Probabilities for Britannia battles

Torben Mogensen
email: torbenm@diku.dk

October 13, 2005

Abstract

This article will analyse the probabilities of outcomes of battles in the game "Britannia" with different forces and terrain.

1 The combat rules

We first summarize the combat rules for Britannia:

Combat in an area lasts one or more combat rounds until there is only one nation in the area or the defending nation submits to the attacker. A combat round consists of the following phases:

The die roll phase Here both sides roll one die per army, burh or fort in the area. Each die has a chance of killing an opponent, see below.

The defender retreat phase If both nations still have forces in the area, any subset of the remaining defenders can retreat.

The attacker retreat phase If both nations still have forces in the area, any subset of the remaining attackers can retreat.

If, at the end of a combat round, both nations still have forces in the area and the defender hasn't submitted, another combat round is initiated.

A die is rolled for each army, burh or fort in the area. On each 5 or 6, one of the opponent's armies is removed. Losses are taken simultaneously on both sides, i.e., after both sides have rolled their dice.

Exceptions:

- When fighting in a difficult-terrain area, it takes a 6 to kill a defending army, burh or fort. Attackers are killed as normal.
- Cavalry and Romans kill other armies on 4, 5 or 6 in normal terrain. A 6 is required to kill a Roman or cavalry army regardless of terrain.
- A leader does not roll any dice, but increases the die roll of all friendly armies, burhs and forts in the area by 1. A 6 still counts as a 6.

So depending on circumstances, anything from a 3 to a 6 may be required to kill an opponent.

2 Probabilities for 1:1 battles

Let us start with the simplest possible battle: Two normal armies of different nations facing each other in non-difficult terrain with no leaders present. Each roll one die, so we can tabulate the results after one round by the following table:

Defender roll	Attacker roll 1	2	3	4	5	6
1	1A+1D	1A+1D	1A+1D	1A+1D	1A	1A
2	1A+1D	1A+1D	1A+1D	1A+1D	1A	1A
3	1A+1D	1A+1D	1A+1D	1A+1D	1A	1A
4	1A+1D	1A+1D	1A+1D	1A+1D	1A	1A
5	1D	1D	1D	1D	none	none
6	1D	1D	1D	1D	none	none

where the entries in the table indicate the survivors so, for example, "1A+1D" means one attacker and one defender surviving. We can count the number of occurrences of each result and get:

1A+1D	16
1A	8
1D	8
none	4

The total is 36 (6×6), so we get 16/36 chance of getting the result 1A+1D, and so on.

So these give us the results for one round of combat. But what about the final results if noone retreats? It turns out we can just ignore the result that changes nothing (i.e., 1A+1D), as these get rerolled, and only look at the remaining outcomes. The new total is 20, so we get a probability of 8/20 for an end result of 1A or 1D and 4/20 for no survivors.

Since many rolls have the same outcome, we can simplify the table somewhat:

Defender roll	Attacker roll 1–4	5–6
1–4	1A+1D	1A
5–6	1D	none

and just multiply each entry by the number of outcomes, i.e., 1A+1D has $4 \times 4 = 16$ possibilites, 1A or 1D has $4 \times 2 = 8$ and "none" has $2 \times 2 = 4$ outcomes. Furthermore, we can divide the number of occurences of each result by the largest common factor, so we get

1A+1D	4
1A	2
1D	2
none	1

These simplifications make it easier to analyse the other cases. For example, a one-on-

one battle in hilands give this table:

Defender roll	Attacker roll 1–5	6
1–4	1A+1D	1A
5–6	1D	none

which translates to

1A+1D	10
1A	2
1D	5
none	1

after redocing by the largest common factor.

Again, we can ignore the 1A+1D row to get the probabilities for battles run to completion. In similar ways, we can get Roman army vs. normal army in non-difficult terrain (Roman is attacker):

1A+1D	5
1A	5
1D	1
none	1

and in difficult terrain:

1A+1D	25
1A	5
1D	5
none	1

I'll not go through the calculations for battles with leaders or attacks on Romans in difficult terrain, but leave these as an exercise for the reader.

