LGP Lumieres

LGPL

Problem 4

The apprentice and the witch

PROBLEM 4

the Apprentice and the Witch

Summary

This problem involves the most optimal and most optimally inefficient methods of combining runes on a workbench in various different ways to create different 'spells,' having varying properties, across distinct initial configurations. In this solution, results were found on maximum number of healing spells, minimum and maximum mana cost to create those spells; generalizing those results across different initial configurations; minimum mana costs for blasting spells in varying initial configurations, values of α, and relations between power and weight. Conjectures were made regarding the minimum mana cost for creating fire spells, and 2 additional research directions were explored.

Question	Result
1	Solved
2	Solved
3	Mostly Solved
4	Mostly Solved
5	Solved
6	Some Results

Contents

Problem Statement: A witch is forging spells by assembling runes on her workbench. Her workbench looks like a huge square plate that we identify with $[1;\ell]^2$ for some integer $\ell \geq 1$, with a notch for a rune on each point with integer coordinates. She starts by preparing her workbench by inserting runes in some of the notches. Each notch may contain no or one rune. The description of the way the notches are filled is called the **initial configuration**. Moreover, each rune has a *weight*, which depends on the type of the rune.
Once the initial configuration is set, the witch may no longer add or remove runes. She may only move them in order to combine them, respecting the following rule: she moves runes one by one, in a horizontal or vertical line, only from one notch to another. (One rune may be moved several times, following a path formed by a broken line. A rune cannot jump above another.) The **mana cost** of such an operation is the product of the distance being travelled by the weight of the rune.

1 Healing Spells

Question 1: The witch starts by forging healing spells. She works only with one type of rune, of weight equal to 1. Once a rune is moved into a notch containing another rune, they merge to create one healing spell, and vanish from the board. For instance, after the move in Figure 6, the notches A and B will be empty, and the witch will have one healing spell in hand. In the initial configuration, every notch of the workbench is filled by a rune.

a) How many healing spells can she create from this initial configuration?
b) What is the minimum mana cost needed to create those spells?
c) The witch wants to play a bad trick to one of her enemies, who asked her the recipe for the healing spell, by indicating her the most expensive way to create healing spells. To make the trick not so obvious to detect, the same configuration should not arise twice on the board. What is the maximum mana cost that can be achieved this way? Try to give an estimation as precise as possible.

(a) Maximum Number of Healing Spells

There are ℓ^2 total runes in the initial configuration. Since each healing spell uses exactly 2 runes, the maximum number of healing spells that can be created is:

$$\left\lfloor \left(\frac{\ell^2}{2} \right) \right\rfloor$$

(b) Minimum Mana Cost for the Healing Spells

Each healing spell can be created by moving one rune onto another by traversing over 1 notch. The mana cost of such an operation would be 1, and evidently the minimum mana cost of creating a healing spell.

Thus, the theoretical minimum mana cost of creating $\lfloor (\frac{\ell^2}{2}] \rfloor$ healing spells will also be $\lfloor (\frac{\ell^2}{2}) \rfloor$.

This mana cost can be achieved through the following algorithm:

- In the top row, choose the leftmost rune and move it one to the right.
- Repeat the previous step until there are less than two runes left in this row.
- Move to the row below and repeat the previous two steps.
- Repeat the previous three steps until all rows are done.
- If ℓ is even, the algorithm is complete. If ℓ is odd, the runes in the rightmost column of the workbench will remain.
- Choose the top rune of the column and move it one down.
- Repeat the previous step until there are less than 2 runes left.

