及的一些基本术语给出相应解释:

在类图中,节点是最基本的图元,代表某个特定的类。所有关联于某节点的节点个数总和,称为该节点的扇入度,用 Div(in)表示。某节点关联到的所有节点个数总和,称为该节点的扇出度,用 Div(out)表示。节点的连接度 Div=Div(in)+Div(out)−Div(in∩out),其中 Div(in∩out)表示扇入节点和扇出节点所构成集合的公共元素个数。

2.2 算法的核心思想

基于通道的螺旋型算法由以下步骤构成:

1)按照 Div 的计算公式,计算所有节点的 Div 值;

2)从所有节点中,选择具有最大 Div 值的节点(如果存在多个节点,具有相同的最大 Div 值,则从中随机选取),所选节点即作为逻辑中心节点,并且分配以逻辑坐标原点(0,0);

3)与该中心节点具有关联关系的所有节点构成集合 T,对 T 集合中的节点按照 Div 值,以中心节点为中心,按螺旋线方向,并且兼顾均匀对称分布的要求,分配逻辑坐标(节点逻辑坐标分配方案在3.1节中详细讨论)。分配时,存在以下两种情况:

a)若最优逻辑坐标未分配给其他任何节点:此时即将该逻辑坐标分配给当前节点;

b)否则:按照由内层到外层、并按照3.1节讨论的节点编码顺序选择尚未分配逻辑坐标,分配给当前节点。

4)设已分配逻辑坐标的节点数为 N_1,节点总数为 N_0。若 $N_1<N_0$,则存在不连通的子图,设 X 为所分配的逻辑横坐标的最大值,则可以确定下个子图的中心为 X+1,从剩下未分配逻辑坐标的节点中选取具有最大 Div 值的节点,并分配以逻辑坐标(X+1,0),重复步骤3;

若 $N_1=N_0$,算法终止。

3 关键技术及其实现

在本节中,将着重介绍基于通道的螺旋的布图算法实现细节以及涉及的关键技术。

3.1 节点逻辑坐标的分配

节点的逻辑坐标描述了节点间的相对位置,不同的节点逻辑坐标分配方案直接影响最终的布图效果。下面给出本算法中实际采用的逻辑坐标分配基本方案。

在算法中,将节点周围八个相位分别命名为:N、S、E、W、NE、SE、SW 和 NW。如图1所示,图中的数字标明了八个相位的分配顺序。下面将结合图1,详细阐述节点逻辑坐标的分配策略。

首先假定当前处理的中心节点 T,其逻辑坐标为(X,Y),与该节点存在关联关系的节点集合为 $t=\{T_1,T_2,T_3,\cdots,T_m\}$,集合 t 中的第 n 个节点 T_n 为当前需要分配逻辑坐标的节点。具体的逻辑坐标分配策略如下:

1)设 T_n 的逻辑坐标为(X′,Y′),利用公式计算相应的 X′,Y′理论取值。对于 X′:

*)国家九五重点攻关项目(98-760-01-02)。周 俊 硕士研究生,主要研究方向为软件工程、软件测试技术。孙昌爱 博士研究生,主要研究方向为软件测试、软件体系结构技术、构件技术。金茂忠 教授,博士生导师,主要研究领域为软件工程环境、软件测试。

$$X'=\begin{cases} X & (n \bmod 8=1、2) \\ X-(n/8+1) & (n \bmod 8=4、6、7) \\ X+(n/8+1) & (n \bmod 8=3、5) \\ X+n/8 & (n \bmod 8=0) \end{cases} \quad (1)$$

对于 Y′：

$$Y'=\begin{cases} Y & (n \bmod 8=3、4) \\ Y-(n/8+1) & (n \bmod 8=2、5、7) \\ Y+(n/8+1) & (n \bmod 8=1、6) \\ Y+n/8 & (n \bmod 8=0) \end{cases} \quad (2)$$

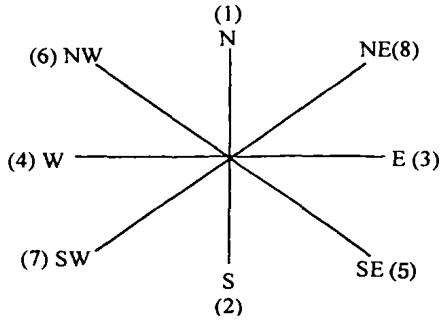

图1　逻辑坐标分配顺序示意图

2)如果计算出的节点理论坐标已经分配，则该坐标不可用。此时按照螺旋线方向由内向外循环地考察中心节点周围的每个坐标，直到首次找到尚未分配的可用坐标时，算法终止，将该坐标分配给节点。

3.2　多个子图中心节点的逻辑坐标分配方案

一般情况下，集合中的所有节点构成若干的子图，而一个子图和多个子图的处理方式从本质上说没有区别。子图布图的关键在于：如何合理而有效地安排子图的中心节点的逻辑坐标。因此，为了合理地安排各个子图的位置，有必要着重讨论子图中心节点的逻辑坐标分配方案。

