Quorsum Endgames:
A Markov Model

William Q. Erickson

December 6, 2017

Abstract

Since the game's inception seven years ago, Quorsum players have had only one consistent source for endgame strategy: their gut. A board game characterized by its wild down-to-the-wire finishes ought to have some mathematical rules of thumb to guide late-game decisions, but until now, beyond the basic table of single-roll probabilities, any long-term calculations have proved elusive due to the unique "chain rule" mechanic in Quorsum. As a long-awaited remedy, in this paper I derive a Markov model to calculate the expected number of turns until victory given a single-pawn endgame position.

1 Quorsum and its (Relevant) Rules

When you first glance at a Quorsum board, all you see is numbers. And yet, somewhat maddeningly for its enthusiasts, the game has till now resisted efforts to calculate probabilities beyond a single roll of the dice. The main reason for this difficulty is the "chain rule" for movement—a rule which, while giving the game much of its strategy and excitement, heavily complicates the long-term probabilities of success. In this section we familiarize the reader with the rules of Quorsum; for the curious, links to the full rules, video tutorials, and sample games are listed in the references.

Quorsum is played on a randomized 4-by-4 grid of square **tiles**. (See Figure 1.) Each tile is marked with a certain **value**—a whole number between 2 and 6 inclusive, usually depicted as pips on a die. We refer either to a "tile of value 4" or simply a "4-tile." Players quickly learn to regard the value of a tile as a measure of its "resistance"—it will soon become clear that the higher the value, the more difficult it is to move onto that tile.

Quorsum is a two-player racing game. Each player controls two **pawns**, and each of the four pawns begins in one of the four corner tiles. The goal of the game is to move each of your two pawns across the board to its **home tile** (the corner tile opposite its starting tile) before your opponent can do the same; thus each pawn is marked with an arrow that designates its homeward direction. The early and middle stages of the game involve the four pawns jockeying for key tiles and blocking each other along the way, but typically by the endgame some of the pawns have cleared the fray and have an open path home; these are the situations explored in this paper.

The final mechanic to address is how the pawns actually move from tile to tile. (Quorsum players will notice that we pass over the mechanic of flipping tiles, along with any notion of defense, in this introduction; we will address this aspect near the end of the paper, after developing a model for purely offensive scenarios.) On each turn, a player has four standard

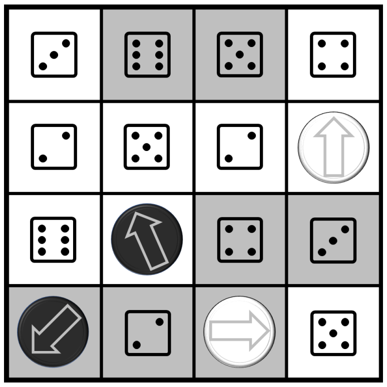

Figure 1: A typical endgame in Quorsum. One pawn (the black pawn in the lower-left) is already home.

(The fact that the tiles are reversible, both light and dark, is deferred until the end of the paper, for simplicity's sake.)

six-sided dice which he can assign to attempt to move one or both of his pawns. Pawns can be moved only onto a vacant orthogonally adjacent tile. To attempt to move a pawn, a player declares "I am using d dice to move *this* pawn to *that* tile," where d is some number between one and four. Then he rolls his allotted dice all at once to determine whether his move succeeds. A move onto a tile of value t **succeeds** iff at least one of the dice shows t or greater.

Example: In Figure 1, your opponent is attempting to move her upper-right white pawn onto its home tile just above, which has value 4. She is using three dice. If she rolls ⚁ ⚂ ⚄, then she has succeeded (because of the ⚄) and she moves her pawn onto the 4-tile; if, on the other hand, she rolls ⚀ ⚀ ⚂, then she has failed because none of the dice shows 4 or greater.

