form of a heuristic function that estimates how far a given state is from the goal, or if we precompute partial solutions involving patterns or landmarks.

- Before an agent can start searching, a well-defined problem must be formulated.
- A problem consists of five parts: the initial state, a set of actions, a transition model describing the results of those actions, a set of goal states, and an action cost function.
- The environment of the problem is represented by a state space graph. A path through the state space (a sequence of actions) from the initial state to a goal state is a solution.
- Search algorithms generally treat states and actions as atomic, without any internal structure (although we introduced features of states when it came time to do learning).
- Search algorithms are judged on the basis of completeness, cost optimality, time complexity, and space complexity.
- Uninformed search methods have access only to the problem definition. Algorithms build a search tree in an attempt to find a solution. Algorithms differ based on which node they expand first:
  - Best-first search selects nodes for expansion using an evaluation function.
  - Breadth-first search expands the shallowest nodes first; it is complete, optimal for unit action costs, but has exponential space complexity.
  - Uniform-cost search expands the node with lowest path cost, $g(n)$, and is optimal for general action costs.
  - Depth-first search expands the deepest unexpanded node first. It is neither complete nor optimal, but has linear space complexity. Depth-limited search adds a depth bound.
  - Iterative deepening search calls depth-first search with increasing depth limits until a goal is found. It is complete when full cycle checking is done, optimal for unit action costs, has time complexity comparable to breadth-first search, and has linear space complexity.
  - Bidirectional search expands two frontiers, one around the initial state and one around the goal, stopping when the two frontiers meet.
- Informed search methods have access to a heuristic function $h(n)$ that estimates the cost of a solution from $n$. They may have access to additional information such as pattern databases with solution costs.
  - Greedy best-first search expands nodes with minimal $h(n)$. It is not optimal but is often efficient.
  - $A^*$ search expands nodes with minimal $f(n) = g(n) + h(n)$. $A^*$ is complete and optimal, provided that $h(n)$ is admissible. The space complexity of $A^*$ is still an issue for many problems.
  - Bidirectional $A^*$ search is sometimes more efficient than $A^*$ itself.
  - IDA* (iterative deepening $A^*$ search) is an iterative deepening version of $A^*$, and thus addresses the space complexity issue.
  - RBFS (recursive best-first search) and SMA* (simplified memory-bounded $A^*$) are robust, optimal search algorithms that use limited amounts of memory; given enough time, they can solve problems for which $A^*$ runs out of memory.