2010

Practical Play of the Dice Game Pig

Todd W. Neller
Gettysburg College

Clifton G.M. Presser
Gettysburg College

Neller, Todd W. and Clifton G.M. Presser. "Practical Play of the Dice Game Pig." The UMAP Journal 31.1 (2010), 5-19.

Practical Play of the Dice Game Pig

Abstract
The object of the jeopardy dice game Pig is to be the first player to reach 100 points. Each turn, a player repeatedly rolls a die until either a 1 is rolled or the player holds and scores the sum of the rolls (i.e., the *turn total*). At any time during a player's turn, the player is faced with two choices: *roll* or *hold*. If the player rolls a 1, the player scores nothing and it becomes the opponent's turn. If the player rolls a number other than 1, the number is added to the player's turn total and the player's turn continues. If the player instead chooses to hold, the turn total is added to the player's score and it becomes the opponent's turn.

In our original article [Neller and Presser 2004], we described a means to compute optimal play for Pig. However, optimal play is surprisingly complex and beyond human potential to memorize and apply. In this paper, we mathematically explore a more subjective question:

What is the simplest human-playable policy that most closely approximates optimal play?

While one cannot enumerate and search the space of all possible simple policies for Pig play, our exploration will present interesting insights and yield a surprisingly good policy that one can play by memorizing only three integers and using simple mental arithmetic. [*excerpt*]

Keywords
dice game, pig, probability, optimal play, strategy

Disciplines
Computer Sciences

Practical Play of the Dice Game Pig

Todd W. Neller
Clifton G.M. Presser
Department of Computer Science
300 N. Washington St.
Campus Box 402
Gettysburg College
Gettysburg, PA 17325–1486
tneller@gettysburg.edu

Introduction to Pig

The object of the jeopardy dice game Pig is to be the first player to reach 100 points. Each turn, a player repeatedly rolls a die until either a 1 is rolled or the player holds and scores the sum of the rolls (i.e., the *turn total*). At any time during a player's turn, the player is faced with two choices: *roll* or *hold*. If the player rolls a 1, the player scores nothing and it becomes the opponent's turn. If the player rolls a number other than 1, the number is added to the player's turn total and the player's turn continues. If the player instead chooses to hold, the turn total is added to the player's score and it becomes the opponent's turn.

In our original article [Neller and Presser 2004], we described a means to compute optimal play for Pig. However, optimal play is surprisingly complex and beyond human potential to memorize and apply. In this paper, we mathematically explore a more subjective question:

> *What is the simplest human-playable policy that most closely approximates optimal play?*

While one cannot enumerate and search the space of all possible simple policies for Pig play, our exploration will present interesting insights and yield a surprisingly good policy that one can play by memorizing only three integers and using simple mental arithmetic.

First, we review the criterion for optimality and discuss our means of comparing the relative performance of policies. Then we describe and evaluate several policies with respect to the optimal policy.

The UMAP Journal 31 (1) (2010) 5–19.

Maximizing the Probability of Winning

Let $P_{i,j,k}$ be the player's probability of winning if the player's score is i, the opponent's score is j, and the player's turn total is k. In the case where $i + k \geq 100$, we have $P_{i,j,k} = 1$ because the player can simply hold and win. In the general case where $0 \leq i, j < 100$ and $k < 100 - i$, the probability of a player winning who is playing optimally (i.e., an *optimal player*) is

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

where $P_{i,j,k,\text{roll}}$ and $P_{i,j,k,\text{hold}}$ are the probabilities of winning by rolling or holding, respectively. These probabilities are given by

$$\begin{aligned} P_{i,j,k,\text{roll}} &= \frac{1}{6}\left[(1 - P_{j,i,0}) + \sum_{r\in[2,6]} P_{i,j,k+r}\right], \\ P_{i,j,k,\text{hold}} &= 1 - P_{j,i+k,0}. \end{aligned}$$

The probability of winning after rolling a 1 or holding is the probability that the other player will not win beginning with the next turn. All other outcomes are positive and dependent on the probabilities of winning with higher turn totals.

These equations can be solved using value iteration as described in Neller and Presser [2004]. The solution to Pig is visualized in **Figure 1**. The axes are i (player 1 score), j (player 2 score), and k (the turn total). The surface shown is the boundary between states where player 1 should roll (below the surface) and states where player 1 should hold (above the surface).

Comparing Policies

Throughout the paper, we measure the performance of policies against the optimal policy of Neller and Presser [2004]. In this section, we describe the technique.

