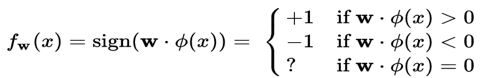
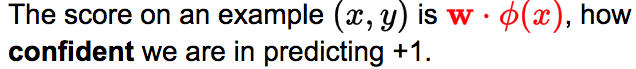


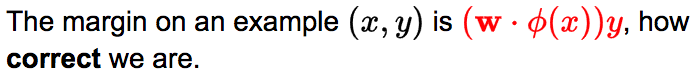
Feature Vector



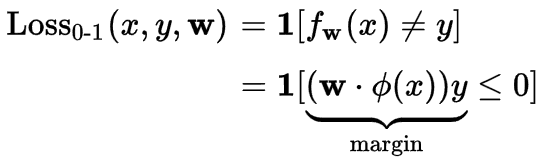
**Linear Classifier**



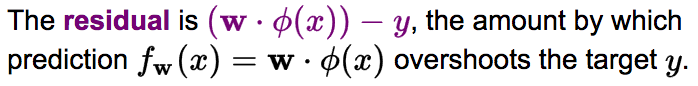




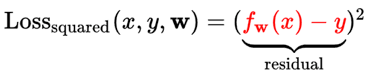
Zero One Loss for binary classifier



**Linear Regression**

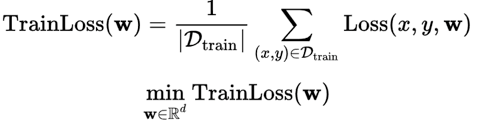


**Squared Loss and absdev**

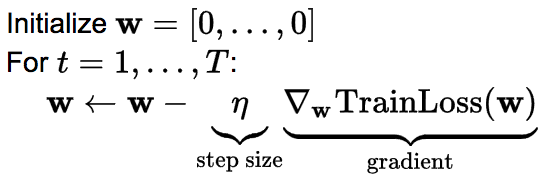


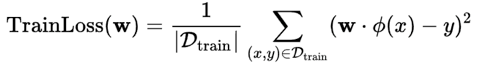


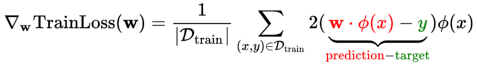
**Objective : Minimize the Train Loss**



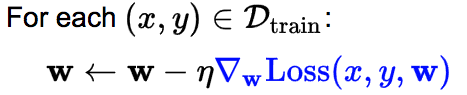
**Gradient Descent**







**Stochastic Gradient Descent**



**Hinge Loss**

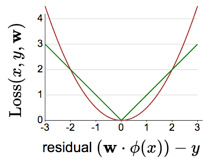
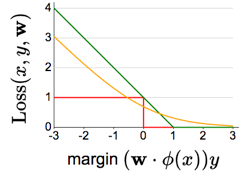


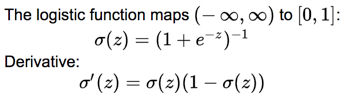
**Gradient of Hinge Loss**

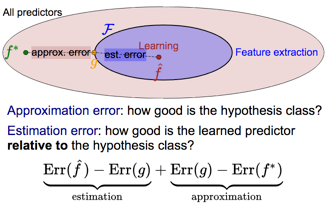


**Logistic Regression**

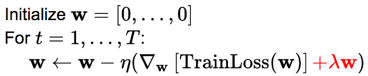


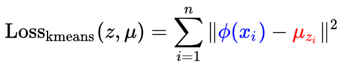


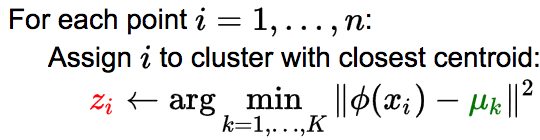


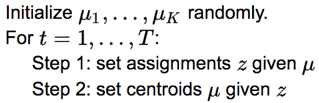
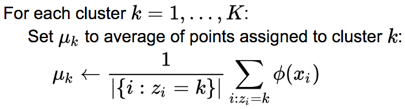