The Chain Rule: In one sense, pawns move "one tile at a time," BUT they can nonetheless link together several such moves on the same turn. If your move succeeds, then you may immediately attempt to move that pawn again, but using only the dice that succeeded in the previous roll. You may continue this "chain" until all your dice fail.

Example: In Figure 1, you are attempting to move your remaining black pawn toward its home tile in the upper-left corner, using all four dice. You first attempt to move onto the 5-tile just above. (You could also attempt the 6-tile to the left, but the probability of succeeding at a 6 is much lower.) You roll ⚁ ⚃ ⚄ ⚅, thus succeeding with two of the dice. You move your pawn onto the 5-tile, and may now attempt to continue moving, using only those two surviving dice. You now try for the 2-tile to the left; you roll ⚂ ⚄ and hence succeed with both dice surviving, so you move your pawn onto the 2-tile, and may continue trying to move. The next adjacent tile you want (your pawn's home tile) has value 3; but you roll ⚀ ⚁ and thus do not succeed with either die, so your move has ended for this turn.

2 What Every Quorsum Player Wants To Know

Because Quorsum, if not *the* racing game, is certainly *a* racing game, the key questions concern time; and in a game, time is measured in turns. Specifically, given a position, a player will naturally ask, "How many turns can I expect until this pawn reaches home?" In other words, we want to determine the expected number of turns until victory. Because of the complexity stemming from the varying tile values and number of dice being rolled, we will have to build up our model by first answering some simpler questions:

- **What is the probability of success in moving from one tile to another?** This will be simple to compute, and of course depends both on the value of the attempted tile and on the number of dice being rolled.
- **What is the probability of a pawn progressing at least n tiles during one turn?** This answer will use the previous result, and moreover will depend both on the number of dice rolled *and* on each of the tile values comprising the pawn's desired path.
- **What is the probability of a pawn progressing *exactly* n tiles during one turn?** This crucial result will condition on the previous one.
- **What is the expected number of turns until a pawn reaches home?** To answer this, the most pressing question of all, we will use our previous results to create a Markov chain modeling a pawn's progress toward its home tile.

3 Answers in the Form of Functions

We address each of the aforementioned questions, en route to constructing our ultimate goal, the Markov model.

3.1 The Probability of Success For One Move

Let $s(d,t)$ denote the probability of success, using d dice, attempting to move onto a t-tile. Here we have $d,\ t \in \mathbb{N}$ with $1 \leq d \leq 4$ and $1 \leq t \leq 6$. Since of course success is the complement of failure, we simply take the complement of the probability of all d dice rolling less than t, giving the straightforward formula

$$s(d,t) = 1 - \left(\frac{t-1}{6}\right)^d \tag{1}$$

(Remark: Although there are no tiles with value 1, still we will want to include 1 as a possible t-value for later—see section 4.2 below. It should be clear that although a player never literally "moves onto a 1-tile," mathematically we still have $s(d,1) = 1$ for any d, since a die cannot roll less than 1.)

Given the restrictions on the domain, s takes only 20 possible values, comprising a table which any decent Quorsum player has committed to memory (at least to the nearest hundredth):

t \ d	1	2	3	4
⚁	.833	.972	.995	.999
⚂	.667	.889	.963	.988
⚃	.500	.750	.875	.938
⚄	.333	.556	.704	.802
⚅	.167	.306	.421	.518

3.2 The Probability of Exactly k Dice Surviving a Roll

To account for the loss of dice caused by the chain rule, we are interested not only in the success/failure of an attempted move, but also how many dice survive each roll, since this affects the probability of moving to the next tile within the same turn.