Let $Roll^A_{i,j,k}$ and $Roll^B_{i,j,k}$ be Boolean values indicating whether or not player A and player B, respectively, will roll given a score of i, an opponent score of j, and a turn total of k. Then we can define the respective probabilities of winning in these states, $P^A_{i,j,k}$ and $P^B_{i,j,k}$, as follows:

$$P^A_{i,j,k} = \begin{cases} \frac{1}{6}\left[(1 - P^B_{j,i,0}) + \sum_{r\in[2,6]} P^A_{i,j,k+r}\right], & \text{if } Roll^A_{i,j,k}; \\ 1 - P^B_{j,i+k,0}, & \text{otherwise.} \end{cases}$$

$$P^B_{i,j,k} = \begin{cases} \frac{1}{6}\left[(1 - P^A_{j,i,0}) + \sum_{r\in[2,6]} P^B_{i,j,k+r}\right], & \text{if } Roll^B_{i,j,k}; \\ 1 - P^A_{j,i+k,0}, & \text{otherwise.} \end{cases}$$

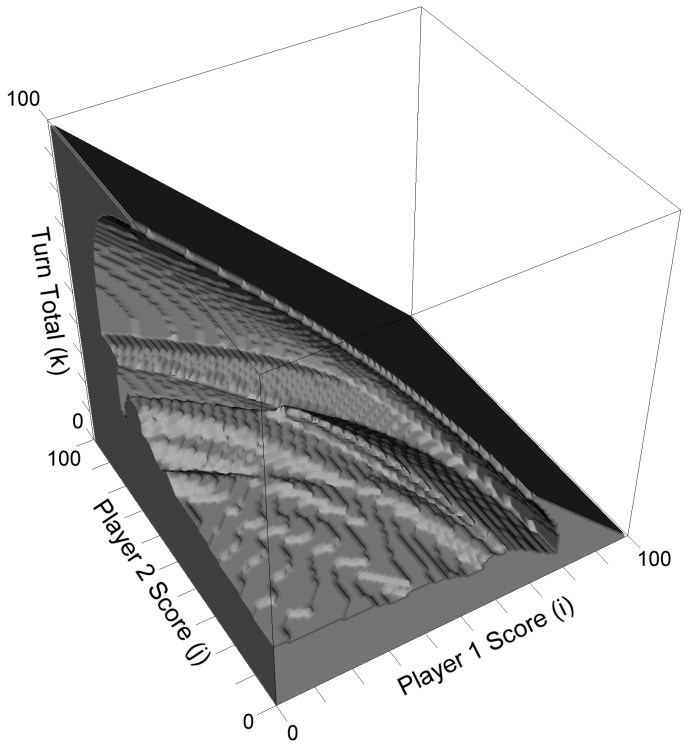


Figure 1. Roll/hold boundary for optimal Pig policy.

There is an advantage to going first: When two optimal players compete, the first player has a 53.06% chance of winning. The probability of player A (respectively, B) winning when going first is $P^A_{0,0,0}$ (respectively, $P^B_{0,0,0}$). Since there are no draws, the probability of A winning when B goes first is $(1 - P^B_{0,0,0})$. Assume that the first player is determined by an odd/even die roll. Then the average probability of a player A win is

$$\tfrac{1}{2}\left[P^A_{0,0,0} + (1 - P^B_{0,0,0})\right].$$

Once the system of equations above is solved, we use this average probability to evaluate relative policy strength.

Those equations can be solving using a process similar to *value iteration* [Bellman 1957; Bertsekas 1987; Sutton and Barto 1998], by which we iteratively improve estimates of the value of being in each state until our estimates are "good enough." Put simply, we begin with arbitrary estimates for all unknown probabilities; all our initial estimates are 0. Then we iteratively go through all equations, updating our left-hand-side probabilities with new estimates computed from the right-hand-side expressions. These estimate revisions continue until they converge, that is, until no single estimate is changed significantly. This procedure can be viewed as a generalization of the Jacobi iterative method for solving linear systems.

Let $\mathcal{S}$ be the set of all nonterminal game states (i, j, k) where $i, j \in [0, 99]$ and $k \in [0, 99 - i]$. Then our policy-comparison algorithm is given as **Algorithm 1**.

Algorithm 1 Policy Comparison

For each $(i,j,k) \in \mathcal{S}$, initialize $P^A_{i,j,k}$ and $P^B_{i,j,k}$ arbitrarily.
Repeat
$\Delta \leftarrow 0$
For each $(i,j,k) \in \mathcal{S}$,

$$p1 \leftarrow \begin{cases} \frac{1}{6}\left[(1 - P^B_{j,i,0}) + \sum_{r\in[2,6]} P^A_{i,j,k+r}\right], & \text{if } Roll^A_{i,j,k}; \\ 1 - P^B_{j,i+k,0}, & \text{otherwise.} \end{cases}$$

$$p2 \leftarrow \begin{cases} \frac{1}{6}\left[(1 - P^A_{j,i,0}) + \sum_{r\in[2,6]} P^B_{i,j,k+r}\right], & \text{if } Roll^B_{i,j,k}; \\ 1 - P^A_{j,i+k,0}, & \text{otherwise.} \end{cases}$$

