Let's Summon Math (temp title)

Nora Lambert

December 26, 2025

Abstract

This is not a serious or professional article, just the rambling of an ex-math student.

1 Introduction

Let's Summon Demons is a board game created by Steven Rhodes Games where children and their pets sacrifice themselves to summon demons. I bought it a few months ago on a whim and ended up loving it. My roommates and I played it a lot and experimented with the multiple game modes, be it in 1v1, 4 player, starting with a demon, increasing the number of candles, etc. I want to explore the mathematics of this game, see if you can evaluate the power of cards objectively and maybe even identify a metric that might help win games.

This project is mainly for me to learn LaTeX. Somehow the girl with a bachelor in mathematics has never used LaTeX! Although, this can also be viewed as a way for me to justify having spent the time and money to get such a degree in the first place.

The following report should be accessible to anyone with knowledge of 1st year probability and calculus. Having played the game is useful but not necessary.

2 Rules of the Game

Players will start with 5 souls and a random candle card each. The *block deck* and *demon deck* are shuffled, 5 block cards are put in the center of the table face-up (this is the *Block*) while 3 demons are distributed to every player face-down. Players may look at their demons at any moment.

Every turn, a player can buy a face-up card from the *Block*, snuff 3 cards to summon a demon and must roll the 2 dice. Any player that has cards with the number corresponding to the totals of the dice can activate their effects. These will have various effects, such as gaining souls, card, demons or other unique effects.

To win, a player must possess at least 3 demons and at least 10 souls.

A player can order the three available actions during their turn in any way they choose, but cannot pass the turn without having rolled the dice. When a card is acquired from the *Block*, either by buying or by an effect, it is not replaced with a new one until the end of the turn. Also, if multiple effects are activated with the same roll, players proceed in turn order, ordering their effects as they wish.

3 Contents of the Deck

Here are the contents of the game:

- 5 candle cards
- 100 block cards
- 20 demon cards

A more detailed list of card will be useful for this analysis. Here are in figure 1 every card with their count and characteristics. Followed by figure 2 and 3 which breakdown the counts per type and subtypes.

Card Name	Copies	Number	Type	Subtype
Adam	6	8	Boy	-
Alice	2	4	Girl	Sweet
Alligator	2	4	Animal	-
Annie	2	9	Girl	Rotten
Ara Macao	2	7	Animal	-
Calvin	2	9	Boy	Rotten
[insert rest of table here]	[...]	[...]	[...]	[...]
Skunk	1	7	Animal	-

Figure 1: Contents of the Block Deck

Card Type	Copies	Ratio
Boy	34	.34
Girl	34	.34
Animal	32	.32
Total	100	1

Figure 2: Counts of the Block Deck per Card Type

Card Type	Copies	Ratio
Rotten	24	.24
Sweet	24	.24
Untyped	52	.52
Total	100	1

Figure 3: Counts of the Block Deck per Card Subtype

4 Dice Probabilities

4.1 Sum of Dice

Will need to summarize this article and cite it.

	1	**2**	**3**	**4**	**5**	**6**
1	2	3	4	5	6	7
2	3	4	5	6	7	8
3	4	5	6	7	8	9
4	5	6	7	8	9	10
5	6	7	8	9	10	11
6	7	8	9	10	11	12

Figure 4: Totals of 2 Dice

Before we can begin to evaluate cards, we need to be certain of the odds of these cards activating. In *Let's Summon Demons*, every player rolls a pair of 6-sided dice and sums up the total. Cards with

the corresponding number will then activate. We could take the time to write out the usual matrix of sums and manually calculate the odds (as seen in figure 4), but since we are already over-analyzing a simple board game, we will seek a more generic solution. The following section is a summary of Tom Leyson's article found here.

Lets first layout the basics. We will be using $\binom{a}{b} = \frac{a!}{b!(a-b)!}$ to shorthand combinatorials with $a, b \in \mathbb{N}$. We will also label each face x^i, where i is the number on that side. We can then define a lot of n 6-sided dice with a multinomial.

$$f(x,n) = (x^1 + x^2 + x^3 + x^4 + x^5 + x^6)^n \tag{1}$$

Let $n = 2$ and observe the resulting equation:

$$(x^1 + x^2 + x^3 + x^4 + x^5 + x^6)^2 \tag{2}$$

$$= x^2 + 2x^3 + 3x^4 + 4x^5 + 5x^6 + 6x^7 + 5x^8 + 4x^9 + 3x^{10} + 2x^{11} + x^{12} \tag{3}$$

Treating again the exponent as the sum of our dice, we now get the count of combinations for each total in the coefficient as seen in figure 4. We can use a more general equation to represent the sums of n s-sided dice. The probability of getting any factor will be given by taking the coefficient multiplied with the probability of getting this total $1/s^n$.

$$f(x,n) = (x(\sum_{i=0}^{s-1} x^i))^n \tag{4}$$

Now we can express the sum as a geometric series.

$$\sum_{i=0}^{s-1} x^i = \frac{1-x^s}{1-x} \tag{5}$$

Substituting in equation 4 we get.

$$(x(\sum_{i=0}^{s-1} x^i))^n = (x\frac{1-x^s}{1-x})^n \tag{6}$$

$$= x^n(1-x^s)(1-x)^{-n} \tag{7}$$

We now have two binomial terms that can be expanded as binomial series. We will be using the following identity.

$$(p+q)^n = \sum_{k=0}^{n} \frac{n!}{(n-k)!k!} p^k q^{n-k} \tag{8}$$

By substituting we get:

$$(1-x^s)^n = \sum_{i=0}^{n} \frac{n!}{(n-k)!i!}(-x^s)^i 1^{n-i} \tag{9}$$

$$= \sum_{i=0}^{n} \frac{n!}{(n-i)!i!}(-1)^i x^{si} \tag{10}$$

$$= \sum_{i=0}^{n} \binom{n}{i}(-1)^i x^{si} \tag{11}$$

For the 3rd term, $(1-x^s)^{-n}$, we will need to use another tool since the exponent is negative. Using the Maclaurin series expansion of the function, we will be able to expand it,

$$f(x) = f(0) + \frac{f'(0)}{2!}x + \frac{f''(0)}{3!}x^2 + \frac{f'''(0)}{4!}x^3 + ... \tag{12}$$

We now just need to find the derivatives of $(1-x^s)^{-n}$, which are as follows.