To this end, for $k = 0, \ldots, 4$, we define the family of functions

$$s_k(d,t) := \mathbb{P}(\text{exactly } k \text{ dice survive} \mid d \,\text{dice are rolled at a}\, t\text{-tile}).$$

This is merely a binomial distribution; the first parameter is d since the number of trials is the number of dice rolled, and the second parameter is $s(1,t)$, the probability of succeeding with a single die. Hence we have

$$s_k(d,t) = \binom{d}{k} \left(s(1,t)\right)^k \left(1 - s(1,t)\right)^{d-k} \tag{2}$$

3.3 The Probability of a Pawn Moving Exactly j Tiles on a Turn

So far our two functions, s and the family s_k, pertain to a single movement; but of course multiple movements can occur on the same turn, and moreover the essence of strategy is looking ahead by *several* turns. Thus it is now time to consider *paths.*

3.3.1 Paths

Informally, a path is simply a chain of adjacent tiles, along which a player intends to move a pawn. Formally, we define a **path** as a finite sequence $(t_n)_{n=1,2,\ldots\ell}$ over $\{1,2,3,4,5,6\}$, corresponding to the values of the tiles comprising such a chain, the first tile being adjacent to the pawn. (Again, note that although there are no 1-tiles in the game, still we include 1 among possible values of a path—this will be useful later.) The length ℓ of a path is the number of tiles in it, which typically ranges between 1 and 6 inclusive, since all pawns begin 6 tiles away from home (although in a desperate position, a path could extend up to 10 tiles before becoming officially absurd).

Example: In Figure 1, the remaining black pawn will be interested in the path $t_n = (5,2,3)$, since completing that path will bring it home. In this case, $\ell = |t_n| = 3$, with $t_1 = 5$, $t_2 = 2$, and $t_3 = 3$.

3.3.2 Intermediate Step: Probability of a Pawn Moving <u>At Least</u> j Tiles on a Turn

Ultimately, using d_1 dice for the first move attempt, and given a path t_n, we want to determine the probability q_j of the pawn ending the turn *exactly* on the tile t_j. So we are seeking to find the values of

$$q_j(d_1, t_n) := \mathbb{P}(\text{ending the turn on } t_j)$$

Hence if $\ell = |t_n|$, then q_ℓ is the probability of completing the entire chain this turn. On the other extreme, q_0 denotes the probability of all the dice failing on the very first roll. This at least makes q_0 easy to compute, since it is just the probability of failing with all d_1 dice at the first tile t_1 in the path:

$$q_0 = 1 - s(d_1, t_1)$$

But with this easy case out of the way, calculating directly for $j > 0$ is much harder. Instead, to find the remaining q_j, we will first find $\hat{q}_j$, the probability of ending *at least* on tile t_j. Specifically we define

$$\hat{q}_j(d_1, t_n) := \mathbb{P}(\text{ending the turn on } t_x \mid x \geq j).$$

Clearly, for $j = 0$ and 1, we have

$$\hat{q}_0(d_1, t_n) = 1$$
$$\hat{q}_1(d_1, t_n) = s(d_1, t_1).$$

For $j = 2$ we use a conditioning argument:

$$\begin{aligned}
\hat{q}_2(d_1, t_n) &:= \mathbb{P}(\text{ending on at least } t_2) \\
&= \mathbb{P}(\text{1st roll succeeds and 2nd roll succeeds}) \\
&= \mathbb{P}(\text{1st roll succeeds}) \cdot \mathbb{P}(\text{2nd roll succeeds}|\text{1st roll succeeds}) \\
&= \sum_{d_2=1}^{d_1} \mathbb{P}(d_2 \text{ dice survive 1st roll}) \cdot \mathbb{P}(\text{2nd roll succeeds using } d_2 \text{ dice}) \\
&= \sum_{d_2=1}^{d_1} s_{d_2}(d_1, t_1) \cdot s(d_2, t_2)
\end{aligned}$$

Iterating the same conditioning argument leads to (naturally more complicated) formulas for higher j-values:

$$\hat{q}_3(d_1, t_n) = \sum_{d_2=1}^{d_1} s_{d_2}(d_1, t_1) \sum_{d_3=1}^{d_2} s_{d_3}(d_2, t_2) \cdot s(d_3, t_3)$$

$$\hat{q}_4(d_1, t_n) = \sum_{d_2=1}^{d_1} s_{d_2}(d_1, t_1) \sum_{d_3=1}^{d_2} s_{d_3}(d_2, t_2) \sum_{d_4=1}^{d_3} s_{d_4}(d_3, t_3) \cdot s(d_4, t_4)$$

