Homework 4

Thomas Kim tsk389 51835

David Munoz dam2989 51840

CS331 Algorithms and Complexity

Problem Q1. Suppose you're consulting for a company that manufactures PC equipment and ships it to distributors all over the country. For each of the next n weeks, they have a projected *supplys*$_i$ of equipment (measured in pounds), which has to be shipped by an air freight carrier.

Each week's supply can be carried by one of two air freight companies, A or B.

- Company A charges a fixed rate r per pound (so it costs $r \cdot s_i$ to ship a week's supply s_i)
- Company B makes contracts for a fixed amount c per week, independent of the weight. However, contracts with company B must be made in blocks of four consecutive weeks at a time.

Give a polynomial-time algorithm that takes a sequence of supply values $s_1, s_2, \ldots, s_n$ and returns a *schedule* of minimum cost. $Ship(i)$

if $i \leq 0$ **then**
  return 0
end if
return $min(rs_i + Ship(i-1), 4c + Ship(i-4))$

Correctness:

Proof. □

Runtime:

Proof. □

Problem Q2(a). Assume that you have to make change for N, and that you have an infinite supply of each $C = c_1, c_2, \ldots, c_n$ valued coins where $1 \leq c_i \leq N-1$. Compute the minimum number of coins required to make the change. Provide an algorithm which solves the problem using dynamic programming, prove correctness and runtime. $Change(C, N)$

$C = c_1, c_2, \ldots, c_k$
if $C = \{\}$ **then**
  return $INVALID$
end if
if $N = 0$ **then**
  return 0
end if
return $min(Change(C - \{c_k\}, N), 1 + Change(C, N - c_n))$

Proof of correctness

Proof. □

Proof of runtime

Proof. □

Problem Q2(b). Solve the same problem, but this time assume you have a limited supply p_i of each coin c_i Provide a dynamic programming algorithm, do not prove correctness or runtime. $Setup(C)$

$C' = \emptyset$
for $\forall c_i \in C$ **do**
  $c_{i,1}, c_{i,2}, \ldots, c_{i,p_i} = c_i$

    $C' = C' \cup \{c_{i,1}, c_{i,2}, \ldots, c_{i,p_i}\}$
end for

$Change(C, N)$

$C = c_1, c_2, \ldots, c_k$
if $C = \{\}$ **then**
    return $INVALID$
end if
if $N = 0$ **then**
    return 0
end if
return $min(Change(C - \{c_k\}, N), 1 + Change(C - \{c_k\}, N - c_n))$

The solution is $Change(Setup(C), N)$

Problem Q2(c). Assume $C = 1, 2, 4, \ldots, 2^m$ and $N < 2^{m+1}$. For this special case, give an $O(m)$ time algorithm that solves the problem in (a).

$count = 0$
Express N as a binary number $n_1 n_2 n_3 \ldots n_k$
for $1 \leq i \leq 2^m$ **do**
    if $n_i = 1$ **then**
        $count := count + 1$
    end if
end for

Correctness

Proof. Thm: For every element $x \in Q_p$, where Q_p is a field of $p - adic$ integers, there exists a unique representation $x = \sum_{i=0}^{k} a_i p^i, a_k \neq 0$
By expressing N as a 2-adic integer, it is guaranteed that there exists some representation of N in the form shown above.
Note that k in the above sum is bounded by $k < m + 1$, since $2^{m+1} > N$
$\forall 0 \leq i \leq m, 2^i \in C$ Therefore, there exists a unique representation of N as a sum of the elements in C. □

Runtime

Proof. Expressing N as a binary number is $O(m)$ time, since N expressed in binary has at most m digits.
Counting the 1's in the binary representation of N takes $O(m)$ time, for the above reason.
The complexity is therefore $O(2m) = O(m)$ □

Problem Q3. An opportunity cycle is one where the product of the ratios along the cycle is greater than 1. Give a polynomial-time algorithm to find an opportunity cycle in a graph, if one exists.

Let $G(V, E)$ be the graph of possible trades
Let $w(e_i)$ be the weight of edge $e_i \in E$
Let $G'(V', E') | V' = V$
for $\forall e_i(u, v) \in E$ **do**

```
5:     e'_i = (u, v) | w(e'_i) = -log(w(e_i))
6:     E' = E' ∪ {e'_i}
7: end for
8: for ∀v' ∈ V' do
9:     Run Bellman-Ford on G' starting on v' to detect negative cycles, return TRUE if one is detected
10: end for
```

Correctness:

Proof. Suppose there is some cycle $e_1, e_2, \ldots, e_k$ such that $\Pi_{i=1}^{k} e_i > 1$ $log\left(\Pi_{i=1}^{k} e_i\right) > log(1)$
$\implies \sum_{i=1}^{k} log(e_i) > 0$
$\implies \sum_{i=1}^{k} -log(e_i) < 0$
Therefore, by line , a negative cycle corresponds to an opportunity cycle □

Runtime:

Proof. Constructing G' takes $O(|V| + |E|)$ time.
Running Bellman-Ford takes $O(|V||E|)$ time.
Bellman-Ford is run $|V|$ times.
Therefore, the total complexity is $O(|V|^2|E|)$ which is polynomial in the number of vertices/edges. □