IA Cheat Sheet

1 Reactive Agents

1.1 Markov Decision Processes (MDP)

- State transitions are undeterministic
  $$T(s, a, s') = p(s'|s, a)$$
- State values in MDP:
  $$V(s) = \max_{\pi} E\left(\sum_{t=0}^{\infty} \gamma^t R(s, a_\pi(t))\right)$$
- **Value iteration**:
  - First compute values
    $$Q(s, a) \leftarrow R(s, a) + \gamma \sum_{s' \in S} T(s, a, s')V(s')$$
    $$V(s) \leftarrow \max_a Q(s, a)$$
  - Guaranteed to converge
  - Stop criterion: $\max_{s \in S} |V'(s) - V(s)| \leq \epsilon$
  - Compute policy: $\pi(s) = \mathrm{argmax}_a\, Q(s, a)$
- **Policy iteration:**
  - Optimize policy directly.
    $$V_\pi(s) \leftarrow R(s, \pi(s)) + \gamma \sum_{s' \in S} T(s, \pi(s), s')V_\pi(s')$$
    $$\pi'(s) \leftarrow \underset{a}{\mathrm{argmax}}\, V_\pi(s)$$
  - Stop when policy doesn't change anymore.
- **Q-learning**
  - Unknown models (R(..) and T(..))
  - Learn table $Q(s, a)$, starting with arbitrary initial values
    $$Q(s, a) \leftarrow (1 - \alpha)Q(s, a) + \alpha(r + \gamma \max_{a'} Q(s', a'))$$
    with α the learning factor
  - Optimal policy
    $$\pi(a) = \underset{a}{\mathrm{argmax}}\, Q(s, a)$$
- **Partially Observable MDP, Belief MDP**
  - Uncertainty of the current state
  - observations o and belief b
  - transition function
    $$\tau(b, a, b') = \sum_{o|SE(b,a,o)=b'} p(o|a, b)$$
  - reward function
    $$r(b, a) = \sum_{s \in S} b(s)R(s, a)$$
  - Solvable with few states and actions
  - Can be transformed into MDPs and then solved.

2 Deliberative Agents

- Compute strategy for a particular scenario avoiding computations on unnecessary states.
- Reward only given in goal state
- **Depth-first**: always expand the first node found until there are no more successors.
- **Depth-limited**: Impose a depth limit so we don't follow a dead-end path too far.
- Optimization: Keep track of the best goal node and ignore branch with higher cost
- **Breadth-first**: exploring the tree layer by layer (takes too much memory)
- **Minimax search** others actions minimize payoff and our action maximize payoff.
- **Alpha-Beta pruning**: Use DFS and abandon a branch as soon as
  1. Opponent wouldn't allow us to get there
  2. Already found a branch where opponent can do us less harm
- Doesn't work on games with chances.
- **Expectiminimax**: game with chance, maximize expected return. Evaluation needs to average over all possible outcomes.
- **Regret**: The difference in outcome between playing move i and playing the optimal move.
- **Monte-Carlo Tree search**: concentrates on analysing the most promising moves, basing the expansion of the search tree on random sampling of the search space.
- **Multi-Armed Bandit**: Maximize the sum of the reward over several plays
- More sampling on more promising moves
- **Bounding Regret**
  - Initialization: play each arm once
  - Loop: play the jth arm that maximizes
    $$\mu_j + \sqrt{\frac{2\ln(n)}{n_j}} \text{ where } \mu_j \text{ is the average}$$
    reward, n the total number of plays so far and n_j the number of plays with j

3 Planning with Fact. Repr.

- States as combination of (important) features
- State as a vector of k state variables
- Formulate successor functions and rewards as functions on the vector of state variables.
- Usefull for multi-agent (exchange features)
- **Bayesian Networks**:
  - Nodes = events e.i. $x_3 = c$
  - Edges = causation e.i. $(x_3 = c) \rightarrow (x_7 = d)$
  - Causation is uncertain
    $$(x_i \rightarrow x_j) \Rightarrow p(x_j|x_i)$$
  - Model transitions by a separate Dynamic Bayesian Network for each action
- Value function factored into basis functions
  $$V(X) = \sum w_i b_i$$
- b_i are programmed by analysis of the system and w_i are determined so that the overall mean square error is minimized
- Better to use policy iteration.
- **Situation Calculus (STRIPS)**
  - States modelled by a set of propositions
  - *preconditions*, *postconditions*: ADD-LIST, DELETE-LIST