We could continue and write $\hat{q}_5$ and $\hat{q}_6$ explicitly, but instead we can arrive at a general formula for an arbitrary j-value. Rearranging our last result, we get

$$\hat{q}_4(d_1, t_n) = \sum_{d_1 \geq d_2 \geq d_3 \geq 1} s(d_4, t_4) \prod_{m=1}^{3} s_{d_{m+1}}(d_m, t_m)$$

which extends to the general formula

$$\hat{q}_j(d_1, t_n) = \sum_{d_1 \geq \cdots \geq d_j \geq 1} s(d_j, t_j) \prod_{m=1}^{j-1} s_{d_{m+1}}(d_m, t_m) \tag{3}$$

for $1 \leq j \leq \ell$. (For $j > \ell$, we will define $\hat{q}_j \equiv 0$, since it is impossible to end beyond one's home tile.)

Equation (3) is the work of a human, but as far as actually computing these values, software can take over from here.

3.3.3 From "At Least" to "Exactly"

This subsection will be mercifully short. Since we now have the probability $\hat{q}_j$ of a pawn ending on *at least* the jth tile of a given path, we can at last pin down our coveted function q_j:

$$\begin{aligned} q_j &:= \mathbb{P}(\text{ending exactly on } t_j) \\ &= \mathbb{P}(\text{ending on } t_x \mid x \geq j) - \mathbb{P}(\text{ending on } t_x \mid x \geq j+1) \\ &= \hat{q}_j - \hat{q}_{j+1} \end{aligned}$$

Thus, beginning with d_1 dice and given a path t_n, we have

$$q_j(d_1, t_n) = \hat{q}_j(d_1, t_n) - \hat{q}_{j+1}(d_1, t_n) \tag{4}$$

4 The Markov Model

We can already answer some interesting and strategic questions using the functions s, s_k, $\hat{q}_j$, and q_j in Equations (1) through (4), but these are really just ingredients for the answer to our overwhelming question: **What is the expected number of turns until a pawn reaches home?** Before proceeding, we should verify that this situation—a pawn attempting to move along a path home—truly is a stochastic process describable by a Markov model.

4.1 Verifying the Process is Markovian

Suppose we are given a Quorsum position in which we want to analyze a certain pawn and its path home t_n consisting of ℓ tiles, i.e., $|t_n| = \ell$. Then we have a state space $X = \{0, 1, \ldots, \ell\}$ where $X = j$ corresponds to the pawn occupying the tile t_j. (We let $X = 0$ correspond to the pawn having not moved at all along the path.) The steps in this process measure turns, so that $X_r = j$ if the pawn ends its r^{th} turn on tile t_j.

We assume a pure offensive strategy, meaning that the player will use all four dice to move this pawn on every future turn; thus we can set $d_1 = 4$ in Equations (3) and (4) above. (**NB: From now on we will drop the dice-argument d in all the functions we have constructed, with the understanding that $d = d_1 = 4$.**)

Then, since the player begins every turn with four dice, and since the tile values in the path never change, the transition probabilities from one state to the next (i.e., from one tile in the path to the tile at the end of the next turn) depend only on the pawn's current state (i.e., the tile it occupies). Hence we do have a Markov process, and so we can proceed to construct a Markov chain.

4.2 A Slight Modification

... We can *almost* proceed, that is. We have just said that the tile values in the path never change; this is true, but from the pawn's perspective, a path naturally "shrinks" with every successful move. As a pawn progresses along its path, it "leaves behind" the tiles it has already reached; whereas a pawn might begin facing a path of 4 tiles, once it occupies the first tile t_1, it now faces a path of only 3 tiles. Hence we need a way to encode this "leaving tiles behind," for otherwise the probability of reaching home will not even depend on which tile the pawn currently occupies, as if the pawn had to traverse the entire path anew on every single turn, regardless of its progress.

