Baby Tetris

ERGIN Seçkin Yağmur and GROUSSON Dylan

January 11, 2026

Introduction

Project Goal

This project explores optimal strategies for playing "Baby Tetris" using Markov Decision Processes (MDPs).

1. **Player's Optimal Policy**: Find the best strategy for a player assuming a random opponent.
2. **Adversarial Policy**: Design an opponent that actively tries to minimize the player's score.
3. **Robust Player Policy**: Develop a player strategy that performs well against any type of opponent.

Player's Discounted MDP Model

MDP Components

- **States**: $S :=$ (current grid, next piece to place)
- **Actions**: $A :=$ (position, orientation)
- **Transition**: $P(s'|s,a) := 1/2$ (uniform distribution for I or L piece)
- **Reward**: $R(s,a,s') :=$ coeff $\times$ lines cleared

Value Function (Bellman Equation)

The optimal policy is found using value iteration to solve for $V(s)$:

$$V(s) = \max_{a \in A} \sum_{s' \in S} P(s'|s,a) \cdot \left(R(s,a,s') + \lambda V(s')\right)$$

Implementation Choices

- **Pieces teleported**: To reduce intermediate states, pieces are placed directly in their final position and orientation. This simplifies the state space.
- **Reward Calculation**: The reward is calculated based on the number of completed lines in the state *after* a piece is placed.
- **Game Ending**: The game terminates when there are no available (i.e., valid) actions for the current state.
- **Max Iteration Bound**: A maximum number of iterations is set for the value iteration algorithm. This ensures termination even if the convergence to the specified ϵ is slow.

Optimization: Reachable States Algorithm

Algorithm 1 Generate Reachable States

```
Q: a queue; H: a hashMap
s ← s_0; Q.push(s)
while Q.isNotEmpty() do
    s ← Q.pop()
    for a: availableActions(s) do
        s_After ← applyAction(s, a).completeLines()
        if Q, H do not contain s_After then
            Q.push(s_After)
        end if
    end for
    H.push(s)
end while
return H
```

Executions (With Reachable States)

```
i=0, delta=6.10288
i=1, delta=0.352625
i=2, delta=0.0250445
i=3, delta=0.0010013
i=4, delta=4.28143e-05
i=5, delta=1.79071e-06
i=6, delta=4.8676e-08
i=7, delta=1.07541e-09

Score: 7632
in 10000 actions
--------------------
Exec in    7.69 secs
```

height: 4 and λ: 0.1

```
i=0, delta=8.36976
i=1, delta=5.32365
i=2, delta=3.54512
...
i=87, delta=1.33224e-08
i=88, delta=1.0733e-08
i=89, delta=8.64692e-09

Score: 11118
in 10000 actions
--------------------
Exec in   26.53 secs
```

height: 4 and λ: 0.9

```
i=0, delta=6.20552
i=1, delta=0.477762
i=2, delta=0.0334845
i=3, delta=0.00181086
i=4, delta=6.29689e-05
i=5, delta=2.6382e-06
i=6, delta=1.32337e-07
i=7, delta=6.63914e-09

Score: 7620
in 10000 actions
--------------------
Exec in   46.73 secs
```

height: 5 and λ: 0.1

```
i=0, delta=9.7563
i=1, delta=6.7369
i=2, delta=4.43912
...
i=88, delta=1.44415e-08
i=89, delta=1.16674e-08
i=90, delta=9.42616e-09

Score: 13305
in 10000 actions
--------------------
Exec in  402.82 secs
```

height: 5 and λ: 0.9

Adversarial MDP Model

Adversary MDP

- **Actions**: $A_{adv} := \{I, L\}$ (Choice of the next piece).
- **Transition**: $P(s'|s,a,t) := 1/|A_t(s)|$, assuming a uniform player response.

Adversary Value Functions

- **Lowest Maximal Reward (Min-Max)**:

$$V(s) = \min_{t \in \{I,L\}} \max_{a \in A_t(s)} \sum_{s'} P(s'|s,a,t) \cdot \big(R(s') + \lambda V(s')\big)$$

- **Lowest Average Reward (Min-Avg)**:

$$V(s) = \min_{t \in \{I,L\}} \frac{1}{|A_t(s)|} \sum_{a \in A_t(s)} \sum_{s'} P(s'|s,a,t) \cdot \big(R(s') + \lambda V(s')\big)$$

Robust Policy Model

Approach 1: Max-Min Value Iteration

Player maximizes their reward assuming the adversary makes the worst possible choice for them.

$$V(s) = \max_{a \in A} \min_{t \in \{I, L\}} P(s'|s, a, t) \cdot \big(R(s, a, s') + \lambda V(s')\big)$$

Approach 2: Parameterized Reward Function

Adjust weights for different reward components to tune behavior.

$$R(s') := \mathrm{lw} \cdot \mathrm{lines} - \mathrm{hw} \cdot \mathrm{height} + \mathrm{sw} \cdot \mathrm{score} - \mathrm{gr} \cdot \mathrm{gaps}$$

Results and Observations

L.W.	H.W.	S.W.	G.R.	Rand	MinMax	MinAvg	GapAvg	Min
Max-Min VI (Fixed Parameters)				10685	9999	9999	10356	9999
0	0	3	0	11090.70	9999	10000	10622	9999
0	0	1	1	10837.25	9999	10000	9999	9999
10	0	3	1	9833.09	9998	11248	7500	7500
10	0	4	0	10611.33	9998	10000	7500	7500
0	1	4	1.5	9810.65	9999	10000	9999	9810.65
0	2	1	1	0.05	0	0	0	0

Observations

- Robustness vs. High Score
- Score and Line Weights Matter
- Max-Min VI Performance
- Variability

Conclusion

- Developed MDP models and value iteration for player, adversary, and robust Baby Tetris policies.
- Improved efficiency with reachable states, enhancing policy analysis.
- Robust policies provide strong performance against various adversaries.
- Key drivers for good policy: prioritize line clears/score and avoid height penalties.