‹∈S → GOAL?(x) =T or F, Step Cost, and Path Cost. The fringe is the set of all search nodes that haven't been expanded yet.

Jninformed Search: Complete? Optimal? Complexity

Complete: is the algorithm guaranteed to find a solution when there is one? Optimality: does the strategy find the optimal solution? Time complexity: how long does it take to find a solution? Space complexity: how much memory is needed to perform the search?

BFS (complete, optimal if step cost is 1, complexity O(b^d) where d is depth of shallowest goal node), DFS (Complete only for finite search tree Not optimal, time complexity O(b^m) Space Complexity O(m) where m: maximal depth of a leaf node), ID visits the nodes in the search tree in the same order as depth-first search, but the cumulative order in which nodes are first visited, assuming no pruning, is effectively breadth-firs (complete, optimal if step cost is 1) time complexity O(b^d) and Space Complexity O(m).

Heuristic Search: an evaluation function f maps each node N of the search tree to a real number $f(N) \geq 0$. Best-first (greedy) search sorts FRINGE n increasing f (f(N) = h(N)). f(N) = g(N) + h(N), where, g(N) is the cost of the path from the initial node to N, h(N) is an estimate of the cost of a path from N to a goal node.

The heuristic function h(N) is admissible if: $0 \leq h(N) \leq h^*(N)$ where $h^*(N)$ is be the cost of the optimal path from N to a goal node, a.k.a., neve overestimates or it is optimistic. h(G) = 0, where G is the goal state.

An admissible heuristic can usually be seen as the cost of an optimal solution to a relaxed problem (one obtained by removing constraints).

A*: greedy order by f(N) = g(N) + h(N), where: g(N) = cost of best path found so far to N, h(N) = admissible heuristic function. For all arcs: c(N,N') ≥ ε > 0. A* is complete and optimal. This result holds if nodes revisiting states are not discarded

An admissible heuristic h is consistent (or monotone) if for each node N and each child N' of N: h(N) ≤ c(N,N') + h(N'). A consistent heuristic is also admissible.

Let h_1 and h_2 be two consistent heuristics such that for all nodes N: $h_1(N) \leq h_2(N)$. h_2 is said to be more accurate (or more informed) than h_1

Iterative Deepening A* (IDA*). Idea: Reduce memory requirement of A* by applying cutoff on values of f. Consistent heuristic function h. Algorithm IDA*: Initialize cutoff to f(initial-node). Repeat: [Perform depth-first search by expanding all nodes N such that f(N) ≤ cutoff. Reset cutoff to smalles value f of non-expanded (leaf) nodes] Advantages: Still complete and optimal, requires less memory than A* and avoid the overhead to sort the fringe. Drawbacks: Can't avoid revisiting states not on the current path, available memory is poorly used.

Adversarial Search: MIN-MAX search. (1)Using the current state as the initial state, build the game tree uniformly to the leaf nodes. (2) Evaluate whether leaf nodes are wins (+1), losses (-1), or draws (0). (3) Back up the results from the leaves to the root and pick the best action assuming the worst from MIN. Minimax algorithm. Complete? Yes, if tree is finite. Optimal? Yes, against optimal opponent. Otherwise…?. Time complexity? $O(b^h)$. Space complexity? O(bh)

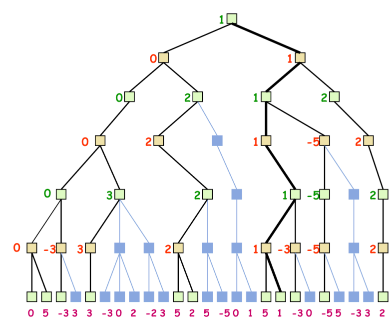
Minimax examines $O(b^h)$ nodes, so does alpha-beta in the worst-case Alpha beta pruning

The gain for alpha-beta is maximum when:

The children of a MAX node are ordered in decreasing backed up values

The children of a MIN node are ordered in increasing backed up values

Then alpha-beta examines $O(b^{h/2})$ nodes



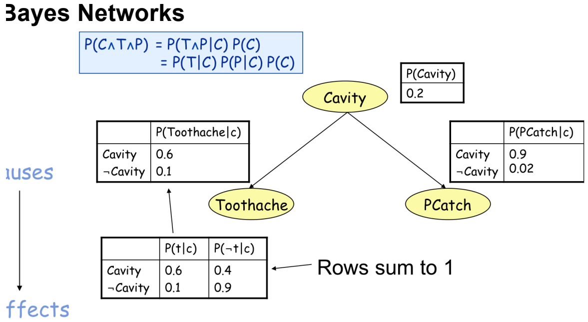
Probabilistic Reasoning: P(A,B) = P(A|B)P(B). Bayes Rule: P(A|B) = P(B|A) P(A) / P(B). Marginalization: $P(C) = S_t S_p P(C \wedge t \wedge p)$

$P(a \vee b) = P(a) + P(b) - P(a \wedge b)$. P(a|b) is the posterior probability of a given knowledge that event b is true.