在本算法中，采用了横向铺开的排列方式，使每个子图沿横向并列排放。在这里，将不再赘述子图中节点的逻辑坐标分配方案，而侧重于中心节点坐标的选择。假设算法当前需要为下一子图的中心节点 T 分配逻辑坐标(X_n，Y_n)。分配算法如下：

1)检索所有已分配逻辑坐标的节点，统计横坐标的最大值，记为 MaxX；

2)此时可以计算得：X_n=MaxX+1，Y_n=0，即该中心节点的逻辑坐标为(MaxX+1，0)；

3)按照3.1节中介绍的节点逻辑坐标分配算法，为该子图的节点分配坐标。

3.3　逻辑坐标到物理坐标的映射

逻辑坐标反映了节点间的相对位置关系，但是最终将完整的布图效果反映在屏幕上，呈现给用户，则需要完成从逻辑坐标到物理坐标的映射步骤。在这里，我们给出一个将节点的逻辑坐标转化为物理坐标的映射算法。在算法中，将用到预先设定的四个常量：X_SPACE、Y_SPACE、NODE_WIDTH 和 NODE_HEIGHT，它们分别表示节点间的横向间距、节点间的纵向间距、节点宽度和节点高度。

1)首先任意选定某点，作为起始点，记为(X，Y)，并且设定其初始取值为：X=X_SPACE，Y=Y_SPACE；

2)统计所有节点的逻辑坐标，设 X 坐标的最大值为 right，最小值为 left，Y 坐标的最大值为 top，最小值为 bottom。根据所获得的四个参数值即可确定整个布图的位置关系。

3)采用下面的公式计算所有节点的层次号，其中公式中出现的 Y 表示节点的逻辑纵坐标：level=top−Y+1；

4)遍历所有的节点，利用公式计算每个节点所在行列，记为 row 和 col：

row=X−left+1

col=top−Y+1

然后再根据计算所得的 row 和 col，确定节点的物理坐标(X′，Y′)，公式如下：

X′=X+(X_SPACE+NODE_WIDTH)×(row−1)

Y′=Y+(Y_SPACE+NODE_HEIGHT)×(col−1)

4　应用实例

在北京航空航天大学软件工程研究所研制开发的 Java 软件测试工具的类图中，应用了以类之间的继承关系为布图基本依据的层次型布图算法，在同一类图中反映类之间的三种关系(继承、关联和聚集)，如图2所示。

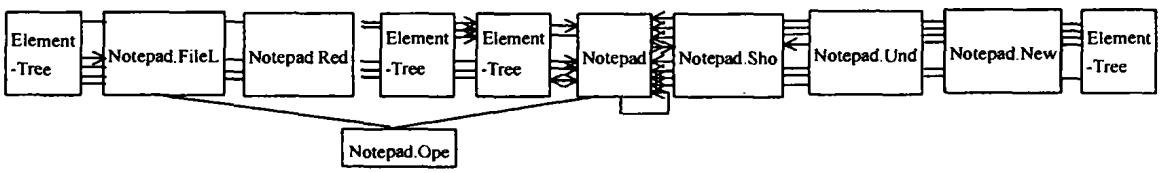

图2　基于继承关系的层次型布图算法实例

其优点在于可以比较清楚、明了而且全面地掌握所有类的关系，但存在的问题是：由于类之间的继承关系是简单的树型结构，而关联和聚集关系是复杂的网状结构，因此将继承关系作为布图依据，必然很难满足网状结构的布图要求。在该类图中，只有两个节点之间存在继承关系，而其他节点由于相互不存在继承关系，因此成为孤立的根节点，被放置在第一层。这样导致整个类图横向宽度过大，而纵向深度不足。产生这种情况的根本原因就在于，以树型结构的继承关系为依据进行复杂网状结构的关联和聚集关系布图。

对于同样一个 Java 程序，采用本文提出的基于通道的螺旋布图算法后结果如图3所示。

采用基于通道的螺旋型的类关联和聚集关系布图算法后，具有较高连接度的节点被放置在类图的中心，与中心节点存在关联关系的其他节点分别按照连接度的大小，围绕中心呈螺旋型分布。因此，本算法能够针对类关联和聚集关系网状结构特点，达到均衡和对称的布图要求。

总结　本文提出了一种基于通道的螺旋型的布图算法，着重论述了算法的核心思想，以及相关的关键技术，并结合具体的应用实例，比较分析了层次型算法与本文提出的算法的

(下转第105页)