The above algorithm has been illustrated below for $\ell = 5$

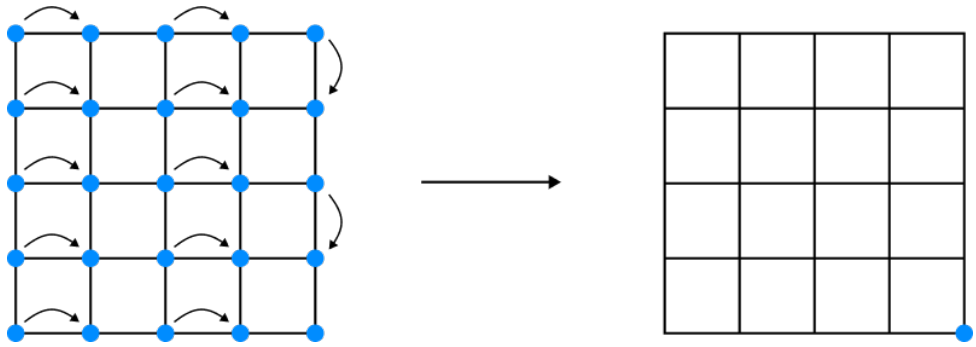

Figure 1: Algorithm for maximum mana cost, $\ell = 5$

(c) Maximum Mana Cost for the Healing Spells

Whenever a healing spell is created, 2 notches are emptied. Thus, the largest possible traversal distance to form another healing spell increases by exactly 2.

In the given initial configuration, the maximum mana cost for making the first spell is evidently always 1.

$\therefore$ Mana Cost to create the k^{th} healing spell after $(k-1)$ healing spells have already been created is $(2k-1)$.

$\therefore$ Theoretical maximum mana cost to create $\lfloor\left(\frac{\ell^2}{2}\right)\rfloor$ healing spells:

$$\sum_{i=1}^{\left\lfloor\frac{\ell^2}{2}\right\rfloor}(2\ell-1) = \left(\left\lfloor\frac{\ell^2}{2}\right\rfloor\right)^2$$

This can be seen by using the greedy algorithm for maximum distance between two runes without

notches in between. This will always maximize the mana cost of each healing spell.

2 Initial Configurations Changed by Apprentice

Question 2: During the night, the child apprentice of the witch comes in her workshop, and to prank her, he picks up some runes and hides them in the garden. The next day, the witch thus has to start with a new initial configuration, where some notches have been emptied by the apprentice. In the following situations, how many healing spells may she create each day, and what is the minimum mana cost for doing so? Of particular interest is the asymptotic ratio between the number of spells and the mana cost when $\ell \to +\infty$, in case no explicit formula is available.

a) The first night, the apprentice takes out every rune which is in a notch whose coordinates are either both even or both odd.

b) The second night, the apprentice is faced with a new fully filled workbench whose side length ℓ is a power of two, and finds himself to be especially inspired. He starts by dividing the whole workbench into four equal squares, and picks up all the runes that are in the bottom left one. Then, he divides the three remaining squares into four equal smaller squares, and in each of them, he picks up all runes in the bottom left one. He continues so until he finds himself dividing squares of side length 2 into squares of side length 1, at which moments he picks the bottom left rune in each square and leaves.

(a) Runes in locations with both even or both odd coordinates

Total number of runes in initial configuration: $\left\lfloor \frac{\ell^2}{2} \right\rfloor$, as each rune can be equated to a healing spell as in Part 1(a).
$\therefore$ Maximum Number of Healing Spells:

$$\frac{\left\lfloor \frac{\ell^2}{2} \right\rfloor}{2}$$

The minimum distance from any rune to another is 2 steps/notches, so the theoretical minimum mana cost is: $\left\lfloor \frac{\ell^2}{2} \right\rfloor$

This can be achieved by the following algorithm:

- Start with the leftmost rune on the bottom row. Move it one left and one up to create a spell.
- Repeat the above steps for all runes with odd coordinates y for which it is possible.
- If ℓ is even, we are done. If ℓ is odd, the top row and right-most column will have the only remaining runes.
- Take the left-most rune of the to row and move it two notches to the right.
- Repeat the previous step till the top row has less than 2 runes left.

- Take the bottom-most rune of the right-most column, and move it 2 notches up.
- Repeat the previous step till the right-most column has less than 2 runes left
- If the runes are remaining, combine the right-most rune on the top row and the top rune on the right-most column with a mana cost of 2.