3 Battles with multiple armies

If one or both sides in a battle have more than one army, things get a tad more complicated, in particular if there is a mixture of cavalry and normal armies (or Romans and forts) on one side. But we can use the same basic technique by setting up a table of outcomes. Here, for example is the table for 2:2 in non-difficult terrain:

Defender	Attacker roll			
roll	1–4/1–4	1–4/5–6	5–6/1–4	5–6/5–6
1–4/1–4	2A+2D	2A+1D	2A+1D	2A
1–4/5–6	1A+2D	1A+1D	1A+1D	1A
5–6/1–4	1A+2D	1A+1D	1A+1D	1A
5–6/5–6	2D	1D	1D	none

where the "/" is used to separate the two dice that are rolled by a player. Note that, for example, 1–4/1–4 represents $4 \times 4 = 16$ outcome and 1–4/5–6 represents $4 \times 2 = 8$ outcomes, so the entry that is cross-indexed by these two represents a total of $16 \times 8 = 128$ outcomes. Counting all outcomes and dividing by the common factor (16) gives us:

2A+2D	16
2A+1D	16
1A+2D	16
1A+1D	16
2A	4
2D	4
1A	4
1D	4
none	1

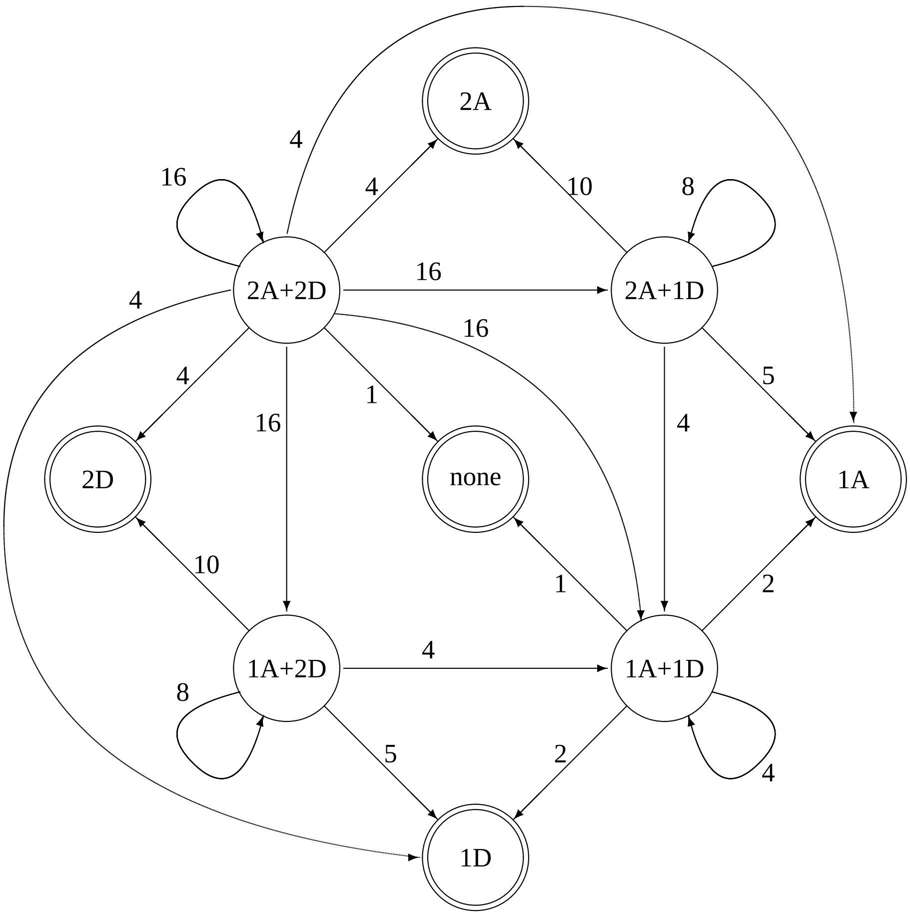
The total is 81 ($(6^4)/16$). So one round of battle will have, for example, 16/81 chance of ending with one survivor on each side. If we assume noone retreats until a loss is taken, we can ingore the first row (2A+2D) to find the probabilities of the results after at least one loss is taken on either side (or both). The new total is $81 - 16 = 65$ so, for example, the chance of ending with 1A+1D after the first loss is 16/65. If the battle is fought to completion (without any retreats), we must consider the further battle if we end with 1A+1D, 2A+1D or 1A+2D. We already have the numbers for 1A+1D, so we need to consider 2A+1D and 1A+2D. 2A+1D gives the table:

Defender	Attacker roll			
roll	1–4/1–4	1–4/5–6	5–6/1–4	5–6/5–6
1–4	2A+1D	2A	2A	2A
5–6	1A+1D	1A	1A	1A

which summarizes to:

2A+1D	8
1A+1D	4
2A	10
1A	5

out of a total of 27 ($(6^3)/8$) outcomes. If we ignore the "no effect" result of 2A+1D, the total is 19. The 1A+2D case is symmetric (we just swap A and D), so we can combine the results into the diagram below. Double circles indicate "final states" (no more battles). The number on an arrow indicate the number of occurrences of going from the origin state to the destination state, including "self-transitions".



We can use ths diagram for several different calculations. We have already seen how we can find the probabilities for a single round of battle by dividing the number on each outgoing transition by the total on all outgoing transitions, for example showing that the chance of all being killed in the first round of battle in a 2:2 fight is 1/81. Similarly, we can find the result of battling until the first loss(es) by dividing only by the total of the transitions *except* the self-transitions. This gives, for example, a chance of 1/65 for a 2:2 battle stopping at no survivors after the first loss.

We can also use the diagram to find the probabilities of each outcome when a battle is fought to completion with no retreats on either side. To do this, we first find the probability of each non-self transition by dividing the numbers on non-self transitions by the sum of numbers on the non-self transitions out of the same state (like we did above to find the results after first loss). So, for example, each number out of 2A+2D is divided by 65 and each number out of 2A+1D is divided by 19. Now find all paths (not using self-transitions) from the initial state to each final (double-circled) state, multiplying the probabilities on the trasitions of each path and adding all the products that lead to the same final state. As each non-self transition reduces the number of armies, all paths are finite and there are only a finite number of these.

Example

Starting from 2A+2D, we can get to 2A either directly with probability 4/65 or through 2A+1D with probability $16/65 \times 10/19$ for a total probability of $4/65 + 16/65 \times 10/19 = 236/1235$. Due to symmetry, this is the same as the probability of ending in 2D.

Again from 2A+2D, we can get to 1A either directly with probability $4/65$, trough 2A+1D with probability $16/65 \times 5/19$, through 1A+1D with probability $16/65 \times 2/5$ or through either 1A+2D or 2A+1 and then 1A+1D each with probability $16/65 \times 4/19 \times 2/5$. This adds up to $4/65 + 16/65 \times 5/19 + 16/65 \times 2/5 + 2 \times 16/65 \times 4/19 \times 2/5 = 1644/6175$. 1D is symmetric, so it has the same probability.

From 2A+2D, we can get to "none" directly at probability $1/65$, through 1A+1D with probability $16/65 \times 1/5$ or through either 1A+2D or 2A+1D and then 1A+1D each with probability $16/65 \times 4/19 \times 1/5$. This adds up to $1/65 + 16/65 \times 1/5 + 2 \times 16/65 \times 4/19 \times 1/5 = 527/6157$.

So we get the following table of final states and probabilities:

2A	1180/6175
2D	1180/6175
1A	1644/6175
1D	1644/6175
none	527/6157

Converting to percentages, we get:

2A	19.11%
2D	19.11%
1A	26.62%
1D	26.62%
none	8.53%

By starting at 2A+1D, 1A+2D or 1A+1D we can get probabilities for battles at these odds as well. If we need larger number of armies, difficult terrain, leaders or Romans/cavalry, we need to make new diagrams like the above. This is not terribly complicated, just a lot of work (and error-prone). Hence, it makes sense to make a program to do the calculations.

