Aaron Brown

Monopoly® 101, Part II

In the second part of this study of the world's favorite boardgame, firing up the great engine of Monopoly

Last issue, we went through a fair bit of analysis to propose the Monopoly property pricing formula

$$\frac{1}{\beta}\left(\Re_i + \frac{\beta_i \Phi}{(n-1)\beta_i + \beta}\right)$$

where $\Re_i$ is the expected rent collected per opponents' roll of the dice from property i, β_i is the expected rate of return on housing investment per opponents' roll of the dice for property i, n is the number of players in the game, Φ is the expected net payment from the Bank per roll of the dice and β is the geometric mean of the β_is for all players.

The games people play

Before applying this model to game play I want to take a step back. If the goal were to determine the optimal strategy for Monopoly, to solve the game, we would have to take a different approach. If we knew the playing strategies of the other players, the problem would be straightforward, but computationally intensive. We could simulate every possible Monopoly game and at every decision node take the path that gives the highest probability of winning. With a few computational tricks that is feasible. Since typical Monopoly games involve a few hundred dice rolls with an average of perhaps a hundred decisions per roll, a Monopoly game is about as complex as a mortgage derivative or long-term strongly path-dependent exotic option.

However, we don't know the other players' strategies. The usual game theory approach is to search for the strategy that gives the maximum chance of winning, assuming all other players know your strategy and act in the worst possible way from your point of view. This strategy has the maximum (over the set of your possible strategies) minimum (over the set of opponents' possible responses) probability of winning and is called "minimax." It may not be a good strategy, in the sense of winning frequently against real opponents.

In Monopoly, the minimax strategy has a very low guaranteed chance of winning. The reason is that if all other players act to minimize your chance of winning, there is very little you can do except hope for extraordinary luck. It should be possible to get minimax style strategies by putting restrictions on the strategies of opponents, but to my knowledge it has never been done.

Rather than a game theory approach, I am taking a financial approach to the game, solving it as if it were a real world financial problem. My solution is not perfect, but it can be derived with a relatively small amount of work. Unlike a typical minimax strategy, it does not depend crucially on tiny details of the game, it does not have to be recomputed from scratch for every minor rule change. If there were an elimination tournament of computer programs to see which played the best Monopoly game, contestants would be well advised to take a minimax approach to development. But my approach is much more practical for human Monopoly play. It gives general insight into the game, even if it doesn't always recommend the optimal decision. Moreover, it's easy to improve for people who want to beat other people who have read this article.

Garbage in, diamonds out

My goal is to illustrate the power of the financial toolkit, not solve the game of Monopoly. So rather than taking a rigorous approach, I started by asserting that valuation is the key to the game. Then I made a series of false assumptions so I could get a closed-form expression for the value, something similar to the Capital Asset Pricing Model or the Gordon Model.

There are obvious objections to this approach, in Monopoly and real world finance. Some students cannot get over the fact that the assumptions are false. Others point out that we're moving backward—we started with one thing we don't know (the value of a property) and wrote down an expression for it that requires two things we don't know (the rate of return for the property and for the game as a whole), which are

unobservable to boot. Still other students are unhappy because we haven't defined value clearly—are we talking about exchange value for a trade or contribution to probability of winning or cash value for a purchase or something else? These things are not the same either theoretically or quantitatively.

Nonetheless, we have done something important. We have provided a foundation to answer questions that are otherwise unanswerable. Suppose the first player in a Monopoly game rolls a 6, lands on Oriental Avenue and declines to buy it. The property goes up for auction. How much should you be willing to bid? Or, late in the game, there is only one more house available to buy. How much should you bid for it? These are the kinds of questions that casual Monopoly players cannot begin to answer. Our financial model is not valuable because it gives accurate answers to these questions, they don't come up enough in game play to make a difference and reasonable arbitrary answers are probably almost as good as the model. Instead the model is valuable because it gives rational answers to these questions.

