Puzzles for Programmers and Pros

What you will learn from this book
- How to expand your puzzle-solving abilities and tackle new challenges
- Proven steps that will help you quickly progress from basic puzzles to a more advanced level
- How to prepare for various types of puzzles presented during a programming interview
- Techniques for determining the best solution to a puzzle
- Methods for solving puzzles using decryption and combinatorics

Who this book is for
This book is for programmers who need to brush up on their puzzle-solving skills as they prepare for the programming job interview. It is also for anyone who love puzzles and challenges.

Puzzles demands a mindset that with a vulnerable openness follow a rigorous drive to find a solution. If you’re preparing for a programming review or just like a challenge, this likes you on a tour of problem-solving exercises so you can dramatically improve your skills. You’ll learn how to conquer simulation puzzles like Sudoku and how heuristic techniques to far more x problems.

Shasha provides you with the tools to several classes of puzzles by hand. These include scheduling, logic, geometric, and probabilistic techniques. You’ll also find a mystery involving bank accounts, and geography that a solve the chance to win a prize. Approaches and techniques in this book p you solve the kind of application and combinatorial problems involved.

www.wrox.com

Recommended Computer Book Categories
- Programming
- General

$24.99 USA/$29.99 CAN

ISBN 978-0-470-43278-1

Dennis E. Shasha
Part I: Mind Games

**Preferential Romance**

A certain marriage counseling service called Marriage Success, or MS for short, advises couples on how to get along better. MS’s idea is simple: Each spouse writes down his/her preferences about various criteria of common interest. “Our criteria go beyond those elements of physical appearance and passion that guide early romance and tend to blind judgment. We want to understand your values as you live day to day. The happy couple is the one whose preferences are compatible or can be made compatible.”

Here are some of the positive qualities each person might wish a spouse to have: biker (B), cultured (C), enthusiastic (E), foodie (F), hiker (H), juggler (J), kayaker (K), movies (M), organized (O), puzzles (P), rich (R), theatre (T), and windsurfer (W).

Suppose X and Y are qualities of people. X→Y implies a preference for X to Y. Preferences are transitive, so X→Y and Y→Z implies X→Z. Two people are fully compatible if their preferences are consistent. The test for consistency is that there is some list of qualities that reflects both of their preferences. If two people aren’t fully compatible, then perhaps at least they are passably compatible, which means they would be fully compatible if each spouse dropped at most one preference.

**Warm-Up**

Suppose Bob’s preferences are

\[ K \rightarrow M \rightarrow P, \text{ and } R \rightarrow T \]

and Alice’s preferences are

\[ O \rightarrow P \rightarrow R \text{ and } W \rightarrow T. \]

Then would they be fully compatible?

**Solution to Warm-Up**

Yes. Ignoring the qualities not mentioned (because they can be put anywhere), here is a consistent ordering for these preferences: KOPRMWT. This is consistent in the sense that for every preference \( X \rightarrow Y \), \( X \) comes before \( Y \) in the list. On the other hand, if Alice added the preference \( R \rightarrow M \), then the couple would not be compatible because \( P \rightarrow R \) and \( R \rightarrow M \) implies \( P \rightarrow M \), but Bob has \( M \rightarrow P \). If Bob drops his \( M \rightarrow P \) preference, then we would be left with:

- Bob: \( K \rightarrow P, \text{ and } R \rightarrow T \)
- Alice: \( O \rightarrow P \rightarrow R \rightarrow M \text{ and } W \rightarrow T. \)

This could yield the following consistent ordering, among others: KOPRMWT. So Bob and Alice would then be passably compatible.
Part I: Mind Games

Figure 1-13: Alice’s preferences in life. Note that she has no preference about whether Bob is rich or a biker.

1. Are Bob and Alice compatible, passably compatible, or neither? After dropping the smallest number of edges in zero, one, or both spouse’s preferences, try to find a consistent ordering.

2. [For the entrepreneurial at heart] Can you describe an algorithm for the counseling company to use to help marriages in distress? That is, try to find a method so spouses have to drop as few preferences as possible.
Revisiting a Traveling Salesman

The Traveling Salesman Problem, affectionately known as TSP, lies at the heart of many problems, including optimizing the deliveries of trucks, scheduling tour stops, and laying out wires. TSP yields very nicely to heuristics, but it also admits some beautiful theory.