Two events a and b are independent if $P(a \wedge b) = P(a) P(b)$ hence P(a|b) = P(a) .

Two events a and b are conditionally independent given c, if $P(a \wedge b|c) = P(a|c) P(b|c)$ hence P(a|b,c) = P(a|c)

Bayes Networks



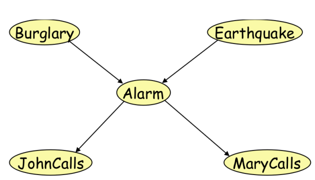
Factorization of the joint distribution

$$p(x) = \prod_{v \in V} p(x_v \mid x_{\mathrm{pa}(v)})$$

A node is independent of its non-descendants given its parents

Inference: Marginalization

$P(A) = \Sigma_{b,e} P(A,b,e)$



glary ⊥ Earthquake
nCalls ⊥ MaryCalls | Alarm
nCalls ⊥ Burglary | Alarm
nCalls ⊥ Earthquake | Alarm
yCalls ⊥ Burglary | Alarm
yCalls ⊥ Earthquake | Alarm

A node is independent of its non-descendents, given its parents

Evidence on the (directed) road between two variables makes them **independent** Evidence on ar "A" node makes descendants **independent** Evidence on a "V" node, or below the V, makes the ancestors of the variables **dependent** (otherwise they are independent)

P(M) = P(M,A) + P(M,-A) = P(M|A)P(A) + P(M|-A)P(-A) [1. Marginalization, 2. Conditioning]

with $P(A) = sum_{b,e} P(A,b,e) = sum_{b,e} P(A|b,e)P(b)P(e)$ [1. Marginalization, 2. Conditioning+independence assumptions]

and P(-A) = 1-P(A)./// P(M|J) = P(M,A|J) + P(M,-A|J) = P(M|A,J)P(A|J) + P(M|-A,J)P(-A|J) = P(M|A)P(A|J) + P(M|-A)P(-A|J) [1) marginalization, 2) conditioning, 3) conditional independence o M and J given A]

P(A|J) = P(J|A)P(A)/P(J) [Bayes rule]

P(-A|J) = 1-P(A|J) P(A) is given in Q1, P(J) is computed in the same way as P(M) in Q1.

laximum Likelihood: Likelihood of data **d**=$\{d_1,\ldots,d_N\}$ given q. P(**d**|q) = P_j P(d_j|q). Log is monotonically increasing function l(q) = log P(**d**|q). dl/dq(q = 0 at the maximum likelihood estimate. P(q|**d**) is known as the **maximum a posteriori** (MAP) estimate

lachine Learning: Agent has made observations (data). Now must make sense of it (hypotheses). Basic form: learn a function from examples. s the unknown target function. An example is a pair (*x*, *f(x)*). Problem: find a hypothesis *h* such that *h* ≈ *f*, given a *training set* of examples D nstance of *supervised learning*: Classification task: f → {0,1,…,C} (usually C=1). Regression task: f → reals. (KIS).

SVM: let y^i = -1 or 1. Boundary $w^Tx+b = 0$, $||w||=1$, geometric margin is $y^i(w^Tx^i+b)$. SVMs try to optimize the minimum margin over all examples

3ayesian learning (find parameters of a probabilistic model) [Maximum likelihood, Maximum a posteriori]. Classification [Decision trees (discrete attributes, few relevant), Support vector machines (continuous attributes)]. Regression [Least squares (known structure, easy to interpret), Neura nets (unknown structure, hard to interpret)] Nonparametric approaches [k-Nearest-Neighbors, Locally-weighted averaging / regression]

Cross-validation: Take out some of the training set. Train on the remaining training set. Test on the excluded instances

Agents: Simple reflex (aka reactive, rule-based), Model-based, Goal-based, Utility-based (aka decision-theoretic, game-theoretic), Learning (aka adaptive).

Types of Environment: Observable / non-observable, Deterministic / nondeterministic, Episodic / non-episodic, Single-agent / Multi-agent.

$U(s) = R(s) + \max_{a\in Appl(s)}\Sigma_{s'\in Succ(s,a)} P(s'|s,a)U(s')$

Action Uncertainty: Each action representation is of the form: Action: a(s) -> $\{s_1,\ldots,s_r\}$ where each s_i, i = 1, ..., r describes one possible effect of the action in a state s