Monopoly as a Markov process

Our first task is to improve our formula by revisiting some of the simplifying assumptions. We started by saying all squares are visited with the same frequency. This is not the case and it affects the definition of rent roll (rather than dividing the rent by 40 we should multiply by the frequency with which the property is visited) and the value of Φ.

To compute the actual frequencies we number the squares from 0 (Go) to 39 (Boardwalk). Next we define a transition matrix such that the number in row i and column j is the probability that one roll of the dice will take a player's token that starts on square i to square j. The eigenvector of this transition matrix, with entries scaled to add up to 1, will be the long-run frequency distribution of visiting squares (if that's not clear, it isn't essential to understand it for this article, but I recommend you learn a little linear algebra, it's often handy). We can compute the eigenvector either with a mathematics package, or by repeated matrix multiplication in Excel (start with a vector of all 0.025 and keep multiplying it by the transition matrix until it stops changing, it doesn't take long).

There's one detail that will make life easier. Square 10, jail, can be visited in two states, "In Jail" and "Just Visiting" with different rules applying to the next dice roll. Square 30, Go to Jail, is never a starting or ending state (if you land on it at the end of one roll, you go to square 10) so we don't need it in our transition matrix. If we make square 30 the jail instead of square 10, so 10 means "Just Visiting" jail and 30 means "In Jail," every row in the transition matrix corresponds to exactly one state.

We populate this matrix with dice probabilities. For example, there is one chance in 18 that a player will roll a three, so the entry in the first row, fourth column is 1/18. This same number will be in the second row, fifth column; and the third row sixth column and so on (wrapping around at the end, so it will be in the 37th row, first column and so on).

The trouble is three squares labeled "Chance" and three more labeled "Community Chest." Landing on these squares requires the player to draw a card, which can send the token to another square. We could try to adjust the transition matrix directly, but it's easier to compute a Dice Matrix as in the previous paragraph and a Card Matrix. The product of Dice times Card (in that order) gives us the Transition Matrix. The Card Matrix will be the Identity matrix except for the six rows 2, 7, 17, 22, 33 and 36 which correspond to Chance and Community Chest squares.

For example, square 2 is Community Chest. Fourteen of the 16 cards do not move the token so the (2,2) matrix entry is 14/16. One card sends the token to Go (square 0), another to card sends the token to Jail ("In Jail," so square 30). Therefore (2,0) and (2,30) are both 1/16 and all other Card Matrix entries in row 2 are zero. One important detail: some Chance cards instruct the player to "Advance token to nearest . . ." The rules say to move the token forward to the first qualifying square, not to move around to board to the qualifying square nearest the starting point. For example, if you are on square 7 (Chance) and draw "Advance token to nearest railroad" you go to square 15 (Pennsylvania Railroad) rather than square 5 (Reading Railroad).

Another problem is the rule that players roll again if they get doubles, but three doubles in a row ends the turn and sends them to jail. We are going to compute all probabilities per roll rather than per turn so a player can start from any square in one of three states: first roll in the turn, roll after one double and roll after two doubles (there is no roll after three doubles because the turn ends). This innocent-looking little rule forces us to use nine times as many entries in our transition matrix. Rows and columns 0 to 39 represent starting off a new turn, 40 to 79 represent squares 0 to 39 after one double, 80 to 119 represent squares 0 to 39 after two doubles.

We're going to need two 40x40 Dice Matrices, one for doubles and one for nondoubles. There are six possible doubles (2, 4, 6, 8, 10 and 12), each with probability 1/36. Without doubles the possible rolls are 3, 4, 10 and 11 with probability 2/36 each; 5, 6, 8 and 9 with probability 4/36 each; and 7 with probability 6/36.

Now we can put together our Transition Matrix. We take the Nondoubles Dice Matrix times the Card Matrix and put that in the Northwest sector. We take the Doubles Dice Matrix times the Card Matrix and put it in the 40 columns to the right (that is the middle 40 columns in the top 40 rows). The remaining 40 columns of the top 40 rows are zeros.

