Outline

- CSPs:
  - Definitions, DFS search, DFS with backtracking, Forward chaining, Constraint propagation, heuristics: variable and value ordering
- Local Search:
  - Hill climbing, stochastic hill climbing
- This lecture:
  - Local search for CSPs
  - Problem structure in CSPs
Note that in CSPs, the path doesn’t matter, only the solution.

Important things in a graph coloring problem: values (colors), variables (nodes), and the topology of the graph (constraints).

- Consider $N$ nodes in a graph
- Assign values $V_1, \ldots, V_N$ to each of the $N$ nodes
- The values are taken in $\{R,G,B\}$
- Constraints: If there is an edge between $i$ and $j$, then $V_i$ must be different from $V_j$
Eval is # of pairs attacking queens.

Neighbors are states where one queen has been moved.

Hill climbing will try the move that leaves the fewest remaining conflicts. It is a greedy algorithm. Therefore hill climbing doesn’t backtrack, not even for random search between ties. However, you may not want to keep track because a) it takes memory, b) if there are only global maxima/minima, or c) we’re likely to revisit states.
Generalize hill climbing algorithm that we used on N-Queens for all CSP’s. Other methods (DFS, forward search, constraint propagation) did not assign all variables up front.
Local Search for CSPs

• During search:
  – States have unsatisfied constraints
  – Successors mean to reassign variable values

• Variable selection
  – randomly select any conflicted variable

• Value selection
  – min-conflicts heuristic

Min-conflicts heuristic - Select variable at random, and then give it the value that results in the fewest conflicts.
Min-Conflicts Algorithm

• Start with a complete assignment of variables
• Repeat until a solution is found or maximum number of iterations is reached:
  – Select a variable $V_i$ randomly among the variables in conflict
  – Set $V_i$ to the value that minimizes the number of constraints violated
If you don’t choose which queen to move randomly, then it is easy to get stuck in a local minimum.
(Zebra is a complicated murder mystery-like problem.)

This is why AI is awesome. Because simple ideas produce disturbingly large improvements.

MRV is Minimum Remaining Value

<table>
<thead>
<tr>
<th>Method</th>
<th>USA (4 coloring)</th>
<th>N-Queens (1&lt;N&lt;=50)</th>
<th>Zebra</th>
</tr>
</thead>
<tbody>
<tr>
<td>DFS Backtracking</td>
<td>&gt; $10^6$</td>
<td>&gt; 40 $10^6$</td>
<td>3.9 $10^6$</td>
</tr>
<tr>
<td>+ MRV</td>
<td>&gt; $10^6$</td>
<td>13.5 $10^6$</td>
<td>1,000</td>
</tr>
<tr>
<td>Forward Checking</td>
<td>2,000</td>
<td>&gt; 40 $10^6$</td>
<td>35,000</td>
</tr>
<tr>
<td>+ MRV</td>
<td>60</td>
<td>817,000</td>
<td>500</td>
</tr>
<tr>
<td>Min-Conflicts</td>
<td>64</td>
<td>4,000</td>
<td>2,000</td>
</tr>
</tbody>
</table>

(Data from Russell & Norvig)
Deterministic bias is bad…it is generally much better to choose randomly when choosing which queen to move.

Discussion

• N-queens is easy for local search
  – Solutions are densely distributed
• Advantages of local search applied to CSPs
  – Online settings when the problem changes
    • Weekly’s airline schedule; flights, personnel assignments
  – Local search from current is faster than backtracking
  – Backtracking could find a solution with many changes from the current schedule
Outline

• Last Monday – CSPs:
  – Definitions, DFS search, DFS with backtracking, Forward chaining,
    Constraint propagation, heuristics:
    variable and value ordering
• Last lecture – Local Search:
  – Hill climbing, stochastic hill climbing
• This lecture:
  – Local search for CSPs
  – Problem structure in CSPs

With local search each state is a complete assignment.
How do you generate initial assignment? It is domain specific, though random is generally pretty good.
Constraint Graph

- Observations about *structure* of the graph
  - Components, independence, connectivity.....

T can be anything
Everyone except T has to be different than SA
Independence

• Why important?
• Suppose each component has c variables, each takes d values, from a total of n variables; then n/c subproblems, with time d^c therefore O(d^c * n/c) linear in n, versus O(d^n).

• n=80 boolean CSP, four with c=20, worst case from a lifetime down to less than 1s.
Simplest Case: Constraint Trees

- Any two variables are connected by at most one path

- Complexity of solving tree-structured CSPs?
  - Time *linear* in the number of variables.
You can really choose any node as the root of the tree.

Intuition: If all the values in the parent’s domain are consistent with the values in all the children’s domains, it is easy to choose consistent values, starting from the root of the tree.

Order the variables such that the parent of a node appears always before that node in the list.
Constraint Tree Algorithm

1. Up from leaves to root:
   - For every variable $V_i$, starting at the leaves:
     - $V_j = parent(V_i)$
     - Remove all the values $x$ in $D(V_j)$ for which there is no consistent value in $D(V_i)$.

2. Down from root:
   - Assign a value to the root of the tree.
   - For every variable $V_i$:
     • Choose a value $x$ in $D(V_i)$ consistent with the value assigned to $parent(V_i)$.

Visit each variable once: $N$

Worst case: Need to check all pairs of values: $d^2$

Total time: $O(N d^2)$
After we give $V_6$ a color, the graph can be treated as a tree, where $V_5$ and $V_3$ start off with one fewer acceptable colors.

Here the complexity of the tree is $(N-1 \times d^2)$, done $d$ times, so we have $(N-1) \times d^3$
For the HWs/midterm: you should think about these structural ideas, and how the algorithms are impacted by structure.
Complexity: $O((N-p) d^{p+2})$

Worst case: Need to check all possible assignments in $G \rightarrow d^k$.

- For every consistent assignment of values to variables in $G$:
  - Apply the tree algorithm to the rest of the variables

Tree algorithm $\rightarrow (N-p) d^2$
Worst case: Need to check all possible assignments in $G \rightarrow d^p$.

- For every possible consistent assignment of values to variables in $G$:
  - Apply the tree algorithm to the rest of the variables

Note: Unfortunately, it is impossible to find the minimum $p$ in polynomial time.
CSPs – Summary

• Definitions
• Standard DFS search
• Improvements
  – Backtracking
  – Forward checking
  – Constraint propagation
• Heuristics:
  – Variable ordering
  – Value ordering
• Examples
• Local search for CSP problems
• Problem structure in CSPs

What you need to know