3 Decision Problems
3.1 Utility Theory and 3.2 Decision Networks

$$EU(a|o) = \sum_{s'} (P(s'|a,o)U(s'))$$
$$EU^*(o) = \max_a \{EU(a|o)\}$$
$$VOI(O'|o) = \left(\sum_{o'} (P(o'|o)EU^*(o,o')) \right) - EU^*(o)$$

3.3 Games

$$U([a_1 : p_1; \cdots ; a_n : pn]) = \sum_{i=1}^{n} (p_i U(a_i))$$

Dominant Strategy - strategy that is best response to all possible opposing strategies s_{-i}.
Dominant Strategy Equilibrium - All agents have a dominant strategy.
Nash Equilibrium - No agent can benefit by switching strategies when all other agents keep their strategy.
Logit level k strategy
level 0 choose uniformly
level k assume other agent acting at level k-1 and choose based on logit distribution

$$P(a_i) = e^{\lambda U_i(a_i, s_{-i})}$$

4 Sequential Problems
4.1 Formulation MDP

$$U = \sum_{t=0}^{n-1} (r_t)$$
$$U = \sum_{t=0}^{\infty} (\gamma^t r_t)$$
$$U = \lim_{n\to\infty} \left(\frac{1}{n} \sum_{t=0}^{n} (r_t) \right)$$

4.2 Dynamic Programming
Policy Evaluation

$$U_t^{\pi}(s) = R(s, \pi(s)) + \gamma \sum_{s'} \left(T(s'|s, \pi(s)) U_{t-1}^{\pi}(s') \right)$$

Policy Iteration

$$\pi_{k+1}(s) = \text{argmax}_a \left\{ R(s,a) + \gamma \sum_{s'} (T(s'|s,a)U^{\pi_k}(s')) \right\}$$

Value Iteration

$$U_{k+1}(s) = \max_a \left\{ R(s,a) + \gamma \sum_{s'} (T(s'|s,a)U_k(s')) \right\}$$
$$\pi(s) = \text{argmax}_a \left\{ R(s,a) + \gamma \sum_{s'} (T(s'|s,a)U^*(s')) \right\}$$

Bellman Equation

$$U(s) = \max_a \left\{ R(s,a) + \gamma \sum_{s'} (T(s'|s,a)U(s')) \right\}$$

Closed and Open - Loop Planning
Closed Loop accounts for future actions
Open Loop uses expected utility only
4.3.1 Factored MDP
4.5 Approximate Dynamic Programming
4.5.1 Local Approximation

$$U(s) = \sum_{i=1}^{n} (\lambda_i \beta_i(s))$$
$$\lambda_i = U(s_i)$$
$$\beta_i(s) \approx d(s, s_i)$$

4.5.2 Global Approximation

$$U(s) = \sum_{i=1}^{m} (\lambda_i \beta_i(s))$$

$\lambda_i \beta_i$ from linear regression

4.6 Online Methods
Forward Search
Branch and Bound Search

$$\underline{U}(s) = \text{Lower Bound}$$
$$\overline{U}(s,a) = \text{Upper Bound}$$

Sparse Sampling
Sample using generative model instead of T and R.
Monte Carlo Tree Search
5 Model Uncertainty
5.1 Exploration and Exploitation

$$\rho_i = P(win_i | w_i, l_i)$$

ε-greedy, choose random with probability ε otherwise greedy.
Softmax choose action with logit-model, probability $e^{\lambda \rho_i}$.
5.2 Maximum Liklihood Model-Based Methods

$$N(s,a,s') = \text{Counts}$$
$$\rho(s,a) = \sum (r(s,a))$$
$$N(s,a) = \sum_{s'} (N(s,a,s'))$$
$$T(s'|s,a) = \frac{N(s,a,s')}{N(s,a)}$$
$$R(s,a) = \frac{\rho(s,a)}{N(s,a)}$$

5.2.1 Randomized Updates - Dyna

$$Q(s,a) = R(s,a) + \gamma \sum_{s'} \left(T(s'|s,a) \max_{a'}\{Q(s',a')\} \right)$$

5.2.2 Prioritized Updates 5.3 Bayesian Model-Based Methods

$$b_0(\theta) = \prod_s \left(\prod_a \left(Dir(\theta_{(s,a)} | \alpha_{(s,a)}) \right) \right)$$
$$b_t(\theta) = \prod_s \left(\prod_a \left(Dir(\theta_{(s,a)} | \alpha_{(s,a)} + m_{(s,a)}) \right) \right)$$
$$T(s',b'|s,b,a) = \delta_{\tau(s,b,a,s')}(b') P(s'|s,b,a)$$
$$P(s'|s,b,a) = \int_\theta b(\theta) P(s'|s,\theta,a) \, \mathrm{d}\theta$$

5.4 Model-Free Methods
5.4.1 Incremental Estimation

$$\hat{x}_n = \hat{x}_{n-1} + \frac{1}{n}(x_n - \hat{x}_{n-1})$$
$$\hat{x} = \hat{x} + \alpha(x - \hat{x})$$

5.4.2 Q-Learning

$$Q(s,a) = Q(s,a) + \alpha(r + \gamma \max_{a'}\{Q(s',a')\} - Q(s,a))$$

5.4.3 SARSA

$$Q(s_t,a_t) = Q(s_t,a_t) + \alpha(r_t + \gamma Q(s_{t+1},a_{t+1}) - Q(s_t,a_t))$$

5.4.4 Eligibility Traces