Design Automation Renegades

GLOBETROTTING DIVISION

Boilerplate Code: Data Structures and Algorithms for Design Automation

Zhiyang Ong [1]

REPORT ON
Common Data Structures and Algorithms
Found in Bolierplate Code for
Design Automation Software

September 17, 2015

[1]Email correspondence to: ✉ ongz@acm.org

Abstract

This report describes the design and implementation of common data structures and algorithms, as well as "computational engines" that are found in electronic design automation (EDA) software.

Data structures and algorithms for digital VLSI and cyber-physical system design include: binary decision diagrams (BDDs), AND-inverter graphs (AIGs), and their associated algorithms for optimization, traversal, and other operations (such as graph matching). Common computational engines for digital systems would include: optimization and verification engines for deterministic and nondeterministic finite state machines; decision procedures for the boolean satisfiability problem (SAT solvers) and satisfiability modulo theories (SMT solvers); quantified boolean formula (QBF) solvers; and SAT and SMT solvers for maximum satisfiability (i.e., Max-SAT and Max-SMT solvers).

Regarding EDA problems that require numerical computation (in digital, analog, or mixed-signal VLSI design), the data structures and algorithms for circuit simulation based on sparse graph would be required. In addition, techniques for model order reduction shall be implemented.

Computational engines for statistical and probabilistic analyses or stochastic modeling can include data structures and algorithms for partially observable Markov decision processes (POMDPs) and Markov chains. Tools for analyses of queueing systems (based on queueing theory) should be included.

Regarding cyber-physical systems and mixed-signal circuits, hybrid automata can be used to represent these circuits and systems.

Optimization engines for EDA include: solvers for different types of mathematical programming, such as linear programming (LP), integer linear programming (ILP), mixed-integer linear programming (MILP), quadratic programming (QP), convex programming (CP), geometric programming (GP), and second-order conic programming (SOCP); solvers for pseudo-boolean optimization (PBO solvers) and weighted-boolean optimization (WBO); and meta-heuristics (e.g., evolutionary algorithms, simulated annealing, and ant colony optimization).

Algorithms shall be implemented using parallel programming, in a scalable style. In addition, considerations shall be given to the use of constraint programming.

More stuff to be included…

Revision History

Revision History:

1. Version 0.1, December 23, 2014. Initial copy of the report.
2. Version 0.1.1, September 16, 2015. Added sections for mathematics and statistics, and the abstract.

Contents

Chapter 1

Algorithms

This section documents algorithms that I have implemented for my C++ -based boilerplate code repository.

A template for typesetting algorithms is shown in PROCEDURE 1.

```
NAME OF THE ALGORITHM(ARGUMENTS)
       // Input ARGUMENT #1: Definition1
       // Input ARGUMENT #2: Definition2
 1     BODY OF THE PROCEDURE
       // A while loop.
 2     while [condition]
 3         [Something]
       // A for loop.
 4     for $Var$ = [initial value] to [final value]
 5         [Something]
       // An if-elseif-else block.
 6     if [Condition1]
 7         Blah...
 8     elseif [Condition2]
 9         Blah...
10     elseif [Condition3]
11         Blah...
12     else
13         Blah...
       // A variable assignment.
14     $blah = A[j]$
           // This is indented with a tab.
       // What is the output of this procedure?
15     return
```

Chapter 2

Data Structures

2.1 Graphs

2.1.1 Directed Graphs

2.1.1.1 Functions that need to be implemented

2.1.1.2 Binary Decision Diagrams (BDDs)

2.1.1.3 AND-Inverter Graphs (AIGs)

2.1.2 Undirected Graphs

Chapter 3

Mathematics

Math symbols that I use frequently:

1. $\mathbb{N}$
2. $\sum\limits_{n}^{i=1}$
3. $f(x) = \lim\limits_{n \to \infty} \dfrac{f(x)}{g(x)}$
4. $\varnothing$
5. q

A 3×3 matrix: $\begin{pmatrix} 11 & 12 & 13 \\ 21 & 22 & 23 \\ 31 & 32 & 33 \end{pmatrix}$

Here is an equation:

$$\iint_{\Sigma} \nabla \times \mathbf{F} \cdot \mathrm{d}\mathbf{\Sigma} = \oint_{\partial \Sigma} \mathbf{F} \cdot \mathrm{d}\mathbf{r}. \tag{3.1}$$

Here is an equation that is not numbered.

$$\nabla \times \mathbf{E} = -\frac{\partial \mathbf{B}}{\partial t}$$

Here is the set of Maxwell's equations that is numbered.

$$\nabla \cdot \mathbf{E} = \frac{\rho}{\varepsilon_0} \tag{3.2}$$

$$\nabla \cdot \mathbf{B} = 0 \tag{3.3}$$

$$\nabla \times \mathbf{E} = -\frac{\partial \mathbf{B}}{\partial t} \tag{3.4}$$

$$\nabla \times \mathbf{B} = \mu_0 \left(\mathbf{J} + \varepsilon_0 \frac{\partial \mathbf{E}}{\partial t} \right) \tag{3.5}$$

$$\begin{gathered}\text{minimize} \sum_{\mathrm{i}=1}^{\mathrm{c}} \mathrm{c_i} \cdot \mathrm{x_i} \\ \underline{x} \in S \\ \text{subject to}: \\ x_1 + x_4 = 0 \\ x_3 + 7 \cdot x_4 + 2 \cdot x_9 = 0\end{gathered}$$

$$f(n) = \begin{cases} case - 1 & : \text{n is odd} \\ case - 2 & : \text{n is even} \end{cases} \tag{3.6}$$

Proof. This is a proof for BLAH ... □

Theorem 3.1. *TITLE of theorem. My theorem is...*

Axiom 3.1. *TITLE of axiom. Blah...*

Cases of putting a bracket/parenthesis on the right side of the equation.

$$\left.\begin{aligned} B' &= -\partial \times E, \\ E' &= \partial \times B - 4\pi j, \end{aligned}\right\} \quad \text{Maxwell's equations}$$

Labeling an arrow: $\xrightarrow{ewq}$

Chapter 4

Statistics

Chapter 5

STL Resources

Some C++ STL resources are:

1. [2]: http://www.tutorialspoint.com/cplusplus/cpp_stl_tutorial.htm
2. [1] and CplusplusCom2015: http://www.cplusplus.com/reference/stl/
3. http://en.cppreference.com/w/cpp/container
4. http://www.cs.wustl.edu/~schmidt/PDF/stl4.pdf

5.1 Computational Complexity of C++ Containers

Table 5.1: Computational Complexity of C++ Containers

Container \ Complexity	add	remove	search	size	empty	begin	end
vector	O(1)	O(n)	O(n)	O(1)	O(1)	O(1)	O(1)

Acknowledgments

I would like to thank Ms. Deepika Panchalingam for motivating me to revise basic data structures and algorithms for internship and job interviews.

Bibliography

[1] cplusplus.com. The C++ resources network: C++ reference. Available online at: `http://www.cplusplus.com/`; April 2, 2014 was the last accessed date, 2014.

[2] Mohtashim. C++ STL tutorial. Available online at *Tutorials Point: C++ Tutorial: C++ STL Tutorial*: `http://www.tutorialspoint.com/cplusplus/cpp_stl_tutorial.htm`; September 17, 2015 was the last accessed date, 2015.