1 handout: slides
asst 1 milestone was due
EOLQs

Heuristic Search
CSPs
Heuristic Search
## Comparison

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Time</th>
<th>Space</th>
<th>Complete</th>
<th>Admissible</th>
</tr>
</thead>
<tbody>
<tr>
<td>Depth-first</td>
<td>$b^m$</td>
<td>$bm$</td>
<td>If $m \geq d$</td>
<td>No</td>
</tr>
<tr>
<td>Breadth-first</td>
<td>$b^d$</td>
<td>$b^d$</td>
<td>Yes</td>
<td>if ops cost 1</td>
</tr>
<tr>
<td>Uniform-cost</td>
<td>$b^d$</td>
<td>$b^d$</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>IDDFS</td>
<td>$b^d$</td>
<td>$bd$</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Greedy</td>
<td>depends</td>
<td>$\approx$ time</td>
<td>depends</td>
<td>No</td>
</tr>
<tr>
<td>A*</td>
<td>depends</td>
<td>$\approx$ time</td>
<td>if $h$ adm.</td>
<td>if $h$ adm.</td>
</tr>
</tbody>
</table>

- **Branching factor**: $b$
- **Maximum depth**: $m$
- **Solution depth**: $d$
Simplified problem must give lower bound on original!

1. Relaxation: fewer and/or weaker constraints
   - Sometime efficient closed form
2. Abstraction: simplify token identity
   - Smaller search space

Want highest value

- If $h_1(n) \leq h_2(n)$ for all $n$, $h_2$ dominates $h_1$
Other Shortest-path Algorithms

- IDA*
- SMA*, IE
- RBFS
- RTA*, LRTA*

Course projects!
Search Algorithms

- **Uninformed methods**
  - Depth-first
  - Breadth-first
  - Uniform-cost

- **Informed methods**
  - Greedy
  - A*

- Bounding memory
  - Iterative deepening
  - Beam search
Search Algorithms, Take 2

- Depth-first
  - Depth-first

- ‘Best-first’
  - Breadth-first
  - Uniform-cost
  - Greedy
  - A*

- Bounding memory
  - Iterative deepening
  - Beam search
Constraint Satisfaction Problems
Map coloring: Given a map of $n$ countries and a set of $k$ colors, color every country differently from its neighbors.

$n$-queens : Given an $n \times n$ chessboard, arrange $n$ queens so that none is attacking another.

What algorithm would you use?
Types of Search Problems

- **Shortest-path (vacuum, tile puzzle, M&C)**
  - given operators and their costs
  - want least-cost path to a goal
  - goal depth/cost unknown

- **Constraint satisfaction (map coloring, \( n \)-queens)**
  - any goal is fine
  - fixed depth
  - explicit constraints on partial solutions
Do not expand any partial solution that violates a constraint.
office hours

asst 1

milestone:

you know it all now

don't forget: beauty, write-up

blog entries due on Wednesdays, 8am, 400 words

projects (Apr 2), UROP (Mar 1)
When assigning a variable, remove the conflicting values for all connected variables. Backtrack on domain wipeout.

Arc consistency: for every value in the domain of $x$, there exists a value in the domain of $y$ that satisfies all the constraints.
Variable choice: choose most constrained variable (smallest domain)
- want to keep tree small, failing quickly

Value choice: try least constraining value first (fewest removals)
- might as well succeed sooner if possible
## Example Results

<table>
<thead>
<tr>
<th></th>
<th>BT</th>
<th>FC</th>
<th>FC+MCV</th>
</tr>
</thead>
<tbody>
<tr>
<td>USA</td>
<td>&gt;1M</td>
<td>2K</td>
<td>60</td>
</tr>
<tr>
<td>n-Queens</td>
<td>&gt;40M</td>
<td>&gt;40M</td>
<td>820K</td>
</tr>
<tr>
<td>Zebra</td>
<td>3.9M</td>
<td>35K</td>
<td>500</td>
</tr>
<tr>
<td>Random 1</td>
<td>420K</td>
<td>26K</td>
<td>2K</td>
</tr>
<tr>
<td>Random 2</td>
<td>940K</td>
<td>77K</td>
<td>15K</td>
</tr>
</tbody>
</table>
Ensure every value for $x$ has a legal value in all neighbors $y$. If one doesn’t, remove it and ensure consistency of all $y$. 
Ensure every value for $x$ has a legal value in all neighbors $y$. If one doesn’t, remove it and ensure consistency of all $y$.

while $Q$ is not empty

$(x, y) \leftarrow \text{pop } Q$

if $\text{revised}(x, y)$ then

if $x$’s domain is now empty, return failure

for every other neighbor $z$ of $x$

push $(z, x)$ on $Q$

$\text{revised}(x, y)$

$\text{revised} \leftarrow \text{false}$

foreach $v$ in $x$’s domain

if no value in domain of $y$ is compatible with $v$

remove $v$ from $x$’s domain

$\text{revised} \leftarrow \text{true}$

return $\text{revised}$
Other Algorithms for CSPs

- (Conflict-directed) Backjumping
- Dynamic backtracking
- Randomized restarting

Course projects!
Please write down the most pressing question you have about the course material covered so far and put it in the box on your way out.

*Thanks!*