  Clearly, this combines every pair of runes using exactly 2 mana, giving us the desired minimum mana cost. This is illustrated below for $\ell = 7$

  This gives the exact ratio between the number of spells and mana cost as $\frac{1}{2}$

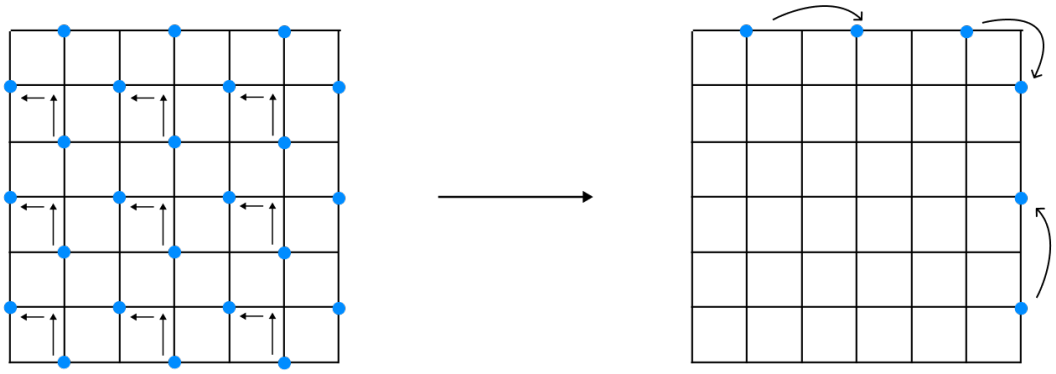

Figure 2: Minimum mana cost for $\ell = 7$.

(b) The case where $\ell = 2^k$

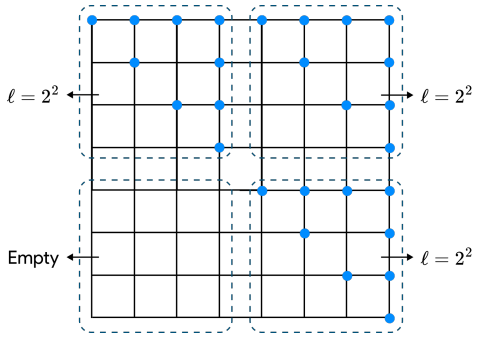

Figure 3: Divide of $\ell = 2^3$ into $\ell = 2^2$ and Empty squares.

Due to the nature of how the apprentice creates the initial configuration, splitting a board of $\ell = 2^k$ into 4 equal squares will yield one of those squares contains no runes, and each of the others containing the initial on figuration of $\ell = 2^{k-1}$. Specifically, the bottom left square would be empty, while the other three would contain the runes as described. This is illustrated in the below diagram for $k = 3$:

Starting from $\ell = 2^0$, which has exactly one rune $\ell = 2^k$ has 3^k total runes, which is always odd. Hence, the total number of spells created is always $\left\lfloor \frac{3^k}{2} \right\rfloor$ For every k, the top most one must be used in creating a healing spell. This is due to two Unwanted configurations being created it not used, as shown below: In each of these configurations, all runes except the centre rune (circled in diagram)

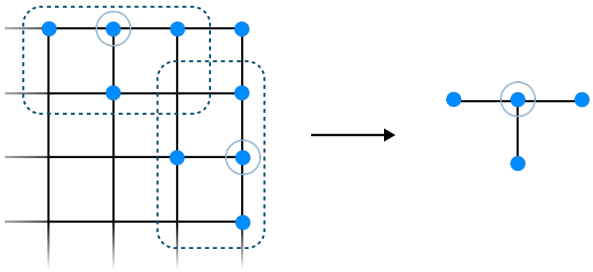

Figure 4: Unwanted configurations

can on by make a healing spell with mana cost of one by combining with the centre rune. Thus, combining 4 runes takes a man a cost of 3 . This is true even when placed within the $\ell = 2$ initial configuration, as the minimum distance between any runes from one of the configurations to the other is 2 , evident through the diagram. Thus, not using the top-right rune makes combining 9 runes into 4 healing spells (with 1 rune left over) have a mana cost of 6 , instead of 5 when using that runner.

In each $\ell = 2^k, k > 0$, the closest runes between the distinct subdivided squares are 1 notch apart. However, this only occurs with the top-right runes in the top-left and bottom-right squares, which as established. need to be used within their squares, Thus, the most efficient way to combine all the runes is to combine them most efficiently in each of the three smaller squares, then use 2 mana cost to combine 2 of the 3 remaining runes. An example of this can be seen in figure?.

