Chapter 3 Solving Problems by Searching
3.5 – 3.6 Informed (heuristic) search strategies
More on heuristics

CS5811 - Advanced Artificial Intelligence

Nilufer Onder
Department of Computer Science
Michigan Technological University
We have seen that A* search

- May have exponential time and space complexity but will perform well with a good heuristic
- Is complete
- Finds the optimal solution

We will look at another property that affects how the search proceeds.
A heuristic is consistent if
\[ h(n) \leq c(n, a, n') + h(n') \]

If \( h \) is consistent, we have
\[
\begin{align*}
f(n') &= g(n') + h(n') \\
&= g(n) + c(n, a, n') + h(n') \\
&\geq g(n) + h(n) \\
&= f(n)
\end{align*}
\]

We get \( f(n') \geq f(n) \), i.e., \( f(n) \) is nondecreasing along any path.

Consistency is the triangle inequality for heuristics.
Progress of $A^*$ with an inconsistent heuristic

Note that $h$ is admissible.
It never overestimates.
The root node was expanded.
Note that $f$ decreased from 6 to 4.
Progress of A* with an inconsistent heuristic

The suboptimal path is being pursued.
The right hand side path is suboptimal.
Goal found, but it appears as a child now. Remember that we cannot goal-test a node until it is selected for expansion.
The node with $f = 7$ is selected for expansion. After expansion, the lower node of the diamond gets a new, lower cost.
The optimal path to the goal is found. But nodes had to be reopened.
Iterative deepening A* (IDA*) search

- Idea: perform iterations of DFS. The cutoff is defined based on the $f$-cost rather than the depth of a node.
- Each iteration expands all nodes inside the contour for the current $f$-cost, peeping over the contour to find out where the contour lies.
The progress of IDA*

The blue nodes are the ones A* expanded. For IDA*, they define the new f-limit.
**IDA* algorithm**

**function** IDA* *(problem)*

**returns** a solution sequence (or failure)

initialize the **frontier** using the initial state of **problem**

\[
f\text{-}\text{limit} \leftarrow \text{f-cost}(\text{root}) \quad // f\text{-}\text{limit}: \text{current f}\text{-}\text{cost limit}
\]

**loop** do

\[
solution, \ f\text{-}\text{limit} \leftarrow \text{DFS}\text{-}\text{Contour}(\text{root, f}\text{-}\text{limit})
\]

**if** solution is non-null **then return** solution

**if** f-limit = ∞ **then return** failure
IDA* algorithm (cont’d)

function DFS-Contour (node, f-limit)
returns a solution sequence (or failure) and a new f-cost limit

// next-f is initialized to $\infty$

if node.f-cost > f-limit then return null, node.f-cost
if the node contains a goal state then return node, f-limit
for each child $n$ in node.Child-Nodes do
    solution, new-f ← DFS-Contour($n$, f-limit)
    if solution is not null then return solution, f-limit
    next-f ← Min(next-f, new-f)
return null, next-f
F-contours for A* search
Properties of IDA*

- **Complete**: Yes, similar to A*.
- **Time**: Depends strongly on the number of different values that the heuristic value can take on. 8-puzzle: few values, good performance TSP: the heuristic value is different for every state. Each contour only includes one more state than the previous contour. If A* expands $N$ nodes, IDA* expands $1 + 2 + \ldots + N = O(N^2)$ nodes.
- **Space**: It is DFS, it only requires space proportional to the longest path it explores. If $\delta$ is the smallest operator cost, and $f^*$ is the optimal solution cost, then IDA* will require $b \times f^* / \delta$ nodes to be stored.
- **Optimal**: Yes, similar to A*
Summary

- Consistency enforces the triangle inequality
- If an admissible but not consistent heuristic is used for graph search, we need to adjust path costs when a node is rediscovered
- Heuristic search usually brings dramatic improvement over uninformed search
- Keep in mind that the f-contours might still contain an exponential number of nodes