$\Delta \leftarrow \max\left\{\Delta,\ \left|p1 - P^A_{i,j,k}\right|,\ \left|p2 - P^B_{i,j,k}\right|\right\}$
$P^A_{i,j,k} \leftarrow p1$
$P^B_{i,j,k} \leftarrow p2$
until $\Delta < \epsilon$
return $\left[P^A_{0,0,0} + (1 - P^B_{0,0,0})\right]/2$

Hold at n (or at Goal)

Perhaps the best-known simple policy is the "hold at 20" policy, where a player holds for any turn total that is greater than or equal to 20, or would reach the goal total of 100. In his book *Dice Games Properly Explained*, Reiner Knizia presents an odds-based argument for why holding at 20 maximizes the expected points per turn, viewing each roll as a bet that a 1 will not be rolled:

> ...we know that the true odds of such a bet are 1 to 5. If you ask yourself how much you should risk, you need to know how much there is to gain. A successful throw produces one of the numbers 2, 3, 4, 5, and 6. On average, you will gain four points. If you put 20 points at stake this brings the odds to 4 to 20, that is 1 to 5, and makes a fair game. ...Whenever your accumulated points are less than 20, you should continue throwing, because the odds are in your favor.
>
> [Knizia 1999, 129]

One might expect that, since holding at 20 maximizes the expected points per turn, then this strategy would have a greater expected win probability against optimal than any other "hold at n" policy for $n \neq 20$.

Comparing optimal play versus "hold at n" for $n \in [15, 35]$, we find a surprising result, shown in **Figure 2**. In fact, we minimize the average probability of an optimal player win to $.521$ when $n = 25$. The optimal

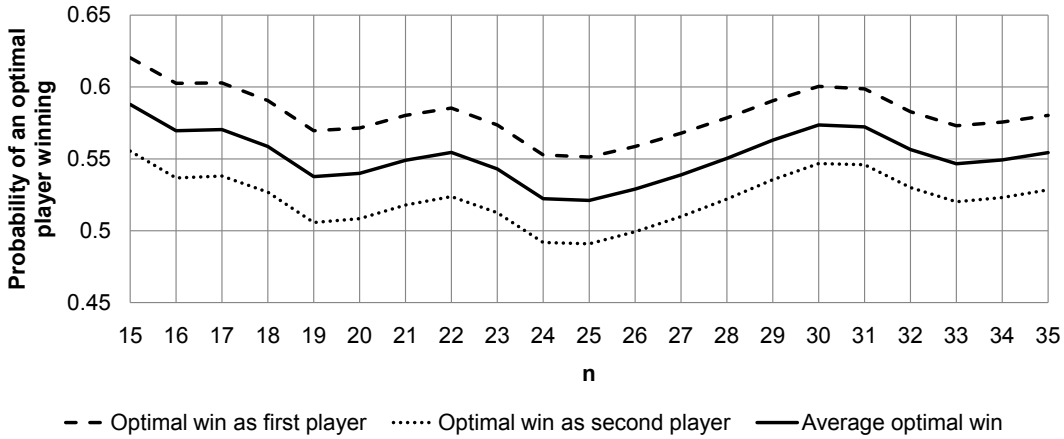


Figure 2. Probability of an optimal player winning against a player using the "hold at n" policy for different values of n.

player has a 4.2% win probability advantage over the "hold at 25" player, compared to an 8.0% advantage over the "hold at 20" player. This is an unexpected and significant improvement.

Considering the average optimal win probabilities of **Figure 2**, we observe local minima at 16, 19, 25, and 33. As fractions of the goal score 100, these approximate $1/6$, $1/5$, $1/4$, and $1/3$. This suggests a slightly different class of policies that seek to reach the goal score through a fixed number of *scoring turns*—turns that do not end with the roll of a 1 (pig) but instead increase the player's score—each of which achieves some desired minimum hold value.

What a Turn Scores

To understand the problem with a simplistic single hold value, first consider that the actual outcome of a "hold at 20" scoring turn increases the score by a minimum of 20 and at most by 25. For example, a player may roll a total of 19, roll a 6, and then hold with 25. To put forward an extreme case, consider a "hold at 20" game where a player has 4 scoring turns of 25, 25, 25, and 24, yielding a score of 99. In general, stopping just short of the goal is inadvisable, since doing so provides the opponent more opportunity to reach the goal and win.

How likely is the extreme case? More generally, how can we calculate the probable outcomes of a turn where we hold at n? As it turns out, this can be calculated by hand in stages. For large n, we would wish to automate the process to avoid error; but for small $n = 4$, a worked example illustrates the technique.

In **Table 1**, we proceed in steps. Initially, we start with a turn total $k = 0$ with probability 1. On each step s, we remove the probability p from turn

Table 1. Worked example for $n = 4$.