输出 $S_n(D(i,j))$；

如果 $S_n(D(i,j))=1$

置 $F_D(i,j)$为1；

对于 O(i,j)中的每个 C(k,l)，调用过程 Zerotree(k,l,n)进行零树编码；

(5)将 n 减1，转移至(3)；

(6)结束。

零树编码过程—Zerotree(i,j,k)可具体描述如下：

(1)开始；

(2)如果 $F_C(i,j)=1$，输出|C(i,j)|的第 n 个重要位；

否则

输出 $S_n(C(i,j))$；

如果 $S_n(C(i,j))=1$，输出 C(i,j)的符号，置 $F_C(i,j)$为1；

(3)如果像素点(i,j)不是第一级小波分解内像素点

如果 $F_D(i,j)=1$，对于 O(i,j)中的每个 C(k,l)，调用过程 Zerotree(k,l,n)进行零树编码；

否则

输出 $S_n(D(i,j))$；

如果 $S_n(D(i,j))=1$

置 $F_D(i,j)$为1；

对于 O(i,j)中的每个 C(k,l)，调用过程 Zerotree(k,l,n)进行零树编码；

(4)结束。

不难看出，与 SPIHT 算法[3]相比，本文算法主要进行了如下两点改进：

·重新修改了零树结构，以2个特征图替换掉了原来的3个集合列表，从而大大降低了内存需求量，为编码算法的硬件实现创造了条件；

·修改了排序策略，提高了小波系数重要性测试速度，从而大大降低了编码时间。

4 实验结果与结论

为了验证本文算法的高效性，以下在 Pentium Ⅰ/350计算机上，以512×512×8bit 标准图像 Lena 为例，进行了3级小波分解与重构实验，并与 SPIHT 算法进行了比较，实验结果如表1和图2所示。其中，小波分解重构采用了常见的双正交9/7小波滤波器[1]。

本文以 SPIHT 算法为基础，通过修改零树结构、完善排序策略等措施，提出了一种新的低内存零树小波图像编码算法。实验结果表明：(1)利用本文算法所得到的复原图像不产生任何方块效应，图像复原质量较高；(2)相同压缩比下，本文算法与 SPIHT 算法具有相似的峰值信噪比；(3)本文算法的编解码速度明显快于 SPIHT 算法。

表1 两种图像编码算法的整体性能比较

比特率(bpp)	峰值信噪比(dB)		编码时间(s)		解码时间(s)	
	本文算法	SPIHT	本文算法	SPIHT	本文算法	SPIHT
0.125	31.02	31.08	0.47	0.51	0.06	0.07
0.25	34.03	34.10	0.56	0.71	0.12	0.15
0.5	37.09	37.21	0.65	0.90	0.19	0.25
0.75	38.81	38.90	0.88	1.15	0.34	0.40
1.0	40.18	40.39	0.94	1.26	0.41	0.49

(a)原图像 (b)SPIHT 算法复原图像 (c)本文算法复原图像

图2 标准图像重构复原效果对照(0.5bpp)

参考文献

1 Antonini M, Barlaud M, Mathieu P, Daubechies I. Image coding using wavelet transform[J]. IEEE Trans on Image Processing, 1992,1(2):205～220

2 Shapiro J M. Embedding image coding using zerotrees of wavelet coefficients[J]. IEEE Trans. on Signal Processing,1993,41(12):3445～3462

3 Said A, Pearlman W A. A new, fast, and efficient image codec based on set partitioning in hierarchical trees[J]. IEEE Trans on Circuits and Systems for Video Technology,1996,6(3):243～250

4 王向阳，杨红颖. 基于多阈值与嵌入零树小波的图像压缩算法[J]. 通信学报,2001,22(12):88～93

5 Li J, Lei S. An embedded still image coder with rate-distortion optimization[J]. IEEE Trans on Image Processing,1999,8(7):913～924

6 王向阳，杨红颖. 基于小波变换的有损图像压缩算法研究[J]. 计算机工程与应用,2001,37(15):82～85

(上接第127页)

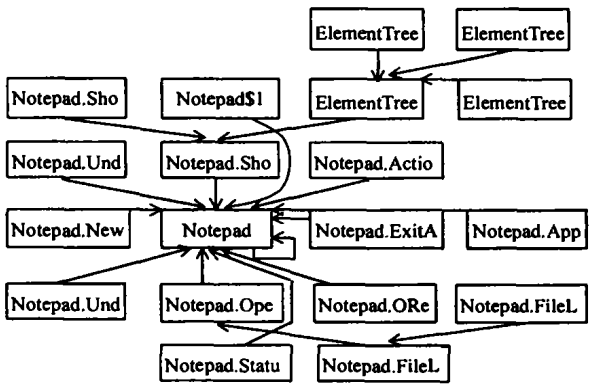

图3 基于通道的螺旋型布图算法实例

应用效果。结果表明：该算法比较好地克服了层次型布图算法的不足，适用于类关联和聚集关系的布图。

参考文献

1 Pressman R S. Software Engineering: A Practitioner 's Approach. McGraw-Hill, Fourth Edition, 1997

2 Tilley S R. The Canonical Activities of Reverse Engineering. Baltzer Science Publishers, The Netherlands, Feb. 2000

3 孙昌爱，刘超，金茂忠. 一种有效的软件结构图的布图算法. 北京航空航天大学学报,2000,26(6):1305～1309

4 刘超，李健，沈海华. 面向对象程序可视化类图的逆向自动生成. 北京航空航天大学学报,1998,24,411～414