Thus, the minimum mana cost for the initial configuration for $\ell = 2^k$ can be defined by: $a_k = 3a_{k-1} + 2$, with $a_1 = 1$ when $\ell = 2$.

Since this is a linear non - homogeneous recurrence relation we solve for homogeneous part $a_h h$ and particular solution $a_n^{(r)}$:

$$
\begin{aligned}
&a_k^{(h)} = 3a_{k-1}^{(h)} \Rightarrow a_k^{(h)} = A \cdot 3^k \\
&a_k^{(p)} = 3a_k^{(p)} + 2 \Rightarrow a_k^{(p)} = -1 \\
&\therefore a_k = a_k^{(h)} + a_k^{(p)} = 3 \cdot A - 1 \\
&\Rightarrow a_1 = 1 = 3A - 1 \Rightarrow A = 2/3. \\
&\therefore a_k = 2 \cdot 3^{k-1} - 1.
\end{aligned}
$$

This can be verified manually for $k = 1, 2, 3$, and 4 , for which the values match $(1, 5, 17, 53)$.

$\therefore$ Ratio of minimum mana cost to number of healing spells $\frac{2 \cdot 3^{k-1}-1}{\left\lfloor \frac{3^k}{2} \right\rfloor}$. As $\ell \to \infty, k \to \infty$, meaning the limit is:

$$
\lim_{k \to \infty} \frac{2 \cdot 3^{k-1} - 1}{\lfloor \frac{3^k}{2} \rfloor} = \lim_{k \to \infty} \frac{2 \cdot 3^{k-1}}{\left(\frac{3^k}{2}\right)} = \frac{4}{3}
$$

3 Maximising Mana

Question 3: We denote by n the number of runes that are left on the bench after the visit of the apprentice. Given a configuration, we denote by C the minimal mana cost needed to merge all runes.

a) What are the possible values of n/C?

b) If $n = 2k$, how can the apprentice choose how to remove the runes in order to maximize the mana cost of creating k healing spells? (You can start by studying $k = 1, 2, 3$ and $\ell = 2k$.)

(a) Possible Values of n/C

To find the possible range of values for $\frac{n}{c}$ across all values of e, we must consider the maximum and minimum possible values of c : $c_{\max}$ and $c_{\min}$, depending on the initial configuration $c_{\min}$ is obviously just $\lfloor \frac{n}{2} \rfloor$, following similar logir as in part 1) b), with each healing spell costing exactly one mana. $c_{\max}$, however, dearly increases without bound for a fixed n, as ℓ goes to infinity. Clearly, by shifting runes one notch at a time, it is possible to go form the configuration of $c_{\min}$ (where all runes are next to each other) to that of $c_{\max}$. Thus, for a given n, the range of values of $\frac{n}{c}$ is:

$$\frac{n}{c} \in S := \left\{ \frac{n}{x} \middle| \, x \in \mathbb{Z}^+, x \geqslant \left\lfloor \frac{n}{2} \right\rfloor \right\}.$$

(b) Maximizing mana with $n = 2k$

For small k, it is not too difficult to find the initial configurations which yield the maximal mana cost. We check for $k = 1, 2, 3$ in the cases that ℓ is odd or ℓ is even (assuming $\ell \geq 3$.

The first two runes always must go in opposite corners, with the next two always going into the remaining corners. Where the next two runes go, depends on the parity of ℓ.

Case 1: ℓ is odd. The next two runes must go in the midpoint of one of the sides, and the centre of the workbench, $(\lfloor \frac{\ell}{2} \rfloor, \lfloor \frac{\ell}{2} \rfloor)$. This maximises the mana cost by maximising the distance between the remaining runes after any first two are removed.

Case 2: ℓ is even. First, put one of the runes in one of the corners of the central 2×2 square, such as $(\frac{\ell}{2}, \frac{\ell}{2})$. Then the other rune should go to (x, y), where x is the column second furthest away from the previous rune, and y is the row number of the central 2×2 square that the previous rune did not occupy working on a similar logic to case 1.

