A Taxonomy and Evaluation of Algorithms used in the ACM Programming Competition

Literature Review

Supervisor: A.J. Ebden

Researcher: Douglas Hobson

Abstract

Several classifications of algorithms used in the ACM Programming Competition and other programming competitions have been developed by individuals or companies as well as collections, summaries and definitions of algorithms. These have proved invaluable in the development of my own classification of the algorithms found in the ACM Programming Competitions run in South Africa over the last six years

1. Introduction

I have found several good sources in the area of algorithm classification. Most of the work in informal in nature, but very similar to the work that I am doing and so is particularly useful. In this paper I will be looking at several classifications that I have found, some specifically dealing with the ACM, another derived by Hal Burch from the USA Computer Olympiad [Burch, 1999] and the classification given by the Top Coder website [Top Coder, 2007]. I will also present my own classification (thus far) of ACM problems from the South African competition from the last six years.

The other thing that will be looked at in this paper is the different types of algorithms already defined and sources that list and explain these. One such source is the Stony Brook Algorithm Repository maintained by Steven Skiena and another is the Dictionary of Algorithms and Data Structures maintained by NIST, the National Institute of Standards and Technology [Black, 2007].

These two broad categories of sources overlap a little. The classifications often include well defined algorithms and explanations of how those algorithms work similar to that found in the algorithm resources.

2.1 Other People's Classifications

2.1.1 Howard Cheng's Classification

Two of the classifications are derived from the ACM Programming Competition. One of these is very good and looks as though it will be quite useful. It was developed by Howard Cheng from the University of Lethbridge [Cheng, 2006]. Cheng has been heavily involved with the ACM Programming Competition as a contestant, a judge, a problem designer, a system administrator, and a local organizer. Cheng's classification does not give a clear description of any of the algorithms, but instead has provided a multitude of examples of each problem type. Fortunately several of the names that Cheng has given to categories in his classification are commonly used and are well defined elsewhere.

Classification from Howard Cheng:

Arithmetic
Backtracking
Big Number
Combinatorics
Data Structure
Dynamic Programming
Geometry
Graph
Greedy
Miscellaneous
Number Theory
Parsing
Permutations
Probability
Recursion
Search
Simulation
Sorting
Straightforward
String

A problem or difficulty with this classification is that Cheng alternates between the type of algorithm used to solve the problem and the sort of problem. As an example, Dynamic Programming is a way of programming that takes a very computationally intensive problem and removes some of the complexity by using up more memory. It is quite a well defined way of solving specific types of problems. Simulation on the other hand, simply means that you have to simulate some game or system to solve the problem. Each problem will have a solution very different to the next, with the solution type depending wholly on problem statement.

The other classification derived from the ACM Programming Competition is very basic, not suitable for use and will receive no further mention.

2.1.2 Hal Burch's Classification

The classification developed by Hal Burch from USACO [Burch, 1999] looks to be the most useful of all the classifications that I have come across. It has a fairly good list of algorithms as well as good descriptions of how each works. It also provides examples of problems that fall into each category. Another good thing about this classification is that the problem set that it was derived from is similar to the sort of problems given in the ACM Programming Competition.

Hal Burch's classification:

Ad Hoc Problems
Approximate Search
BigNums
Complete Search
Computational Geometry
Dynamic Programming
Eulerian Path
Flood Fill
Greedy
Heuristic Search
Knapsack
Minimum Spanning Tree
Network Flow

Recursive Search Techniques
Shortest Path
Two-Dimensional Convex Hull

2.1.3 Top Coder's Classification

Most classifications were developed by one person or a small group working together. An exception to this is the Top Coder classification. Top Coder is a commercial site so it is likely that their classification is a little more reliable, but it is derived from problems found on the Top Coder site. This means that while it is useful for defining algorithm types, it does not cover the same set of questions as the ACM Programming Competition.