4 Making a program

The purpose of the program is, given the number of attackers and defenders in a battle, to determine the probability of each possible final result should the battle be fought to completion.

To do this we will set up a diagram similar to the above and calculate a probability for each node. The probability for the starting node is 1 and for the remaining nodes the probability is found as the sum of the probabilities of each possible predecessor, each multiplied by the probability of the transition from the predecessor to the node in question.

Given that the battle started with A attackers and D defenders, we can write the following equations for the probability og getting to a attackers and d defenders during the battle:

$$
\begin{array}{lcll}
p(A,D) & = & 1 & \\
p(a,d) & = & \sum_{i,j=a,d}^{A,D} p(i,j) \times t(i,j,a,d) & \text{if } (a,d) \neq (A,D) \\
\\
t(i,j,a,d) & = & 0 & \text{if } (i,j) = (a,d) \\
t(i,j,a,d) & = & 0 & \text{if } i < j-d \text{ or } j < i-a \\
t(i,j,a,d) & = & \frac{q(i,j,d) \times q(j,i,a)}{6^{(i+j)} - 4^{(i+j)}} & \text{otherwise} \\
\\
q(i,j,0) & = & \sum_{m=j}^{i} 2^m \times 4^{(i-m)} \times \binom{i}{m} & \\
q(i,j,d) & = & 2^{(j-d)} \times 4^{(i-j+d)} \times \binom{i}{j-d} &
\end{array}
$$

The first equation just says that the initial state is certain. The second adds up the probabilities of predecessors as described above. $t(i,j,a,d)$ is the probability of getting from (i,j) to (a,d). The first rule for this excludes self-transitions. The second excludes transitions where there are not enough attackers to kill as many defenders as the transition indicates or vice-versa. The third rule has the "meat" of the calculation. It calculates how many rolls can get you from (i,j) to (a,d) and then divides this by the total number of rolls *excluding* the number of rolls that don't kill anything. $q(i,j,d)$ calculates how many ways i armies can reduce j opponents to d. There is a special case for $d = 0$ to handle "overkills", i.e., having more kills on the i dice than required to reduce j to 0.

$\binom{n}{m}$ is the number of ways you can pick m out of n items, and can be calculated as $n!/m!/(n-m)!$, where $n!$ is the factorial of n.

The probabilities above are calculated using the standard rule of kills on 5-6, so 2 out of 6 are kills and 4 out of 6 aren't. Hence, the use of 2 and 4 in the formula. If ak out of 6 kills for the attacker and dk out of 6 kills for the defender, we can use the generalized equations below. q has been split into qa and qd, as they must use different probabilities for kills for attacker and defender.

$$
\begin{aligned}
p(A,D) &= 1 && \\
p(a,d) &= \textstyle\sum_{i,j=a,d}^{A,D} p(i,j) \times t(i,j,a,d) && \text{if } (a,d) \neq (A,D) \\
t(i,j,a,d) &= 0 && \text{if } (i,j) = (a,d) \\
t(i,j,a,d) &= 0 && \text{if } i < j-d \text{ or } j < i-a \\
t(i,j,a,d) &= \frac{qa(i,j,d) \times qd(j,i,a)}{6^{(i+j)} - (6-ak)^i \times (6-dk)^j} && \text{otherwise} \\
qa(i,j,0) &= \textstyle\sum_{m=j}^{i} ak^m \times (6-ak)^{(i-m)} \times \binom{i}{m} && \\
qa(i,j,d) &= ak^{(j-d)} \times (6-ak)^{(i-j+d)} \times \binom{i}{j-d} && \\
qd(j,i,0) &= \textstyle\sum_{m=i}^{j} dk^m \times (6-dk)^{(j-m)} \times \binom{j}{m} && \\
qd(j,i,a) &= dk^{(i-a)} \times (6-dk)^{(j-i+a)} \times \binom{j}{i-a} &&
\end{aligned}
$$