Illustrations of both these cases are given below with $\ell = 7$ and $\ell = 8$ respectively:

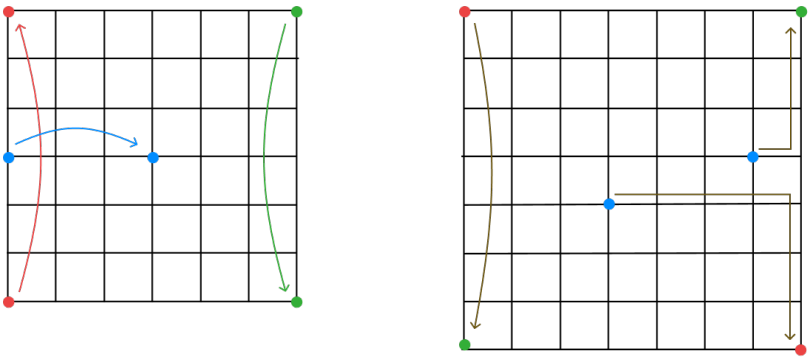

Figure 5: Case 1 and Case 2 for $\ell = 7$ and $\ell = 8$, respectively.

The figure 5 represents the runes when $k = 1$, (red rune), $k = 2$ (green rune), and $k = 3$ (blue rune).

While the other cases beyond $k = 3$ generally became too variable to generalize how the initial configuration would look, we were able to describe it beyond a certain value of n, (depending on ℓ):

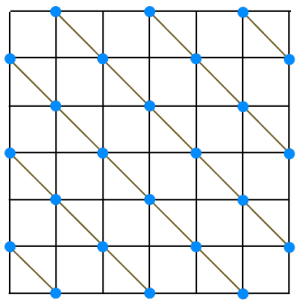

Figure 6: Diagonals giving perfect distance of 2 between all runer in $\ell = 7$.

As seen in the above figure, drawing all the 45° diagonals which cover an even number of points gives the maximum number of points such that the minimum distance between each point is exactly 2. Shifting any point would clearly lower the minimum mana cost required, and thus this initial configuration is the maximal case.

One may notice this configuration is equivalent to that in 2)a), and thus the total number of runes is $n = \lfloor \frac{\ell^2}{2} \rfloor$. However, the diagonal argument has further applications (such as the kth diagonal from any notch representing all notches a distance k away from it), although none further in these solutions so it has been mentioned.

Every n after this can be represented by just adding runes anywhere to the initial configuration in the above figure, as they will always be 1 notch apart from at least one other rune, no matter the location. Thus, the initial configurations which maximize the mana cost can be found for $k \geq \frac{1}{2}\left(\lfloor \frac{\ell^2}{2} \rfloor\right)$.

4 The Blasting Spell

Question 4: The witch now wants to forge a powerful blasting spell. She works with new runes, which are labeled by a positive integer number, called their *power*. In the initial configuration, every rune has a power of 1. When two runes come together, they merge into a new rune whose power is the sum of the powers of both runes. The weight of a rune of power k is k^α, with $\alpha \geq 0$ given. For instance, after the move in Figure 6, the notch A will be empty, and the notch B will contain a rune of power 2 and weight 2^α. The goal of the witch is to obtain a rune with the highest possible power, to get a devastating blasting spell. First, assume that the witch starts from the initial configuration where every notch contains a rune.

a) What is the minimal mana cost of forging a rune of power ℓ^2? You can start by studying the cases $\alpha = 0$, $\alpha = 1$. In case it is impossible to obtain an exact formula, it will be of particular interest to study the asymptotic of the ratio of the mana cost by the power of the rune that is obtained as $\ell \to +\infty$.

b) Fix ℓ and let $\alpha \to +\infty$. What is the behaviour of the total mana cost of forging a rune of power ℓ^2? Try to give an asymptote as precise as possible.

c) Now, explore the general case where the initial configuration is not necessarily full.