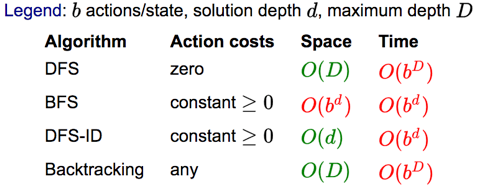
**Regularization**



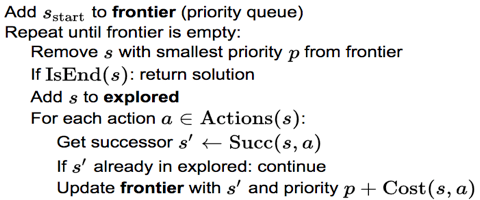
**K Means**



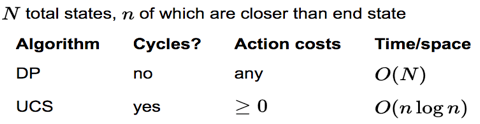




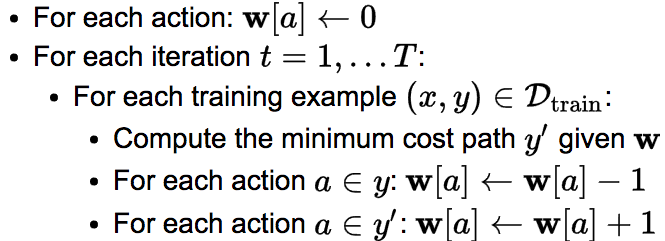
**UCS Algorithm**



**DP vs UCS**

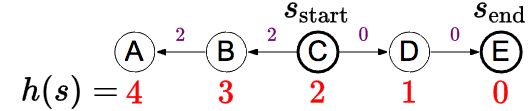


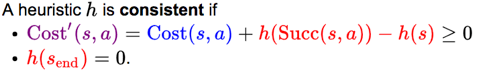
**Structured Perceptron Alogrithm**



A\* Algorithm

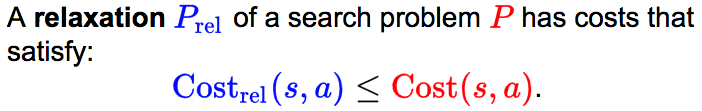


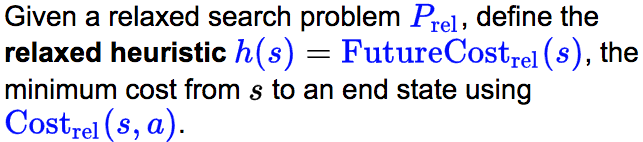




Heuristic is admissible if 

**Relaxation**



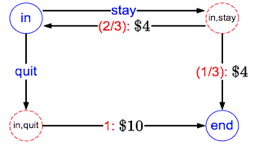


And also that means h(s) is consistent.

Note : Heuristic Tradeoff: efficiency(relaxed) vs tightness(not too relaxed)

Note:if h1 and h2 are consistent then max (h1,h2) is too.

**MDP**



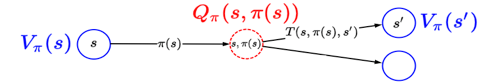
**Search State plus Reward and Transition Prob, Disc factor**



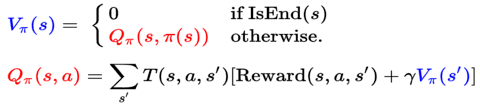


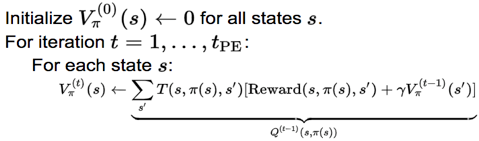






**Policy Evaluation: recurrences and Algorithm**





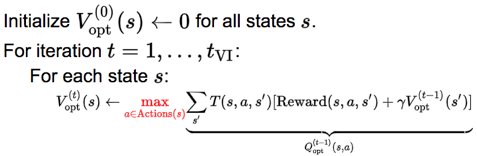
Convergence: 

