A Fully Polynomial Time Approximation Scheme for Subset Sum

CLRS – Chapter 35

Last Revised – 16/11/07

Approximation Algorithms

For any given optimization (minimization) problem and approximation algorithm $\mathcal{A}$ to solve it:

- Let Π = Set of all *instances* of the problem.
- For all instances $I \in \Pi$ define $size(I)$.
- For all instances $I \in \Pi$ define

$$\begin{aligned} OPT(I) &= \text{cost of optimal solution for } I \\ A(I) &= \text{cost of solution produced by } A \text{ on } I. \end{aligned}$$

Let $\rho(n)$ be a function such that

$$\forall I \in \Pi, \text{with } size(I) = n, \quad \frac{A(I)}{OPT(I)} \leq \rho(n).$$

Then A is a
factor $\rho(n)$ approximation algorithm
($\rho(n)$-approximation algorithm) .

(there is a similar definition for *maximization* problems)

The Subset-Sum Problem

<u>Definition:</u> An instance of the *subset-sum decision problem* is (S, t) where:
$S = \{x_1, x_2, \ldots, x_n\}$ a set of positive integers;
t a positive integer.

The problem is whether some subset of S adds up exactly to t. This problem is NP-complete

The *subset-sum optimization problem* is to find a subset of S whose sum is as large as possible but no greater than t.

We will define a class of algorithms A_ϵ, such that, $\forall \epsilon > 0$,

- **A_ϵ is an ϵ-approximation algorithm for subset-sum.**
- **A_ϵ runs in time polynomial in n, $\log t$ *and* $\frac{1}{\epsilon}$.**

Such a class of algorithms is known as a

A fully polynomial-time approximation scheme.

An Exponential Time Algorithm

If $S = \{x_1, x_2, \ldots, x_n\}$ is a set or list and x a real number then define

$$S + x = \{x_1, x_2, \ldots, x_n\} = \{x_1 + x, x_2 + x, \ldots, x_n + x\}.$$

If $L = \{x_1, x_2, \ldots, x_n\}$ and $L' = \{u_1, u_2, \ldots, u_m\}$ are both sorted lists then define **Merge-Lists**(L, L') to be the procedure that returns the sorted union of the two lists. This procedure runs in time $O(|L'| + |L|)$.

Exact-Subset Sums

$n \leftarrow |S|$
$L_0 \leftarrow < 0 >$
for $i = 1$ **to** n
  $L_i =$ **Merge-Lists**$(L_{i-1}, L_{i-1} + x_i)$
  remove from L_i **all elements bigger than** t.
return largest element in L_n.

Let P_i be the set of all values that can be obtained by selecting some subset of $\{x_1, x_2, \ldots, x_i\}$ and summing its members. Then L_i is a sorted list containing all elements in P_i of size no greater than t.
The algorithm therefore returns the correct answer.

Since L_i can have as many as 2^i items this algorithm can take $\Theta(2^n)$ time!

Trimming

Let $L\{x_1, x_2, \ldots, x_m\}$ be a list. To **trim** the list by parameter δ means to remove as many elements from L as possible in such a way that the list L' of remaining elements has the following property:

For every $y \in L$ there exists a $z \in L'$ such that
$$(1-\delta)y \le z \le y.$$

Example:
$L =< 10, 11, 12, 15, 20, 21, 22, 23, 24, 29 >$ **and** $\delta = 0.1$.
A trimmed list would be $L' =< 10, 12, 15, 20, 23, 29 >$.

$\underline{Trim(L, \delta)}$
$L' =< x_1 >$
$last = x_1$
for $i = 2$ **to** m
  if $last < (1-\delta)x_i$
    then append x_i **onto end of** L'**.**
      $last = x_i$
return L'

This algorithm returns a trimmed list in $O(m)$ time.
(It assumes that input list is sorted in non-decreasing order.)

The Actual Approximation Algorithm

Approx-Subset-Sum(S, t, ϵ)
$n \leftarrow |S|$.
$L_0 =< 0 >$.
for $i = 1$ **to** n
  $L_i \leftarrow$ **Merge-Lists**$(L_i, L_{i-1} + x_i)$
  $L_i \leftarrow Trim(L_i, \epsilon/n)$
  remove from L_i **all elements bigger than** t
return largest element in L_n

Note that when list L_i is trimmed we introduce a relative error of at most ϵ/n between the representative values remaining and the elements of the list. By induction can show that, $\forall y \in P_i$ there exists some $z \in L_i$ such that

$$\left(1 - \frac{\epsilon}{n}\right)^i y \le z \le y.$$

Let $\overline{z}$ be the largest element in L_n. If y^* is a solution to the exact subset-sum problem then there exists a $z^* \in L_n$ such that

$$\left(1 - \frac{\epsilon}{n}\right)^n y^* \le z^* \le \overline{z} \le y^*.$$

But $\forall n > 1$,

$$1 - \epsilon \le \left(1 - \frac{\epsilon}{n}\right)^n \Rightarrow (1 - \epsilon) y^* \le \overline{z},$$

and A_ϵ is an ϵ-approximation algorithm.

Running Time

The running time of the ith stage of the algorithm is $O(|L_i|)$.

After trimming, successive elements $z', z \in L_i$ have the property

$$z' < z\left(1 - \frac{\epsilon}{n}\right).$$

Therefore the total number of elements in L_i is at most

$$\begin{aligned} \log_{\frac{1}{1-\frac{\epsilon}{n}}} t &= \frac{\ln t}{-\ln\left(1 - \frac{\epsilon}{n}\right)} \\ &\leq \Theta\left(\frac{n \ln t}{\epsilon}\right) \end{aligned}$$

The running time of A_ϵ is proportional to

$$\frac{n^2 \ln t}{\epsilon}$$

and the A_ϵ form a

Fully Polynomial Time Approximation Scheme

for subset-sum.