(a) Minimum mana cost for ℓ^2 for $\alpha = 0, 1$

For $\alpha = 0$, the weight of all runes is always 1 , and to reach a power of ℓ^2, all runes need to be combined. Thus, the theoretical minimum. mana cost becomes $\ell^2 - 1$, which can be achieved easily by following a snake-like pattern, shown below for $\ell = 4$:

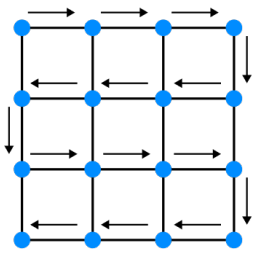

Figure 7: Minimum mana cost for $\ell = 4$, $\alpha = 0$

This gives a ratio of mana cost to power as $\frac{\ell^2-1}{\ell^2}$ which tends to 1 as $\ell \to +\infty$

When $\alpha = 1$, the weight and power of the runes are equal. It becomes equally efficient to move a rune with power greater than 1 compared to moving the same distance with a rune of power 1: moving two runes of power 1 across 2 and 3 notches is as efficient as combining them, then moving the rune of power 2 by 2 notches. The minimum mana cost of moving a power 1 rune from its initial position (x_i, y_i) to some final position (x_f, y_f) is $|x_i - x_f| + |y_i - y_f|$ Fixing one rune to become the one with power ℓ^2, the total mana cost of moving all other runes to that position, say (x, y), will be:

$$\sum_{i=1}^{\ell}\sum_{j=1}^{\ell}|x-i|+|y-j|$$

this will be minimised when $x=\lfloor\frac{\ell}{2}\rfloor$ or $\lceil\frac{\ell}{2}\rceil, y=\lfloor\frac{\ell}{2}\rfloor$ or $\lceil\frac{\ell}{2}\rceil$ For each of those cases, the sum comes out to be:

$$S_{min}=\begin{cases}2n(2n+1)(n+1) & \text{if } \ell=2n+1\\ 4n^3 & \text{if } \ell=2n\end{cases}$$

Proof. Define

$$S(x,y)=\sum_{i=1}^{\ell}\sum_{j=1}^{\ell}\big(|x-i|+|y-j|\big)=\ell\sum_{i=1}^{\ell}|x-i|\ +\ \ell\sum_{j=1}^{\ell}|y-j|.$$

Set

$$S_1(x)=\sum_{i=1}^{\ell}|x-i|,\qquad S_2(y)=\sum_{j=1}^{\ell}|y-j|.$$

Then

$$S(x,y)=\ell\,S_1(x)+\ell\,S_2(y),$$

Clearly, $S_1(x)=S_2(x)$, so it suffices to minimize S_1 on $\{1,\dots,\ell\}$.

Minimizing $S_1(x)$. It is well known that $\sum_{i=1}^{n}|x-a_i|$ is minimized when x is the median of $\{a_1,\dots,a_n\}$.

Thus, choosing $x=\lfloor\ell/2\rfloor$ or $\lceil\ell/2\rceil$ and $y=\lfloor\ell/2\rfloor$ or $\lceil\ell/2\rceil$ yields S_{min}. Let $\ell=2n+1$. Then,

$$\begin{aligned}S_{\min}&=\ell\big(S_1(x)+S_2(y)\big)\\&=\ell(2\sum_{i=1}^{2n+1}|n+1-i|)\\&=\ell(4(1+2+...+n))\\&=2n(2n+1)((n+1))\end{aligned}$$

If $\ell=2n$,

$$\begin{aligned}S_{\min}&=\ell(4\sum_{i=1}^{2n}|n-i|)\\&=\ell(4(\frac{n(n+1)-n}{2}))\\&=(2n)(2n^2)\\&=4n^3\end{aligned}$$

as claimed.

This gives the ratio of mana cost to power as:

$$\text{Ratio} = \begin{cases} \frac{2n(n+1)}{2n+1} & \text{if } \ell = 2n+1 \\ n & \text{if } \ell = 2n \end{cases}$$

□

(b) Minimum mana cost for ℓ^2 with $\alpha \to \infty$

The logic of efficiency as in part 4) a) applies even stronger, now making it less efficient to move a rune with power greater than 1 compared to moving the same distance with a rune of power 1. Now, moving two runes of power 1 across 2 and 3 notches is more efficient than combining them, then moving the rune of power 2 by 2 notches. This means the same method as for $\alpha = 1$ applies, wither the same mana cost.

(c) The General Case

When $\alpha = 0$, we only rued to find the most efficient way to combine all of the runes, without any additional restrictions. When $\alpha \geqslant 1$, we again need only find the runes such that the sum of the distances from such that the sum of the distances from all other runes to it is minimised.