$$f(x) = \frac{1}{(1-x)^n} \tag{13}$$
$$f'(x) = \frac{n}{(1-x)^{n+1}} \tag{14}$$
$$f''(x) = \frac{n(n+1)}{(1-x)^{n+2}} \tag{15}$$
$$f'''(x) = \frac{n(n+1)(n+2)}{(1-x)^{n+3}} \tag{16}$$
$$... \tag{17}$$
$$f^{(j)}(x) = \frac{(n+j-1)!}{(n-1)!}(1-x)^{n+j} \tag{18}$$
$$f^{(j)}(0) = \frac{(n+j-1)!}{(n-1)!} \tag{19}$$

Lets substitute this into our series.

$$(1-x)^{-n} = \frac{1}{(1-0)^n} + \frac{n}{2!(1-0)^{n+1}}x + \frac{n(n+1)}{3!(1-0)^{n+2}}x^2 + ... \tag{20}$$
$$= 1 + \frac{n}{2!}x + \frac{n(n+1)}{3!}x^2 + ... \tag{21}$$
$$= \sum_{j=0}^{\infty} \frac{(n+j-1)!}{j!(n-1)!}x^j \tag{22}$$
$$= \sum_{j=0}^{\infty} \binom{n+j-1}{j} x^j \tag{23}$$

Substituting 11 and 23 into equation 7, we get:

$$f(x,n) = x^n(1-x^s)(1-x)^{-n} \tag{24}$$
$$= x^n(\sum_{i=0}^{n} \binom{n}{i}(-1)^i x^{si})(\sum_{j=0}^{\infty} \binom{n+j-1}{j} x^j) \tag{25}$$

Now we include every possible exponent of x^T. Where T is the sum of our dice.

$$T = n + si + j \tag{26}$$

see This **(This probably should be more justified ngl)**
Substituting $j = T - n - sk$ in equation ... we get:

$$f(x,n,T) = \sum_{i=0}^{n} \binom{n}{i}(-1)^i \sum_{j=0}^{\infty} \binom{n+T-n-si-1}{T-n-si} x^T \tag{27}$$
$$= \sum_{i=0}^{n} \binom{n}{i}(-1)^i \sum_{j=0}^{\infty} \binom{T-si-1}{T-n-si} x^T \tag{28}$$
$$= \sum_{i=0}^{n} \binom{n}{i}(-1)^i \sum_{j=0}^{\infty} \binom{T-si-1}{n-1} x^T \tag{29}$$

But we need to consider that $T - si - n > 0 => i < (p-n)/s$ and since $i \in \mathbb{N}$ the sum's maximum becomes $\lfloor (T-n)/s \rfloor$ where $\lfloor x \rfloor$ is the floor function.

$$= \sum_{i=0}^{\lfloor (T-n)/s \rfloor} \binom{n}{i} (-1)^i \sum_{T-si-n=0}^{\infty} \binom{T-si-1}{n-1} x^T \tag{30}$$

Since T, s and n do not vary and i varies in the first sum we can remove the second summation.

$$= \sum_{i=0}^{\lfloor (T-n)/s \rfloor} (-1)^i \binom{n}{i} \binom{T-si-1}{n-1} x^T \tag{31}$$

Assuming the dice are fair with probability $1/s$ for each face, we can express the probability of rolling a sum T as the product of the multinomial coefficient and the probability of each face to the power of number of dice rolled n.

$$P(n, s, T) = (\frac{1}{s})^n \sum_{i=0}^{\lfloor (T-n)/s \rfloor} (-1)^i \binom{n}{i} \binom{T-si-1}{n-1} x^T \tag{32}$$

4.2 Macro Implementation

Show work of the custom function in Libreoffice Calc maybe include equivalent code for Excel too.

This project was done using LibreOffice Calc, the following macros were created to enable the calculations in Basic. First, we implemented a simple factorial function as Libreoffice doesn't have a built-in option (as far as we know).

```
Function Factorial(n As Integer) As Double
    Dim result As Double
    Dim i As Integer

    If n < 0 Then
        Factorial = 0
        Exit Function
    End If

    result = 1
    For i = 1 To n
        result = result * i
    Next i

    Factorial = result
End Function
```

Second, we created a function that could be used directly in the spreadsheet's cells for the probability distribution of the sum of n s-sided dice. This will be referred as **D_DICETOTAL**, which takes as parameters n dice, s faces and a total T. The code is as follows.

```
Function D_DiceTotal(n As Integer,s As Integer,T As Integer)
        Dim k as integer
        dim highLimit as integer
        dim preSum as double
        dim combinatory as double
        dim mclauren as double
        dim prob as double

        highLimit = Int((T–n)/s)

        for k = 0 to highLimit
```

```
            combinatory = Factorial(n)/(Factorial(n-k)*Factorial(k))
            mclauren = Factorial(T-s*k-1)/(Factorial(T-s*k-n)*Factorial(n-1))
            preSum = preSum + ((-1)^k)*combinatory*mclauren
        next k

        prob = preSum*(1/s)^n

        D_DiceTotal = prob
End Function
```

Finally, to be thorough, we added the cumulative probability function of **D_DICETOTAL**, named **F_DICETOTAL**, which takes the same parameters. The function has some safety checks and sums multiple calls of **D_DICETOTAL**. This can be fairly computationally expensive with bigger parameters, optimizing the function could be the subject of more research.

```
Function F_DiceTotal(n As Integer,s As Integer,T As Integer)
        dim k as integer
        dim preSum as double

        preSum =0

        if n<0 then
            F_DiceTotal = 0
            Exit function
        elseif s<0 then
            F_DiceTotal = 0
            Exit function
        elseif T<0 then
            F_DiceTotal = 0
            Exit function
        end if


        if   T>n*s then
            F_DiceTotal = 1
            Exit function
        end if

        if   T<n then
            F_DiceTotal = 0
            Exit function
        end if


        for k = n to T
            preSum = preSum + D_DiceTotal(n,s,k)
        next k

        F_DiceTotal = preSum

End Function
```

The results of these functions can be seen in the spreadsheet included with this report.

5 Expected Value

5.1 Metrics

5.1.1 Souls

Souls are the principal resources of *Let's Summon Demons*. The players start with 5 and can spend 3 souls to acquire card from *The Block* which will generate more souls, accelerating the speed at which they can access more cards. This will be our base unit for this analysis. Examples of cards that generate souls are the starting candles, Adam and the Owl.

