Announcements

• Reading:
  – 6.1-6.2, Weighted Interval Scheduling
  – 6.3 Segmented Least Squares
  – 6.4 Knapsack and Subset Sum
Dynamic Programming

- Weighted Interval Scheduling
- Given a collection of intervals $I_1, \ldots, I_n$ with weights $w_1, \ldots, w_n$, choose a maximum weight set of non-overlapping intervals.

Intervals sorted by end time

$$P[I_1] = 0$$

$$P[I_2] = 0$$

$$P[I_3] = 1$$

$$P[I_4] = 0$$

$$P[I_5] = 1$$

$$P[I_6] = 2$$
Optimality Condition

- $\text{Opt}[j]$ is the maximum weight independent set of intervals $I_1, I_2, \ldots, I_j$
- $\text{Opt}[j] = \max(\text{Opt}[j-1], w_j + \text{Opt}[p[j]])$
  - Where $p[j]$ is the index of the last interval which finishes before $I_j$ starts
Iterative Algorithm

```java
int[] M = new int[n+1];
char[] R = new char[n+1];

M[0] = 0;
for (int j = 1; j < n+1; j++) {
    v1 = M[j-1];
    v2 = W[j] + M[P[j]];
    if (v1 > v3) {
        M[j] = v1;
        R[j] = 'A';
    }
    else {
        M[j] = v2;
        R[j] = 'B';
    }
}
```
Computing the solution

Opt\[ j \] = max (Opt\[ j - 1 \], w\_j + Opt\[ p[ j ] \])

Record which case is used in Opt computation

```
2 4 9 9 9 16 16
B B B A A B A
```
Optimal linear interpolation

\[
\text{Error} = \sum (y_i - ax_i - b)^2
\]
What is the optimal linear interpolation with three line segments?
What is the optimal linear interpolation with two line segments
What is the optimal linear interpolation with n line segments
Notation

- Points $p_1, p_2, \ldots, p_n$ ordered by x-coordinate ($p_i = (x_i, y_i)$)
- $E_{i,j}$ is the least squares error for the optimal line interpolating $p_i, \ldots, p_j$
Optimal interpolation with two segments

- Give an equation for the optimal interpolation of $p_1, \ldots, p_n$ with two line segments

- $E_{i,j}$ is the least squares error for the optimal line interpolating $p_i, \ldots, p_j$
Optimal interpolation with $k$ segments

- Optimal segmentation with three segments
  - $\text{Min}_{i,j}\{E_{1,i} + E_{i,j} + E_{j,n}\}$
  - $O(n^2)$ combinations considered

- Generalization to $k$ segments leads to considering $O(n^{k-1})$ combinations
Opt_{k}[ j ] : Minimum error approximating \( p_1 \ldots p_j \) with \( k \) segments

How do you express Opt_{k}[ j ] in terms of Opt_{k-1}[1], \ldots, Opt_{k-1}[ j ]?
Optimal sub-solution property

Optimal solution with k segments extends an optimal solution of k-1 segments on a smaller problem
Optimal multi-segment interpolation

Compute $\text{Opt}[k, j]$ for $0 < k < j < n$

for $j = 1$ to $n$
    $\text{Opt}[1, j] = E_{1,j};$

for $k = 2$ to $n-1$
    for $j = 2$ to $n$
        $t = E_{1,j}$
        for $i = 1$ to $j-1$
            $t = \min(t, \text{Opt}[k-1, i] + E_{i,j})$
        $\text{Opt}[k, j] = t$
Determining the solution

- When \( \text{Opt}[k,j] \) is computed, record the value of \( i \) that minimized the sum
- Store this value in an auxiliary array
- Use to reconstruct solution
Variable number of segments

- Segments not specified in advance
- Penalty function associated with segments
- Cost = Interpolation error + C x #Segments
Penalty cost measure

• $\text{Opt}[j] = \min(E_{1,j}, \min_i(\text{Opt}[i] + E_{i,j} + P))$