Artificial Intelligence Exam Preparation Guide

Mumbai University | Subject Code: 48893 | Semester V

Document Overview

This comprehensive preparation package contains:

1. Analysis of 6 previous year question papers (2022-2025)
2. Top 30 most important questions with frequency analysis
3. Three practice papers following Mumbai University format
4. One predicted paper for 2026 examination
5. Study strategy, comparison tables, and important algorithms

Paper Analysis Summary

Papers Analyzed:

- December 2022 (QP Code: 10014546)
- May 2023 (QP Code: 10029721)
- December 2023 (QP Code: 10038167)
- May 2024 (QP Code: 10055372)
- December 2024 (QP Code: 10066783)
- May 2025 (QP Code: 10083059)

Total Questions Analyzed: 89 questions across 6 papers
Unique Topics Identified: 36 topics

Exam Pattern

Duration: 3 hours
Maximum Marks: 80

Structure:

- **Q1:** Compulsory - Solve any 4 out of 5 sub-questions (5 marks each) = 20 marks
- **Q2-Q6:** Attempt any 3 questions (2 sub-questions per question)
  - Each sub-question = 10 marks
  - Total from Q2-Q6 = 60 marks

Note: Unlike Computer Network which has "any 3 from Q2-Q6", AI typically asks "any 3 from remaining 5 questions".

Topic-wise Frequency Analysis

Most Important Topics (Top 20):

Rank	Topic	Frequency	Marks
1	AI Agents (Types & Architecture)	9	10 marks
2	PEAS Descriptor	6	5 marks
3	Hill Climbing Algorithm	6	10 marks
4	Genetic Algorithm	5	10 marks

Rank	Topic	Frequency	Marks
5	Forward & Backward Chaining	5	10 marks
6	Environment Types	5	10 marks
7	Expert System	4	5-10 marks
8	Bayesian Belief Network	4	10 marks
9	Alpha-Beta Pruning	4	10 marks
10	Prolog Programming	4	5 marks
11	Bayes Theorem	4	10 marks
12	Iterative Deepening Search	4	10 marks
13	First Order Predicate Logic (FOPL)	4	5-10 marks
14	AI Applications	3	5 marks
15	A* Algorithm	3	10 marks
16	Types of Learning	3	10 marks
17	Resolution Theorem Proving	3	10 marks
18	Partial Order Planning	3	10 marks
19	Total Order Planning	3	10 marks
20	Intelligent Agent	3	5 marks

Question Type Distribution

Q1 (Compulsory) - Common Topics:

- PEAS descriptor (different systems)
- AI perspectives (Acting/Thinking humanly/rationally)
- Expert System architecture
- FOPL conversions
- Prolog programs
- Short definitions (AI, learning types)
- Hill climbing problems
- Supervised vs unsupervised learning

Q2-Q6 (10 Marks Questions) - Core Topics:

Agents & Environments:

- AI agents architecture and types
- Environment classification
- PEAS detailed descriptor

Search Algorithms:

- A* algorithm
- Hill Climbing and Simulated Annealing
- Alpha-Beta pruning
- Search strategy comparison (BFS, DFS, IDS)

- Iterative Deepening

Knowledge Representation & Logic:

- Forward and Backward chaining
- Resolution theorem proving
- FOPL statements
- CNF conversion
- Knowledge representation techniques

Optimization & Learning:

- Genetic Algorithm
- Types of learning in AI
- Reinforcement learning

Planning:

- Partial Order Planning
- Total Order Planning

Probability:

- Bayesian Belief Networks
- Bayes Theorem applications
- Conditional probability

Problem Solving:

- 8-Puzzle formulation with A*
- Missionaries and Cannibals problem

High-Priority Algorithm Questions

1. Alpha-Beta Pruning

Always includes numerical example - Practice with different game trees.