5.1.2 Cards

Cards are the 2nd type of resource we will need to account in this analysis. They are a "refined" resource, as in they are acquired by consuming lower worth resources, in this case souls. Since cards can be bought for 3 souls, the base value of a card from the *Block Deck* will be 3 souls. Some cards can get more cards with their effect, examples of such cards include the Cat, Alice and Annie.

5.1.3 Demons

Demons are the most refined resource available in this game. They also are a win condition, since when a player has obtained 3 demons and 10 souls, this player wins the game. Since demons are made by snuffing 3 cards, they will inherit the value of 9 souls. Some effects in this game can make demons without having to use the base mechanic of snuffing. Damien, Deguelivre and the Goat are examples of this kind of effect.

5.2 Modifiers and Conditions

5.2.1 Stealing

Most resource gains in this game are in the form of generating resources, but in rare cases effects will allow players to steal from each other. Souls, cards and even demons can be stolen. Stealing can be very powerful, as it lets a player slow down it's opponents while still advancing their game state. This potency is dependent on the number of opponents, stealing in a 1v1 game is much more powerful as there cannot be another opponent that advances while the other is slowed. In 3-5 player games, the stealing effect might also be spread out, reducing the effective hurdle on opponents. -+Therefore, we use this stealing multiplier m_s which takes into account the number of active players in a game.

Let N_p be the number of active players in the game.

$$m_s = 1 + 1/(N_p - 1) \tag{33}$$

5.2.2 Discarding

Discarding a card means removing it from a player's board and putting it into the discard pile. If an effect involves removing on of the perspective player's card, the expected value of the effect will have a penalty applied. If the effect discards an opponent's card, a bonus will be added. As in the previous section, the value of removing an opponent's card is modulated by number of players in the game N_p. Since cards are worth 3 souls since thats for how much they can be bought, we will include 3 as a conversion factor. This value d_{value} will be calculated as follow:

Let c_{opp} be the number of cards removed from opponents and c_{player} be the number of cards removed from the perspective player.

$$d_{value} = 3(\frac{c_{opp}}{N_p - 1} - c_{player}) \tag{34}$$

5.2.3 Demon's Effect

Summoning demons is how this game is won. Demons also have some very powerful effects, but they have a downside: their effects only activate if the owner of the card rolls the specific number. Since the

roll and the current player are independent, we have the probability of activating p_{demon} be calculated as follows:

$$p_{demon} = \mathbb{P}(X = x, CurrentPlayer = Owner) \tag{35}$$
$$= \mathbb{P}(X = x)\mathbb{P}(CurrentPlayer = Owner|X = x) \tag{36}$$

Since the total of the dice and the current player's turn are independent (we will consider an exception in section 5.4.5).

$$= \mathbb{P}(X = x)\mathbb{P}(CurrentPlayer = Owner) \tag{37}$$
$$= \mathbb{P}(X = x)/N_p \tag{38}$$

5.2.4 Conditional Effects

A classic game design trick is to put conditions or restrictions on powerful effects. This rewards players for planning around the conditions or having a big picture strategy. In this analysis, we will represent the expected value of a conditional effect this way:

$$E[Card] = \begin{cases} \sum x\mathbb{P}(X = x) \text{ if the condition is met} \\ 0 \text{ otherwise} \end{cases}$$

Conditions will be explored more in section 7 when we will need to consider the game state to evaluate cards.

5.2.5 Other

5.3 Evaluating Archetypes

There are multiple cards in this game who share very similar effects. We will group such effects in *archetypes* to avoid redundancy. A full list of the cards in every archetype with their expected values will be included at the end of each subsection.

5.3.1 Simple Soul Generators

The most basic type of effect you can find in this game are the soul generators. A card is a **soul generator** if it ONLY outputs souls. Such a generator is said to be *simple* if the amount of generated souls does not depend on the game state. Calculating such a card's expected value is simple.

Let x be the number on the card, $\mathbb{P}(X = x)$ be the probability of this total on two die (see section 4.1) and n the number of souls created by this effect.

$$E[Generator] = n\mathbb{P}(X = x) \tag{39}$$

Stealing
If the effect *steals* souls from another player (see section 5.2.1) the equation needs to include the stealing multiplier m_s (equation 33).

$$E[Generator] = m_s n\mathbb{P}(X = x) \tag{40}$$

Discarding
If the effect *discards*, which is typically in cases of simple generators, itself. Then we need to include the discard value d_{value} (equation 34).

$$E[Generator] = (n + d_{value})\mathbb{P}(X = x) \tag{41}$$

Stealing and Discarding
If an effect steals and discards at the same time, we combine the additions from 40 and 41.

$$E[Generator] = m_s(n + d_{value})\mathbb{P}(X = x) \tag{42}$$

Glen will not be present in the table of EVs as the card gives away a soul when activated. We will discuss this special case in section 5.4.1

Card Name	Probability	Gains (Souls)	EV (Souls)	Discard	Steal	Cond
Adam	.1389	1	0.1389			
Ara Macao	.1667	2	0.3333			x
Calvin	.1111	2	0.4444*		x	x
Caroline	.0278	5	0.1389			
Carrie	.0556	2	0.1111			
Chuck	.0833	2	0.1667			x
Dog	.1667	1	0.1667			x
Dillinger	.0833	1	0.1667*		x	
Eve	.1389	1	0.1389			
Goldfish	.1667	5	0.3333	x		
Jesus	.056	2	0.1111			
Louis	.1111	1	0.1111			
Owl	.1667	4	0.1667*	x		
Rabbit	.1667	2	0.3333			x
Rabid Dog	.1667	2	0.3333			x
Regan	.1111	1	0.1111			
Sam	.0833	6	0.2500	x		x

Figure 5: EV breakdown for every simple soul generator from the Block

Card Name	Probability (2 player game)	Gains (Souls)	EV (Souls)	Discard	Steal	Cond
Antechrist	.0416	5	0.4167*		x	
Porcus	.0556	5	0.2778			

Figure 6: EV breakdown for every simple soul generator demons

5.3.2 Complex Soul Generators

Some soul generators do not give a static number of souls as we saw in section 5.3.1. Cards like Lola, Romeo and Donnie give a number of soul s based on a variable in the game state. This part is mandatory for section 7 where we will put to use our analysis. The EV of such effects will be very similar to section 5.3.1, but the number of souls n will be replaced by the conditional count s. The results table below will include the variable that gives s its value.

$$E[Generator] = s\mathbb{P}(X = x) \tag{43}$$

Stealing

$$E[Generator] = m_s s\mathbb{P}(X = x) \tag{44}$$

Discarding

$$E[Generator] = (s + d_{value})\mathbb{P}(X = x) \tag{45}$$

Stealing and Discarding

$$E[Generator] = m_s(s + d_{value})\mathbb{P}(X = x) \tag{46}$$

It is important to note that complex soul generators are inherently conditional, as s can and will often be 0 in game states.

