CS 486/686
Recap, Modern-AI Landscape & Exam Prep

Yuntian Deng

Lecture 24

Putting the term together: search, uncertainty, decisions, learning

The course in one picture

Search I — uninformed

Problem formulation

- States, actions, transition model, initial & goal states, path cost.
- Search tree explores states by expanding the **frontier**.

Pick the data structure for the frontier → pick the algorithm.

DFS vs BFS vs IDS

- **DFS:** stack → $O(bm)$ space, not complete.
- **BFS:** queue → complete, $O(b^d)$ time and space.
- **IDS:** DFS in increasing depth → BFS-completeness, DFS-space.

IDS combines the best of both.

Search II — heuristic + A*

The three frontier rules

- **LCFS:** min $\mathrm{cost}(n)$.
- **GBFS:** min $h(n)$.
- **A*:** min $f(n) = \mathrm{cost}(n) + h(n)$.

Heuristic properties

- **Admissible** ($h \leq h^*$) ⇒ A* is optimal with multi-path pruning off.
- **Consistent** ($h(n) \leq c(n, n') + h(n')$) ⇒ safe to use multi-path pruning.
- Consistent ⇒ admissible (not the other way around).

Search strategy summary

Strategy	Frontier choice	Complete?	Space	Time
Depth-first	Last added	No	Linear	Exp
Breadth-first	First added	Yes	Exp	Exp
Lowest-cost first	min $\text{cost}(n)$	Yes	Exp	Exp
Greedy best-first	min $h(n)$	No	Exp	Exp
A*	min $\text{cost}(n) + h(n)$	Yes	Exp	Exp

A* dominates when the heuristic is informative enough to prune most of the tree.

CSPs + local search

Constraint satisfaction

- Generate-and-test is exponential — exploit problem structure.
- **Backtracking:** incremental, prune as soon as a constraint fails.
- **Arc consistency / AC-3:** shrink domains before/during search; complexity $O(cd^3)$.

Local search

- Trade completeness for very large state spaces.
- **Greedy descent / hill-climbing:** always move to best neighbour.
- Beats local minima with **random restarts** + **simulated annealing** (accept worse moves with prob $e^{-\Delta/T}$).

Probability + independence

The four rules

- **Product:** $P(A, B) = P(A \mid B)\, P(B)$
- **Sum:** $P(A) = \sum_b P(A, B{=}b)$
- **Chain:** $P(X_1, \ldots, X_n) = \prod_i P(X_i \mid X_{<i})$
- **Bayes:** $P(H \mid e) \propto P(e \mid H)\, P(H)$

Independence

- **Unconditional:** $P(A, B) = P(A)\, P(B)$.
- **Conditional:** $P(A, B \mid C) = P(A \mid C)\, P(B \mid C)$.
- Tells us which terms drop out of the chain rule — the key to scaling up inference.

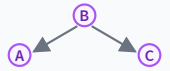
Bayesian networks + d-separation

Construction recipe

- Pick a variable order (causal often works best).
- Add each node; choose the smallest set of parents that makes it conditionally independent of the rest.
- Joint factorizes as $P(X_1, \ldots, X_n) = \prod_i P(X_i \mid \mathrm{pa}(X_i))$.

3 d-separation patterns

Chain $A \to B \to C$: blocked by observing B.



Common cause $A \leftarrow B \to C$: blocked by observing B.

Common effect $A \to B \leftarrow C$: blocked by *not* observing B or descendants.

A path is blocked ⇒ the endpoints are conditionally independent.

Inference — VE & HMM filtering

Variable elimination

- **Define** a factor for each CPT.
- **Restrict** factors to evidence.
- **Multiply** factors sharing a variable; **sum out** the hidden one.
- **Normalize** to get a posterior.

HMM filtering

- Recursion: $P(S_t \mid o_{0:t}) \propto P(o_t \mid S_t) \sum_{s_{t-1}} P(S_t \mid s_{t-1}) P(s_{t-1} \mid o_{0:t-1})$
- One forward pass: $O(t \cdot |S|^2)$ time.

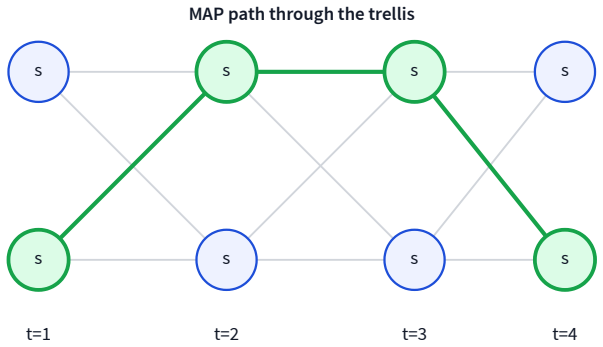
HMM decoding — smoothing & Viterbi

Forward×backward smoothing

- Forward: $\alpha_k = P(S_k \mid o_{0:k})$.
- Backward: $\beta_k = P(o_{k+1:t-1} \mid S_k)$.
- Posterior: $P(S_k \mid o_{0:t-1}) \propto \alpha_k \, \beta_k$.

Viterbi

- DP over the trellis, max instead of sum.
- Backtrack the argmax pointers to recover the MAP sequence.



Decision networks

Three node types

- **Chance** nodes — like a Bayes net (CPTs).
- **Decision** nodes — rectangles, no CPT, chosen by us.
- **Utility** node — diamond, deterministic function of its parents.

Maximum expected utility

- Policy $\pi(d \mid \text{parents})$; MEU picks π^* maximizing $\mathbb{E}[U]$.
- Evaluate by enumeration or by running VE on the DN.
- **Value of information:** EU gain from observing a variable before deciding.