Example Problem Structure:

- Given game tree with values at leaves
- First node is MAX
- Show α and β values at each step
- Mark pruned branches
- Find optimal move

2. Resolution Theorem Proving

Common proof scenarios:

Example 1: "Someone is smiling"

- All graduating people are happy
- All happy people smile
- Someone is graduating
- Prove: Someone is smiling

Example 2: "Robert is criminal"

- It's crime to sell weapons to hostile nations
- Country A is hostile
- Robert sold weapons to Country A
- Prove: Robert is criminal

Example 3: "X likes peanuts"

- X likes all food
- Peanuts are food (through transitive properties)
- Prove: X likes peanuts

Steps:

1. Convert to FOPL
2. Convert to CNF
3. Negate goal
4. Apply resolution
5. Derive empty clause ⊥

3. Forward & Backward Chaining

Practice proving "Robert is criminal" with both methods:

- Show derivation tree for forward chaining
- Show goal tree for backward chaining

4. A Algorithm with 8-Puzzle *

Always asked with example:

- Given initial and goal state
- Calculate heuristic (Manhattan distance or misplaced tiles)
- Show g(n), h(n), f(n) for each node
- Expand nodes in order
- Show solution path

5. Genetic Algorithm

Common problem types:

- String matching
- Function optimization
- Show:
  - Initial population
  - Fitness calculation
  - Selection process
  - Crossover operation
  - Mutation
  - Next generation

Important Comparison Tables

1. AI Agent Types

Agent Type	Characteristics	Example
Simple Reflex	Condition-action rules	Thermostat
Model-based Reflex	Maintains internal state	Vacuum with map
Goal-based	Plans to achieve goals	GPS navigation
Utility-based	Maximizes utility function	Autonomous vehicle

Agent Type	Characteristics	Example
Learning	Improves with experience	Spam filter

2. Environment Classifications

Dimension	Types	Chess	Crossword	Taxi
Observable	Full/Partial	Full	Full	Partial
Deterministic	Det/Stoch	Det	Det	Stoch
Episodic	Ep/Seq	Seq	Ep	Seq
Static	Static/Dynamic	Static	Static	Dynamic
Discrete	Disc/Cont	Disc	Disc	Cont
Agents	Single/Multi	Multi	Single	Multi

3. Search Algorithm Comparison

Algorithm	Time	Space	Complete	Optimal
BFS	$O(b^d)$	$O(b^d)$	Yes	Yes*
DFS	$O(b^m)$	$O(bm)$	No	No
IDS	$O(b^d)$	$O(bd)$	Yes	Yes*
Bidirectional	$O(b^{d/2})$	$O(b^{d/2})$	Yes	Yes*
A*	$O(b^d)$	$O(b^d)$	Yes	Yes**

*With unit step cost
**With admissible heuristic

where: b = branching factor, d = depth of solution, m = maximum depth

4. Learning Types Comparison

Type	Data	Feedback	Example	Algorithms
Supervised	Labeled	Labels	Classification	Decision Tree, SVM, NN
Unsupervised	Unlabeled	None	Clustering	K-means, PCA
Reinforcement	Environment	Rewards	Game playing	Q-learning, Policy gradient
Semi-supervised	Partially labeled	Partial	Web classification	Co-training

5. Planning Approaches

Feature	Total Order	Partial Order
Order	Complete linear order	Partial ordering
Flexibility	Rigid	Flexible

Feature	Total Order	Partial Order
Parallelism	Sequential only	Allows parallel actions
Commitment	Early commitment	Least commitment
Efficiency	Less efficient	More efficient
Implementation	Simpler	Complex
Example	Forward state-space search	POP algorithm

PEAS Descriptor Examples

Practice writing PEAS for:

1. Medical Diagnosis System

- **Performance**: Accurate diagnosis, patient satisfaction, cost
- **Environment**: Hospital, patient symptoms, medical history
- **Actuators**: Diagnosis output, treatment recommendations
- **Sensors**: Patient data, test results, symptoms

2. Autonomous Vehicle

- **Performance**: Safety, speed, legality, comfort, efficiency
- **Environment**: Roads, traffic, weather, pedestrians
- **Actuators**: Steering, accelerator, brake, signals
- **Sensors**: Cameras, GPS, radar, lidar, speedometer