Card Name	Probability	Value of s	Discard	Steal
Donnie	.0278	count of rotten children possessed		x
Fifi	.0833	count of animals possessed		
Lisa	.0556	count of sweet children possessed		
Lola	.1389	count of boys possessed		
Romeo	.1389	count of girls possessed		

Figure 7: EV breakdown for every complex soul generator

Card Name	Probability (2 player game)	Value of s	Discard	Steal
Molosse de Bael	.0695	count of animals possessed		x
Nounou sommes legion	.0695	count of cards possessed		

Figure 8: EV breakdown for every complex soul generator demon

5.3.3 Card Generators

Another very common type of card found in this game is *card generators*. These are effects that let a player directly acquire cards, either from the Block or the Deck without having to pay souls. These are some of the most powerful effects in the game and even let players bypass the notion of *tempo* that we will explore in section 6. The following formulas will assume no information from the deck, but for optimal play a player must keep count of every publicly available card to know the current composition of the deck.

Card generators can check either the top of the deck or the Block. We will start with analyzing the former. Since we simultaneously reveal the top cards without replacement, we can use the typical hypergeometric formula to model the probabilities of success.

Let n be the number of cards the effect reveals from the top of the deck, N the number of cards in deck, K the number of successes in the deck and k the number of successes observed. Then the probability $\mathbb{P}(X = k)$ of observing k success is such:

$$\mathbb{P}(X = k) = \frac{\binom{K}{k}\binom{N-K}{n-k}}{\binom{N}{n}} \tag{47}$$

We can also find the expected value of this function. But first a useful lemma.

$$\binom{q}{p} = \frac{q}{p}\binom{q-1}{p-1} \tag{48}$$

Proof:

$$\binom{q}{p} = \frac{q!}{p!(q-p)!} \tag{49}$$

$$= \frac{q}{p}\frac{(q-1)!}{(p-1)!(q-p)!} \tag{50}$$

$$= \frac{q}{p}\frac{(q-1)!}{(p-1)!(q-1-p+1)!} \tag{51}$$

$$= \frac{q}{p}\frac{(q-1)!}{(p-1)!(q-1-(p-1))!} \tag{52}$$

$$= \frac{q}{p}\binom{q-1}{p-1} \tag{53}$$

Mean of a hypergeometric random variable proof:

$$E[X] = k\mathbb{P}(X=k) = k\sum_{k=1}^{K}\frac{\binom{K}{k}\binom{N-K}{n-k}}{\binom{N}{n}} \tag{54}$$

$$= k\sum_{k=1}^{K}\frac{\frac{K}{k}\binom{K-1}{k-1}\binom{N-K}{n-k}}{\frac{N}{n}\binom{N-1}{n-1}} \tag{55}$$

$$= \frac{nK}{N}\sum_{k=1}^{K}\frac{\binom{K-1}{k-1}\binom{N-K}{n-k}}{\binom{N-1}{n-1}} \tag{56}$$

$$\text{Let j = k-1} \tag{57}$$

$$= \frac{nK}{N}\sum_{j=0}^{K-1}\frac{\binom{K-1}{j}\binom{N-K}{n-j-1}}{\binom{N-1}{n-1}} \tag{58}$$

We can identify an hypergeometric probability formula

$$Hyp(N-1, K-1, n-1)$$

.
Therefore

$$\sum_{j=0}^{K-1}\frac{\binom{K-1}{j}\binom{N-K}{n-j-1}}{\binom{N-1}{n-1}} = 1 \tag{59}$$

And

$$E[X] = \frac{nK}{N} \tag{60}$$

Now lets take Alice as an example. On a 4 she reveals the top 2 cards of the deck and adds any Sweet child found this way to its owner's board. As seen in figure 3, there are 24 Sweet children in the deck.

$$E[Alice|X=4] = \frac{nK}{N} = \frac{2*24}{100} = 0.48 \text{ card} \tag{61}$$

This gives us the expected value of each activation of Alice. Combining this value with the probability of activation, we get the expected value of the card itself.

$$E[Alice] = \frac{nK}{N} * \mathbb{P}(X=x) = 0.48 * \mathbb{P}(X=4) = 0.48 * 0.0833 = 0.039984 \text{ card} \tag{62}$$

Converting to souls to ease comparison to other cards gives us:

$$0.039984 * 3 = 0.1200 \text{ souls}$$

Lets now take the 2nd case, card generating effects that take from the *Block*. In most game states, there are 5 cards in the *Block* available to be purchased by souls or acquired via card effects. We will be again using the hypergeometric distribution again, but with some differences. $n = 5$ for every effect and only one card may be gathered even if $k > 1$.

Evaluating the probability distribution of this new function. Let X be the sum of two die and $Y \sim Hyp(100, K, 5)$ be a random variable where K is the number of valid targets in the *Block*.

$$\mathbb{P}(X = x, Y \geq 1) = \mathbb{P}(X = x)\mathbb{P}(Y \geq 1|X = x) \tag{63}$$

$$\text{Since X and Y are independent} \tag{64}$$

$$= \mathbb{P}(X = x)\mathbb{P}(Y \geq 1) \tag{65}$$

$$= \mathbb{P}(X = x)(1 - \mathbb{P}(Y = 0)) \tag{66}$$

Now we can evaluate the expected value. Since we can only get 1 card from the Block if we succeed, we can treat it like a binomial function.

$$E[X, Y] = 0 * \mathbb{P}(X = x, Y = 0) + 1 * \mathbb{P}(X = x, Y \geq 0) \tag{67}$$

$$= \mathbb{P}(X = x)(1 - \mathbb{P}(Y = 0)) \tag{68}$$

As in section 5.3.1, we can include effect that discard themselves, steal, or are conditional. These are the variants for deck generators:

Stealing

$$E[card] = \frac{m_s n K}{N}\mathbb{P}(X = x) \tag{69}$$

Discarding

$$E[card] = \frac{(n + d_{value})K}{N}\mathbb{P}(X = x) \tag{70}$$

Stealing and Discarding

$$E[card] = \frac{m_s(n + d_{value})K}{N}\mathbb{P}(X = x) \tag{71}$$

Same variants for the block generators:

Stealing

$$E[card] = m_s\mathbb{P}(X = x)(1 - \mathbb{P}(Y = 0)) \tag{72}$$

Discarding

$$E[card] = (1 + d_{value})\mathbb{P}(X = x)(1 - \mathbb{P}(Y = 0) \tag{73}$$

Stealing and Discarding

$$E[card] = m_s(1 + d_{value})\mathbb{P}(X = x)(1 - \mathbb{P}(Y = 0) \tag{74}$$

We will discuss Annie in section 5.3.5.

