Marked Graph Classes

Tomokazu ARITA

Nihon University

Last modification: 29, August, 2003

1 はじめに

プログラム仕様書の作成・編集支援システムの開発は，仕様書の定式化，システム構造の設計，編集操作の定式化，データ構造の提案及システムの構築が段階的に行われてきている．現在，プログラム仕様書のガイドラインである ISO6592 に記述されている全ての項目を含む仕様書の定式化が，[1] で行われている．この定式化には属性グラフ文法が用いられている．また，その結果に伴い，システム構造の設計及び編集操作の定式化の提案がなされてきた [2].

本解説書は，仕様書の構造を表すグラフを Java 言語により実装する方法について述べる．

2 準備

2.1 プログラム仕様書 Hiform とマーク付きグラフ

プログラム仕様書 Hiform は，プログラム仕様書のガイドライン ISO6592 の項目を全て含む表形式の仕様書である．その項目間の関係はマークつきグラフで表される．

仕様書を表すマーク付きグラフでは次の 3 種類の辺を使用する．(1)“in” は within を表し，項目間の包含関係・グループ化を表す．(2)“ov” は over を表し，項目間の上下関係を表す．(3)“lf” は left of を表し，項目間の左右の関係を表す.

3 Marked Graph Classes

Marked Graph Classes(MGC) は，マーク付きグラフ (Marked Graph) を JAVA により実装する際に使用するクラスファイルのセットである．この MGC は，パッケージ hiformed.grammar に属しており，次の 2 つのクラスにより

構成される．(1) クラス MGNode は，マーク付きグラフの１頂点とその周辺の辺を実装する際に使用する．マーク付きグラフは MGNode を繋いだリスト構造で表される．(2) クラス MarkedGraph は，MGNode でつながれたリストの１頂点へのポインタを保持するクラスである．次の節では，各クラスのデータ構造について述べる．

4 Data Structures for Marked Graph Classes

4.1 Marked Graph Class(hiformed.grammar.MarkedGraph)

コンストラクタ

public MarkedGraph()
MarkedGraph クラスを生成する．

メソッド

setStartNode
public void setStartNode(MGNode node)
　リストの先頭頂点 node を設定する．

getStartNode
public MGNode getStartNode()
　設定されているリストの先頭頂点を返す．設定されていない場合は null を返す．

4.2 MGNode Class(hiformed.grammar.MGNode)

　マーク付きグラフを表すリストの各頂点.（各頂点へのポインタがグラフの辺を表す）

コンストラクタ

public MGNode()
　MGNode を生成する．生成後は，各メソッドで値を設定する必要がある．

メソッド

setNodeID
public void setNodeID(int id)
　ノードの id(¿0) を設定する．

getNodeID
public int getNodeID()
　ノードの id を返す．設定されていない場合は-1 を返す．

setInStartNode
public void setInStartNode(MGNode node)
“in” とラベル付けられた辺が，この頂点に接続されている時にこの頂点が辺の終端頂点ならば，その辺の開始頂点として隣接頂点 node へのポインタを設定する．

setInEndNode
public void setInEndNode(MGNode node)
“in” とラベル付けられた辺が，この頂点に接続されている時にこの頂点が辺の開始頂点ならば，その辺の終端頂点として隣接頂点 node へのポインタを設定する．

setOvStartNode
public void setOvStartNode(MGNode node)
“ov” とラベル付けられた辺が，この頂点に接続されている時にこの頂点が辺の終端頂点ならば，その辺の開始頂点として隣接頂点 node へのポインタを設定する．

setOvEndNode
public void setOvEndNode(MGNode node)
“ov” とラベル付けられた辺が，この頂点に接続されている時にこの頂点が辺の開始頂点ならば，その辺の終端頂点として隣接頂点 node へのポインタを設定する．

setLfStartNode
public void setLfStartNode(MGNode node)
“lf” とラベル付けられた辺が，この頂点に接続されている時にこの頂点が辺の開始頂点ならば，その辺の終端頂点として隣接頂点 node へのポインタを設定する．

setLfEndNode
public void setLfEndNode(MGNode node)
“lf” とラベル付けられた辺が，この頂点に接続されている時にこの頂点が辺の開始頂点ならば，その辺の終端頂点として隣接頂点 node へのポインタを設定する．

getInStartNode
public MGNode getInStartNode()
“in” とラベル付けされた辺がこの頂点に接続されている時にこの頂点がその辺の終端頂点ならば，隣接する開始頂点を返す．もし，辺が存在しない場合は null を返す.

getInEndNode
public MGNode getInEndNode()

“in” とラベル付けされた辺がこの頂点に接続されている時にこの頂点がその辺の開始頂点ならば，隣接する終端頂点を返す．もし，辺が存在しない場合は null を返す．

getOvStartNode
public MGNode getOvStartNode()
“ov” とラベル付けされた辺がこの頂点に接続されている時にこの頂点がその辺の終端頂点ならば，隣接する開始頂点を返す．もし，辺が存在しない場合は null を返す．

getOvEndNode
public MGNode getOvEndNode()
“ov” とラベル付けされた辺がこの頂点に接続されている時にこの頂点がその辺の開始頂点ならば，隣接する終端頂点を返す．もし，辺が存在しない場合は null を返す．

getLfStartNode
public MGNode getLfStartNode()
“lf” とラベル付けされた辺がこの頂点に接続されている時にこの頂点がその辺の終端頂点ならば，隣接する開始頂点を返す．もし，辺が存在しない場合は null を返す．

getLfEndNode
public MGNode getLfEndNode()
“lf” とラベル付けされた辺がこの頂点に接続されている時にこの頂点がその辺の開始頂点ならば，隣接する終端頂点を返す．もし，辺が存在しない場合は null を返す．

setNodeLabel
public void setNodeLabel(int label)
頂点のラベル label を設定する．

getNodeLabel
public int getNodeLabel()
頂点のラベルの値を返す．

setAttX
public void setAttX(int x)
属性 x の値を設定する．

getAttX
public int getAttX()
属性 x の値を返す．

setAttY
public void setAttY(int y)

属性 y の値を設定する.

getAttY
public int getAttY()
属性 y の値を返す

setAttWidth
public void setAttWidth(int width)
属性 width の値を設定する.

getAttWidth
public int getAttWidth()
属性 width の値を返す.

setAttHeight
public void setAttHeight(int height)
属性 height の値を設定する.

getAttHeight
public int getAttHeight()
属性 height の値を返す.

setNodeName
public void setNodeName(java.lang.String name)
頂点の名前 name を設定する.

getNodeName
public java.lang.String getNodeName()
頂点の名前を返す.

4.3 おわりに

本解説書では，Marked Graph Classes についてそのデータ構造について述べた．現在，これに基づいた仕様書作成・編集支援系が開発中である．

参考文献

[1] 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)*

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