We repeat the process for the next set of 40 rows, except that the Doubles Dice Matrix times

the Card Matrix goes in the last set of 40 columns, rather than the middle. For the last set of 40 rows we again put the Nondoubles Dice Matrix times the Card Matrix in the first 40 columns but we don't use the Doubles Dice Matrix. Instead we add 1/16 to every number in column 30, from row 80 to 119.

One more complication and we're done. We have to define what happens to players starting "In Jail," square 30 in our model. Players have the option of paying $50 to the Bank to get out of jail, in which case we could eliminate row 30 altogether. Players sent to Jail would just convert automatically to "Just Visiting" and go to square 10. This typically happens early in the game when players want to get out to buy available properties and collect money from the Bank for passing Go. Later in the game, there are few or no available properties and the rents extracted by other players more than offset the income from the Bank. Since this phase is the important one for valuation, we are going to assume players stay in jail as long as possible.

The Monopoly rules are not clear about how this works, so I'm going to use the tournament interpretation. Yes, there are Monopoly tournaments. The other common source of fully-specified rules is the set used in Hasbro's official Monopoly computer game, these differ slightly from the tournament rules.

In tournaments, the jailed player rolls the dice when it is her turn. If she gets doubles, she gets out of jail, moves the number of squares on the dice and ends her turn (i.e. she does not roll again for doubles as she normally would). On the first and second turn in jail, if she does not get doubles, she stays in jail. On the third turn, if she does not get doubles she pays $50 to the Bank and moves the indicated number of squares.

So for square 30 we are going to interpret the three states as first turn in jail (row 30), second turn in jail (row 70) and third turn in jail (row 110). For rows 30 and 70, we use the Doubles Dice Matrix probabilities to set the chances of going to squares 12, 14, 16, 18, 20 and 22, with 5/6 chance of going from row 30 to row 70 (staying in jail but moving from first to second turn) or 70 to 110 (moving from second to third turn in jail). For row 110 we use the sum of the Doubles and Nondoubles Matrix probabilities to set the chances of going to rows 12, 13, 14, 15, 16, 17, 18, 19, 20, 21 and 22 (we always go to the first set of columns, even if we roll doubles).

Passing Go and not passing Go

Transitions that end up below the diagonal generally involve passing Go but there are a few exceptions. First, you have to look at each of the nine submatrices individually, for example, row 37 (starting from Park Place on the first roll), column 45 (ending at Reading Railroad after rolling doubles) does pass Go, although the final column is larger than the row. The entry is below the diagonal in the submatrix, however. Also transitions to Jail never pass Go, by rule. Backward moves from the "Go Back Three Spaces" Chance card are also never passes, these are easy to spot in the matrix. Finally, a fraction of the moves from rows 35-39 to row 5 (or a multiple of 40 increment) involve the one case of passing Go twice on a roll (if you land on Chance and draw "Take a Ride on the Reading," you collect $200).

Throwing all this in, with a few minor adjustments, gives a value for Φ of $23.62, much lower than the $30.40 we computed assuming equal frequencies. The biggest difference is the reduced frequency of passing Go due to time spent in Jail. Unfortunately, Φ is no longer a constant. We must subtract $0.29 for each house on the board and $0.96 for each hotel, divided by the number of players in the game. This accounts for one Chance and one Community Chest that require payments per building. These cards are irrelevant early in the game, but often decisive in late games (good players remember where they are in the stack, by the time they become important their exact positions are usually known). When the game gets down to two players, if it is fully-developed (all 32 houses and 12 hotels are on the board), Φ goes down to $13.17. A smaller adjustment to Φ could be made for the Income Tax square, but it is quantitatively insignificant to valuation.