**Optimal value and Policy**





**Value Iteration Algorithm:**

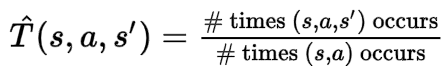


Note: converges if mdp is acyclic or disc factor < 1

**Reinforcement Learning: no T or R defined in MDP.**

**Model based Monte Carlo(optimal policy)**

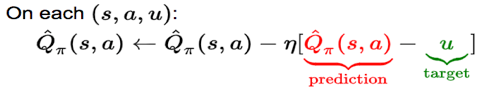
****

****

**Model Free Monte Carlo**

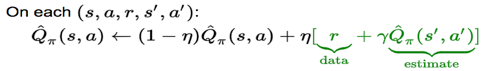
****

****

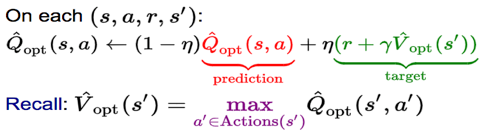
****

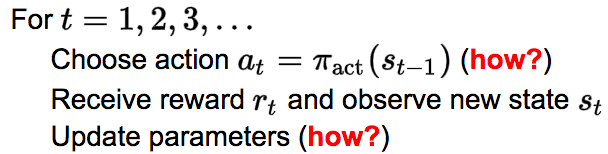
**Bootstrapping Methods: SARSA**

**The main advantage that SARSA offers over model-free Monte Carlo is that we don't have to wait until the end of the episode to update the Q-value.**

****

**Q-Learning Algorithm(Off Policy and no Succ States reqd)**

****

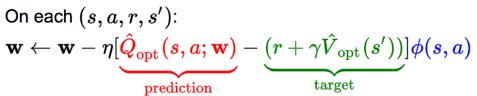
**Exploration Policy: Exploration vs Exploitation(RL tmplat)**

**Epsilon Greedy Policy for choosing actions**

****

**Function Approx. :Large State Spaces, hard to explore**

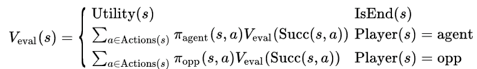
****

****

**Adversarial Games: search state plus these:**

****

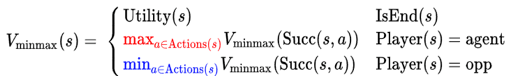
**Game Evaluation Recurrence(Generic)**

****

**Expectimax Recurrence(opponent takes average)**

****

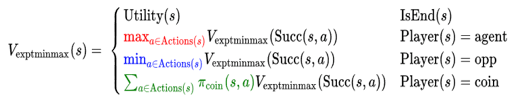
**Minimax Recurrence**

****

**Relationship between game values**

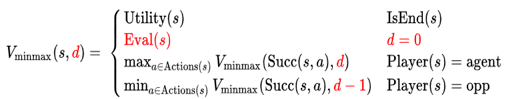
**V(Ex,pi7) >= V(max,pi7) >= V (max,pimin) >= V(Ex,pimin)**