4 Multiagent Systems

- Delegation: central planner computes a plan for several agents
- Mediated: each agent makes its own partial plans; mediator coordinates them
- Distributed: each agent makes its own plans and coordinates through message exchange
- **Blackboard systems**: centralized blackboard with current goals, states and agent's plans
- **Partial-global-planning (PGP)**: each agent inserts its partial plans in the goal tree. $\Rightarrow$ discover joint goals and combine plans.
- **Publish-subscribe systems**: identify potential conflicts and create explicit objects for them. When an agent's plan involves the resource, all others are notified $\rightarrow$ detection of conflicts/synergies. It uses peer-to-peer negotiations for optimal joint plan $\rightarrow$ D.CSP
- **Ontologies**: shared concepts and vocabulary. $\Rightarrow$ Communication among heterogenous agents
- **Contract Nets (CN)**: task-sharing protocol:
  - Agents can be *contractors* or?and *managers*.
  - When an agent cannot solve a task
    $\Rightarrow$ breaks it down into sub-tasks
    $\Rightarrow$ announces the sub-task to the CN
    $\Rightarrow$ acts as a manager for this sub-task.
  - Bids are received from contractors and the winning contractor are awarded the job.
- **Market-based CNs**: managers set the prices

5 Distributed Multiagent Systems

- **Social laws**: Common rules that all agents follow to avoid conflicts. Laws must allow A to achieve goals. Find social laws is NP
- **Distributed CNs**: Managers distribute tasks asynchronously and contacts agents directly.
- **Marginal cost** to A_i of task t given a remaining set of tasks T:
  $$c_{add}(A_i, t) = cost(A_i, T \cup t) - cost(A_i, T)$$
- A bid higher that marginal cost: make profit

5.0.1 Constraint Satisfaction Problems

- Variables, domains, constraints, relations
- Find solution such that for all constraints, value combinations are allowed by relations.
- Solving a CSP:
  - backtrack search: assign one variable at a time, backtrack when no assignment without satisfying constraints
  - dyn. prog.: eliminate variables and replace by constraints until a single one remains
  - local s.: start with random assignment, make changes to reduce number of constraint violations

5.0.2 Distributed CSP

- task allocation, resource sharing, scheduling, all can be expressed as constraint satisfaction.
- Each variable belongs to one agent
- **Centralized Backtr'**: gather all info' into a leader which solves the problem.
- **Sync. Backtr'**: A_0 generates a partial sol', A_i generate an extension to this partial sol'
- Allows common CSP heuristics such as forward checking and dynamic variable ordering yielding in strong efficiency gains
- **Asynchronous Backtr'**: Agent work in parallel without sync. Global priority ordering among variables.
- **Dynamic Programming**: replace variables by constraints
- **Dist. local s.**: init variables arbitrarly and iteratively make local improvements. Low complexity but sub-opt. solutions
- **Min-conflicts**:
  - random value to variables in parallel
  - At each step, find the change in variable assigment which most reduces the number of conflicts.
- **Breakout algorithm**:
  - Like local s. for solving CSP
  - Agents repeatedly improve their tentative and bad sets of assignments for variables simultaneously while communicating such tentative sets with each other until finding a solution to an instance of the distributed CPS.
  - similar to min-conflict, but assign dynamic priority to every conflict, initially = 1.

6 Game Theory

- **Zero-sum game**: for every outcome, sum of rewards = 0 $\Rightarrow$ pure competition
- **Strategy**: recipe by which each player chooses its actions.
- **Pure strategy**: for each state, the action is chosen in a deterministic way.
- **Mixed strategy**: choose action following a probabilty distribution
- **Dominant strategy**: strategy which is best for every action of the other player.
- If both players have a strictly dominant strategy $\rightarrow$ unique Nash equilibrium
- **Weakly dominant strategy**: for every action of the other player, the strategy is at least as good as any other, and it is strictly better for at least one action of the other player.
- **Very weekly dominant strategy**: for every action of the other player, the strategy is at least as good as any other
- **Minimax stra.**: maximize gains supposing that the opponent minimize its losses.
- **Minimax theorem**: In a zero-sum game with 2 players, the average gain v_A of player A using the best mixed minimax strategy is equal to the average loss v_B of player B using its best mixed minimax strategy.
- **Lottery**: payoff is uncertain
- **Nash equilibrium**: no player has an interest to change given that the other doesn't change.
- $\exists$ set of mixed Nash equ. strategies
- Properties of the Nash equ. for player A:
  - A gets expected payoff $v(A)$
  - $\forall a_j \in s(A)$ have expected payoff $v(A)$:
    $$v(A) = \sum_{a_k \in s(B)} p(a_k)R_A(a_j, a_k)$$
  - $\forall a_j \notin s(A)$ have smaller payoff
- Action a_i strictly dominates a_j if for all strategies of the other players, the expected payoff for a_i is greater than that for a_j
- **Computing Nash equ.**: Eliminate all dominated actions, search through all possible supports and solve then for Nash equ.
- **Bayes-Nash equ.**: Use expected utilities

