データ構造可視化のためのオブジェクトレイアウト言語の設計

Design of the Object Layout Language to Visualize Data Structure

松崎　駿[*1]，　酒井　三四郎[*2]，　松澤　芳昭[*3]
Shun MATSUZAKI[*1]，　Sanshiro SAKAI[*2]，　Yoshiki MATSUZAWA[*3]
*1*2*3 静岡大学情報学部
*1*2*3 Faculty of Informatics, Shizuoka University
Email: matsuzaki-s@sakailab.info

あらまし：プログラムの可視化を行うツールの問題点として，配置が期待どおりに行われないという点が挙げられる．そこで図として表示されるオブジェクトの配置を決めるためのレイアウト言語，及びプログラム実行時にそのレイアウト言語で記述された定義の通りにオブジェクトの配置を行うツールを開発した．
キーワード：データ構造，可視化，レイアウト，言語

1. 研究の背景と目標

データ構造を構築するオブジェクトが持つ変数には意味が込められている．例えば，ある二分木を構成するプログラムで生成されるオブジェクトの right と名付けられた変数は自身の右部分木の根を参照する変数とされており，連結リストを構築するプログラムで生成されるオブジェクトの変数 next は自身のオブジェクトの次のデータを参照する変数である．

プログラミング教育では，データ構造理解を補助する手段の一つとしてオブジェクト構造を図示するという方法がしばしば取られるが，右部分木を左に表示するような変数の意味を無視したオブジェクトの配置を行うとかえって学習者の混乱を招く．

そこで本研究では，データ構造のオブジェクトが持つ変数の意味に則した配置を行うためのオブジェクトレイアウト言語の設計を行った．

2. 関連研究

Demian らはプログラムデバッグ時の各オブジェクトの変数の値，参照先，実行中のメソッドといった情報を図で表示する Eclipse プラグイン Jive[(1)]を開発した．

Jive で描かれる図では，他のオブジェクトに参照されているオブジェクトは参照しているオブジェクトの下に配置される．オブジェクトが複数のオブジェクトを参照している場合，参照されているオブジェクトは参照しているオブジェクトの下に横一列で並べられる．このとき，横に並ぶ順番はオブジェクトが生成された順番となる．そのため，二分木で右部分木を参照する変数が参照しているにもかかわらず，対象のオブジェクトが左側に配置されるといったことが起こりうる．

3. システムの提案

3.1 概要

本研究では，オブジェクトレイアウト言語：DAST の開発を行った．

データ構造を構成するオブジェクトのクラス名，変数名の付け方はプログラムの作成者によって微妙に異なる．そのため各変数の意味を可視化ツールに理解させるということは困難である．

そこで DAST では，データ構造を構成するクラスが持つ各変数に対して，自身のオブジェクトから見てその変数が参照するオブジェクトをどの方向に配置するかを指定する．これにより，各オブジェクトをどの位置に配置するかが決定できる．

3.2 アーキテクチャ

本システムのアーキテクチャを図 1 に示す．

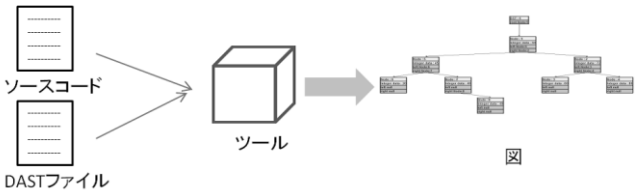

図 1 システムのアーキテクチャ

このシステムでは，データ構造を構成するプログラムのソースコードと DAST ファイルを読み込ませることで通常のプログラムの実行結果に加え，各オブジェクトの状態が明示された図が得られる．

3.3 DAST の文法

DAST による二分木のレイアウトの例を以下に示す．

```
Class:BST, Node
BST{
        root:v
}
Node{
        right:v>
        left:v<
}
```

プログラムは木の節となるオブジェクトのクラスである Node と，根となる節を参照しているオブジェクトのクラス BST で構成されている．

1 行目の"Class:"に続けて描画するオブジェクトのクラスを入力し，2 行目以降でそれらのクラスが持

つ参照型変数または配列について，その参照するオブジェクトを自分の周囲のどの方向に配置するかを指定する．方向の指定には，v, ^ , < , > の 4 種の文字を用い, うち 2 種類を組み合わせたもの(< と > , v と^の組み合わせを除く）を含め 8 方向のいずれかを指定する．

