Backtracking and Branch and Bound

Module 11
CSE5311 Fall 2008

Backtracking

- Using Backtracking
  - Large instances of difficult combinatorial problems can be solved
  - Worst case complexity of Backtracking can be exponential
- Typically, a path is taken to check if a solution can be reached
  - If not, the path is abandoned and another path taken
  - The process is repeated until the solution is arrived at
N-Queens problem

- Place n-queens on an n x n chess board so that no two queens attack each other.

  - A queen can attack another if the latter is on the same row, column or diagonal.
Hamiltonian Circuit Problem

Start at a vertex and visit all the other vertices in the graph exactly once and return to the start vertex.
Hamiltonian Circuit Problem

Subset Sum Problem

- Given a Set S = \{s_1, s_2, \ldots, s_n\} and a positive integer \(d\) find a subset of the given set S such that the sum of the positive integers in the subset is equal to \(d\).
- Let S = \{3, 7, 9, 13, 26, 41\}; \(d = 51\).
- Note – the list should be sorted.
Let \( S = \{3, 7, 9, 13, 26, 41\} \); 
\( d = 51 \)
Branch and Bound

- With backtracking
  - The search space is can be very large
  - It is an exhaustive search
  - Worst case complexity is exponential

- Branch and bound technique
  - Limits the search space
  - Through an estimate of the
    - Upper bound or
    - Lower bound

Scheduling problem

- The problem of assigning n people to n jobs such that the total cost is as small as possible

<table>
<thead>
<tr>
<th>Job Person</th>
<th>J1</th>
<th>J2</th>
<th>J3</th>
<th>J4</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>9</td>
<td>2</td>
<td>7</td>
<td>8</td>
</tr>
<tr>
<td>B</td>
<td>6</td>
<td>4</td>
<td>3</td>
<td>7</td>
</tr>
<tr>
<td>C</td>
<td>5</td>
<td>8</td>
<td>1</td>
<td>8</td>
</tr>
<tr>
<td>D</td>
<td>7</td>
<td>6</td>
<td>9</td>
<td>4</td>
</tr>
</tbody>
</table>
Branch and Bound

- Find a Lower Bound on the cost of the solution
- The lower bound is only an estimate
  - *This is only an estimate*
  - *The LB may not be a legitimate solution*
- In this case, consider the lowest cost from each row
  - 2 + 3 + 1 + 4 = 10
  - *This is our LB*
Start
LB = 2 + 3 + 1 + 4 = 10

A → J1
LB = 9 + 3 + 1 + 4 = 17

A → J2
LB = 2 + 3 + 1 + 4 = 10

A → J3
LB = 7 + 4 + 5 + 4 = 20

A → J4
LB = 8 + 3 + 1 + 6 = 18

Person
A 9 2 7 8
B 6 4 3 7
C 5 8 1 8
D 7 6 9 4

Kumar CSE531
Knapsack Problem

- We wish the maximize the profit in the knapsack
- Maximization
- Use Upper bound

- \[ UB = v + (W-v)(v_{i+1}/w_{i+1}) \]
- When we start \( v = 0 \)

<table>
<thead>
<tr>
<th>Item</th>
<th>Weight</th>
<th>Value</th>
<th>Value/weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>4</td>
<td>$40</td>
<td>10</td>
</tr>
<tr>
<td>B</td>
<td>7</td>
<td>$42</td>
<td>6</td>
</tr>
<tr>
<td>C</td>
<td>5</td>
<td>$25</td>
<td>5</td>
</tr>
<tr>
<td>D</td>
<td>3</td>
<td>$12</td>
<td>4</td>
</tr>
</tbody>
</table>
Knapsack

\[ \text{UB} = v + (W-v)(v_{i+1}/w_{i+1}) \]

**Start**
- \( W=0, v=0 \)
  - UB = 100

**W/o A**
- W=4 v=4
  - UB = 76
- W=0, v=0
  - UB = 60

**Which is better?**

**With A**
- W=11 Invalid

**W/o B**
- W=4 v=40
  - UB = 70
- W=4 v=40
  - UB = 64

**W/o C**
- W=9 v=65
  - UB = 69

**W/o D**
- W=12 Invalid
- W=9, v=65
  - UB = 65

Traveling Salesperson Problem

- \( \text{LB} = \sum_{\text{over all cities}} \text{(distance to two nearest cities)}/2 \)
- \( \sum \text{over all cities} \)

[Diagram of cities and distances]
### Problems

<table>
<thead>
<tr>
<th>Item</th>
<th>Weight</th>
<th>Value</th>
<th>Value/weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>10</td>
<td>$100</td>
<td></td>
</tr>
<tr>
<td>B</td>
<td>7</td>
<td>$63</td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>8</td>
<td>$56</td>
<td></td>
</tr>
<tr>
<td>D</td>
<td>4</td>
<td>$12</td>
<td></td>
</tr>
</tbody>
</table>

Diagram:

- a connected to b with weight 2
- b connected to c with weight 7 and to d with weight 8
- c connected to d with weight 1
- a connected to c with weight 5
- a connected to d with weight 3

Kumar  
CSE5311