#### CS F364 Design & Analysis of Algorithms

#### **ALGORITHM DESIGN TECHNIQUES**

**0/1 Knapsack Problem: Dynamic Programming Algorithm:** 

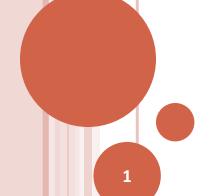
**Time Complexity** 

**Pseudo-Polynomial Time Algorithms** 

**Space Complexity** 

Limitations

**Problem Variants** 



```
Known (Atomic) Solutions: P(0, w)=0 for all w and P(k, 0)=0 for all k
Recursive structure: P(k,w) =
      Profit(k,w)
// assume output array Pf[0..N][0..Wmax]
// assume array wt[1..N] of weights and p[1..N] of prices
   for (k=0; k<=N; k++) Pf[k,0] = 0;
    for (w=0; w<=Wmax; w++) Pf[0,w] = 0;
    for (k = 1; k \le N; k++)
      for (w=1; w<=Wmax; w++)
        Pf[k,w] = (wt[k] > w) ? Pf[k-1,w] :
                 max(Pf[k-1,w], Pf[k-1,w-wt[k]]+p[k]);
```

2/4/2016

```
CSIS, BITS, Pila
```

- Time Complexity: O (N\*Wmax)
  - Is this polynomial time? Why or Why not?
  - What if Wmax is O(2<sup>N</sup>)?
- Pseudo-polynomial time algorithms
  - Complexity is defined in terms of max. input size
    - o e.g. N\*Wmax is polynomial in the size of the set of items

```
EXAMPLE - 0/1 KNAPSACK - DP SOLUTION
Profit(k,w)
// assume output array Pf[0..N][0..Wmax]
  assume array wt[1..N] of weights and p[1..N] of prices
   for (k=0; k<=N; k++) Pf[k,0] = 0;
    for (w=0; w<=Wmax; w++) Pf[0,w] = 0;
    for (k = 1; k<=N; k++)
      for (w=1; w<=Wmax; w++)
          Pf[k,w] = (wt[k] > w) ? Pf[k-1,w] :
                   max(Pf[k-1,w], Pf[k-1,w-wt[k]]+p[k]);
```

- Space Complexity: O(N\*Wmax)
  - Can this be reduced? If so, how? If not why not?
- P[k,\_] is dependent only on P[k-1]
  - At any time only 2 rows (index k and k-1) are needed.
- <u>Exercise</u>: Rewrite the procedure after pruning unwanted rows in the profit matrix.

```
EXAMPLE - 0/1 KNAPSACK - DP SOLUTION
```

- P[k,\_] is dependent only on P[k-1]
  - At any time only 2 rows (index k and k-1) are needed.
- What about columns? Can they be pruned?
  - Number of columns needed at any time: 1+max<sub>j</sub> w<sub>j</sub>
- Exercise: Rewrite the procedure after pruning unwanted rows and columns in the profit matrix

- Validity of assumptions:
  - What if weights are not integers?
  - Rational numbers? Real numbers?

- Validity of assumptions:
  - What if weights are not integers?
  - Rational numbers? Real numbers?
- Consider weights to be rationals
  - o i.e. normalized fractions of the form  $(p_i / q_i)$
  - Multiply all weights by lcm<sub>i</sub> (q<sub>i</sub>)
    - o All (scaled) weights are integers:
      - Scaling weights does not affect profits.
  - Impact on complexity:
    - o Time :  $N * (Icm_i(q_i) * Wmax)$
    - o Space:  $2 * (1+max_j(w_j)) * lcm_j(q_j)$

- Integer weights can also be normalized (i.e. scaled)
  - If Integer weights are divided by  $gcd_j(w_j)$  the time and space complexities can be reduced by the same factor.
    - o When is this useful?
  - Are there ways reducing the complexity factor dependent on weights?
    - o Relook at the recurrence relation.