この場合は BST クラスの変数 root が参照しているオブジェクトを下に，Node クラスの変数 right が参照するオブジェクトを右下に，left が参照するオブジェクトを左下に配置すると定義している．

実際に二分探索木のプログラムを実行し，先ほどの二分探索木の定義を読み込んだ結果得られる図が図 2 である．

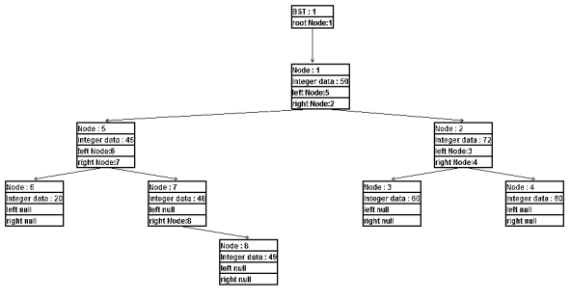

図 2 二分木の例

ハッシュのチェイン法の場合，DAST の記述は次のようになる．

```
Class:HashC, Cell, MyKey
HashC{
        table[]:v
}
Cell{
        key:>
        next:v
}
```

プログラムはハッシュ表を持つオブジェクトのクラス HashC，格納されるデータとなるオブジェクトのクラスである Cell，キーとなるオブジェクトのクラスである MyObject で構成される．

この記述にしたがって配置を行った結果得られるのが図 3 である．

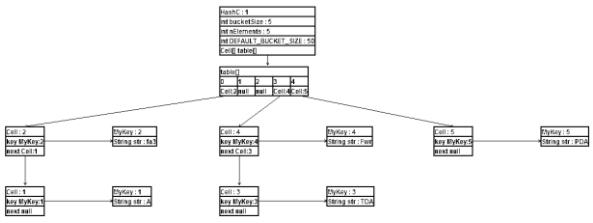

図 3:ハッシュ　チェイン法の例

4. 評価

実際に java で組まれているデータ構造のうち，DAST による定義がどの程度の範囲を網羅できるのかを評価する．

4.1 評価方法

DAST による配置定義が行えるかを確認するために,「静岡大学情報学部の講義アルゴリズムとデータ構造Ｉ－ＣＳ」のテキストとして用いられる書籍[2]にて掲載されているデータ構造のプログラムの配置を DAST で記述した．

さらに，DAST がプログラムの実装方法に依存することなく対応することができるかを確認するために，書籍(2)取り扱われていた 5 種類のデータ構造について，別の書籍(3)(4)に掲載されている同じデータ構造で実装の異なるプログラムを用い同様の確認を行った．

4.2 結果

表 1:結果

	書籍(2)	書籍(3)	書籍(4)
連結リスト	○	○	○
二分木	○	○	○
B 木	○	○	-
ハッシュ（オープンアドレス）	○	-	○
ハッシュ（チェイン）	○	-	○

表 1 に実験結果を示す（○:DAST を用いて配置を定義することができた， - :該当プログラムが非掲載）

5. 結論

今回用いた全てのデータ構造のプログラムで DAST による配置定義をすることができた．

このことから，本研究で設計した DAST によるレイアウト定義はデータ構造の学習者が最低限理解すべき範囲を全て網羅できていると考えられる．

参考文献

(1) Demian Lessa, Jeffrey K. Czyz, Bharat Jayaraman :JIVE: A Pedagogic Tool for Visualizing the Execution of Java Programs, Technical report, State University of New York at Buffalo (2010), http://www.cse.buffalo.edu/tech, (2014/02/01 アクセス)

(2) 近藤嘉雪[著]，『定本 Java プログラムのためのアルゴリズムとデータ構造』，2011，ソフトバンククリエイティブ，487pp

(3) 奥村晴彦，首藤一幸，杉浦方紀，土村展之，津留和生，細田隆之，松井吉光，光成滋生[著]，『Java によるアルゴリズム辞典』，2003，技術評論社，500pp

(4) Robert Lafore[著]，岩谷宏[訳]，『Java でまなぶアルゴリズムとデータ構造』，1991，ソフトバンク株式会社，628pp