	Turn Total (k)									
Step	0	1	2	3	4	5	6	7	8	9
0	1/1									
1	1/6		1/6	1/6	1/6	1/6	1/6			
2	1/6		1/6	1/6	1/6	1/6	1/6			
3	7/36			1/6	7/36	7/36	7/36	1/36	1/36	
4	2/9				7/36	2/9	2/9	1/18	1/18	1/36

total $k = s$. Consider this the probability of "passing through" turn total k. (For the single case $k = 0$, we do effectively return to this turn total when a 1 (pig) is rolled.) When "passing through" turn total k, $p/6$ is added to probabilities for $k = 0, k + 2, k + 3, \ldots, k + 6$. Consider step 0. From the initial position, we remove the 1, and we distribute it in sixths according to the probable roll outcomes. One-sixth of the time, we roll a 1 (pig) at the beginning of the turn and contribute to a 0 score outcome. For rolls of 2 through 6, we contribute $1/6$ each to the "passing through" turn total probabilities of 2 through 6.

In step 1, there is no change, since it is impossible to have a turn total of 1; so there is no contribution passing through to other turn totals. In step 2, we pass through the turn total of 2 with probability $1/6$; we remove this value and then contribute $1/6 \times 1/6 = 1/36$ to turn totals 0, 4, 5, 6, 7, and 8. In step 3, we similarly contribute the same amount to turn totals 0, 5, 6, 7, 8, and 9.

Now all nonzero entries at turn totals ≥ 4 are the probabilities of such outcomes for a "hold at 4" turn. We are no longer "passing through" these totals; we hold for each. Further, the probability for turn total 0 is the probability of rolling a pig for a "hold at 4" turn.

This process can be continued to compute probable outcomes of a "hold at n" turn for any n. Continuing the process, the probable outcomes for a "hold at 25" turn are shown in **Figure 3**.

We observe that while most "hold at 25" scoring turns will be close to 25, a turn total outcome of 30 is more than $1/6$ as likely as an outcome of 25. If the "hold at 25" player has multiple high-scoring turns, that player will overconservatively stop just short of the goal. It would be desirable to "pace" the scoring turns so that a high-scoring turn benefits *all* future scoring turns.

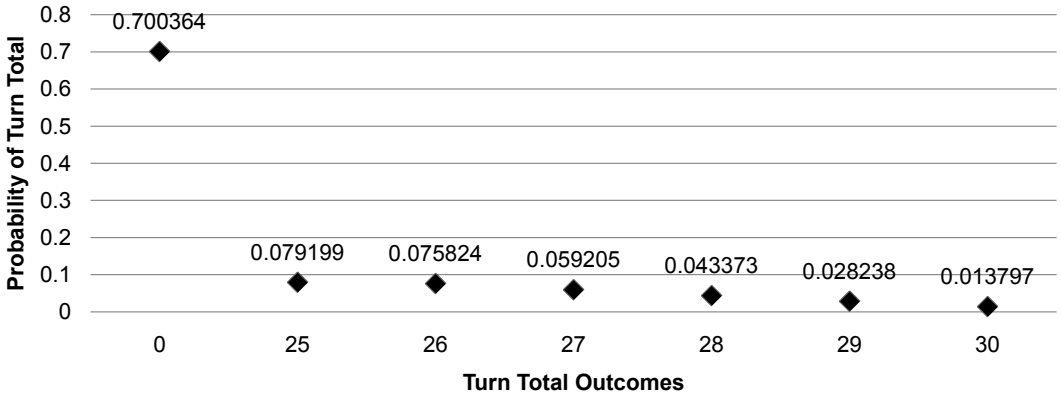


Figure 3. Probability of each possible scoring outcome of a single turn using a "hold at 25" policy.

t Scoring Turns

In contrast to the "hold at n" policy, a "t scoring turns" policy allows us to vary hold values according to a desired pace towards the goal. For example, if $t = 5$, we might initially hold at 20. Then, scoring 25 on the first turn, we might choose lesser hold values henceforth.

Let t_s be the number of scoring turns so far, i.e., turns that have increased a player's score. One scheme chooses a hold value that approximately divides the remaining points to the goal, $(100 - i)$, by the remaining number of scoring turns in the policy, $t - t_s$. Letting $h(i, t_s)$ be the hold value when a player has score i and has taken t_s scoring turns, then we have

$$h(i, t_s) = \left\lfloor \frac{100 - i}{t - t_s} \right\rfloor .$$

For example, suppose that a player is playing such a policy with $t = 4$. If the player's score is 51 after 2 scoring turns, the player would hold at

$$h(51, 2) = \left\lfloor \frac{100 - i}{t - t_s} \right\rfloor = \left\lfloor \frac{49}{2} \right\rfloor = \lfloor 24.5 \rfloor = 24.$$

In **Figure 4**, we compare optimal play versus "t scoring turns" play for $t \in [3, 6]$. Since "hold at 25" was superior to other "hold at n" policies, we would expect that $t = 4$ would be the best of these "t scoring turns" policies. This expectation is correct. The optimal player has a 3.3% win probability advantage over this "4 scoring turns" player, compared to a 4.2% advantage over the "hold at 25" player. A "hold at 25" player's scoring turn will increase the score in the range [25, 30], so "hold at 25" is one kind of "4 scoring turns" player. However, by pacing the scoring across the game with this simple policy, we reduce the optimal player advantage further by almost 1%.