**Expectiminimax Algorithm**

****

**Speeding up minimax : Eval func and Alpha Beta pruning**

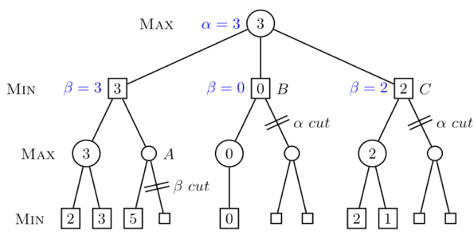
**Depth Limited Search**

****

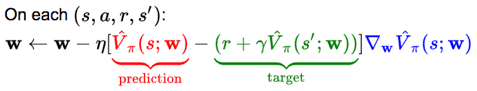
**AlphaBeta Pruning**

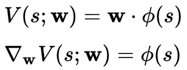
**alpha: lwr bound on val of max node**

**beta : uppr bound on val of min node**

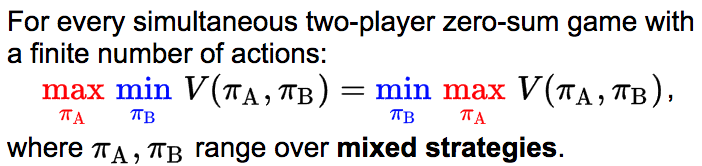
****

**TD Learning(On policy and need Succ states )**

****

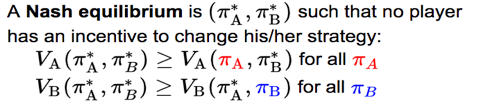
****

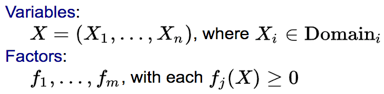
**Simultaneous Games(Rock paper scissors)**

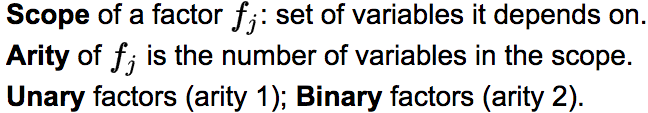
****

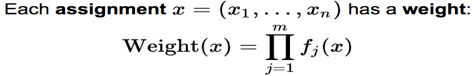
**Non Zero Sum Games(Prisoner’s dilemma)**

**In any finite-player game with finite number of actions, there exists at least one Nash equilibrium.**

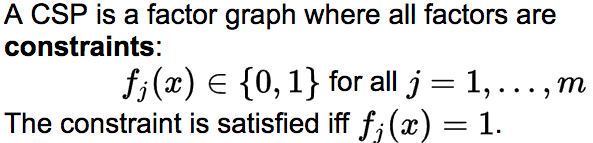
****

**Factor graph**

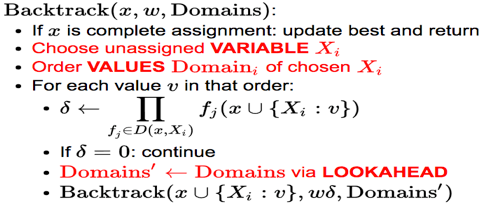




Objective 

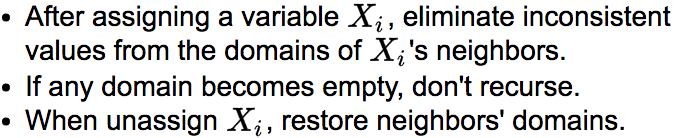


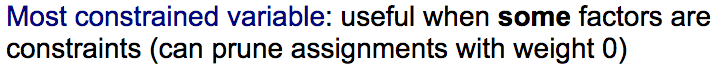
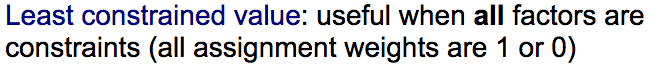






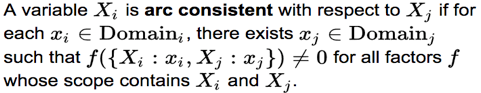
**Forward Checking(Look Ahead)**

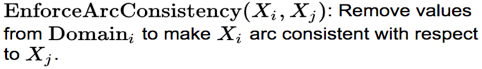




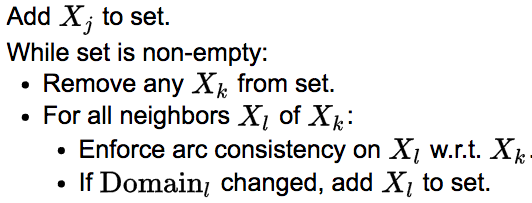


**Arc Consistency: eliminate values from domains-> reduce branching**





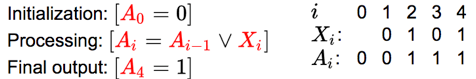
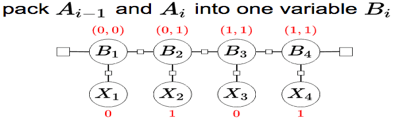
AC 3 Algorithm