A certain salesman, we’ll call him Bob, starts at a certain city C and wants to visit a certain set of other cities by car at least once and then return to C. (Factoid: What is the preferred profile of salespeople from pharmaceutical companies who call on doctors? Answer: Former cheerleaders.) Travel times and costs may vary, but we will assume that the costs are positive, symmetric, and that the triangle inequality holds. Symmetry means that going from X to Y costs the same as going from Y to X. The triangle inequality means that going from X to Y to Z cannot be cheaper than going directly from X to Z (it could be the same price, but not cheaper).

The question is whether Bob can visit all his cities for a certain price or less. As you can see, if someone proposes a solution, you can verify that it meets the price budget easily, but finding a solution below a certain price can be hard — it may require exploring all possible paths.

Many people hear this problem and think: “Why can’t a greedy approach work? That is, from each city, I’ll just go to the next nearest one.”

How bad can this get? That is, can you find a set of points where the greedy approach fails to find the minimum length route for the traveling salesman (assuming for the moment that cost is proportional to length)? Figure 2-13 lays out the visual for the problem.

Figure 2-13: The salesman starts at C and ends at C. Each time he goes to the closest unvisited city. How well will Bob do?

Using the greedy heuristic, if you start at point C, you will go down, right, up, right, down, right, up, right, down and then all the way back to C, as illustrated in Figure 2-14. In fact however, the best route would be to go directly right from C until the upper right hand point, then down, left and back to C, as in Figure 2-15.

Figure 2-14: What happens when Bob visits the nearest previously unvisited city first?
When greed doesn’t work, heuristics might help, but first there is the question of how bad
greed could be. Concretely, can you find a situation, again assuming that cost is propor­
tional to length, when the greedy “go to the next nearest city” strategy could be more than
twice as costly as optimal?

I’m not going to answer this right away, because I want to present you with a strategy that
is surely no more than twice as costly as optimal, is also greedy, and can easily be
improved by heuristics. It also applies to every set of cities.

As you may already know, a spanning tree is a graph without cycles that connects all ver­
tices (cities in our case). The cost of a spanning tree is the sum of the costs of its edges
(remember that we assume that the cost to go from A to B is the same as the cost to go from
B to A — symmetry). A minimum cost spanning tree is a spanning tree having the property
that no other spanning tree has lower cost. To construct a minimum spanning tree starting
at city C, we use the following algorithm:

1. Call an edge “useful” if it connects a vertex in a tree to a vertex
   outside the tree.
2. Initialize the tree T to contain the city C and no edges
   until there are no more cities to include.
   a. Add edge E to the edges of T if E is useful
   and for every other useful edge E’,
      the cost of E’ is greater than or equal to the cost of E
   b. Add the node of E not already in T to the nodes of T
   end until

Given the nodes in Figure 2-16, Figure 2-17 shows a minimum spanning tree (again where
cost is proportional to distance).

Now, here comes the key insight. The lowest cost route that a traveling salesman uses to
visit all the cities is a spanning tree (though not necessarily a minimum cost one) if one
ignores the final edge that brings the salesman back to his own city. The reason is that the
route will touch all cities exactly once, because the triangle inequality makes it unprofitable
to revisit a city. Because the route is a spanning tree (plus an extra edge), it must be at least
as costly as the minimum spanning tree.

The consequence of this observation is that the cost of the minimum spanning tree pro­
vides a lower bound on the possible cost of the optimum traveling salesman route, even if
we never find that optimal route.

We are not done, however. Every route is a spanning tree, but most minimum spanning trees
are not routes. Fortunately, we can create a route from any minimum spanning tree that will
never cost more than twice as much as the optimal route, as illustrated in Figure 2-18.

Suppose we start at the center point and go right, then up, then down, then up, then
left back to the center. We continue similarly on the left. The point is that every edge will be
traversed twice. By symmetry, this means that the route costs twice as much as the minimum
spanning tree. Because the minimum spanning tree is less expensive than the (unknown)
optimal route, this route is less expensive than twice the cost of the optimal route.

Many improvements are now possible. Some are illustrated in Figure 2-19 where the
returns to the center are done via diagonal lines. Beyond such heuristics, there are some
guarantees. A very clever algorithm by N. Christofides showed that one could combine
minimal matchings with spanning trees to obtain a route that is no more than 3/2 as
expensive as the optimal one.
As clever as these approaches are, however, they don't necessarily extend to other problems, because they depend strongly on symmetry and the triangle inequality. Or do they?

![Figure 2-19: Some simple heuristic improvements to the minimum spanning tree-based route that will either leave the cost the same or reduce it (assuming the triangle inequality holds).]

1. Can you find a case where, if symmetry and the triangle inequality do not hold, even the factor of two guarantee may not either? If not, can you prove that the spanning tree will always guarantee a cost that is no more than twice the optimal?