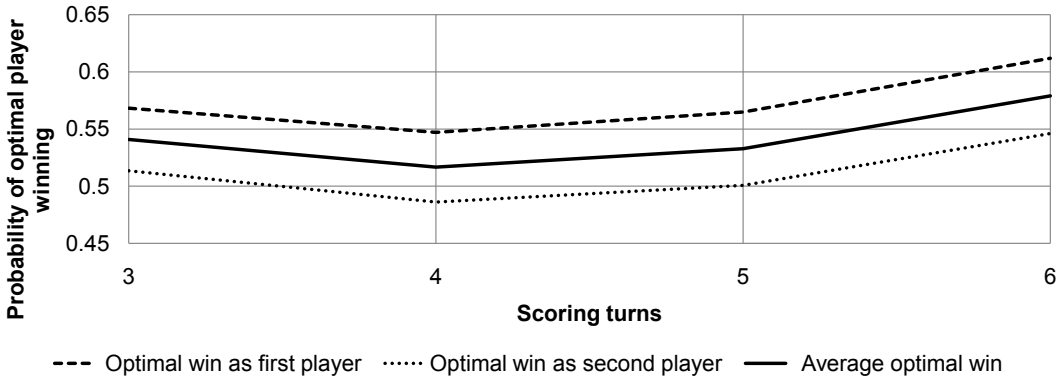


Figure 4. The probability of an optimal player winning against a player using the "t scoring turns" policy for different values of t.

Optimal t Scoring Turns

We do not claim that our previous scheme is the *best* means of pacing the scoring. Testing the entire space of "t scoring turns" policies is well beyond what is computationally feasible. However, with a measure of "best," we can solve for the "'best" policy, using value iteration techniques.

First, we note that a "t scoring turns" policy does not take the opponent's score (j) into account. Rather, it concerns itself only with the player's score (i) and the turn total (k). The player is essentially playing solo, blind to the opponent's progress. If constrained to have 4 scoring turns, what measure should we optimize to win most often against an ignored opponent?

Our objective is to choose hold values that *minimize the expected number of remaining turns to the goal*. The expectation can be expressed as

$$E_{i,k} = \min\left\{E_{i,k,\text{roll}},\ \ E_{i,k,\text{hold}}\right\},$$

where $E_{i,k,\text{roll}}$ and $E_{i,k,\text{hold}}$ are the expected number of future turns if one rolls or holds, given by:

$$E_{i,k,\text{roll}} = \tfrac{1}{6}\left[(1+E_{i,0}) + E_{i,k+2} + E_{i,k+3} + E_{i,k+4} + E_{i,k+5} + E_{i,k+6}\right],$$
$$E_{i,k,\text{hold}} = 1 + E_{i+k,0}.$$

In the terminal case where $i \geq 100$, we define $E_{i,k} = 0$. Solving these equations using value iteration yields an intriguing policy, shown in **Figure 5**. This policy, while more difficult to remember, can be summarized by the following instructions for computing hold values:

- **After 0 scoring turns:** Hold at 24.
- **After 1 scoring turn:** From 24 subtract 1 for each 3 points above 24.
- **After 2 scoring turns:** From 25 subtract 1 for each 2 points above 48.
- **After 3 scoring turns:** Hold at 100 − score. (Roll to win!)

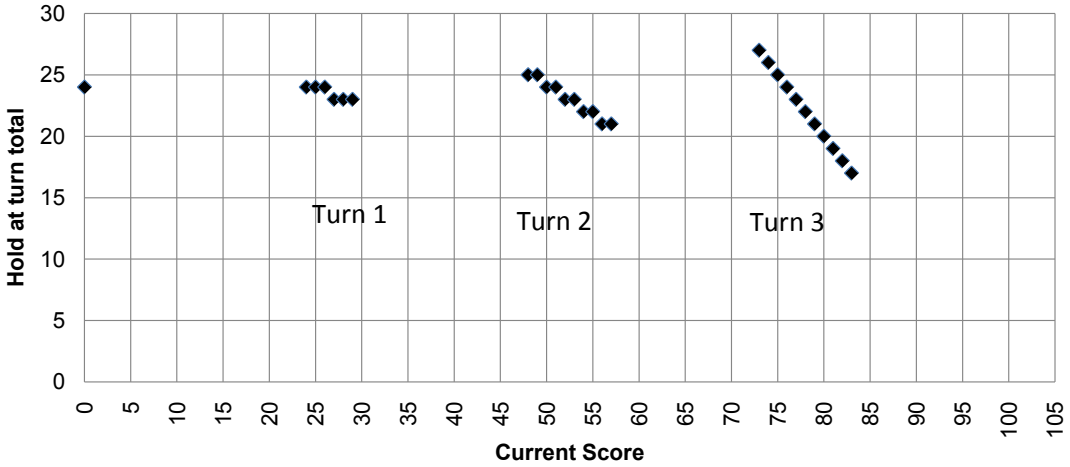


Figure 5. Hold values for each possible score using the "optimal 4 scoring turns" policy.