Card Name	Probability	Expected Gains (Cards)	EV (Souls)	Discard	Steal	Cond
Alice	.0833	.48	0.1200			
Cat	.1667	.32	0.1600			
Feral Cat	.1667	1	0.5000*			x
Falcon	.1667	2.24	0.6200	x		
Tommy	.1111	.34	0.1133			

Figure 9: EV breakdown for every card generator pulling from the deck

Card Name	Probability	Expected Gains (Cards)	EV (Souls)	Discard	Steal	Cond
Dolores	.0833	.75	0.1887			
Jane	.1389	.86	0.3590			
Marilyn	.1111	.88	0.2938			

Figure 10: EV breakdown for every card generator choosing from the Block

Card Name	Probability	Expected Gains (Cards)	EV (Souls)	Discard	Steal	Cond
Père Fouet-tard	.0417	2	.2500			

Figure 11: EV breakdown for every card generator demon

5.3.4 Demon Generators

Generating demons can be extremely beneficent in this game as demons are worth 9 souls and because of the tempo gain (see section 6). There are only two *block* cards that can create demons, the goat and Damien. Just as in section 5.3.1, calculating the expected value of demon generators is easy.

Let x be the number on the card, $\mathbb{P}(X = x)$ be the probability of this total on two die and n the number of demons created by this effect.

$$E[Generator] = 9n\mathbb{P}(X = x) \tag{75}$$

Discarding

The most common cost on cards that summon demons is sacrificing a set number of cards. We have already defined d_{value} that we will use here.

$$E[Generator] = (9n + d_{value})\mathbb{P}(X = x) \tag{76}$$

Example

Lets calculate the EV for the goat. Since its effect requires the player to discard 2 other cards we will use equation 76. First we need de calculate d_value

$$d_{value} = 3(c_{opp} - c_{player}) = 3(0 - 2) = -6 \tag{77}$$

Now for the EV:

$$E[Goat] = (9 * 1 + (-6)) * 0.1667 = 0.5000 \text{ souls} \tag{78}$$

Card Name	Probability	Gains (Demons)	EV (Souls)	Discard	Steal	Cond
Damien	.0556	1	.3333	x		
Goat	.1667	1	.50	x		x

Figure 12: EV breakdown for every demon generator

Card Name	Probability (2 player)	Gains (Demons)	EV (Souls)	Discard	Steal	Cond
Degueulivre des demons	.0278	1	.2500			
Demon Egg	N/A	1	0	x		

Figure 13: EV breakdown for every demon generating demon

5.3.5 Removal

There are ways for players to interact and remove their opponent's cards, either by stealing or discarding them. The alligator, the skunk or Annie have such effects. The main value source for those cards will be d_{value} and m_s defined in section 5.2. When they force discards, the EV of removal cards is then defined as:

$$E[Removal] = d_{value}\mathbb{P}(X = x) \tag{79}$$

When they steal and discard, as with Annie, the formula is:

$$E[Removal] = m_s d_{value}\mathbb{P}(X = x) \tag{80}$$

For the most optimal use of removal, one should calculate the greatest EV of their opponents card when using targeted removal effects. Mass removal, like the skunk, should only be used when one is behind in the game.

Card Name	Probability	Cards removed	EV (Souls)	Discard	Steal	Cond
Alligator	.1667	2	.5000	x		
Annie	.1111	1	.3333	x	x	
Skunk	.1667	Variable	Variable	x		

Figure 14: EV breakdown for every removal card from the Block

Card Name	Probability (2 players)	Cards removed	EV (Souls)	Discard	Steal	Cond
Mephistopheles	.0278	1	.5000**		x	
The Devil	.0139	ALL	A lot		x	

Figure 15: EV breakdown for every removal demon
*** Assuming an opponent has a demon to steal*

5.3.6 Clones

The last of the prominent card archetypes, clones are cards that can copy the effect of one or more cards. Such effects are all conditional, but combined with the proper card, clones can be very powerful.

They mostly serve to increase the odds of activating an effect or doubling the effect when it activates. For optimal use, clones should copy the most powerful effect on the board. This will be calculated by taking the greatest gain in souls/cards/demons, regardless of the probability of this effect activating. If the clone happens to activate multiple cards, we will sum these gains instead. In the *Block deck*, there are only 2 clone effects: Destiny and Irwin. The former activating one sweet child, while the later activates all of your animals.

Single Target Clone

Calculating Destiny's EV will require us to use the maximum of gains. let g_i be the value, in souls, gained when card i activates. We will define a function $G(i)$

$$G(i) = \begin{cases} g_i \text{ if the card } i \text{ is a valid target for the clone} \\ 0 \text{ otherwise} \end{cases}$$

The best gain G_{max} is then defined as:

$$G_{max} = max(G(1), G(2), ...G(n)) \tag{81}$$

Meaning the EV is:

$$E[SingleClone] = G_{max}\mathbb{P}(X = x) \tag{82}$$

Multi-Target Clone

When multiple cards are activated via a clone, such as with Irwin, we can use G_{Total} defined as follows.

$$G_{Total} = \sum_{i=1}^{n} G(i) \tag{83}$$

This gives us the EV:

$$E[MultiClone] = G_{Total}\mathbb{P}(X = x) \tag{84}$$

Card Name	**Probability**	**Target Type**	**Number of targets**
Destiny	.1111	Sweet Children	1
Irwin	.1389	Animals	All

Figure 16: Clone probability and targets

Card Name	**Probability (2 player**	**Target Type**	**Number of targets**
Belze'bzz	.0834	Children	All

Figure 17: Cloning demon probability and targets

5.4 Special Cases

5.4.1 Giving Souls

When we studied simple soul generators in section 5.3.1, we glossed over Glen, which has the unique effect of giving a soul to an opponent when generating souls. We can treat this gift as the opposite of stealing souls.