Notice that if a state is reached in which one path (which can overlap itself) can be drawn covering all runes without covering any empty notches, the minimum mana cost to combine all the runes from that configuration, equals the number of runes in it minus 1

This occurs in all initial configurations in 2) b), making the total mana cost equal $3^k - 1$ (for $\ell = 2^k$). However, the initial configurations in 2) a) are far more complicated, as the minimum distance from any nine to the other is 2 notches. Thus, some runes must be moved from the initial configuration in order to create a structure which can be chained together, as desired.

This brings up the general formula for creating a ℓ^2 power rune with $\alpha = 0$:

Let $s :=$ Minimum number of moves to create a connected chain across all runes, $r :=$ Total number of runes in the initial configuration Then, minimum mana cost $= r + s - 1$. However, finding s is difficult, as just shifting the runes from the initial configurations is often suboptimal, as see by the below configuration:

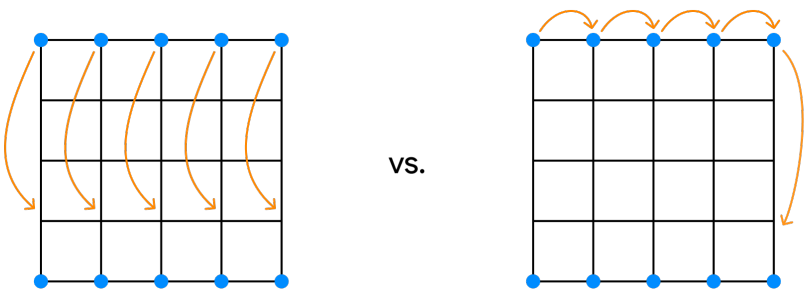

Figure 8: Suboptimality of moving from initial configurations

Evidently, combining the top row first is more efficient This added layer of complexity makes it very difficult to create a closed form for s, and we have not been able to find one in the case of 2)a), where manual checking for $\ell = 5, 6, 7, 8, 9$, and 10 give the minimal mana costs as $15, 24, 32, 44, 53$, and 69 , respectively. Now taking $\alpha \geqslant 1$, the same logic as in parts 4) a) and 4) b) of fixing one round then moving all other runes into it, one by one, The minimum mana cost is the minimum element of the set of the sum of distances from all other runes to a particular rune, taken across all runes.

However, a closed form for this could not be found in the cases of 2)a), 2)b), or the general case.

5 Mana Cost for Shapes of Functions

Question 5: In the previous situation, the weight of a rune of power k is given more generally by $f(k)$, for some function $f : \mathbb{R}_+ \to \mathbb{R}_+$ (The previous situation corresponds to $f(x) = x^\alpha$.) Try to generalise the answers to the previous question for various shapes of the function f, as general as possible.

Consider first functions $f(x)$ that are superadditive. That is: $f(m+n) \geqslant f(m) + f(n), \forall m, n \in \mathbb{R}$

Then, a similar concept of efficiency as in parts 4) a) and 4) b) comes into play, and gives similar results, as $f(x) = x^\alpha$ is suprodditive for $\alpha \geqslant 1$.

However, if $f(x)$ is subadditive, that is, $f(m+n) \leqslant f(m) + f(n) \quad \forall m, n \in \mathbb{R}$, the most efficient method is akin to the case of $\alpha = 0$ in part 4) a) , as it costs less mana to combine runes and then move them rather than moving each rune individually by the same distance.

There are other types of functions such as convex, concave, supermultiplicative, submultiplicative, etc, but those can be superadditive or subadditive over different intervals, making it impossible to formulate a most efficient method for such generated functions in general. To find the specific values of mana cost for superadditive and subadditive functions in the case of a full configuration is not difficult:

When $f(x)$ is superadditive, the mana cost of creating a rune of power ℓ^2 is just $f(1)$ multiplied by the mana cost in parts 4) b), as moving a rune of power 1 across a notch now has a mana cost of exactly $f(1)$. When $f(x)$ is subadditive:

If $2f(m+n) \leq f(m) + f(n)$, the minimal mana cost for creating the rune of power ℓ^2 will be: $f(1) + f(2) + \cdots + f\left(\ell^2 - 1\right) = \sum_{i=1}^{\ell^2-1} f(i)$, which is achieved by following the snake-like pattern as in Figure 3. This is because it always becomes more efficient to combine and create the highest power rune and move it as its weight will be lower than the weights of the runes that created it.