To fix this, let $t_n = (t_1, \dots, t_\ell)$ be a path, and define the modified path

$$_it_n \equiv (\underbrace{1, 1, \dots, 1,}_{i \text{ times}} t_{i+1}, t_{i+2}, \dots, t_\ell).$$

That is, just replace each of the first i terms with the number 1. Since $s(d, 1) = 1$ for all d, this renders the first i tiles in a path "automatic"—which is equivalent to the pawn currently occupying tile t_i, having left behind the i tiles it has already reached.

Now for the finishing touch: define a new probability function

$$q_{ij}(t_n) \equiv q_j(_it_n), \tag{5}$$

which in English gives the probability that a pawn, starting on the i^{th} tile of a path, ends its turn on the j^{th} tile. In other words, this is the probability of a pawn moving from t_i to t_j (i.e., from state i to state j) during one turn.

...Which, if you are about to construct a Markov chain, is incredibly convenient.

4.3 Constructing the Transition Matrix

Now that all the parts are in place, a brief recap of where we stand. Let a path t_n be given, with $|t_n| = \ell$. Our state space is then $X = \{0, 1, \dots, \ell\}$, corresponding to the number of tiles our pawn has moved along the path; meanwhile each step corresponds to one turn (not just a one-tile movement). Hence $X_r = j$ corresponds to the pawn ending its r^{th} turn on tile t_j, with $X_0 = 0$, and the pawn is home as soon as it enters state ℓ, which is an absorbing state. Now then, we have an $(\ell + 1) \times (\ell + 1)$ transition matrix Q, which we know *a priori* should look like this:

$$
Q = \begin{array}{c} \\ \mathit{0} \\ \mathit{1} \\ \mathit{2} \\ \vdots \\ \ell \end{array} \begin{array}{c} \begin{array}{ccccc} \mathit{0} & \mathit{1} & \mathit{2} & \dots & \ell \end{array} \\ \begin{pmatrix} * & * & * & * & * \\ 0 & * & * & * & * \\ 0 & 0 & * & * & * \\ 0 & 0 & 0 & \ddots & * \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix} \end{array}
$$

Clearly Q should be upper-triangular since we are moving our pawn only forward toward home, so the state sequence X_r must be nondecreasing. What are the transition probabilities, i.e., the entries Q_{ij} of matrix Q? We have actually already seen the answer, for as it turned out,

$$Q_{ij} = q_{ij}(t_n) \tag{6}$$

where the q_{ij} are precisely our old friends from Equation (5)! (See closing remarks of previous subsection.)

4.4 Expected Number of Turns Until Victory

Now that we have a transition matrix, we can use it to answer not just a "How likely?" question, but a more important "How long?" question.

We want to determine the expected number of turns until the pawn reaches home. We note that the only recurrent state is the absorbing state ℓ, corresponding to the pawn reaching the home tile; all other states are transient. (A comforting thought during those games when the dice all seem to be against you.) Precisely because states $0, 1, \dots, \ell-1$ are all transient, we can calculate the mean number of turns spent in each of those transient states. Specifically, for $X_0 = i$, the mean number of turns spent in state j is typically denoted s_{ij}, and is calculated using a well-known procedure:

First we isolate the square block of Q corresponding to the transient states $0, 1, \dots, \ell-1$; call this block Q_T. In our case, Q_T is simply Q with the last row and last column deleted. Then

$$s_{ij} = (I - Q_T)^{-1}_{ij}$$

Now, the expected number of turns before victory is simply the sum of the individual mean times spent on each of the tiles before t_ℓ, given that we start at the beginning of the path in state 0. Hence we have our answer:

$$\mathbb{E}(\text{number of turns until victory}) = \sum_{j=0}^{\ell-1} s_{0j}, \tag{7}$$

which is just the sum of the entries in the first row of the matrix $(I - Q_T)^{-1}$. As a bonus, instead of having to update our model every time a pawn moves to a new tile t_i along a path, we can instantly update its expected number of turns until victory simply by replacing s_{0j} with s_{ij} in Equation (7).