Let $n_{generated}$ and n_{given} be the number of souls generated and given to another player respectively. Lets also define $m_g = \frac{1}{N_p - 1}$. Then, the expected value of Glen can be defined as:

$$E[Glen] = (n_{generated} - n_{given} m_g)\mathbb{P}(X = x) \tag{85}$$

Therefore, in a 2 player game, Glen's expected value can be calculated as:

$$E[Glen] = (2 - 1 * \frac{1}{2-1}) * 0.1111 = 0.1111 \text{ souls} \tag{86}$$

Meanwhile, in a 5 player game, Glen is much more effective as the impact of the given soul is spread around the other players.

$$E[Glen] = (2 - 1 * \frac{1}{5-1}) * 0.1111 = 0.1944 \text{ souls} \tag{87}$$

5.4.2 Doubles

Dice-zuzu is a unique effect as it does not trigger on a specific number, but instead triggers every double on the dice, generating souls equal half that total. We will refer back to 4, where each pair as a $\frac{1}{36}$ chance of happening. The expected value for this demon is then defined by:

$$E[Zuzu] = (\frac{1}{s^2}\sum_{i=1}^{s} i)\mathbb{P}(CurrentPlayer = Owner) \tag{88}$$

$$= \frac{1}{s^2}\frac{n(n+1)}{2}\frac{1}{N_p} \tag{89}$$

$$= \frac{1}{6^2}\frac{6(6+1)}{2}\frac{1}{N_p} \tag{90}$$

$$= \frac{7}{12N_p} \tag{91}$$

5.4.3 Instant Win

Players need to summon three demons and acquire ten souls to win a game. Wining instantly skips all of that, functionally gaining 19 to 37 souls, 9 souls per remaining demon and 10 for the stash. Naturally this great amount of resources will be tremendous for tempo (see in section 6).

Let D be the remaining number of demons before winning.

$$E[Snake] = (9D + 10)\mathbb{P}(X = x, CurrentPlayer = owner) \tag{92}$$

$$= \frac{(9D + 10)}{36N_p} \tag{93}$$

So if the first summoned demon is the snake in a two player game, the EV is:

$$E[Snake] = \frac{(9 * 2 + 10)}{36 * 2} \tag{94}$$

$$= 0.3888 \text{ souls} \tag{95}$$

5.4.4 Gender Ideology

We saw that cards have some basic characteristics in figure 1. Types and subtypes determine legal targets for card e ffects. Demongorguignol and Incestuous Demon play around with the types and let some effect break the rules. Both give the cards in a player's possetion a new type or subtype. Of course, these effects will be context dependent. We will elaborate on how to calculate the impact of these demons in an ongoing game.

1. Calculate the current board's EV, EV_c.

2. Evaluate the board's Potential EV, EV_p, with the demon's effect applied.
3. Remove the 3 (or 4 see section 5.4.7) lowest results from EV_p.
4. The expected value gain of the demon is $EV_p - EV_c$.

We will see in section 7 that these cards can simply toggle a value for every card on the board and then execute the evaluation function twice.

5.4.5 Re-rolling

If a player has Rerollifer has one of their cards, they can chose to ignore their first result on their turn and roll again, accepting the new result. The probability of getting a specific total x is:

$$p_{reroll} = \mathbb{P}(X = x) + (1 - \mathbb{P}(X = x))\mathbb{P}(X = x) \tag{96}$$

Therefore, the EV of a card needs to include this specific probability on a player's turn. Since this effect applies to all other cards on a player's field. We can treat the value of Rerollifer as we did for type-changing demons in section 5.4.4.

$$E[X] = x\left(\frac{p_{reroll}}{N_p} + \frac{(N_p - 1)\mathbb{P}(X = x)}{N_p}\right) \tag{97}$$

Same as with most numbered demons, Rerollifer is much more impactful in games with a low player count.

5.4.6 *The World,* stop time!

Satange Mechanique, has a peculiar effect when activated. It is the only card that stops other cards from activating. Once a 9 is rolled, the extra roll given will activate only Satange's owner's cards.

Im not sure how to calculate this ngl. Pistes de solutions: value de 1 + la value des cartes non-démon des autres joueurs. Piste de test: evaluer la qté bloquée et la qté gagnée par des algo qui jouent (prob random) sur le long terme avec satange.

5.4.7 Taxes

A player wanting to slow down their opponents can find their answer in Baphometal, a demon with a passive effect which requires other players to snuff 4 cards to summon a demon instead of the usual 3. Intuitively, we know this demon will be more impactful if played early and in a game with many opponents. We can quantify this impact by calculating the extra "spendings" players will have to pay to win after summoning Baphometal.

Let N_p be the number of players. Let d_i be the number of demons player i has on the field, and D be the number of demons necessary to win the game. Baphometal's value can then be calculated as such:

$$value = 3(DN_p - \sum_{i=1}^{N_p} d_i) \tag{98}$$

We now have a value for this card, be we still need to consider some specifics. An astute reader will have noticed we used "Value" instead of EV for this effect. Indeed, Baphometal's impact is fully realised as soon as it's summoned. Meaning it's value as a card will be seen as 0 on every turn. It will only have value when considered to be summoned or stolen. Of course, any substatial ponctual value will be greater than any EV because such effect are expected to activate over and over during the game. To couteract this fact, we will setup a simulation in section 7.2.1 of the game without Baphometal to try to measure the average game length. Having an estimate of turns in a game will let us spread the instant value of the card to be more in line with other more standard effects.

5.4.8 Cashback on Animals

Similarly to Baphometal, Assmodeus offers a passive effect which can be hard do price accurately. Since it gives a soul every time an animal is acquired, we will need to use section 7.2.1's simulation script again. We can estimate the average number of animals obtained every game and then calculate a turn-by-turn average using the average game length.

Sequencing might also need to be altered to prefer animal based strategies, but this is beyond the scope of this paper.

