Midterm Review

Exam time: **3:15-4:45** (90 minutes) (no aids allowed)

**Search**

- Uninformed search:
  - BFS, UCS, DFS, Depth-limited, Iterative Deepening depth-first search
    - How to perform: search order criteria
    - Evaluation: criteria

<table>
<thead>
<tr>
<th>Criterion</th>
<th>BFS</th>
<th>UCS</th>
<th>DFS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Complete?</td>
<td>Y(^{(i)})</td>
<td>Y(^{(i)})</td>
<td>N</td>
</tr>
<tr>
<td>Optimal?</td>
<td>Y</td>
<td>Y</td>
<td>N</td>
</tr>
<tr>
<td>Time</td>
<td>O(b^d)</td>
<td>O(b^{1+\left\lceil C/\varepsilon \right\rceil})</td>
<td>O(b^m)</td>
</tr>
<tr>
<td>Space</td>
<td>O(b^d)</td>
<td>O(b^{1+\left\lceil C/\varepsilon \right\rceil})</td>
<td>O(bm)</td>
</tr>
</tbody>
</table>

- Heuristic search:
  - Greedy best-first search:
    - Always expand node with lowest h-value
  - A*:
    - \( f = g + h \)
    - Properties:
      - Admissibility: \( h \leq h^* \) (optimistic)
      - Monotonicity: \( h(n1) \leq c(n1,n2) + h(n2) \) (triangle inequality)
      - Simple proof: i.e. f value non-decreasing, etc.

b: maximum branching factor of the search tree  
d: depth of the least-cost solution  
m: maximum depth of the state space (may be \( \infty \))  
i: complete if b is finite;  
ii: complete if step costs \( \geq \varepsilon \) (positive \( \varepsilon \))
Example:
Trace the execution of A* for the graph shown below.

• Show the successive configurations of the frontier where the elements on
the frontier are paths. That is, the path n1 -> n2 -> n3 would be
written [n1, n2, n3]. Under each element of the frontier, indicate the f,
g and h values of the final node in the path (e.g., if g(m) = 5 and
h(m) = 7 write “12=5+7” underneath m. Remember to get the order
right, g(m) followed by h(m).

• Indicate the path that is expanded at each stage.

• Finally show the path to the goal found by A*.

• Is the heuristic admissible? Is it monotonic?

<table>
<thead>
<tr>
<th>A</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>4</td>
</tr>
<tr>
<td>C</td>
<td>5</td>
</tr>
<tr>
<td>D</td>
<td>7</td>
</tr>
<tr>
<td>E</td>
<td>3</td>
</tr>
<tr>
<td>F</td>
<td>10</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>G</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>H</td>
<td>5</td>
</tr>
<tr>
<td>I</td>
<td>8</td>
</tr>
<tr>
<td>J</td>
<td>2</td>
</tr>
<tr>
<td>K</td>
<td>0</td>
</tr>
<tr>
<td>L</td>
<td>7</td>
</tr>
</tbody>
</table>
CSP

- Problem formalization:
  - Variables: definition
  - Domain: possible values
  - Set of constraints (variable scopes): Binary/higher-order constraints

Example: Sudoku
- **Variables**: $V_{11}, V_{12}, \ldots, V_{21}, V_{22}, \ldots, V_{91}, \ldots, V_{99}$
- **Domains**:
  - $\text{Dom}[V_{ij}] = \{1-9\}$ for empty cells
  - $\text{Dom}[V_{ij}] = \{k\}$ a fixed value $k$ for filled cells.
- **Constraints**:
  - Row constraints:
    - All-Diff($V_{11}, V_{12}, V_{13}, \ldots, V_{19}$)
    - All-Diff($V_{21}, V_{22}, V_{23}, \ldots, V_{29}$)
    - ..., All-Diff($V_{91}, V_{92}, \ldots, V_{99}$)
  - Column Constraints:
    - All-Diff($V_{11}, V_{21}, V_{31}, \ldots, V_{91}$)
    - All-Diff($V_{21}, V_{22}, V_{13}, \ldots, V_{92}$)
    - ..., All-Diff($V_{19}, V_{29}, \ldots, V_{99}$)
  - Sub-Square Constraints:
    - All-Diff($V_{11}, V_{12}, V_{13}, V_{21}, V_{22}, V_{23}, V_{31}, V_{32}, V_{33}$)
    - All-Diff($V_{14}, V_{15}, V_{16}, \ldots, V_{34}, V_{35}, V_{36}$)

- Forward Checking algorithm: with heuristics
  - Minimum Remaining Values (MRV)
  - Generalized Arc Consistency (GAC):
    Enforced GAC during search, prune all GAC inconsistent values

GAC enforced example:
- Domains:
  - $\text{Dom}[X, Y, Z] = \{1,2,3,4\}$
  - $\text{Dom}[W] = \{1,2,3,4,5\}$
- Constraints:
  - $C1: X = Y + Z$
  - $C2: W > X$
  - $C3: W = X + Z + Y$
- Give the resultant GAC consistent variable values.
Games

- General sum game: i.e. two-player
  - Min-max strategy
  - Min-max game search tree: alpha-beta pruning
    - Alpha cuts
    - Beta cuts

Example: game tree search (slides 38 in week 5)