Search Trees

- Each node of the search tree for a problem represents a subset of feasible solutions of the problem.
- The root of the tree represents the set of all feasible solutions of the problem.
- The descendants of each node of the tree represent a partition of the set represented by that node.
A collection of subsets $B_i$ of set $A$ (i=1,2,...t) is a partition if

$$B_1 \cup B_2 \cup B_3 \cdots \cup B_t = A$$

and

$$B_i \cap B_j = \emptyset \quad \text{if} \quad i \neq j$$
Example: Ranking Nodes in a Preference Graph

In many experiments (especially in the social sciences, when numerical measurement of attributes are difficult or impossible), one is required to rank a set of objects by comparing only two at a time.

Example

Six different dog foods are to be ranked according to their appeal to dogs. Each day, 2 of the 6 are served to a dog, who indicates his preference by finishing it first.

Preference Graph

Preference Matrix

A is preferred to B, etc.

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>-</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>B</td>
<td>0</td>
<td>-</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>C</td>
<td>0</td>
<td>1</td>
<td>-</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>D</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>-</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>E</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>-</td>
<td>1</td>
</tr>
<tr>
<td>F</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>-</td>
</tr>
</tbody>
</table>
In the dog food example, the dog exhibited some inconsistency: for example,

```
A --> B
  ^   |
  |   v
D
```

he preferred A over B,

B over D,

and D over A!

How can we establish a "good" ranking?

---

**Methods for Ranking**

- ranking by score: the score of an object is the number of pairs in which it is preferred (i.e., the row-sum of the preference matrix).
- ties may occur
- assumes every possible pair was compared

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
<th>score</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>-</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>B</td>
<td>0</td>
<td>-</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>C</td>
<td>0</td>
<td>1</td>
<td>-</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>D</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>-</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>E</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>-</td>
<td>1</td>
</tr>
<tr>
<td>F</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>-</td>
</tr>
</tbody>
</table>

For example, A > C > B > E > F > D

or C > A > F > E > B > D

e etc.
**Methods for Ranking**

- **ranking by Hamiltonian path:** find a path through every node of the preference graph such that each node is preferred over its successor. For example, $A \rightarrow C \rightarrow B \rightarrow E \rightarrow F \rightarrow D$
or $A \rightarrow C \rightarrow E \rightarrow F \rightarrow B \rightarrow D$

  (several such paths may exist!)

- **ranking with minimum discrepancies**

  A discrepancy is an instance in which $X$ is ranked above $Y$, but $Y$ is preferred to $X$.

  For example, the ranking $A > B > D$ has one discrepancy (i.e., $A > D$)

  — does not assume that every pair was compared!
  — is a difficult problem to solve
Using a Search Tree for
Minimum Discrepancy Ranking

Two different methods for partitioning:

• choose a pair of objects X & Y which have not been ranked.

Form two subsets of rankings:
   --those in which X > Y, i.e., X is ranked above Y
   --those in which Y > X, i.e., Y is ranked above X

Second method of partitioning:

• an object is assigned to a position in the ranking e.g., in the first partition, n nodes are created, in each of which one of the n objects is assigned to the first position in the ranking, and

in the second partition, n-1 nodes are created, one for each of the remaining n-1 objects which might be assigned to the second position in the ranking, etc.
Example

First Partitioning Method

\[ \Delta = \# \text{ discrepancies} \]

\[ \text{all rankings} \]

\[ \text{A:B} \]

\[ \Delta = 0 \]

\[ B > A \quad \Delta = 1 \]

We will partition the most promising node, that with no discrepancies.

\[ \text{all rankings} \]

\[ \text{A:B} \]

\[ \Delta = 0 \]

\[ A > B \]

\[ \Delta = 1 \]

\[ B > C \]

\[ \Delta = 1 \]

\[ A > B > C \]

\[ (B > C \text{ is a discrepancy}) \]

Again, we partition the most promising node.

\[ \text{A:B} \]

\[ \Delta = 0 \]

\[ A > B \]

\[ \Delta = 0 \]

\[ B > C \]

\[ \Delta = 1 \]

\[ A > B \]

\[ (B > C \text{ is a discrepancy}) \]

\[ A > B > C \]

\[ \Delta = 0 \]

\[ B > C \]

\[ \Delta = 1 \]

\[ A > B \]

\[ \Delta = 0 \]

\[ B > C \]

\[ \Delta = 1 \]

\[ A > B \]

\[ \Delta = 0 \]

\[ B > C \]

\[ \Delta = 1 \]

\[ A > B \]

\[ \Delta = 0 \]

\[ B > C \]

\[ \Delta = 1 \]

\[ A > B \]

\[ \Delta = 0 \]
There remain 4 leaf nodes which are NOT terminal nodes, but each of these will have descendants with AT LEAST ONE discrepancy!

The ranking A>C>B>D is a minimum-discrepancy ranking.
Example

Second Partitioning Method

all rankings

Δ = # discrepancies

A is # 1
Δ = 1

B is # 1
Δ = 2

C is # 1
Δ = 1

D is # 1
Δ = 2

We will partition the most promising node, that with one discrepancy.

© D.L. Bricker, U. of Iowa, 1998
We will partition the most promising node, that with one discrepancy.

All remaining leaf nodes have at least two discrepancies!