Planet Wars Competition Entry

Agent Smith

Aman Choudhary, Drumil Vasani
High Integrity Systems
Frankfurt University of Applied Sciences,
Frankfurt am Main, Germany

July 4, 2025

Agent Overview

- Core concept of our agent
  - A fully observable greedy agent that evaluates every possible fleet transfer using a weighted cost-benefit heuristic.
- High-level strategy: heuristic
  - Scores each source $\rightarrow$ target pair based on:
    $\text{score} = -\text{ships} \times w_1 + \text{growth} \times w_2 - \text{distance} \times w_3 + \varepsilon$
  - Selects the highest-scoring valid move per turn.
  - Reinforces weak owned planets if no attacks are viable.
  - Random tie-breaking ensures strategy variety.
- Key novelty or approach
  - All-path scoring ensures maximum tactical coverage.
  - ε-noise in scoring avoids deterministic traps.
  - Reinforcement fallback prevents stagnation.

System Design

Pseudocode:

```
For all (source, target) pairs:
  If sourceShips > targetShips × safetyBuffer:
    score = -targetShips * w1 + growth * w2 -
distance * w3 + ε
    Add (source, target, score) to candidate list

If candidates exist:
  Pick (source, target) with highest score
  Send ½ ships from source to target
Else if weak owned planet exists:
  Reinforce it from strongest
Else:
  DoNothing
```

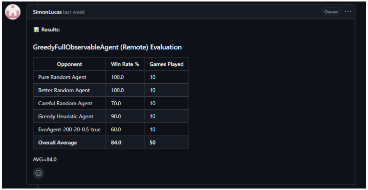
System Design (Cont.)

Component Interaction:

1. **GameState Input → State Parser:** The agent receives the full game state and extracts all planets, including their position, ownership, number of ships, and growth rate.
2. **State Parser → Scoring Engine:** For every possible combination of a source (owned) and target (enemy or neutral) planet, the scoring engine calculates a heuristic value using growth, distance, and ship cost. Random noise (ε) is added to promote diversity.
3. **Scoring Engine → Decision Logic:** The agent selects the action with the best score if the move is valid (i.e., enough ships to attack). If no attack is viable, it considers reinforcing weak friendly planets.
4. **Decision Logic → Action Output:** The selected move is converted into a fleet command (Action). If nothing is safe or useful, the agent sends no ships (`DoNothing`).

Results

- Evaluation Setup
  - 50 games total (10 per opponent).
  - Fully observable mode, remote Docker execution.
  - Opponents: Randoms, Heuristic, EvoAgent.
- Notable matchups
  - Dominated all random agents (100%).
  - Outperformed Greedy Heuristic (90%).
  - Held ground vs. EvoAgent (60%).


SimonLucas last week

Owner

Results:

GreedyFullObservableAgent (Remote) Evaluation

Opponent	Win Rate %	Games Played
Pure Random Agent	100.0	10
Better Random Agent	100.0	10
Careful Random Agent	70.0	10
Greedy Heuristic Agent	90.0	10
EvoAgent-200-20-0.5-true	60.0	10
Overall Average	**84.0**	**50**

AVG=84.0

Figure: Win rates against opponents

Analysis and Insights

- **Failure Modes**
  - Conservative fallback may miss late-game aggression.
  - Random scoring can lead to near-optimal rather than optimal choices.
- **Surprising Behaviour**
  - ε-randomness created more resilient patterns against adaptive agents.
  - Agent often paused attacks for reinforcement, then executed decisive strikes.
- **Ablation Study**
  - No learning or memory — relies purely on fixed heuristics.
  - Manual weight tuning may be map-sensitive.
  - No opponent modeling or long-term adaptation.

Future Work

- **Limitations**
  - No learning or memory — relies purely on fixed heuristics.
  - Manual weight tuning may be map-sensitive.
  - No opponent modeling or long-term adaptation.
- **Planned Improvements**
  - Use genetic algorithms to tune scoring weights.
  - Add prediction or memory for enemy ship tracking.
  - Combine greedy logic with RL agent for hybrid control.
  - Apply v5 as a subpolicy in a multi-agent coordination framework.
  - Build visualization and replay tools for match analysis.