Solving the Dice Game Pig:
an introduction to dynamic programming and value iteration

Todd Neller

Gettysburg College

This work was sponsored in part by NSF DUE CCLI-A&I Award Number 0409497.

Pig

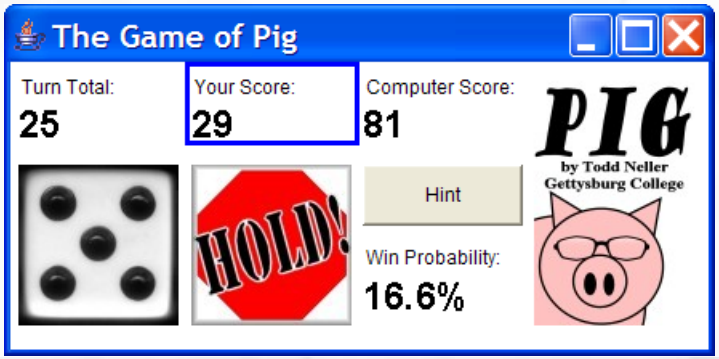
- One of the most fun, extremely simple dice games
  - "Extremely simple" – described in two sentences
  - "Fun" – spawned commercial variants, e.g. Pass the Pigs (a.k.a. Pigmania, 1977)
- High Fun-to-SLOC (source lines of code) ratio
- Simple examples are teaching treasures
  - Currently, Pig is used extensively by math educators and is increasingly being used by CS educators.

Outline

- Pig Rules
- What the optimal player optimizes
- Dynamic programming approximation
  - Exercises
- Value iteration solution
  - Exercises
- Conclusion


Pig Rules

- The first player reaching **100** points wins.
- On each turn, a player rolls a die as many times as desired until either the player **holds** and scores the **sum of the rolls**, or **rolls a 1** and scores **nothing**.



Maximizing Score

- A play maximizing score will roll if the expected gain exceeds expected loss:
  - 5/6 gain: average of 2…6 = 4
  - 1/6 loss: turn total *k*
  - 5/6*4 = 1/6**k* → *k* = 20
  - Hold at 20 maximizes expected score per turn
- When might a player not want to hold at 20? Why?
- Playing to score is not playing to win.

Optimality Equations

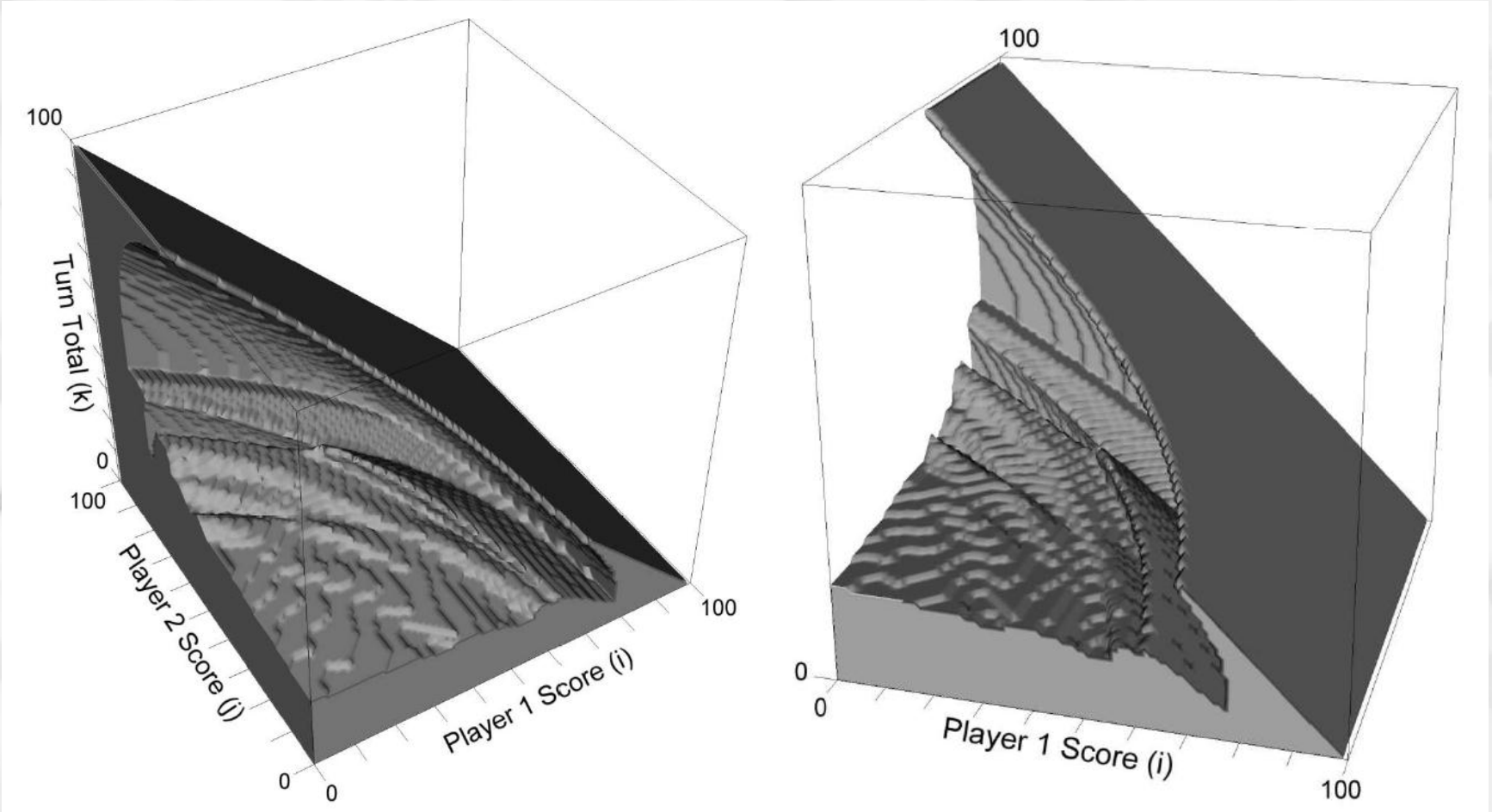
- i = player score, j = opponent score, k = turn total