Classification from Top Coder:

Advanced Math
Brute Force
Dynamic Programming
Encryption/Compression
Geometry
Graph Theory
Greedy
Math
Recursion
Search
Simple Math
Simple Search/Iteration
Simulation
Sorting
String Manipulation
String Parsing

2.1.4 Comparison of Classifications

Both Cheng and Burch's classifications include a catchall category. Cheng's has two: "Straightforward" and "Miscellaneous". Burch's has "Ad hoc". Both seem to think that there are

usually a few problems in each competition that do not conform to any specific algorithm and each problem that falls into these categories needs to be solved on its own. The impression that I got from both Cheng and Burch is that these problems are fairly easy but occasionally appear challenging until properly understood.

There are several categories in Cheng and Burch's classifications that describe the same thing or are related in some way. The following table shows these relationships with comments to clarify where appropriate.

Cheng	Burch
Big Number	BigNums
Combinatorics	Knapsack (specific class of Combinatorics)
Data Structure	Minimum Spanning Tree (subset of Data Structure)
Dynamic Programming	Dynamic Programming
Geometry	Computational Geometry
Graph (Graph Theory)	Shortest Path Network Flow (all are subsets Eulerian Path of Graph Theory) Minimum Spanning Tree
Greedy	Greedy
Miscellaneous Straightforward	Ad Hoc Problems
Search	Heurist Search (not identical, but related) Note that Burch's "Complete Search" is not related to Cheng's "Search" at all but would be better known as Brute Force.

Table 2.1.3: A comparison of Cheng and Burch's classifications

The Top Coder classification was not included in this table, despite it also having several categories that are related to categories in Cheng and Burch's classifications, because its problem set is quite different from ACM problems.

2.2 My Classification

I have developed my own classification derived from all the ACM competitions held in South Africa. There are 36 problems covered by my classification: six years of questions with six problems per year. I have included my classification of all the problems on a problem by problem basis in the Appendix.

My classification:

Backtracking
Complete Search
Dynamic Programming
Encryption
Game/Puzzle
    Chess
    Sudoku
Graph Theory
Greedy
Mathematical
    Bases
    General
    Geometry
    Perfect Squares
    Physics
Parsing
Straightforward
Tree

Many of the 36 problems in my problem set need elements from more than one category in my classification to solve. This makes them more interesting but also harder to classify. Where a problem has needed more than one technique or algorithm to solve I have kept the algorithms separate and simply said that the problem needs a combination of algorithms to solve. This seems better than to try and reduce each problem down to a single algorithm which would result in inaccuracy or try to specify a new algorithm that accomplishes what a combination would have.

In my classification I have divided some fields into sub categories, an example being Mathematics. Many problems need some knowledge of mathematics, but they clearly do not all need the same specific knowledge. Some require an understanding of number systems and how they function as well as how to convert from one base or number system to another. Several other questions require an understanding of basic geometry as well as trigonometry; these have been grouped together as Geometry.

The category Tree needs some clarification. This category refers specifically to those algorithms that require the construction and traversal of a tree data structure. This includes all the possible methods of traversing the tree, i.e. breadth and depth first searches. While they are separate algorithms, they are so closely related that I have put them into one category.

A category that is quite fun and interesting is the Game/Puzzle category. I included this to indicate that basic (or sometimes in-depth) knowledge of a given game or puzzle would be very useful in solving the problem.

I have also included a category like Cheng's called Straightforward to cater for problems that may be too simple to assign to any algorithm class or as a comment to add onto the classification of a problem to indicate that understanding of the specified algorithm, while potentially useful, is not necessary to solve the problem.

3. Algorithm Resources

The Stony Brook Algorithm Repository maintained by Steven Skiena of Stony Brook University [Skiena, 2001] looks as though it will be useful. Steven Skiena is involved with the Stony Brook ACM ICPC and the Stony Brook team has done well under his guidance [Skiena, 2001]. His repository contains several algorithms commonly found in the ACM ICMP with good descriptions of each algorithm type as well as code implementations in several common programming languages.