7 Agent Negociation

- **Mediated Equ.**: can ask the mediator to play
- Mediator can be a mean to enforce a contract.
- **Strategic negotiation**:
  - Agents make and accept/reject offers
  - **Alternating offers (AO)** with discount factors $\Rightarrow$ last agent doesn't have advantage
  - Agree at first step $\rightarrow$ max. joint return.
  - Issue: first agent $\rightarrow$ get a bigger share
- **Axiomatic negotiation**:
  - fix a set of axioms s.t. solution is unique
  - negotiate according to a protocol that guarantees the axioms.
  - u = payoff(A), v = payoff(B)
  - u_*, v_* = minimal payoff (without coop.)
  - $\bar{u}, \bar{v}$ = payoff in case of negotiation
  - **Pareto-Optimality**: there is no other feasible pair (u, v) which is better for both
- **Nash Bargaining Solution**:
  - players demand a portion of some good
  - If sum <, both players get their request
  - if sum >, neither player gets their request
  - max. the product of surplus utilities (NE):
    $$(\bar{u}, \bar{v}) = \sup_{u,v}(u - u_*)(v - v_*)$$
- NE with AO $\rightarrow$ follow framework
  - goal, a worth and cost assign to each goal.
  - Expected utility:
    $$u_i(D_j) = [\sum_{g \in G(D_j)} w_i(g)] - c_i(D_j)$$
- **Monotonic concession protocol**:
  - AO where offers from each agent must $\uparrow$
  - agent with most to lose (high risk) with failure makes next concession
  - risk tolerance of agent i:
    $$risk_i = \frac{u_i(D_i) - u_i(D_j)}{u_i(D_i) - u_i(D_{worst})}$$
    A_i rejects offer D_j and proposes D_i
  - Maximizes product of utility gains and converges towards Nash solution.
- **Task Allocation** using the same framework:
- Cost for computing join
- **Stackelberg games**: a *leader* and a *follower*.

7.1 Social choice (SC)

1. F always returns a result
2. Making d_j more preferable $\forall A_i$ cannot make it less preferable in F
3. If a change doesn't affect relative order of a subset $\alpha = \{d_{i1}, d_{i2}, \ldots\}$, can't change the relative order of choices in α in F.
4. F is surjectif and there is no dictator

- **Arrow's Theorem**: For $k \geq 3$ and $n \geq 2$, there is no SC satisfying all 5 axioms.
- **Mechanisms**: Map agent actions to outcome
- **Individual rationality**: Agent should be better off participating than not.
- Difficulty: true agent utilities are private
- **Truthful mechanisms**: best strat' $\equiv$ truth
- **Revelation principle**: For any mechanism, there is a truthful mechanism with the same outcome and payments.
- **Random dictator** is a truthful mechanism
- **Incentive-compatibility**: when the interaction is structured so that the participant with more information is motivated to act in the interest of the other party (or has less incentive to exploit an informational advantage), the result is incentive compatibility
- **(VCG)**: pay a tax punishing for the damage done to others $\Rightarrow$ truthful
- $\Rightarrow$ waste the tax, not pareto-efficient, hard to apply to large problems, collusion.
- Affine maximizer:
  $$f = \underset{d \in D' \subset D}{\mathrm{argmax}}\, (c_d + \sum_i w_i v_i(d))$$
- **Groves mechanism**: same as VCG tax but return some of the tax payments to the agents
- **Median rule** $\Rightarrow$ never profitable for A_i to not be truthful

8 Auctions

- Forward au.: auc. = seller, highest bid wins
- Reverse au.: auc. = buyer, lowest bid wins
- **Optimal allocation** (Pareto efficiency): resources end up to who value them the most
- Different auction protocols:
  - **Dutch** : Auctioneer continuously lowers price until bidder takes it
  - **English**: Bidders raise their bids until nobody is willing to go any higher
  - **Discriminatory**: Bidders submit one secret bid, item is sold to the highest
  - **Vickrey**: Bidders submit one secret bid, item is sold to the highest bidder, but at the price of the 2nd-highest bid.
- **Optimal bid**: just enough to win
- **Collusion**: buyers coordinate their bidding
- **Information extraction**: extract private information for use in decision mechanisms.