6 Tempo

As of right now, we have considered souls and card as equivalents, as players can transform souls into cards, one card at a time, every turn. This imposes a speed limit. Even with infinite souls, it would still take nine turns for a player to obtain the sufficient cards to summon three demons. Similarly, demons can also only be summoned once each turn by game mechanics. We will define the number of a specific resource that can be obtained in one turn **Tempo**. The principal metrics we will examine are card and soul tempo, but demon tempo will still be touched on.

6.1 Card Tempo

We can use a modified version of expected value as our basis for tempo. We will measure **Card Tempo** T_{card} as a certain number of expected cards.

Definition 6.1. Let G_{card} be the set of cards on the board that generate cards and $EV_{card} = \sum_{i \in G_{card}} E[x_i]$ be the expected value for cards generated by the board and p_{card} be the odds of being able to buy a card each turn. The card tempo T_{card} of the board is then:

$$T_{card} = EV_{card} + p_{card} \tag{99}$$

This equation should be fairly intuitive to anyone, tempo is the average number of card generated each turn and the probability of buying one via game mechanics. We still have to define p_{card}, this is because we first need to introduce more powerful tools to calculate this variable, this will be touched on in section 6.2.

6.2 Markov Chains

The following section is more advanced and uses Markov chains, it is not required to understand section 7. Most of the information is taken from Anne MacKay's notes for course STT489 at Université de Sherbrooke.

Definition 6.2. Let $\{X_n | n = 0, 1, ...\}$ be a stochastic process with discrete states in space E. This process is a *Markov Chain* when for every sequence $i_0, i_1, ..., i_n, i, j$, such that $\mathbb{P}(X_0 = i_0, X_1 = i_1, ..., X_{n-1} = i_{n-1}) > 0$ we have:

$$\mathbb{P}(X_{n+1} = j | X_n = i, X_{n-1} = i_{n-1}, ..., X_0 = i_0) = \mathbb{P}(X_{n+1} = j | X_n = i) \tag{100}$$

To model the speed at which players can buy cards, we will need to know the amount of souls in bank at every turn t. Since turns are discrete, then the time space T will be countable, and since souls are a direct function of the number of turns, we can define $\{X_t | t \in T\}$ as a stochastic process. Furthermore, as we have seen in section 5, no effects in this game required memory of the previous turns. To know the chances of getting to j souls on turn $n + 1$, we just need to know the previous total of souls i on turn n. Therefore,

$$\mathbb{P}(X_{n+1} = j | X_n = i, X_{n-1} = i_{n-1}, ..., X_0 = i_0) = \mathbb{P}(X_{n+1} = j | X_n = i) \tag{101}$$

This means we can use a Markov chain to define the amount of souls a player has each turn, hereby giving us the states in which they can buy cards. Transition matrices can be be used to represent the states and transition probabilities of a chain.

Definition 6.3. A Markov chain is said to be *irreducible* if it has only one *equivalency class.*

TROUVE UNE BELLE DEFINITION DE CLASSE DEQUIVALENCE. In other words, the states of an irreducible chain all communicate with each other.

Definition 6.4. A Markov chain is said to be *stationary* if $\mathbb{P}(X_{n+1} = j|X_n = i)$ is independent of n, that is if

$$\mathbb{P}(X_{n+1} = j|X_n = i) = \mathbb{P}(X_1 = j|X_0 = i) \tag{102}$$

Lets focus on the chain we use to represent souls acquired. We know from chapter 5 that the effects of cards do not change under time. They do the same thing wether it is the first or 20th turn. This means the probabilities of going from i souls to j in bank will be constant for any given board of cards. This means $\mathbb{P}(X_{n+1} = j|X_n = i)\forall n \Rightarrow \mathbb{P}(X_{n+1} = j|X_n = i) = \mathbb{P}(X_1 = j|X_0 = i)$. Therefore, the chain is stationary.

Two chains will be used, the first will be the **Buying Chain**, which will calculate the turns on which the selected player can buy cards. The state number will represent the number of souls in bank. We will assume that cards are bought as soon as available. Each step in the chain will be a turn cycle. To calculate the transition probabilities however, we will need to account for each turn's roll. We will then use a second, smaller chain we will call the **Turn Cycle Chain**. This one will represent the transitions probabilities each turn given by the soul generators of the board. Since we need the transition probabilities for the whole turn cycle, we will need to use the Chapman-Kolmogorov equation's corollary **(To define earlier)**.

Theorem 6.1. ***Chapman-Kolmogorov Equation****. Let $\{X_t|t = 0, 1, ...\}$ a Markov chain with transition matrix P and state space E. For every $m, n \in \mathbb{N}$ and every $i, j \in E$,*

$$p_{ij}^{(n+m)} = \sum_{k \in E} p_{ik}^{(n)} p_{kj}^{(m)}. \tag{103}$$

Corollary 6.1.1. *Let $\{X_t|t = 0, 1, ...\}$ be a Markov chain with transition matrix P and state space E. If E is finite, then $P^{(n+m)} = P^{(n)}P^{(m)}$ for each $n, m \in \mathbb{N}$*

Corollary 6.1.2. *Let $\{X_t|t = 0, 1, ...\}$ be a Markov chain with transition matrix P and state space E. If E is finite, then $P^{(n)} = P^n$ for each $n \in \mathbb{N}$*

Definition 6.5. Let n be the number of players in the game, a *turn cycle* is defined as n sequential turns.

Definition 6.6. Let n be the number of players in the game and P the transition matrix for souls, the *Turn Cycle Chain* P_{cycle} is then defined as:

$$P_{cycle} = P^{(n)} = P^n \tag{104}$$

Now that we have the turn cycle's transition matrix, we have the probabilities of gaining j souls, p_{0j}. When can then construct the buying chain's matrix. **CLEAN THIS UP, MAYBE USE OTHER LETTERS**

$$p_{kl} = \begin{cases} p_{0j} \text{ if } j + k = l \text{ and } k < 3 \\ p_{0j} \text{ if } j + k = l - 3 \text{ and } k >= 3 \\ 0 \text{ otherwise} \end{cases}$$

With this in place, we will be able to calculate the stationary distribution of the turn cycle matrix. To do so, we will need the chain to be finite. The rules of the game do not specify a maximum number of souls a player can have and recommend using any tokens as souls if the players run out of game pieces. But for the use cases of this paper and reducing the computing load, we will cap the size of the transition matrix to the total number of souls a player needs to win. Total Soul Count = $10 + 9 * nb_{demons}$, where nb_{demons} is the number of demons needed to be summoned to win the game (usually 3).

