CS472 Module 7 Part F - Problem Reduction

Athens State University

March 7, 2016

Outline

Contents

1 Problem Reduction

Problem Reduction

- *Problem Reduction*: Find solution to a problem by reducing it to another problem that you know how to solve
- Important method in the theory of computer science as it can be used to classify problems according to their complexity

2 Examples of problem reduction

Examples of problem reduction: counting paths in a graph

- Problem: How many paths exist between two vertices in a graph?
- We can use the principle of mathematical induction to show that the number of different paths of length $k > 0$ from vertex i to vertex j equals the element at position (i, j) of the matrix A^k, where A is the adjacency matrix of the graph
- So, we have reduced the problem of counting paths in a graph to the problem of computing matrix exponents
  - And we can use the same algorithms we've been using for scalar multiplication

Examples of problem reduction: Optimization

- An optimization problem asks that you find a maximum (minimum) of some function $f(x)$
- So, if you have an algorithm for computing the maximum, what must you do make that algorithm work for computing the minimum
- One simply needs to note that $\min f(x) = -\max[-f(x)]$

3 Linear Programming

Linear Programming: What is it?

- Many problems of optimal decision making can be reduced to an instance of the *linear programming* problem:
  - a problem of optimizing a linear function of several variables subject to constraints in the form of linear equations and linear inequalities
- The algorithms for solving these problems are such that they can not deal with linear programming problems that deal only with integers
  - Such problems are defined as integer linear programming problems
  - These problems are particularly painful to solve, and it is thought that no known polynomial time algorithm for solving such problems

Linear programming: example

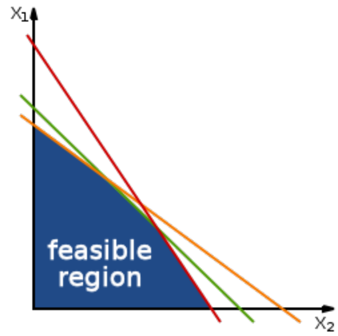
Honking Big State University needs to invest $100M worth of their endowment. This sum needs to be split between three types of investments: stocks (10% return), bonds (7% return), and cash (3% return). To minimize the risk, the rules of the endowment require the investment in stocks to be no more that 1/3 of money invested in bonds. In addition, 25% of the total amount invested in stocks and bonds must be invested in cash. How should the $100M be invested to maximize the return?

Linear Programming: example

- Let s, b, and c be the amounts (in millions of dollars) invested in stocks, bonds, and cash.
- So, we can rephrase our investment problem as
  - Maximize $0.10s + 0.07b + 0.03c$
  - Subject to

$$\begin{aligned}
s + b + c &= 100 \\
s &\leq (1/3)b \\
c &\geq 0.25(s + b) \\
s &\geq 0 \\
b &\geq 0 \\
c &\geq 0
\end{aligned}$$

Linear Programming: Simplex Algorithm



- The feasible region is a convex polyhedron
- The simplex algorithm searches for a minimum value of the objective function that is an extreme point of the feasible region

Linear Programming: A return to knapsacks

- Recall that the knapsack problem asks, for n items with a weight and value for each item, what subset of the items will fit into a sack that can hold W pounds
- This problem can be reduced to an optimization problem
  - Maximize
$$\sum_{j=1}^{n} v_j x_j$$
  - Subject to
$$\sum_{j=1}^{n} w_j x_j \leq W$$

Linear Programming: A return to knapsacks

- But suppose we "simplify" this problem to the *discrete knapsack* problem where we only take a whole item or no item at all
  - Maximize
$$\sum_{j=1}^{n} v_j x_j$$
  - Subject to
$$\sum_{j=1}^{n} w_j x_j \leq W$$

    where $x_j \in 0, 1$
- Turns out this is a much more difficult computational problem that our original continuous problem
  - Will reconsider this issue when talk about computational complexity

4 Key Points

Key Points

- What is problem reduction?
- How do we use this technique?
- Optimization problems and linear programming