8.1 Auction Platform

- **Open-cry**: continously changing state, hard to publish information at the same time.
- Timing impossible to guarantee $\leftarrow$ internet
- Bids must be *authentic* and *confidential*
- Solution: **Interagents**
  - Can identify registered bidders
  - Cryptographic protocol in between
  - Interagent can handle timing functions

8.2 Multi-unit auctions

- **Uniform price auction**:
  n units for sale $\Rightarrow$ pay $n + 1$st highest bid
- **Multi-unit Vickrey auction**: Each agent pays price of the bid it displaced from the set of winning bids $\rightarrow$ significantly lower revenue.
- Still susceptible to manipulation

8.3 Double auctions

- Both buyers and sellers make bids
- **Clearing double auctions**:
  all buy bids $\geq$ sell price $\leq$ M-th highest
- M-th price is IC for sellers but not for buyers
- **McAffee auction**: Price = average of last sell/buy combination $\Rightarrow$ IC

8.4 Bidding strategies

- Truthful protocols $\Rightarrow$ truth is dominant strat'
- Nash equ.: agent can't do better given the others' strat'

8.5 Combinatorial valuations

- Buyers are looking for combinations of items
- **Sequential a.**: auction each item individually
  - bid depends on outcomes of other auctions
  - **Perceived-price bidder**: bid for most profitable comb. based on perceived prices
  - **Straightforward bidding**: bid for all items in most profitable combination
  - **Sunk-aware bidding**: Take into account what bids have already been won. Items already won cannot be returned, so thier price is discounted by factor k.
  - **Price prediction**: predict final prices and select most profitable comb. to bid for.
- **Combinatorial auctions**:
  - Bidders place bids for comb. of items
  - Auctioneer decides on best comb. of bids
  - determining the winning comb. is NP-hard
  - **Generalized Vickrey Auction (GVA)**: Agent pays for difference between best allocation without its bids and best allocation with its bids. Manipulate with fake bidder. $\Rightarrow$ Non Truthful
- **Generalized Second-price auction**: is not IC, not VCG, but prices are more stable.

9 Coalitions and Group Decisions

- **Coalitions**: groups cooperate to optimize utility and use SC to agree on joint decisions.
- **Coalition stability**: coalition N is stable if $\nexists S \subset N$ gives higher utility $\forall A \in S$ than in N
- **SuperAdditive game**: $\forall S, T \in N$ if $S \cap T = \varnothing \rightarrow v(S \cup T) \geq v(S) + v(T)$
- **Convex** games have nonempty core $\forall S, T \in N, v(S \cup T) \geq v(S) + v(T) - v(S \cap T)$
- **Payoff distr.**: how distribute the rewards
- The set of payoff distributions for what the grand coalition is stable is called the **core**
- **Shapley value (SV)**: expected distribution of returns of the game. Unique and in the core.
- **Carrier**: minimal coalition s.t. result is always completely decided by these agents
- Game with efficient SV computation:
  - **Weighted graph**: agents contribute to coalitions either individually or in pairs. SV is the sum of edge weights in subgraph
  - **Marginal contribution nets**: contribution can be in larger groups

9.1 Group decision making

- **Majority voting**: 2 alternatives, agents vote for favourite. Always CW.
- **Majority graph**: directed edge from d_i to d_j $\rightarrow$ majority prefers d_i over d_j.
- **Condorcet winner (CW)**: d_{CW} beats or ties all others pairwise majority. CW $\rightarrow$ PO
- **Plurality voting**: vote for one alternative, order alternatives by number of votes.
- **Borda count**: voters rank options
- **Slater rank.**: among all possible rankings, choose the closest to majority graph.
- **Kemeny Score**: sum of the win pair-wise against other candidates in the orders given by the agents.