Another good resource is the Dictionary of Algorithms and Data Structures maintained by NIST [Black, 2007]. It is very extensive and has excellent descriptions of nearly every algorithm I have ever come across. It also often provides links to other sites to provide further clarification or examples of the algorithm being implemented.

Most of the classifications can also be regarded as algorithm resources, if to a slightly lesser degree than Skiena and Black's.

4. Conclusion

There are several classifications out there, though they are informal, they are very good. They provide excellent reference material and give me something fairly reliable to hold my classification up against for comparison. The Top Coder classification [Top Coder, 2007], while unrelated to the ACM, is still valuable for the purposes of understanding and defining the algorithms involved. Cheng [Cheng, 2006] and Burch's [Burch, 1999] classifications are excellent resources and have been very useful to me in the design of my own classification by providing names for algorithms that are widely used and exposing me to the many formally defined algorithms that are needed for such a classification.

The two algorithm resources, the Stony Brook Algorithm Repository and the Dictionary of Algorithms and Data Structures, provide suitable reinforcement of anything that is not clear in the classifications. They are also very nice to have for their formal treatment of algorithms that I can refer to in this sometimes informal area.

5. Appendix

Problem #	Balloon Colour	Type of Problem
2001-1	Red	Graph
2001-2	Yellow	Complete Search
2001-3	Green	Greedy
2001-4	Blue	Graph Theory + Tree
2001-5	Pink	Mathematical - Bases + Dynamic Programming
2001-6	White	Encryption + Complete Search
2002-1	Pink	Mathematical - Bases + Backtracking
2002-2	Yellow	Mathematical - Geometry
2002-3	Green	Game / Puzzle + Data Structure
2002-4	Purple	Mathematical - Geometry
2002-5	Blue	Mathematical + Complete Search
2002-6	Red	Mathematical - Bases + Dynamic Programming
2003-1	Red	Mathematical / Physics
2003-2	Yellow	Graph Theory
2003-3	Blue	Mathematical - Bases + Parsing + Dynamic Programming
2003-4	Pink	Mathematical - Advanced
2003-5	Green	Encryption
2003-6	White	Mathematical + Complete Search
2004-1	Red	Graph Theory
2004-2	Blue	Complete Search
2004-3	Purple	Mathematical - Geometrical
2004-4	Yellow	Mathematical - Physics
2004-5	White	Mathematical

2004-6	Green	Mathematical
2005-1	Yellow	Mathematical – Primes
2005-2	Red	Complete Search + Straightforward
2005-3	Green	Complete Search
2005-4	Blue	Mathematical - Physics
2005-5	Orange	Graph Theory
2005-6	Purple	Parsing + Mathematical
2006-1	Yellow	Mathematical - Perfect Squares
2006-2	Green	Complete Search (Game - Chess)
2006-3	Red	Puzzle / Sudoku
2006-4	Blue	Parsing + Encryption
2006-5	Orange	Graph Theory + Tree
2006-6	Purple	Mathematical + Complete Search

5. References

Black Paul E, ed., U.S. National Institute of Standards and Technology, *Dictionary of Algorithms and Data Structures*, Published: 2007-02-12, Accessed: 2007-06-24, <http://www.nist.gov/dads>

Burch Hal, Published: 1999, Accessed: 2007-06-24,
<http://ace.delos.com/usacotext2?a=L3LjaPPIHL3&S=probs>
Login Required. Local copy at: <http://www.cs.ru.ac.za/research/g04h2708/USACO.html>

Cheng Howard, *Problem Classification on Spanish Archive*, Published: 2006-12-17, Accessed: 2007-06-24, <http://www.cs.uleth.ca/~cheng/contest/hints.html>

Skiena Steven, *Stony Brook Algorithm Repository*, Published: 2001-03-07, Accessed: 2007-06-24, <http://www.cs.sunysb.edu/~algorith/>

Top Coder, Published: unknown, Accessed: 2007-06-24,
<http://www.topcoder.com/tc?module=ProblemArchive>