A VARIANT ON THE BLOTTO GAME

JULIEN GODAWATTA

This is a report which discusses, with the aid of simulations in Python, the following variant on the traditional Blotto game:

> 10 castles numbered 1 to 10 are worth 1, 2, 3 ... 10 points respectively. We are given 100 soldiers to allocate between the castles. Our opponent independently does the same. Castles are fought in order from 1 to 10. The number of soldiers on each castle is then compared, and for each castle, whoever has the most soldiers on that castle wins its points (no points in case of a tie). If a player wins 3 consecutive castles, they automatically win all the remaining Castles too.

Beyond the enormous number possible distributions, I see two main difficulties:

(a) any strategy will admit a counter-strategy that beats it on average; as such, any sensible solution will rely on an assumption on the expected 'depth of reasoning' of competitors. Much like the famous 'guess $\frac{2}{3}$ of the average' game, iterating too far across counter-strategies leads to a bad solution *in practice*: one that is either too risky or too conservative. For instance if one were convinced on putting all soldiers in 3 castles, starting with the naïve $[33, 33, 34, 0, \dots]$, one might follow this train of thought:

$$[33, 33, 34, 0, \dots] \longrightarrow [0, 34, 35, 31, \dots] \longrightarrow [0, 0, 36, 32, 32, \dots]$$

which can then lead to reverting to the beginning to win all castles, with a less even spread such as $[28, 35, 37, 0 \dots]$. One can then cycle through again:

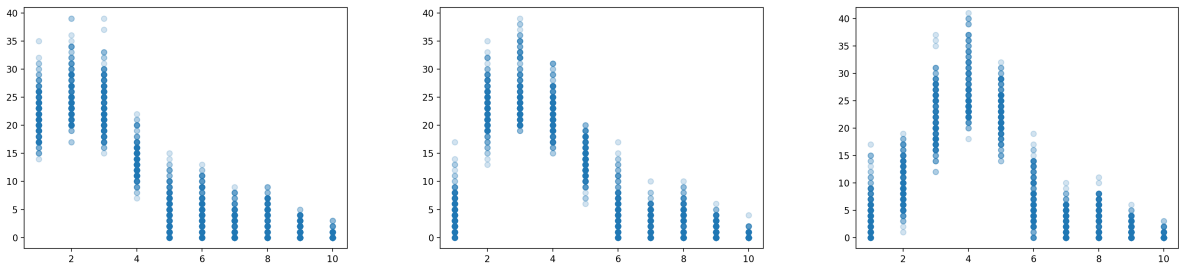
$$[28, 35, 37, 0 \dots] \longrightarrow [0, 36, 38, 26, 0 \dots] \longrightarrow [0, 0, 39, 27, 34, \dots] \longrightarrow [23, 37, 40, 0, \dots] \longrightarrow \cdots$$

and so on, achieving progressively disproportionate solutions.

(b) the game is extremely sensitive to small changes in scores. For instance, while $(27, 38, 35, 0, 0 \cdots)$ wins against $(30, 36, 34, 0, \dots)$ at 5-to-1, moving 1 soldier for the first player to get $(26, 38, 35, 1, 0 \cdots)$ will then win 54-to-1.

I begin by considering two base strategy directions that I call **Gr** for 'greedy', and **Pr** for 'preventative', whose details are as follows (C_i indicates castle i):

Gr strategies: the crucial rule on 3-consecutives steers most of the attention on the first castles. **Gr** strategies aim to secure 3 consecutive wins on these early castles, making for a swift high-score win. Every such arrangement overlaps at C_3, making it an import focal point: I suspect many **Gr**-thinkers are likely to have C_3 as their most populated castle, though a clever opponent may very well purposefully skew the peak to the right or left to escape predictability. To reflect this I denote by **Gr**$[i]$ the strategy which places a peak number of soldiers on castle C_i for $i = 2, 3, 4$, aiming for 3 consecutives. I am making the assumption that very few players would peak at C_1 or C_5. In fact, because focusing on castles C_1-C_3 leaves a lot of the row unguarded, **Gr**$[2]$ is a high-risk high-reward plan, which I would expect to be the least popular as result. Finally, to reduce risk-taking, players will want to pad out an amount of soldiers around the area of focus, with a tapering into later castles to get an edge of an opponent if the fight for C_1-C_5 was close. I modelled these in Python likewise:

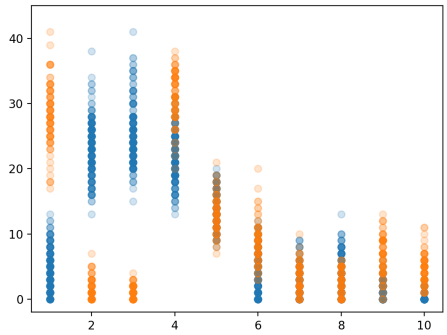


Gr$[2]$ **Gr**$[3]$ **Gr**$[4]$

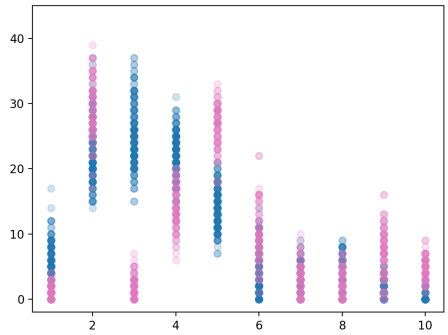
Despite being a 'first port of call', this is actually a very effective approach, and wins against opponents that are too conservative. For instance, simulated against 100,000 semi-randomised soldier allocations, the distribution $[4,\ 26,\ 40,\ 25\ ,3,\ 0,\ 0,\ 0,\ 0,\ 0]$ won 99.311% of its battles with an average score of 53.839.

Pr strategies: the strategies above are able to counter themselves, but due to a high concentration in the first half of the game, sacrifice late-game attack. An interesting response to consider is therefore one which attempts to block **Gr** whilst saving material for castles further down which **Gr** has left poorly populated. This is a risky

idea because **Gr** is versatile, but if done very agressively can be quite successful, especially against a small pool of opponents. **Pr** deliberately avoids C_3 whilst emphasising neighbouring castles, particularly ones expected to be defended only mildly well. Two viable ways of achieving this are to heavily attack C_1 & C_4, or C_2 & C_5, which I denote **Pr**[1] and **Pr**[2]. **Pr**[1] has a very slight edge by placing its attacks earlier. Modelling in Python, these would look like the following against **Gr**[3] for instance:



Pr[1] **vs Gr**[3]



Pr[2] **vs Gr**[3]

Note: small amounts can be left in earlier castles to get an edge over other **Pr**-players. Depending on remaining resources from C_1-C_5, a conservative player may wish to put some effort into C_9 and C_{10} should **Gr**-**Pr** fight closely through C_5-C_8, though arguably the most sensible plan is to emphasise play on the first half nonetheless.

I would not expect all opponents to neatly fall into one of the strategies above, but I would assume the vast majority would broadly be basing their distribution on one of those ideas. Because I am fighting a large group people, I should create an arrangement that responds all of the ideas above sufficiently well. The aim is to secure a strong *average*, rather than maximise wins. As a result, I choose to opt for a **Gr** strategy, since **Pr** blocks very efficiently but does not gather as many marks on average (if I had very few opponents, I might opt for that instead). I decided to run a series of tests in Python to generate some good responses, by creating sets of distributions of the form:

$$\lambda_1\mathbf{Gr}[2] + \lambda_2\mathbf{Gr}[3] + \lambda_3\mathbf{Gr}[4] + \lambda_4\mathbf{Pr}[1] + \lambda_5\mathbf{Pr}[2] + \lambda_6\mathbf{Bnd} + \lambda_7\mathbf{Rnd}$$

where **Gr**[i], **Pr**[j] are large collections of distributions following ideas outlined above, **Bnd** is a set of semi-random blend of those strategies, **Rnd** is a 'noise' set of completely random (uniform) distributions of the soldiers, and λ_i are ratios of these collections with $\sum\lambda_i = 1$. I ran tests for varying ratios. The **Rnd** ratio was kept low as it is less relevant, and most good distributions will beat almost all random distributions. The following table shows good options, where the entries represent the average number of points gained per game, over around 5,000 games against each group.

Distribution	**Gr**[2]	**Gr**[3]	**Gr**[4]	**Pr**[1]	**Pr**[2]	**Bn**
[3, 2, 27, 35, 29, 1, 1, 2, 0, 0]	42.842	43.022	42.084	46.266	41.545	42.872
[0, 4, 28, 35, 31, 2, 0, 0, 0, 0]	45.574	42.079	41.359	46.128	48.851	44.134
[2, 3, 27, 36, 29, 2, 0, 1, 0, 0]	42.857	43.24	40.717	49.496	40.782	43.954

These are all very much **Gr**[4]. The middle option is the boldest and most successful in my tests, by keeping soldiers very close. 'Keep it simple, soldier', as one might say.