Markov decision processes

Setup

- States, actions, transition $P(s' \mid s, a)$, reward $R(s)$, discount γ.
- Policy $\pi : S \to A$; value $V^\pi(s) = \mathbb{E}[\sum_t \gamma^t R(s_t) \mid \pi]$.

Bellman + the two algorithms

- **Bellman optimality:** $V^*(s) = R(s) + \gamma \max_a \sum_{s'} P(s' \mid s, a)\, V^*(s')$.
- **Value iteration:** apply the Bellman update until convergence.
- **Policy iteration:** evaluate (linear system) $\leftrightarrow$ improve (greedy) — converges fast.

Reinforcement learning

Two learning regimes

- **Passive ADP:** follow a fixed policy, estimate $\hat{P}$, $\hat{R}$, solve via PE.
- **Active ADP:** also pick actions — balance exploration vs exploitation (ε-greedy, softmax, optimistic init).

Model-free updates

- **Q-learning** (off-policy): $Q(s,a) \leftarrow Q(s,a) + \alpha\big[R + \gamma \max_{a'} Q(s',a') - Q(s,a)\big]$.
- **SARSA** (on-policy): use the action actually taken instead of $\max_{a'}$.
- ADP converges fast but needs a model; Q-learning is slower but model-free.

Supervised + unsupervised learning

Supervised

- **Model + loss + fit:** pick $h_{\mathbf{w}}$, minimize a loss by gradient descent.
- **Linear** (squared error) & **logistic** regression (cross-entropy + softmax).
- **Generalization:** over/underfitting, cross-validation, regularization (double descent).

Unsupervised & representations

- **k-means:** alternate assign $\leftrightarrow$ recompute centroids; pick k via elbow / silhouette.
- **PCA:** project onto top eigenvectors of the covariance matrix; maximize variance.
- **Autoencoders** and **self-supervised / contrastive** (CLIP) representations.

Decision trees + neural networks

Trees & ensembles

- Greedy top-down split (**ID3**) maximizing **information gain**; or Gini / **CART**.
- Control overfitting with **pruning**.
- **Ensembles:** random forests (bagging) & gradient boosting — the tabular winners.
- Interpretable, great for small tabular data.

Neural networks

- Weighted sums + non-linear activations (ReLU, GELU); MLPs and **CNNs**.
- **Forward pass** computes loss; **backprop** computes gradients via chain rule (δ recursion).
- Train with **SGD** + momentum / **Adam(W)**; regularize (dropout, norm).
- Black box but unmatched for images, audio, text.

Transformers & large language models

Attention (L21)

- **Scaled dot-product:** $\mathrm{softmax}\left(QK^\top/\sqrt{d_k}\right)V$ — every token reads every token.
- **Multi-head** + **positional encodings**; a block = attention + FFN + residual + norm.
- Parallel, long-range, and scales — it replaced RNNs.

LLMs (L22)

- **Autoregressive:** $\prod_t p\left(x_t \mid x_{<t}\right)$, next-token cross-entropy, at scale.
- **Fine-tuning** (SFT) then **alignment** (RLHF / DPO — the RL of L15).
- In-context learning; systems: **agents, tools, RAG**.

Generative AI: diffusion & multimodal

Diffusion (L23)

- Fixed **forward** noising; learned **reverse** denoiser $\varepsilon_\theta(x_t, t)$.
- Loss $\|\varepsilon - \varepsilon_\theta(x_t, t)\|^2$; sample noise → image.
- **Latent diffusion** + guidance → text-to-image.

Families & multimodal

- Autoregressive / VAE / GAN / diffusion — different ways to model $p(x)$.
- **VAE:** ELBO + reparameterization.
- **CLIP** ties images and text in one embedding space.

Notation cheat sheet

Symbols that recur across the four modules. Watch for the two collisions on the right.

Symbol	Meaning	First seen
γ	Discount factor for future rewards	L13 (MDPs)
$\pi, \pi^\star$	Policy / optimal policy	L12–L15
$V(s),\ Q(s,a)$	State / state-action value functions	L13–L15
$H(p),\ IG$	Entropy / information gain ($IG = H_{\text{before}} - H_{\text{after}}$)	L6 (entropy) · L18 (IG)
Σ	Covariance matrix (PCA)	L17
Q, K, V	Query / key / value in attention	L21
ε_θ	Noise-prediction network (diffusion)	L23
α	RL TD step size (L15); learning rate in SGD (L20) — *different meanings*	L15 / L20
ε	Exploration prob in ε-greedy (L15); vs the noise variable ε in diffusion (L23)	L15 / L23

Worth a row on your two-sided study sheet.

How it fits together — and where it's going

Modern AI systems **compose** everything in this course: search + planning, probabilistic reasoning, decision-making under uncertainty, and deep learning — wrapped around large pretrained models.

Agents

LLMs that plan, call tools, and act — search & decisions, revisited.

Tool builders

Turning powerful models into reusable, specialized neural software (my *programasweights* direction).

Generative UIs

Interactive world models like NeuralOS — systems you generate, not just program.

Want more? CS480/680 (ML) and CS485 go deeper; research is moving fast.

Final exam — logistics

What to bring

- Non-programmable calculator.
- One two-sided A4 study sheet (handwritten or typed).

Format (tentative)

- ~8 problems across all four modules; ~84 marks (76+ = 100%); pass at 42.
- L21–L23 (transformers / LLMs / diffusion) examinable at the **conceptual** level — no heavy derivations.

Exact details confirmed on the course page before exam day.

Thanks for a great term — good luck on the final!