Simulating beta

Now we can turn our attention to the β_is, which we are going to estimate by simulation. That is, we're going to guess values for the β_is and simulate Monopoly games in which players make decisions based on the valuations. If our valuations are consistent, the total valuation of all players should grow at the game rate of interest on average and individual players' probability of winning should always increase as the valua-

LONG-RUN CELL FREQUENCIES IN MONOPOLY IF PLAYERS STAY IN JAIL AS LONG AS POSSIBLE (2.5% IS AVERAGE)

#	Square	Freq	#	Square	Freq	#	Square	Freq	#	Square	Freq
0	Go	2.92%	10	Visiting Jail	2.14%	20	Free Parking	2.82%	30	In Jail	9.39%
1	Mediterranean	2.01%	11	St. Charles	2.56%	21	Kentucky	2.61%	31	Pacific	2.52%
2	Community Chest	1.78%	12	Electric Company	2.62%	22	Chance	1.04%	32	North Carolina	2.48%
3	Baltic	2.04%	13	States	2.18%	23	Indiana	2.57%	33	Community Chest	2.23%
4	Income Tax	2.20%	14	Virginia	2.43%	24	Illinois	3.00%	34	Pennsylvania	2.36%
5	Reading Railroad	2.80%	15	Pennsylvania Railroad	2.64%	25	B&O Railroad	2.89%	35	Shortline Railroad	2.29%
6	Oriental	2.13%	16	St. James	2.68%	26	Atlantic	2.54%	36	Chance	0.82%
7	Chance	0.82%	17	Community Chest	2.30%	27	Ventnor	2.52%	37	Park Place	2.06%
8	Vermont	2.19%	18	Tennessee	2.82%	28	Water Works	2.65%	38	LuxuryTax	2.06%
9	Connecticut	2.17%	19	New York	2.81%	29	Marvin Gardens	2.44%	39	Boardwalk	2.49%

tion of their portfolio increases relative to the total valuation. We adjust our guess for the β_is to make these things true, and rerun the simulation. After many iterations we hope to converge on a consistent set of β_is. It's always possible that we have consistent but incorrect β_is, which cause simulated players to play stupidly and justify the wrong values. It's also possible that our model is too simple to capture much of the variation in probability of winning. So this approach is not rigorous. This is similar to Richard Roll's famous criticism of the Capital Asset Pricing Model. We will address this possibility by running our final model against a number of simulated computer players that use entirely different playing algorithms. If our valuations win consistently against other strategies, they're good enough for the purpose. If not, we have to go back and try another model or set of parameters.

In the first cut, we assumed that each monopoly could be improved at a constant linear rate. In reality, there are 10 or 15 discrete jumps per monopoly, and they are highly non-linear. We could increase the number of parameters to try to get a more accurate price. I prefer to do the best one-parameter job I can and see if it's good enough. If you want to beat someone who has read this article, you might consider developing a two-parameter pricing model and picking off the one-parameter errors. Or you could go all the way to an exact 15 parameter model. But each parameter adds considerably to the fitting time. I use only 8 (one per monopoly) parameters while an exact model requires 110.

We're also going to reduce the time by simulating only partial games. Black and Scholes taught us value propagates backward, so we start at the end of the game. All Monopoly games come down to two players, generally owning all the properties between them. I simulated a large number of games by distributing properties and money between two players and playing out to see who won. I concentrated on initial allocations that were close to even. If the valuations are off for situations in which one player has a high probability of beating the other, it won't make much difference. But errors in close situations are harmful.

I needed an assumption for building and mortgage strategy. If you build a house on a monopoly you increase the rent your opponent pays for landing there, so it's good to do. But if you are forced to later sell that house, usually to pay rent to your opponent, you only get half your money back. So it makes sense to invest your cash aggressively, but keep enough liquidity that you don't lose too much on sell-backs. A similar decision is when to mortgage or unmortgage property. If you mortgage, the Bank lends you half the official purchase price of the property, but you cannot collect rent from your opponent on the property. This foregone rent is the interest on the loan. There is also a transaction cost, the Bank demands 10% of the loan amount as a fee when the loan is repaid.