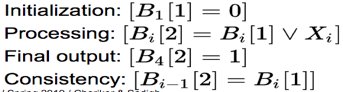


Runtime O(ED^3). D -> largest domain E-> number of edges

**N ARY constraints**



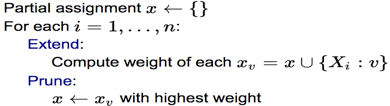
Auxiliary Variables hold intermediate computation.



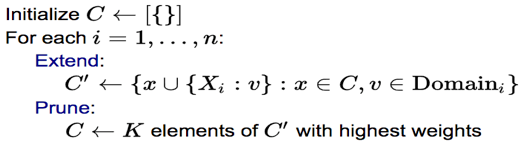


Pruning Techniques only useful for constraints, does not help reduce actual runtime .

Finding Maximum Weight Assignments: Greedy Algorithm

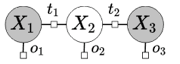


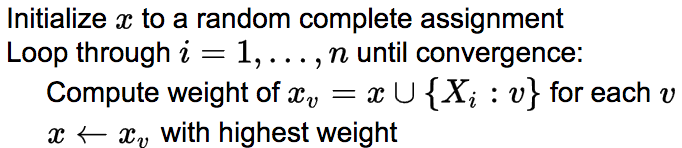
**Beam Search Algorithm**:





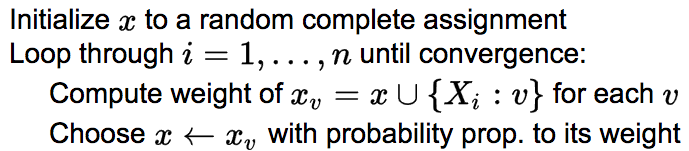
**Iterated conditional modes**



 x1,v,x3

**Gibbs Sampling**: Sample an assign. Prop. to its weight

Note: The Gibbs update depends on the Markov blanket on a variable



**Algorithms for max-weight assignments in factor graphs:**

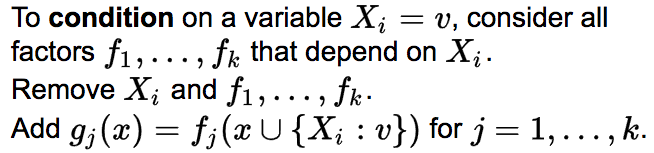
**Backtracking search: exact, exponential time**

**Beam search: approximate, linear time**

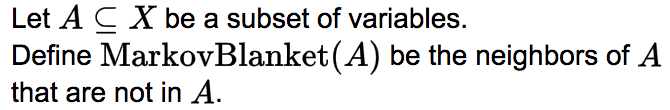
**Iterated conditional modes: approximate, deterministic**

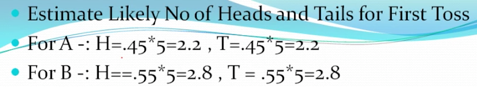
**Gibbs sampling: approximate, randomized**