Example: Again returning to Figure 1, trying to bring the black pawn home, we have the path $t_n = (5, 2, 3)$. Having fed the preceding functions into Mathematica, we let the software compute the transition matrix via Equation (6):

$$Q = \begin{array}{c} \\ \mathit{0} \\ \mathit{1} \\ \mathit{2} \\ \mathit{3} \end{array} \begin{array}{c} \begin{array}{cccc} \mathit{0} & \mathit{1} & \mathit{2} & \mathit{3} \end{array} \\ \begin{pmatrix} .198 & .075 & .169 & .559 \\ 0 & .001 & .038 & .961 \\ 0 & 0 & .012 & .988 \\ 0 & 0 & 0 & 1 \end{pmatrix} \end{array}$$

Next we isolate the transient block

$$Q_T = \begin{pmatrix} .198 & .075 & .169 \\ 0 & .001 & .038 \\ 0 & 0 & .012 \end{pmatrix}$$

and subtract from the identity and invert:

$$(I - Q_T)^{-1} = \begin{pmatrix} 1.247 & .094 & .217 \\ 0 & 1.001 & .039 \\ 0 & 0 & 1.012 \end{pmatrix}$$

Now by Equation (7), we sum the first row and conclude that our expected number of turns until victory is $1.247 + .094 + .217 = \mathbf{1.558}$. Hence, on average, we are about a turn and a half away from reaching home.

4.5 Other Questions—Answered Before They Were Asked

Armed with the transition matrix of a Markov chain, we can suddenly answer even more questions than anticipated, all of which are significant to a frantic Quorsum player assessing his winning chances late in the game. For example:

- **What is the probability of having reached home by the end of my n^{th} turn?** This translates to finding $\mathbb{P}(X_n = \ell)$, which is simply $(Q^n)_{0\ell}$, the upper-right entry of the n^{th} power of matrix Q.

  Example: In the previous example, we see from Q that we have a .559 probability of winning this turn; now to find the probability of having won by the end of the *next* turn, $\mathbb{P}(X_2 = 3)$, we simply take the square of the transition matrix

$$Q^2 = \begin{pmatrix} .039 & .015 & .038 & \mathbf{.909} \\ 0 & .000 & .000 & 1.000 \\ 0 & 0 & .000 & 1.000 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

  and observe that the desired probability is **.909**.

- **Which player is currently favored to win?** Since Quorsum is a racing game, this boils down to which player has the lower expected number of turns until victory, which we calculated in Equation (7). But we also need to account for whose turn it is at the moment, since that player has a chance at victory before the opponent gets to move again. The best criterion seems to be to compare the two expectations after subtracting 1 from the player whose turn it is; the lower result should be the player favored to win. It would be useful in the future to make this intuitive argument rigorous.

- **I still need to move *both* my pawns home; how many of my four dice should I use to move each pawn?** In our calculations above, we assumed that a player devotes every turn exclusively to moving one pawn, and hence we set $d_1 = 4$ in every scenario. But our model can easily be applied when a player still needs to bring both pawns home, and thus might split his dice between the two pawns. We would simply consider *two* paths, t_n and u_n, one for each pawn; if $|t_n| = \ell$ and $|u_n| = m$, then we construct two matrices, $T \in M_{\ell+1}(\mathbb{R})$ and $U \in M_{m+1}(\mathbb{R})$, as above, where

  $$T_{ij} = q_{ij}(d_t, t_n) \text{ and } U_{ij} = q_{ij}(d_u, u_n).$$

  Notice we have put the dice arguments d_t and d_u back into the function q_{ij}, since we certainly cannot use all four dice each turn for both paths. The strategy here depends on finding the respective numbers of dice for each path which minimizes the expected number of turns till victory. This can easily be done using software to check, where the minimum is taken over the whole-number dice values such that $d_t + d_u = 4$. There are only 5 such cases to check.