3. Online English Tutor

- **Performance**: Student learning, engagement, improvement
- **Environment**: Online platform, student responses, curriculum
- **Actuators**: Lesson content, feedback, exercises
- **Sensors**: Student answers, time taken, progress metrics

Prolog Programs to Practice

1. Factorial

```
factorial(0, 1).
factorial(N, F) :-
    N &gt; 0,
    N1 is N - 1,
    factorial(N1, F1),
    F is N * F1.
```

2. Fibonacci

```
fib(0, 0).
fib(1, 1).
fib(N, F) :-
    N &gt; 1,
    N1 is N - 1,
    N2 is N - 2,
    fib(N1, F1),
    fib(N2, F2),
    F is F1 + F2.
```

3. Family Tree

```
% Facts
parent(tom, bob).
parent(tom, liz).
parent(bob, ann).
parent(bob, pat).
male(tom).
male(bob).
female(liz).
female(ann).

% Rules
father(X, Y) :- parent(X, Y), male(X).
mother(X, Y) :- parent(X, Y), female(X).
grandparent(X, Z) :- parent(X, Y), parent(Y, Z).
sibling(X, Y) :- parent(Z, X), parent(Z, Y), X \\= Y.
```

4. List Operations

```
% Length of list
length_list([], 0).
length_list([_|T], N) :- length_list(T, N1), N is N1 + 1.

% Sum of list
sum_list([], 0).
sum_list([H|T], Sum) :- sum_list(T, Sum1), Sum is H + Sum1.

% Member of list
member_list(X, [X|_]).
member_list(X, [_|T]) :- member_list(X, T).
```

FOPL Conversion Practice

Common Patterns:

Universal Quantification:

- "All X are Y" → $forallx(X(x)rightarrowY(x))$
- "Every X is Y" → $forallx(X(x)rightarrowY(x))$

Existential Quantification:

- "Some X are Y" → $existsx(X(x)landY(x))$
- "At least one X is Y" → $existsx(X(x)landY(x))$

Negation:

- "No X are Y" → $forallx(X(x)rightarrownegY(x))$
- "Not all X are Y" → $negforallx(X(x)rightarrowY(x))$

Complex:

- "Only X are Y" → $forallx(Y(x)rightarrowX(x))$
- "X if and only if Y" → $forallx(X(x)leftrightarrowY(x))$

Important Algorithms - Pseudocode

Hill Climbing:

```
function HILL-CLIMBING(problem)
    current ← MAKE-NODE(problem.INITIAL-STATE)
    loop do
        neighbor ← highest-valued successor of current
        if VALUE(neighbor) ≤ VALUE(current) then
            return current
        current ← neighbor
```

Simulated Annealing:

```
function SIMULATED-ANNEALING(problem, schedule)
    current ← MAKE-NODE(problem.INITIAL-STATE)
    for t = 1 to ∞ do
        T ← schedule(t)
        if T = 0 then return current
        next ← random successor of current
        ΔE ← VALUE(next) - VALUE(current)
        if ΔE &gt; 0 then current ← next
        else current ← next with probability e^(ΔE/T)
```

A Search:*

```
function A-STAR(problem)
    OPEN ← priority queue with initial state
    CLOSED ← empty set
    while OPEN not empty do
        current ← node with minimum f(n) from OPEN
        if current is goal then return path
        add current to CLOSED
        for each neighbor of current do
            if neighbor in CLOSED then continue
            g ← g(current) + cost(current, neighbor)
            h ← heuristic(neighbor)
            f ← g + h
            if neighbor not in OPEN or g &lt; g(neighbor) then
                add/update neighbor in OPEN with f-value
```

Study Schedule Recommendation

Week 1: Agents & Search

- Day 1-2: AI Agents, PEAS, Environment Types
- Day 3-4: Search algorithms (BFS, DFS, IDS)
- Day 5-6: A* algorithm with examples
- Day 7: Hill Climbing, Simulated Annealing

Week 2: Knowledge & Logic

- Day 1-2: FOPL conversions
- Day 3-4: Forward & Backward Chaining
- Day 5-6: Resolution theorem proving
- Day 7: CNF conversion, Inference rules