6.2.1 Limit Distribution

To calculate the probability of being able to buy a card on each round, we will need to use the limit distribution of the buying chain. This way we will know the probabilities to be in that state at any given time. To do this, we will need to first verify that the buying chain can have a stationary distribution.

Definition 6.7. Let $\{X_n|n = 0, 1, 2, ...\}$ be a Markov chain and $i \in E$ a recurrent state. The *time of first return* to state i is defined as:

$$T_i = \inf\{n \in \mathbb{N}|X_n = i, X_0 = i\} \tag{105}$$

Definition 6.8. Let $\pi = \{\pi_i\}_{i\in E}$ be a distribution on E. It is said to be *stationary* if for every $j \in E$,

$$\pi_j = \sum_{i\in E} \pi_i p_{ij} \tag{106}$$

Definition 6.9. A state $i \in E$ is a *recurrent state* if exists $n \geq 1$ such as $p_{ii}^{(n)} > 0$.

Definition 6.10. Let $i \in E$ be a recurrent state. The *period* of i, noted d_i, is given by

$$d_i = GCD\{n \geq |p_{ii}^{(n)} > 0\}. \tag{107}$$

Definition 6.11. Let $\{X_n|n = 0, 1, ...\}$ a Markov chain, $i \in E$ a recurrent state and T_i the time of first return to state i,

- If $E[T_i] < \infty$, the state is said to be *positively-recurrent*.
- If $E[T_i] = \infty$, the state is said to be *null-recurrent*.

Corollary 6.1.3. *Let $i, j \in E$ be states. If i is recurring and $i \leftrightarrow j$, then j is recurring.*

Definition 6.12. A state $i \in E$ is *ergodic* if it is positively recurring and aperiodic. An irreducible Markov chain which has only recurring and aperiodic states is said to be ergodic.

Theorem 6.2. *Let $\{X_n|n = 0, 1, ...\}$ be an irreducible and ergodic Markov chain. Then:*

$$\pi_j := \lim_{n\to\infty} p_{ij}^{(n)} \tag{108}$$

exists for each j and every π_j is independent from i. Furthermore, $\{\pi_i\}_{i\in E}$ determines the unique stationary distribution for the Markov chain.

To show that the buying chain is irreducible, lets focus on the simplest possible board: only one candle card. In this case, each turn the player can gain 1 soul with probability p or not with probability $1-p$. For a turn cycle of n players, this means the possibilities for n pulls can be modeled by a binomial distribution. Trivially, this means the function can get from 0 to n souls in a turn cycle. states can thus be incremented by any value from 0 to n, making it so any state is accessible via a lower state. Then, when $i > 2$, the buying chain lowers it's state by 3, representing the acquisition of a new card and can be increased in the same step by souls gained (from 0 to n). This means the state changes by any value from -3 to $n - 3$, letting any state to be accessible from a higher state. Since every state can be accessed by any other state, the chain is irreducible.

Of course, adding any other soul generators to this board will only change the rate and range of changes, but every state will still be accessible, making it irreducible.

Lets now prove that the buying chain is positively recurring: *chain is finite and closed, which means positive recurrent* **PROVE THIS AT SOME POINT**

We still need to prove that the chain is aperiodic ($d_i = 1$). For $i = 0, 1, 2$ the proof is trivial since there is always a chance to not gain souls that turn. For $i > 2$ let n be a return time. Since not gaining souls will result in lowering the total, eventually states 0,1 or 2 will be reached in ,let's suppose, $n - k$ turns. If there are no souls gained for one turn before returning in k turns to state i, then the return time is $n+1$. If this happened for 2 turns instead, it would be $n+2$. This can repeat for ever, meaning

$$d_i = GCD\{n, n + 1, n + 2, n + 3, ...\} = 1 \tag{109}$$

The chain is therefore aperiodic. Since the chain was already positively recurring, it can be said to be ergodic.

Furthermore, since the buying chain is ergodic and irreducible, we can use theorem 6.2 to calculate the limit distribution, but first,

Definition 6.13. Let $\pi = \{\pi_i\}_{i \in E}$ be a distribution on E. It is said to be *stationary* if for every $j \in E$,

$$\pi_j = \sum_{i \in E} \pi_i p_{ij} \tag{110}$$

Using the definition of a stationary distribution 6.13, we can find it by solving the following linear equation system.

$$\begin{cases} \pi_j = \sum_{i \in E} \pi_i p_{ij}, j \in E \\ \sum_{i \in E} \pi_i = 1. \end{cases} \tag{111}$$

In cases such as ours, where the number of m states is finite, we can write $\pi = [\pi_1 ... \pi_m]$. Then we can rewrite 111 as:

$$\pi = \pi P \tag{112}$$

Where P is the transition matrix. Transposing both sides we get.

$$\pi^T = P^T \pi^T \tag{113}$$

Meaning the stationary distribution is given by the eigenvector of eigenvalue 1 from P^T.

6.3 Soul Tempo

Take the transition matrix, change it a bit, then calc expected time to reach ≥ 10 or something similar.

6.3.1 Going for the Win

An algo based only on EV will not try to win the game, it will instead spend all of it's souls to increase tempo over and over and over. Ideas: win condition detection. Maybe lower the weight of card tempo as every card/demon is acquired. ex simple si 2 demons + 4 cartes qui genere des ames, 100/100 devrait etre focus sur uniquement les ames et win, not spending.

7 Building a Game Algorithm

Now that we have evaluated the value of (almost) every card in this game, the least we can do is to try to create an algorithm that plays using these values.

7.1 Making of

Idée de base: 1-check si un result de dé pourrait faire gagner. Si oui, roll first (maybe snuff or buy too? this might be harder to implement) 1.5 check si autre player could win too, check pour option aggressives (ca aussi pourrait etre compliqué) 2-Si peux acheter avant de roll, do it. 3-roll, check pour buy une carte. 4-Check pour demon? (prob mettre une condition que la valeur de souls/tour ne peut pas baisser en dessous)

7.2 Testing

Test it vs random, test les tours pour win en solo? Maybe lets make a tournament of weird algo???

7.2.1 Game Length

Section used to calculate Baphometal and Assmodeus values.

7.3 The King of Demons Tournament

Pretend youre Tomm7 ig

7.3.1 Setup

7.3.2 Meet the challengers

8 Conclusion