MSAI3105 Exam Preparation: Important Topic Clusters with Explanations

Lecture 1 – Introduction to AI

• Definitions of AI: systems that think like humans, think rationally, act like humans, or act rationally.

• Turing Test: A machine is intelligent if it can mimic human responses well enough to fool an interrogator. Limitations include bias, reproducibility, and lack of mathematical rigor.

• Rational Agents: Entities that maximize performance measures given percepts. Rationality ≠ omniscience or clairvoyance.

• History of AI: Dartmouth Conference (1956), Samuel's checkers program, expert systems, AI winter.

• Fields influencing AI: Philosophy, mathematics, psychology, linguistics, neuroscience, economics.

• Narrow AI vs. General AI: Narrow = specific tasks; General = human-level intelligence.

• Risks and Benefits: Autonomous weapons, bias, unemployment, cybersecurity threats vs. efficiency, research acceleration, productivity.

Lecture 2 – Agents & Problem Solving

• Agent: perceives environment (sensors), acts (actuators).

• PEAS Framework: Performance measure, Environment, Actuators, Sensors.

• Agent Types: Simple reflex, Model-based, Goal-based, Utility-based, Learning agents.

• Environment Properties: Observable/partially observable, deterministic/stochastic, episodic/sequential, static/dynamic, discrete/continuous, single/multi-agent.

• Examples: Vacuum world, automated taxi agent, internet shopping agent.

• Problem formulation in search: initial state, goal state, actions, transition model, path cost.

• Romania example: Travel from Arad to Bucharest.

Lecture 3 – Search Strategies & Heuristics

• Evaluation criteria: Completeness, Optimality, Time, Space.

• BFS: Complete, optimal (equal step costs), $O(b^d)$.

• DFS: Incomplete in infinite spaces, not optimal, $O(bm)$.

• Uniform Cost Search (UCS): Handles varying costs, uses priority queue by path cost.

• Dijkstra's Algorithm: Expands lowest cost nodes first.

• Heuristics: Estimates of distance to goal. Admissible if they never overestimate; consistent if they satisfy triangle inequality.

• Examples: Manhattan distance, misplaced tiles for puzzles.

• Greedy Best-First Search: Expands lowest heuristic cost $h(n)$.

• A* Search: $f(n) = g(n) + h(n)$, complete and optimal with admissible heuristics.

Lecture 4 – Advanced Search & Local Optimization

• Designing heuristics: From relaxed problems (removing constraints) or subproblems (pattern databases).

• Dominance: If h2 ≥ h1 everywhere, h2 dominates.

• Local Search: Optimization where path is irrelevant (e.g., TSP, N-Queens).

• Hill Climbing: Iteratively improve state; risks = local maxima, plateaus.

• Variants: Random restart, sideways moves.

• Simulated Annealing: Accepts worse moves probabilistically (P = e^(–ΔE/T)) to escape local maxima.

• Genetic Algorithms: Inspired by evolution. Use selection, crossover, mutation on encoded states.

• Applications: Scheduling, layout problems, large optimization tasks.

Lecture 5 – Game Playing

• Games in AI: Models of competition/cooperation (chess, backgammon, poker).

• Game types: Deterministic/stochastic, perfect/imperfect info, zero-sum/non-zero-sum.

• Minimax Algorithm: Assumes optimal play by both players, computes best move for MAX.

• Alpha-Beta Pruning: Cuts off branches that cannot influence outcome. Reduces complexity to O(b^(m/2)) with perfect ordering.

• Evaluation Functions: Approximate value of states at limited depth (e.g., chess piece values).

• Horizon Effect: Incorrect evaluation when important moves lie beyond cutoff depth.

• Quiescence Search: Extends unstable positions beyond horizon.

• Historical Systems: Chinook (checkers, 1994), Deep Blue (chess, 1997), Hydra (advanced pruning), Go challenges.

• Nondeterministic games: Expectiminimax for chance events (dice, cards).

Lecture 6 – Constraint Satisfaction Problems (CSPs)

• Definition: Assign values to variables from domains subject to constraints.

• Examples: Map coloring, Sudoku, N-Queens, scheduling.

• Types of constraints: Unary (1 variable), Binary (2 variables), n-ary (multiple), Soft (violations allowed at cost).

• Complexity: CSPs and SAT are NP-complete.

• Backtracking Search: DFS assigning variables one by one, undoing invalid assignments.

• Heuristics: MRV (Minimum Remaining Values), MCV (Most Constraining Variable), LCV (Least Constraining Value).

• Forward Checking: Eliminate future domain values that conflict with current assignments.

• Arc Consistency (AC-3): Enforce pairwise consistency across variables.

- Efficiency: Heuristics drastically reduce runtime vs. brute force.