After each dice roll, each simulated player considered each legal decision, and simulated each one forward three full turns, assuming no other building, unmortgaging or unforced selling or mortgaging. The player took whatever decision resulted in the highest average value at the end of the turns. If our valuations were perfect, this step would be unnecessary, whatever action maximized instantaneous valuation would be the optimal decision. But investigating forward corrects many flaws in valuation and means simulated players may make sound decisions even when our valuation parameters are just guesses.

I fit β_is for the eight monopolies that gave the most accurate predictions of which sets of property and cash would win against other sets. This is relatively easy for two players because there is no trading and you do not need to use an overall interest rate, β, you can compute values using only the β_is of the two players.

Monopoly yield curve dynamics

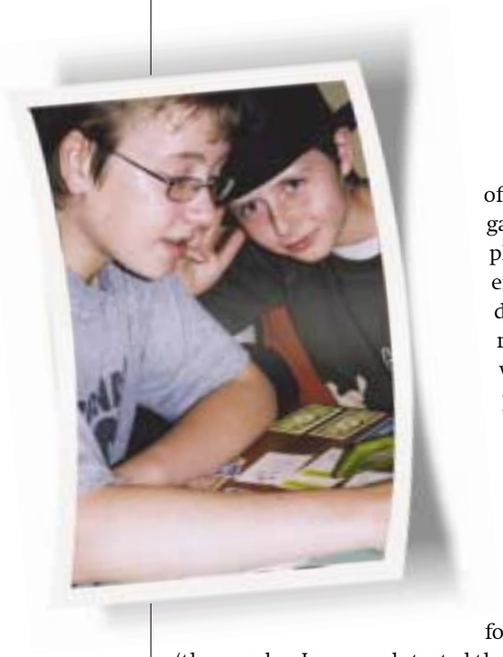
Once I had the eight β_is, I moved to a simulation of a three-player game. The payoff function was no longer win/lose but the total value of the portfolio of cash and properties when one player went bankrupt. For this, I needed to consider trades. It would be very complex to worry about trades while still fitting β_is, so I wanted to use the same ones I fit for the endgame. But these did not give reasonable valuations in the three-person game.

This is not surprising. The argument used for the valuation formula is a long-term equilibrium

that ignored all transient factors. But transients can be very important. For example, the probability is 8.31 per cent that an opponent will land on the St. James+Tennessee+New York color group on a random roll. But the specific probability for a given roll can be 0 per cent to 42 per cent. Obviously this is a significant factor in determining whether to invest in houses on this monopoly on a given turn.

I solved this by introducing a yield curve. At every turn in the game there is an interest rate to apply to expected gains and losses on the next dice roll, a different interest rate for the dice roll after that, and so on until three full rounds of turns have passed and the long-term interest rate applies to all future cash flows. I fit the distribution of these rates the same way I fit the β_is in the endgame. Short-term interest rates are highly volatile, changing both with token positions on the board and as players approach bankruptcy. A player near bankruptcy makes interest rates very high, rates fall sharply after the bankruptcy. Long-term interest rates change slowly.

Now I could incorporate trades into the simulation. After each roll of the dice, each property is evaluated by each player. Whichever player gives it the highest value buys it from the owner for the average of his value and the owner's value. This does not result in much trading, because it is very expensive to trade developed properties (all houses and hotels must first be sold back to the Bank at half price). Generally, once someone started selling properties, she usually lost. However the process is still very important, because it determines who ends up with what at the beginning



of the two person game. In casual play, losing players often go down in flames, resulting in a windfall for the player (or the Bank sometimes) who pushes the bankrupt over the edge.

I repeated this process for four players (the number I assumed started the game). This is the ideal number for Monopoly, more players increase the amount of luck involved, put too much cash in the game and make trading difficult.