5 Summary

5.1 Further Questions

We have made an (unspoken) assumption so far, by shelving one of the distinctive Quorsum mechanics: namely, each tile is actually reversible, with a dark side and a light side (but the same value on both sides). This is significant because it introduces a restriction on pawn movement: a pawn can move only from dark-to-dark or light-to-light, and hence **flipping** tiles is a major part of the game. Some of a player's four dice can always be used to try to flip tile(s) and not just to move pawns, which lends a *defensive* component to Quorsum strategy. In our discussion above, we have assumed a pure offensive strategy, in which we use the same number of dice—typically all four—to move our pawn on every turn, instead of possibly "splitting" our dice between offense and defense.[1]

Furthermore, whenever we mentioned a "path," we implicitly meant a path of tiles, *none of which need to be flipped*; in Figure 1, our black pawn was on a light tile, and its $(5, 2, 3)$ path home consisted entirely of light tiles as well. The white pawn in the lower-right, however, still needs to flip its home tile before it can legally move there. Ultimately, we would like a model that applies to a path of dark *and* light tiles. This model would

[1] There is, however, a variant known as Quorsum Sprint, in which players use only one side of all the tiles and hence there is no flipping. For the Sprint variant, the model in this paper is already comprehensive.

necessarily be more complicated because it would have to account for dice spent to flip certain tiles before moving onto them. Certainly a single Markov chain would no longer suffice, since the transition probabilities now depend on the color of the tiles in the path, not just on the pawn's current state.

Another interesting question: In terms of winning probability, is a path t_n equivalent to the path $t_{\sigma(n)}$ for $\sigma \in S_\ell$? For instance, is our expected number of turns until victory for $(5, 2, 3)$ the same as for $(2, 3, 5)$? Certainly the matrix Q would look very different, since we would expect to spend the most time behind the 5-tile, but does the order affect the final result of the row sum in Equation (7)? If not, then the number of cases to tabulate using software will be dramatically decreased.

5.2 Results

By applying a Markov model to a Quorsum endgame, where the state space corresponds to tiles along a specified path (none of which need to be flipped), we are now able to answer our main question: Given a certain path, how many turns will it take on average for a pawn to reach home? The linchpin of this model is Equation (6), which encodes the entire transition matrix in our Markov chain; from there, Equation (7) gives us the answer to our question. The ancillary Equations (1) through (5), while of use and interest in their own right, were obtained primarily to build up these two main results. We can also answer the natural questions at the end of Section 4: the probability of reaching home after a certain number of turns, which player is currently expected to win, and even the optimal allotment of dice when bringing both pawns home.

Of course, the elegant brevity of Equations (6) and (7) is deceiving, since the ancillary functions $\hat{q}_j$ behind them are far too complicated for a mortal Quorsum player to memorize, much less calculate mentally while in the heat of battle. We will need to use software to compute and tabulate our results from this paper (cf. the example in Section 4.4), but nonetheless these results will be useful to Quorsum players seeking to hone their intuitions while studying endgames, or determining in retrospect whether they truly made the best play at the end of a nail-biter.

References

Complete rules of Quorsum:
`http://quorsum1.wixsite.com/quorsum/rulebook`

Quorsum YouTube Channel:
`https://www.youtube.com/channel/UCodwbgWKNYXar4PTYIUVrSA`
Contains video tutorials and sample games.

Quorsum page on BoardGameGeek:
`https://boardgamegeek.com/boardgame/107958/quorsum`
Contains images, reviews, and soon this paper.