We can code the equations fairly directly in the programming language Haskell (see `http://www.haskell.org`). I use the Hugs implementation of Haskell, as it is portable and easy to use.

```
p0(ak,dk,aa,dd,a,d) = p(a,d) where

  p(a,d) | (a,d)==(aa,dd) = 1
  p(a,d) = sum [p(i,j)*t(i,j,a,d)
               | i<-[a..aa], j <-[d..dd], (i,j)/=(a,d)]

  t(i,j,a,d) | i<j-d || j<i-a = 0
  t(i,j,a,d) = qa(i,j,d) * qd(j,i,a) / (6^(i+j) - (6-ak)^i * (6-dk)^j)

  qa(i,j,0) = sum [ak^m * (6-ak)^(i-m) * k(i,m) | m<-[j..i]]
  qa(i,j,d) = ak^(j-d) * (6-ak)^(i-j+d) * k(i,j-d)

  qd(j,i,0) = sum [dk^m * (6-dk)^(j-m) * k(j,m) | m<-[i..j]]
  qd(j,i,a) = dk^(i-a) * (6-dk)^(j-i+a) * k(j,i-a)

  k(n,m) = fromInt(product [n-m+1..n] `div` product [1..m])
```

`aa` and `dd` are used instead of *A* and *D*, as variables can't start with capital letters. To calculate the probability of reaching `a` attackers and `d` defenders when starting from `aa` attackers and `dd` defenders when the attackers kill on `ak` different numbers and the defenders kill on `dk` different numbers, you just call `p0(ak,dk,aa,dd,a,d)`.

The program isn't very efficient, as it will recalculate `p` for the same values several times, so you must be patient when `aa+dd` is greater than 10, though. You can speed up the calculation by avoiding recomputation. This is done by storing the values of p(a,d) in a data structure and look up in this instead of recomputing. We can do this by rewriting the equations for p(a,d) as follows:

```
p(a,d) | (a,d)==(aa,dd) = 1
p(a,d) = sum [(pp!!i!!j)*t(i,j,a,d)
              | i<-[a..aa], j <-[d..dd], (i,j)/=(a,d)]

pp = [[p(a,d)|d<-[0..dd]]|a<-[0..aa]]
```

The other equations are unchanged. Note that the recursive call to p has been replaced by the lookup (pp!!i!!j). Now you can compute for all realistic battle sizes in reasonable time.

5 A few sample cases

For those of you too lazy to run the above program yourself, I have computed the results of a few common battles. The 1:1 cases were covered in the beginning, so these involve multiple armies.

3A vs. 2D in normal terrain:

3A	26.6%
2A	36.3%
1A	18.8%
none	3.7%
1D	9.8%
2D	4.9%

4A vs. 2D in normal terrain:

4A	31.7%
3A	40.4%
2A	18.8%
1A	5.0%
none	0.9%
1D	2.2%
2D	0.9%

2A vs. 1D in difficult terrain:

2A	37.9%
1A	29.7%
none	5.4%
1D	26.9%

3A vs. 2D in difficult terrain:

3A	13.9%
2A	22.5%
1A	15.2%
none	3.7%
1D	23.0%
2D	21.6%

4A vs. 2D in difficult terrain:

4A	18.6%
3A	29.2%
2A	20.0%
1A	9.2%
none	2.1%
1D	12.1%
2D	8.9%

1Roman vs. 2D in normal terrain:

1R	38.0%
none	7.6%
1D	31.0%
2D	23.4%

6 Conclusion

The general method for calculating probabilities extend also to battles with mixed armies (e.g., with forts or cavalry), but you need to keep track of the numbers of each type of army and the different dice these use, so it is a bit more work. The program can also be extended to handle these cases, but it will add considerably to its complexity.

An alternative to calculating exact probabilities like shown above is to simulate a large number of battles and count the occurences. This is often simpler to program and if you run a sufficiently large number of battles (a few million should do), you can get results that are fairly close to the exact probabilities. This isn't useful for calculation by hand, though.