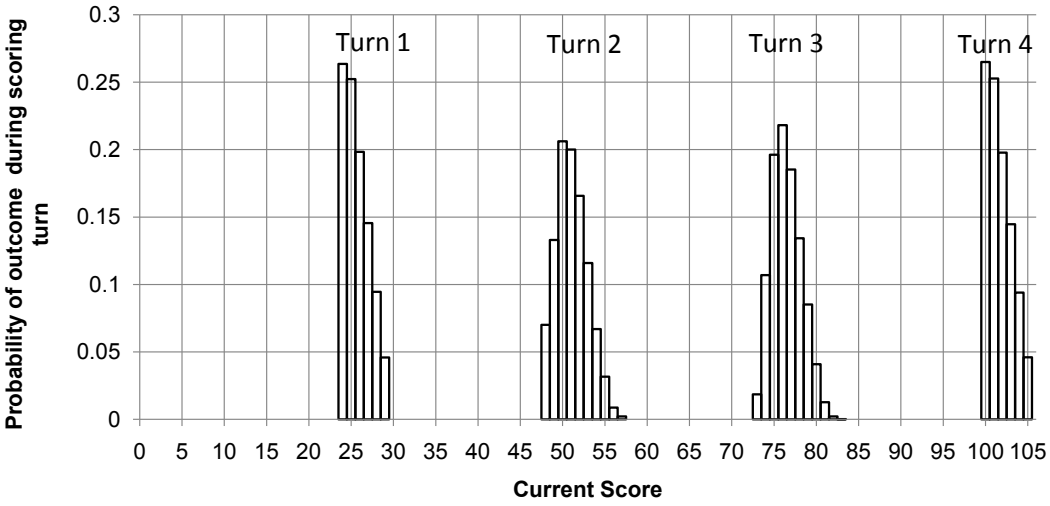


Figure 6. Probability of reaching each possible score during its respective scoring turn.

The probabilities of reaching different scores while playing this policy are shown in **Figure 6**. The player using this policy will expect to reach the goal score of 100 in an average of 12.62 turns.

On average, the optimal player will win versus the "optimal 4 scoring turns" player with probability .514805, yielding a 3.0% win probability advantage, compared to a 3.3% advantage over the simple "4 scoring turns" player.

Score Base, Keep Pace, and End Race

In this section, we introduce a policy also devised by the authors that is simple to remember, yet reacts to a lead by the opponent and has excellent performance relative to the optimal policy.

We begin with a simple two-variable framework for decisions: Roll if

- the turn total k is less than some base value b,
- your score i plus the turn total k is still less than the opponent's score j, or
- either your score i or the opponent's score j is within e points of the goal.

Alternately varying b and e, we find that $b = 18$ and $e = 31$ gives the best performance against the optimal player. On average, the optimal player will win versus this policy with probability .513723, yielding a 2.7% advantage.

Next, we add an additional variable p for keeping pace with a leading opponent; your goal is to end the turn with a score within p of the opponent's score j. You now roll if:

- $k < b$ (you must score at least b),
- $i + k < j - p$ (you must get within p of j), or
- either $i \geq 100 - e$ or else $j \geq 100 - e$ (you roll to win when someone is within e of the goal).

Successively optimizing each of the three parameters, we find that $b = 19$, $p = 14$, and $e = 31$ gives the best performance against the optimal player. On average, the optimal player will win versus this policy with probability .509431, yielding a 1.9% advantage.

Keep Pace and End Race

In this section, we introduce a modification of our previous policy that keeps pace with the opponent differently and has even better performance. With this policy, you roll if:

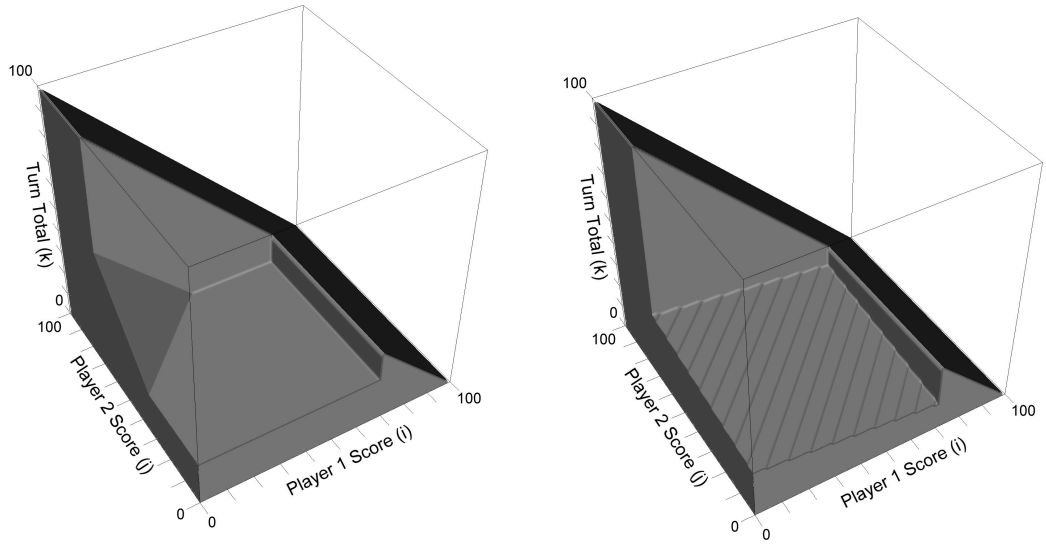


Figure 7. Roll/Hold boundary for "score base, keep pace, and end race" and "keep pace and end race" policies.

- either $i \geq 100 - e$ or $j \geq 100 - e$, or else
- $k < c + \dfrac{j - i}{d}$.

The first condition has a player roll if either player's score is e points or less from the goal. In the second condition, we compute a hold value by taking a constant c and changing it proportionally to the score difference. If your score is ahead or behind, you use a lower or a higher hold value, respectively.

For practical use, we reduce this computation to integer arithmetic. Four common ways of converting a noninteger result of division to an integer are integer division (truncation of digits beyond the decimal point), floor (next lower integer), ceiling (next higher integer), and rounding (closest integer, values halfway between rounded up).

For each of these four cases, we successively optimize each of the three parameters c, d, and e. **Table 2** shows the best policies for each case.

The best of these, utilizing rounding, is indeed the best policy yet, reducing the optimal player's advantage to a surprisingly narrow 0.922%:

- If either player's score is 71 or higher, roll for the goal.
- Otherwise, subtract your score from your opponent's and let m be the closest multiple of 8. (Choose the greater multiple if halfway between multiples.) Then hold at $21 + \frac{m}{8}$.

Table 2. Optimal parameter values for "keep pace and end race" strategy, for the four cases of rounding.

Integer conversion	c	d	e	Average probability of optimal win
$(j-i)$ div d	21	7	29	.504790
floor $\left(\frac{j-i}{d}\right)$	22	8	29	.504799
ceiling $\left(\frac{j-i}{d}\right)$	21	8	29	.504655
round $\left(\frac{j-i}{d}\right)$	21	8	29	.504610

Recent Related Work

Johnson computed exact turn outcome probabilities for two policies: hold at 20 points, and hold after 5 rolls [Johnson 2008].

For the similar jeopardy dice game Pass the Pigs®, Kern contrasted multinomial-Dirichlet inference scoring probabilities with empirical scoring probabilities, giving special attention to the extreme policies of hold at 100 (goal) and hold after 1 roll [Kern 2006].

Tijms [2007], after restating two-player optimality equations for Pig [Neller and Presser 2004] and Hog [Neller and Presser 2005], described a game-theoretic, simultaneous-action version of Hog.

Glenn et al. made recent progress on the analysis of Can't Stop® [Fang et al. 2008a,b; Glenn and Aloi 2009; Glenn et al. 2007a,b], an excellent jeopardy dice game by Sid Sackson in which 2–4 players race to be the first claiming three tracks corresponding to two-dice totals.

Smith published a survey of dynamic programming analyses of games in general [Smith 2007].

Conclusions

Although the "hold at 20" policy for playing Pig is well known for maximizing expected points per turn, it fares poorly against an optimal player. The optimal policy is expected to win an average of 54.0% of games against a "hold at 20" policy, yielding an 8.0% advantage.

We evaluated a variety of policies, including

- hold at n;
- simple "t scoring turns"[1];

[1]Hold at the floor of (points remaining divided by scoring turns remaining).

- optimal "t scoring turns," minimizing expected turns;
- score base (b), keep pace (p), and end race (e); and
- keep pace (c, d) and end race (e).

Through many iterations of policy comparison, we find that the optimally-tuned last policy performs very well with respect to the optimal policy:

If either player's score is 71 or higher, roll for the goal. Otherwise, hold at

$$21 + \text{round}\left(\frac{j-i}{8}\right).$$

Whereas an optimal player holds an 8.0% advantage over "hold at 20" play, this advantage is reduced to 0.922% against this new policy. Although human play of Pig cannot match optimal play, it is interesting to find that simple, good approximations of optimal play exist.

Although we have compared a number of practically-playable policy classes for many parameters to the optimal policy, we do not claim to have found the best human-playable policy for Pig. We invite readers to continue this exploration. Readers can evaluate their policies online at `http://cs.gettysburg.edu:8080/~cpresser/pigPolicy.jsp`.

Given that many jeopardy dice and card games share similar dynamics [Neller and Presser 2005], we believe the "keep pace and end race" policy can provide a good approximation to optimal play of many jeopardy games. Value iteration, policy comparison, and Monte Carlo methods may be used to find the best policy parameters.

