The Formal Design Models of Digraph Architectures and Behaviors

Yingxu Wang, University of Calgary, Canada
Aderemi Adewumi, University of Lagos, Nigeria, and University of Calgary, Canada

ABSTRACT

Graphs are one of the most fundamental and widely used non-linear hierarchical structures of linked nodes. Problems in sciences and engineering can be formulated and solved by the graph model. This paper develops a comprehensive design pattern of formal digraphs using the Doubly-Linked List (DLL) architecture. The most complicated form of graphs known as the weighted digraph is selected as a general graph model, based on it simple graphs such as nondirected and/or nonweighted ones can be easily derived and tailored. A rigorous denotational mathematics, Real-Time Process Algebra (RTPA), is adopted, which allows both architectural and behavioral models of digraphs to be rigorously designed and implemented in a top-down approach. The architectural models of digraphs are created using RTPA architectural modeling methodologies known as the Unified Data Models (UDMs). The physical model of digraphs is implemented using nodes of DLL dynamically created in the memory. The behavioral models of digraphs are specified and refined by a set of 18 Unified Process Models (UPMs) in three categories namely the management operations, traversal operations, and node manipulation operations. This work has been applied in a number of real-time and non-real-time system designs and specifications such as a Real-Time Operating System (RTOS+), graph-based and tree-based applications, and the ADT library for an RTPA-based automatic code generation tool.

Keywords: Abstract Data Types, Digraphs, Formal Design Models, Graph Theory, Real-Time-Process-Algebra (RTPA), Software Engineering, Software Science, System Architecture System Behavior Specification, Unified Data Models, Unified Process Models

1. INTRODUCTION

Data object modeling is a process to creatively extract and abstractly represent a real-world problem by data models based on constraints of given computing resources. Abstract Data Types (ADTs) are a set of highly generic and rigorously modeled data structures in type theory (Guttag, 1977; Broy et al., 1984; Cardelli & Wegner, 1985; Stubbs & Webre, 1985), which is an abstract logical model of a complex data structure with a set of predefined operations. A number of ADTs
have been identified in computing and system modeling such as stack, queue, sequence, record, array, list, tree, file, and graph (Broy et al., 1984; Mitchell, 1990; Mc Dermid, 1991; Wiener & Pinson, 2000; Wang, 2007; Wang, Ngolah, Tan, Tian, & Sheu, 2010). ADTs possess the following properties: (i) An extension of type constructions by integrating both data structures and functional behaviors; (ii) A hybrid data object modeling technique that encapsulates both user defined data structures and allowable operations on them; (iii) The interface and implementation of an ADT are separated where detailed implementation of the ADT is hidden to applications that invoke the ADT and its predefined operations.

Graphs are a typical data object for modeling a system of hierarchical and networked components in computing and system architectural description. Graphs are the most complicated data objects in system architectural and behavioral modeling beyond records, lists and trees. Graphs are a well studied ADT supported by both computer science and graph theory as a mathematical structure (Harary, 1969; Lipschutz & Lipson, 1997; Bondy & Murty, 2008).

Definition 1. A graph (G) is a tuple of a finite nonempty set of nodes N and a finite nonempty set of edges E, i.e.:

\[
G \equiv (N,E), e = (n_i, n_j) \in E, 1 \leq i, j \leq |N| \land i \neq j
\]

where a node, \( n \in N \), is an indexed structure such as a record or array of information and/or functions; and an edge, \( e = (n_i, n_j) \in E \), is a relation between a pair of nodes.

Graphs can be classified into the categories of simple, weighted, directed, and weighted directed graphs. The simple graph has been formally described in Definition 1. The weighted and directed graphs are characterized by the weighted or ordered edges E, respectively. Based on Definition 1, the weighted, directed, and weighted directed graphs can be derived as follows.

Definition 2. The weighted graph \( G^* \), digraph \( \overline{G} \), and weighted digraph \( \overline{G}^* \) are extensions of the simple graph as follows, respectively:

\[
G^* \equiv (N,E^*), e^* = (r(n_i, n_j), w(n_i, n_j)) \in E^*, 1 \leq i, j \leq |N| \land i \neq j
\]

\[
\overline{G} \equiv (N,\overline{E}), \overline{e} = (r(n_i, n_j)) \in \overline{E}, 1 \leq i, j \leq |N| \land i \neq j
\]

\[
\overline{G}^* \equiv (N,\overline{E}^*), \overline{e}^* = (\overline{r}(n_i, n_j), w(n_i, n_j)) \in \overline{E}^*, 1 \leq i, j \leq |N| \land i \neq j
\]

where \( r(n_i, n_j) \) and \( \overline{r}(n_i, n_j) \) denote a normal and ordered edge, and \( w(n_i, n_j) \) the weight of an edge.

It is noteworthy that the weighted digraph as defined in Eq. 4 is a general model of all types of graphs. Other simple graphs such as nondirected and nonweighted ones can be tailored and derived on the basis of digraphs.

As a non-linear hierarchical structure of linked nodes, both graphs and trees have high-level similarity in their architectural and behavioral models (Stubbs & Webre, 1985; Mc Dermid, 1991; Lipschutz & Lipson, 1997; Gosling, Bollella, Dibble, Furr, & Turnbull, 2002, Wang & Tan, 2011). A tree can be perceived as a special case of a graph that is connected and has no loop. A minimum connected subgraph is a special tree known as the minimum spanning tree.
The Formal Design Model of a Real-Time Operating System (RTOS+): Conceptual and Architectural Frameworks
[www.igi-global.com/article/formal-design-model-real-time/43900?camid=4v1a](www.igi-global.com/article/formal-design-model-real-time/43900?camid=4v1a)

Inconsistency-Induced Learning for Perpetual Learners
[www.igi-global.com/article/inconsistency-induced-learning-perpetual-learners/64178?camid=4v1a](www.igi-global.com/article/inconsistency-induced-learning-perpetual-learners/64178?camid=4v1a)