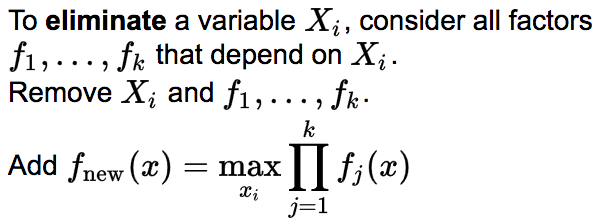
**Conditioning Algorithm**



Conditionally Independent: conditioning on C, makes A and B independent



**Elimination Algorithm**

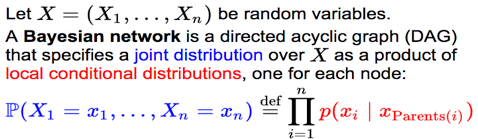


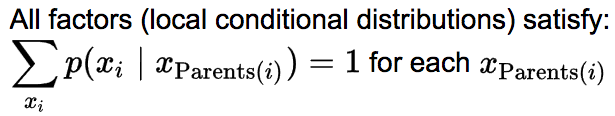


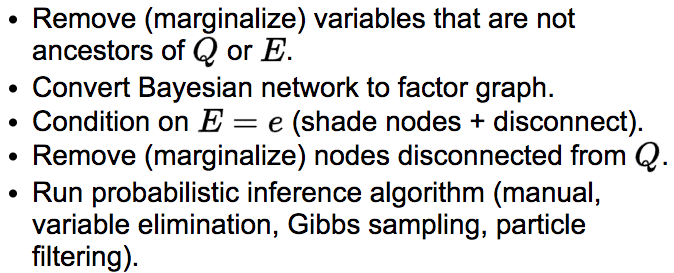
The **treewidth** of a factor graph is the maximum arity of any factor created by variable elimination with the best variable ordering.

**Bayesian Ntwks: Joint ,Marginal(sum) ,Conditional(select rows) Distribution: three types for random variables**

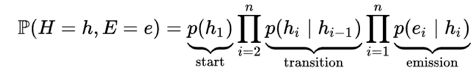
**Explaining away**:Suppose two causes positively influence an effect. Conditioned on the effect, conditioning on one cause reduces the probability of the other cause



Probabilistic Inference Strategy: 



HMM:

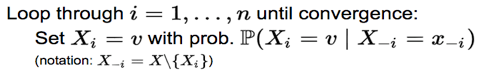


Smoothing Query: Lattice Representation:

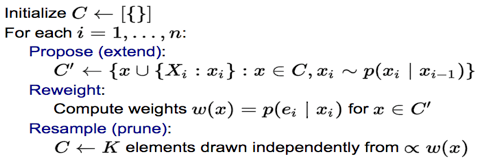


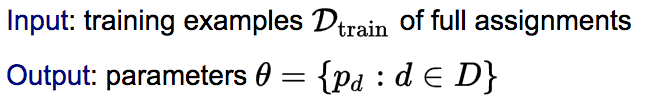


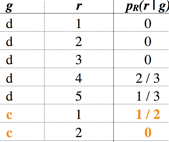
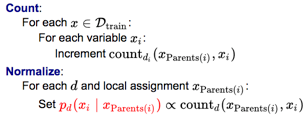
**Gibbs Sampling**



**Particle Filtering**

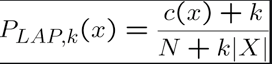


**Max Likelihood Learning Algorithm**

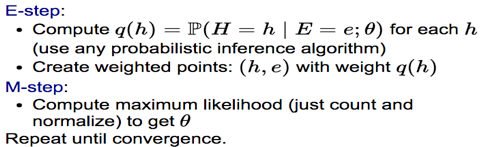


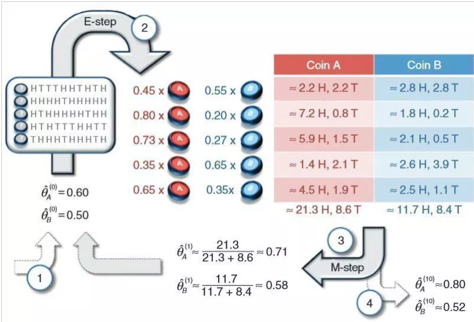
**Regularization:Laplace Smoothing(add 6 for Dice)**

just add to the count for each possible value, regardless of whether it was observed or not.Eg: X = 6 for dice



**Expectation maximization (EM)algorithm**

****

****

Max Likelihood of p^n(1-p)^m = n/(n+m)

E step: for each missing assume all vals,create virtual bags and find probs. Mstep: find Max likelihood for all params.