References

Bellman, Richard E. 1957. *Dynamic Programming*. Princeton, NJ: Princeton University Press.

Bertsekas, Dmitri P. 1987. *Dynamic Programming: Deterministic and Stochastic Models*. Upper Saddle River, NJ: Prentice-Hall.

Fang, Haw-ren, James Glenn, and Clyde P. Kruskal. 2008a. Retrograde approximation algorithms for jeopardy stochastic games. *ICGA [International Computer Games Association] Journal* 31(2): 77–96.

_____. 2008b. A retrograde approximation algorithm for multi-player Can't Stop. In *Computers and Games, 6th International Conference, CG 2008 Beijing, China, September 29–October 1, 2008. Proceedings*, edited by H. Jaap van den Herik, Xinhe Xu, Zongmin Ma, and Mark Winands, 252–263. Berlin: Springer. `http://www.springerlink.com/index/qk87960427872415.pdf`.

Glenn, James, and Christian Aloi. 2009. A generalized heuristic for Can't Stop. In *Proceedings of the Twenty-Second International Florida Artificial Intelligence Research Society Conference (FLAIRS), May 19–21, 2009, Sanibel Island, Florida, USA*, edited by H. Chad Lane and Hans W. Guesgen, 421–426. Menlo Park, CA: AAAI Press.

Glenn, James, Haw-ren Fang, and Clyde P. Kruskal. 2007a. A retrograde approximation algorithm for one-player Can't Stop. In *Computers and Games, 5th International Conference, CG 2006, Turin, Italy, May 29–31, 2006, Revised Papers*, edited by H. Jaap van den Herik, Paolo Ciancarini, and H. (Jeroen) H.L. Donkers, 148–159. Berlin: Springer. http://www.springerlink.com/index/y0k517082438956g.pdf .

_____. 2007b. A retrograde approximation algorithm for two-player Can't Stop. In *MICC Technical Report Series 07-06: Proceedings of the Computer Games Workshop 2007 (CGW 2007)*, edited by H. Jaap van den Herik, Jos Uiterwijk, Mark Winands, and Maarten Schadd. The Netherlands: Universiteit Maastricht. http://citeseerx.ist.psu.edu/viewdoc/downloaddoi=10.1.1.131.3258&rep=rep1&type=pdf .

Johnson, Roger W. 2008. A simple "Pig" game. *Teaching Statistics* 30(1): 14–16.

Kern, John C. 2006. Pig data and Bayesian inference on multinomial probabilities. *Journal of Statistics Education* 14(3). http://www.amstat.org/publications/jse/v14n3/datasets.kern.html .

Knizia, Reiner. 1999. *Dice Games Properly Explained*. Brighton Road, Lower Kingswood, Tadworth, Surrey, KT20 6TD, U.K.: Elliot Right-Way Books.

Neller, Todd W., and Clifton G.M. Presser. 2004. Optimal play of the dice game Pig. *The UMAP Journal* 25(1): 25–47.

_____. 2005. Pigtail: A Pig addendum. *The UMAP Journal* 26(4): 443–458.

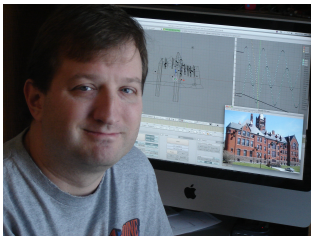
Smith, David K. 2007. Dynamic programming and board games: A survey. *European Journal of Operational Research* 176: 1299–1318.

Sutton, Richard S., and Andrew G. Barto. 1998. *Reinforcement Learning: An Introduction*. Cambridge, MA: MIT Press.

Tijms, Henk. 2007. Dice games and stochastic dynamic programming. *Morfismos* 11(1): 1–14. http://chucha.math.cinvestav.mx/morfismos/v11n1/tij.pdf .

About the Authors

Todd W. Neller is Associate Professor and Chair of the Department of Computer Science at Gettysburg College. A Cornell University Merrill Presidential Scholar, Neller received a B.S. in computer science with distinction in 1993. He was awarded a Stanford University Lieberman Fellowship in 1998, and the Stanford University Computer Science Department George E. Forsythe Memorial Award in 1999 for excellence in teaching. He completed his Ph.D. in computer science with a distinction in teaching in 2000. His dissertation work concerned the extension of artificial intelligence search algorithms to hybrid dynamical systems, and the refutation of hybrid system properties through simulation and information-based optimization. Recent works have concerned the application of artificial intelligence techniques to the design and analysis of games and puzzles.



Clifton G.M. Presser is also Associate Professor of Computer Science at Gettysburg College. He received a B.S. in mathematics and computer science from Pepperdine University in 1993. Clif received his Ph.D. in computer science at the University of South Carolina in 2000. Clif's dissertation research was on automated planning in uncertain environments. Currently, his research interests are in remote and collaborative visualization.