Week 3: Optimization & Probability

- Day 1-2: Genetic Algorithm
- Day 3-4: Alpha-Beta Pruning
- Day 5-6: Bayesian Networks
- Day 7: Bayes Theorem problems

Week 4: Planning & Practice

- Day 1-2: Partial & Total Order Planning
- Day 3-4: Problem formulation (8-puzzle, M&C)
- Day 5-6: Prolog programming
- Day 7: Types of learning, Expert systems

Week 5: Revision & Practice

- Day 1-2: Practice Paper 1
- Day 3-4: Practice Paper 2
- Day 5-6: Practice Paper 3
- Day 7-8: Predicted Paper 2026
- Day 9-10: Revise top 20 questions
- Day 11-12: Algorithm practice
- Day 13-14: Final revision

Exam Day Tips

Time Management:

- Q1 (20 marks): 35-40 minutes (≈8 minutes per sub-question)
- Each 10-mark question: 25-30 minutes
- Reserve 15-20 minutes for review

Answer Writing Strategy:

For 10-mark questions:

1. Introduction/Definition (1 mark)
2. Main explanation with diagram/algorithm (6-7 marks)
3. Example/Application (2 marks)
4. Conclusion (1 mark)

For 5-mark questions:

1. Definition (1 mark)
2. Explanation (3 marks)
3. Example if applicable (1 mark)

*For algorithms (A, Alpha-Beta, Resolution):**

1. Write algorithm/steps (2-3 marks)
2. Apply on given example (5-6 marks)
3. Show all intermediate steps (1-2 marks)
4. Final answer (1 mark)

Common Mistakes to Avoid

1. ✖ Not drawing diagrams for agents
2. ✖ Incomplete PEAS descriptor (missing any component)
3. ✖ Not showing step-by-step in algorithms
4. ✖ Missing resolution tree in resolution problems
5. ✖ Incorrect FOPL syntax
6. ✖ Not explaining limitations of Hill Climbing
7. ✖ Incomplete Genetic Algorithm explanation
8. ✖ Missing pruning marks in Alpha-Beta
9. ✖ Poor time management on Q1

Quick Revision Checklist

Diagrams to Practice:

- [ ] Simple Reflex Agent
- [ ] Model-based Reflex Agent
- [ ] Goal-based Agent
- [ ] Utility-based Agent
- [ ] Learning Agent
- [ ] Expert System Architecture
- [ ] Resolution Tree
- [ ] Forward Chaining Derivation
- [ ] Backward Chaining Goal Tree
- [ ] Bayesian Network
- [ ] Alpha-Beta Game Tree
- [ ] State Space Graph (8-puzzle, M&C)

Algorithms to Master:

- [ ] A* with example
- [ ] Hill Climbing
- [ ] Simulated Annealing
- [ ] Alpha-Beta Pruning
- [ ] Genetic Algorithm
- [ ] Iterative Deepening
- [ ] Resolution

Must-Know Topics:

- [ ] PEAS for 3-4 different systems
- [ ] Environment classification for 3-4 systems
- [ ] FOPL conversion (10+ examples)
- [ ] Prolog programs (3-4 programs)
- [ ] Forward & Backward chaining proof
- [ ] Resolution proof with tree
- [ ] Bayes theorem calculation
- [ ] Search algorithm comparison table

Last-Minute Tips

1 Day Before Exam:

- Revise all formulas
- Practice one game tree (Alpha-Beta)
- Practice one resolution proof
- Review PEAS examples
- Revise agent diagrams
- Go through comparison tables
- Early sleep

On Exam Day:

- Read all Q1 options first
- Choose 4 easiest sub-questions
- For Q2-Q6, scan all and mark 3 easiest

- Start with your strongest question
- Draw neat diagrams
- Show all steps in algorithms
- Manage time strictly

Good luck with your AI exam preparation!

Remember: AI is about problem-solving and reasoning. Focus on understanding concepts rather than memorizing. Practice algorithms with different examples.