$$P_{i,j,k} = \max\left(P_{i,j,k,roll}, P_{i,j,k,hold}\right)$$

$$P_{i,j,k,roll} = \frac{1}{6}\left((1 - P_{j,i,0}) + \sum_{r=2}^{6} P_{i,j,k+r}\right)$$

$$P_{i,j,k,hold} = 1 - P_{j,i+k,0}$$


Optimal Play



Dynamic Programming

- Fibonacci example
- Progressive Pig: score at least 1 per turn → acyclic state space
- Worked example using Knuth's literate programming style, e.g.

⟨*Compute the probability of winning with optimal play*⟩≡

```
public double pWin(int i, int j, int k) {
    if (i + k >= goal) return 1.0;
    if (j >= goal) return 0.0;
    if (computed[i][j][k]) return p[i][j][k];

    ⟨Recursively compute p[i][j][k]⟩
    return p[i][j][k];
}
```


Dynamic Programming (cont.)

- Exercises:
  - Pig solitaire: reach goal score *g* in *n* turns
  - Pig solitaire 2: maximize score in *n* turns
  - THINK solitaire: 2-dice Pig variant in 5 turns
  - Advanced projects:
    - Risk board game analysis
    - Simple Yahtzee variant analysis

Value Iteration

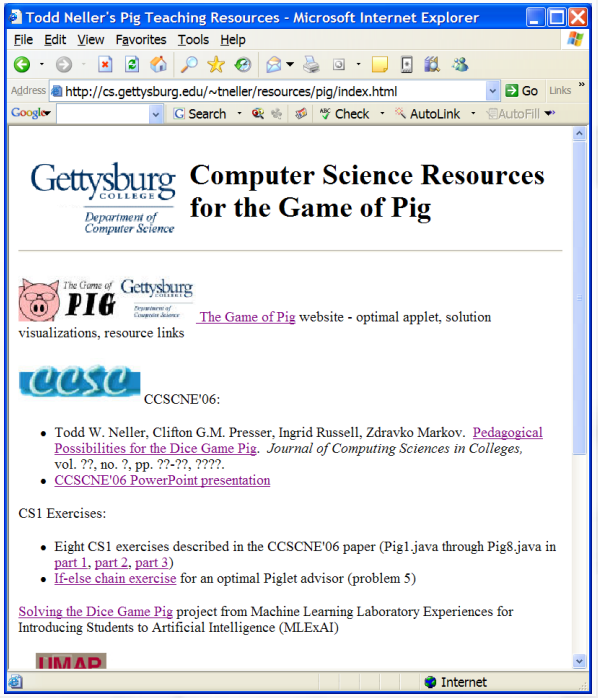
- Worked example: Piglet
  - Pig with a coin and goal score of 10
  - Score heads flipped or nothing if a tail is flipped
- Exercises:
  - Pig
  - Pig Solitaire 3: minimize turns to reach goal score *g*
  - Pass the Pigs
  - Advanced projects:
    - Hog: Pig with single throw of as many dice as desired
    - 10,000: jeopardy dice game

Conclusion

- Simple examples are teaching treasures.
- Pig is a simple, fun teaching example for teaching dynamic programming and value iteration.
- These projects provide many jumping off points for undergraduate research projects.
- There are also applications of Pig to CS1 (algorithm design, OO design), Machine Learning, Graphics, Networking, etc.


Further Resources

- http://modelai.gettysburg.edu
- http://cs.gettysburg.edu/~tneller/resources/pig
  - Game of Pig Website
  - CCSCNE paper, PPT
  - CS1 Exercises
  - MLExAI Pig project
  - 3 UMAP Journal papers



This work was sponsored in part by NSF DUE CCLI-A&I Award Number 0409497.