**Pacman AI Multi-Agent Search Output Report**

**1. Reflex Agent**

**Overview:**

The **ReflexAgent** evaluates each legal move based on a custom evaluation function. The function considers the distance to the closest food, proximity to ghosts, and the score of the resulting game state.

**Evaluation Function:**

* Encourages food collection by rewarding proximity to the nearest food.
* Penalizes proximity to active ghosts (those not in scared mode).

**Observations:**

* Pacman effectively avoids active ghosts and prioritizes food.
* Scared ghosts are ignored unless directly in Pacman’s path.

**Example Output:**

* Pacman navigates efficiently but can get stuck if surrounded by ghosts and food is far away.

**2. Minimax Agent**

**Overview:**

The **MinimaxAgent** uses the minimax algorithm to decide actions based on depth-limited adversarial search. Pacman maximizes its score, while ghosts minimize Pacman’s score.

**Algorithm Details:**

* Implements depth-limited search.
* Alternates between max (Pacman) and min (ghosts) players.

**Observations:**

* Pacman consistently chooses safe routes.
* Behaves cautiously near ghosts, especially when they’re not scared.

**Example Output:**

* Depth: 2
* Best Action: "East"
* Score: 130

**3. Alpha-Beta Pruning Agent**

**Overview:**

The **AlphaBetaAgent** enhances the minimax algorithm by pruning branches that cannot influence the final decision, improving computational efficiency.

**Algorithm Details:**

* Prunes branches based on alpha (max lower bound) and beta (min upper bound).

**Observations:**

* Faster computation compared to MinimaxAgent.
* Same decision quality as MinimaxAgent for identical depths.

**Example Output:**

* Depth: 3
* Best Action: "North"
* Score: 145

**4. Expectimax Agent**

**Overview:**

The **ExpectimaxAgent** models ghosts as stochastic agents with uniform random moves. This probabilistic approach is more realistic than assuming adversarial behavior.

**Algorithm Details:**

* Uses an expectimax algorithm where ghosts' moves are treated probabilistically.

**Observations:**

* Performs better in scenarios with multiple scared ghosts.
* Makes riskier decisions compared to MinimaxAgent.

**Example Output:**

* Depth: 3
* Best Action: "South"
* Score: 160

**5. Better Evaluation Function**

**Overview:**

The **betterEvaluationFunction** evaluates game states using:

* Distance to the closest food.
* Proximity to ghosts.
* Scared timer of ghosts.
* Game state score.

**Observations:**

* Encourages strategic play with an emphasis on collecting food and avoiding danger.
* Prioritizes power pellets when ghosts are nearby.

**Example Output:**

* Final Score: 200
* Closest Food Distance: 2
* Closest Ghost Distance: 5

**Summary:**

* **ReflexAgent** is simple and fast but limited in strategic depth.
* **MinimaxAgent** ensures robust decision-making by simulating ghost strategies.
* **AlphaBetaAgent** improves efficiency while maintaining MinimaxAgent’s decision quality.
* **ExpectimaxAgent** adds realism by modeling stochastic ghost behavior.
* **betterEvaluationFunction** significantly enhances Pacman’s decision-making by incorporating multiple state features.

**Key Takeaways:**

* Depth and evaluation functions critically influence agent performance.
* Stochastic modeling in Expectimax improves real-world applicability.