This still did not get me back to the beginning of the game, because I started the four-player simulation assuming all properties were owned. For the final simulation step, I put all players at Go with $1,500 in cash and no properties. Whenever player A landed on a property, each simulated player in the game computed its value from his perspective. If A had the highest valuation, and it was higher than the purchase price, she bought it. If B had the highest valuation, and it was higher than the purchase price, A bought it from the Bank and immediately sold it to B for the average of their valuations. If the highest value was less than the purchase price, that player bought the property from the Bank at that price. The effect of these rules was usually that the player with the most cash bought the first property available in each color group (or railroads or utilities), and that same person bought all subsequent properties in that group. Again, the payoff was not winning or losing, but portfolio valuation after the last property is purchased. For this stage of the game, I assumed no building or mortgaging and did not consider trades because there are too many possibilities before properties are formed into monopolies.

And the answer is …

Finally, I took these results and tested them against other strategies. I programmed a number of reasonable seeming strategies, and found that mine regularly beat them. I tried perturbing the valuation parameters and running complete games to see if I could develop an improved model. This resulted in some adjustments to my parameters.

At the conclusion of this process, which took about a day of effort spread out over several months (plus about a dozen overnight runs on my computer), I had β_i estimates for each color group and a model of interest rate yield curve and evolution (not a model that can be easily written down, however). Although the pricing model is a simple closed form, the actual valuations require numerical solution on a computer because they depend, sometimes crucially, on all the details of the game state.

There are three situations in which the original closed-form valuation formula can be applied directly. First is before any development has taken place so the transient effects are minor. People often shorten the Monopoly game by distributing the properties among the players instead of cash. The starting values in the table above are appropriate for that purpose. If you distribute the properties and use cash to make everyone up to the same value, the game is fair. Don't use too much cash or you force down the interest rate and change the values.

The second situation is late game values. Suppose you see a game in progress with a lot of houses and hotels showing, and no player in immediate danger of bankruptcy. If you have no information about the positions of the tokens on the board, the values in the end game column should give accurate predictions of who is likely to win.

Finally, we can ask what is the expected maximum value of this monopoly over the course of a random game (minimum values are quite small and not very meaningful). This reveals the important midgame role of the first two monopolies and Park Place+Boardwalk.

Has all this work created an algorithm that can beat a good human player? I don't know, the answer awaits empirical test. But that's not the point of the exercise. If this model doesn't work, we can always go back and add more parameters or tinker with the formulae.

The key is that we have developed what appears to be a reasonable model for the game and solved it (albeit with numerical algorithms rather than simple formulae). We have injected rationality into the play. We can test assertions about strategy. Unless our model is way off, we understand the game better.

The same thing is true of the basic models of quantitative finance. Although they are not precise enough to give useful prices, they point the way toward rational solutions. Practical finance involves a lot of *ad hoc* pricing assumptions and analytic shortcuts. But we can make experiment with confidence because there is a sound theory underlying everything. W

MONOPOLY VALUATIONS IN SOME SPECIFIC SITUATIONS

Properties		Start (1.6% Interest)		End (6% interest)		Maximum
	β_i	$\Re_i$	Value	$\Re_i$	Value	Value
Mediterranean + Baltic	3.33%	$0.24	$435	$14.20	$235	$1,002
Oriental + Vermont + Connecticut	7.65%	$0.87	$510	$36.81	$610	$1,263
St. Charles + States + Virginia	4.36%	$1.53	$529	$57.34	$950	$1,164
St. James + Tennessee + New York	11.89%	$2.44	$618	$80.35	$1,332	$1,592
Kentucky + Indiana + Illinois	2.89%	$3.06	$601	$87.36	$1,448	$1,575
Atlantic + Ventnor + Marvin Gardens	3.41%	$3.40	$632	$87.45	$1,450	$1,619
Pacific + North Carolina + Pennsylvania	0.77%	$3.92	$530	$96.78	$1,604	$1,662
Park Place + Boardwalk	6.44%	$3.93	$694	$80.65	$1,337	$2,606
All four Railroads	0.00%	$23.04	$1,429	$23.04	$382	$1,429
Both Utilities	0.00%	$3.69	$229	$3.69	$61	$229