The Theory of Computation

Bernard M.E. Moret

Addison-Wesley-Longman, 1998

Solution Manual
Chapter 2, Problem 2.8

1. We need a collection of items; let that be a set S. Each item has a size which is a natural number; thus we have an assignment of sizes to the elements of S, i.e., a function from S to $\mathbb{N}$, call it $s\colon S \longrightarrow \mathbb{N}$. Finally, all bins have the same size, but that size must be specified, say $B \in natural$. These items define an instance of the problem.
   Placing an item x in the ith bin is equivalent to assigning to item x an index i; let that be the assignment function $b\colon S \longrightarrow \{1, 2, \ldots, k\}$ when a total of k bins are used. Then we want all items assigned to bin i to fit within the bin, that is, we want to have $\sum_{b(x)=i} s(x) \leq B$. These are the conditions that a solution to the problem must fulfill.
   The objective is then to find the smallest number of bins in which it is possible to pack all items, that is, to identify the smallest value of k for which an assignment function b exists that fulfills the conditions.
   Thus an instance of our problem is given by a finite set S of items, a size function $s\colon S \longrightarrow \mathbb{N}$ that assigns a size to each item, and a bin size $B \in \mathbb{N}$. The objective is to minimize k subject to finding a function $b\colon S \longrightarrow \{1, 2, \ldots, k\}$ obeying $\sum_{b(x)=i} s(x) \leq B$ for each $1 \leq i \leq k$.
2. We need a Boolean formula; this requires a certain number of variables, call them x_1, dots, x_n. Specifying the formula can be done with a suitable alphabet (with symbols for the Boolean operators and for parentheses as well as for the variables) or by encoding everything into a simpler alphabet. A solution is a truth assignment, that is, a function from the set of variables to the set {true,false}, that causes the formula to evaluate to 'true."
   Further details could be given about the evaluation of the formula, but such tend to be messy and uninformative—and we all know how to evaluate a Boolean formula. Most *Satisfiability* problems are restricted to instances that use only certain types of Boolean connectives (e.g., just $\vee$, $\wedge$ and)/and often certain types of formula, in which cases

more formal descriptions of legal instances and of the constraint on the solution are quite reasonable.

3. The input is a sequence of numbers, presumably natural integers; we need a notion of the length of that sequence. So we can formalize the input as an ordered sequence of n natural integers, say $(x_1, x_2, \ldots, x_n)$ with $x_i \in \mathbb{N}$. A subsequence can be viewed as a selection from the set $\{1, 2, \ldots, n\}$ of indices, i.e., a function $s : \{1, 2, \ldots, n\} \longrightarrow \{0, 1\}$, where $s(i) = 1$ denotes the fact that the x_i is part of the chosen subsequence. The condition that must be met by a solution may then be written as

$$i < j \Longrightarrow (s(i) = 0 \vee s(j) = 0 \vee x_i < x_j)$$

The objective of the problem is to identify a selection function s that selects as many items as possible, that is, that maximizes the sum $\sum_{i=1}^{n} s(i)$.
Formally, then, our problem is given by an ordered sequence of n natural integers, $(x_1, x_2, \ldots, x_n)$ with $x_i \in \mathbb{N}$. The objective is to find a function $s : \{1, 2, \ldots, n\} \longrightarrow \{0, 1\}$ that maximizes the sum $\sum_{i=1}^{n} s(i)$ subject to

$$i < j \Longrightarrow (s(i) = 0 \vee s(j) = 0 \vee x_i < x_j)$$

for all $1 \leq i, j \leq n$.