However, if $2f(m+n) > f(m) + f(n)$, it can be more efficient to combine two runes of stronger power than 1 instead, and it is tough to specify the most efficient method in general for this scenario.

6 Mana Costs for Fire Spells

Question 6: In order to create a fire spell, the witch works with slightly different runes than in the previous situation. This time, each rune carries not a number, but a set in $\mathbb{R}^2$. In the initial configuration, every rune carries the square of side length 1 whose center is the notch the rune starts in. When two runes merge, they form a new rune which carries the union of the sets carried by both runes. The power of a rune is the area of the set it carries, while its weight is the perimeter of this set. What is the minimal mana cost of forging a rune of power ℓ^2? In case it is impossible to obtain an exact formula, it will be of particular interest to study the asymptote of the ratio of the mana cost by the power of the rune that is obtained as $\ell \to +\infty$.

Conjecture for the minimum mana cost for $\ell = 2n$ and $\ell = 2n + 1$:

For $\ell = 2n$:

Combine all runes in each row from left-most to the rune in the $\lfloor \frac{\ell}{2} \rfloor^{th}$ column, and from the right-most to the rune in the $\lceil \frac{\ell}{2} \rceil^{th}$ column. Then, combine the runes in the $\lfloor \frac{\ell}{2} \rfloor^{th}$ column from top and bottom separately until both reach the $\lfloor \frac{\ell}{2} \rfloor^{th}$ row. Do similarly for the $\lfloor \frac{\ell}{2} \rfloor^{th}$ column, then combine the final two runes.

For $\ell = 2n + 1$: Combine the runes as in the previous case, but now only to the runes in the $\lfloor \frac{\ell}{2} \rfloor^{th}$ column.

This procedure seems to give a lower total mana cost than most other methods, but we were not able to prove this gave the lowest mana cost.

7 Further Research

Question 7: Suggest and study other research directions.

(a) Research Direction 1: Generalizing to k-dimensions

Here, we take a k-dimensional hypercube defined by $[1, \ell]^k$ as the workbench and solve for parts of each problem based on this new workbench.

Results for Parts of Problems 1, 2, 3, and 4:
1) a) Maximum number of healing spells is now $\lfloor \frac{\ell^k}{2} \rfloor$, as the total number of runes is ℓ^k.
b) Minimum Mana Cost = Number of Healing Spells = $\lfloor \frac{\ell^k}{2} \rfloor$
c) Each healing spell created still increases the length of the longest possible path by 2, starting from 1 when 0 healing spells have been created.

$\therefore$ Total maximum mana cost:

$\sum_{i=1}^{\lfloor \frac{\ell^k}{2} \rfloor} (2\ell - 1) = (\lfloor \frac{\ell^k}{2} \rfloor)^2$

2) a) Total number of runes = Number of healing spells in part 1) a) = $\lfloor \frac{\ell^k}{2} \rfloor$

Hence, Total Mana Cost = Total number of runes = $\lfloor \frac{\ell^k}{2} \rfloor$, as the minimum distance from any rune to another is still exactly 2 notches. Hence, ratio of number of healing spells to mana cost is still exactly equal to $\frac{1}{2}$

3) a) Similar to the case of 2 dimensions, $c_{min} = \lfloor \frac{n}{2} \rfloor$, while c_{max} is again unbounded as $\ell \to +\infty$. Thus, the range of values of $\frac{n}{c}$ is exactly the same as the 2-dimensional case.

4) When $\alpha = 0$, Minimum mana cost = Total Number of Runes - 1 = $\ell^k - 1$, ratio of mana cost to power = $\frac{\ell^k - 1}{\ell^k}$

When $\alpha \geq 1$, once again the minimum mana cost can be defined using a similar sum. Fixing a rune at coordinates $(a_1, a_2, \ldots, a_k)$, the mana cost of moving all other runes to it is:

$\sum_{i_1=1}^{\ell} \sum_{i_2=1}^{\ell} \cdots \sum_{i_k=1}^{\ell} |a_1 - i_1| + |a_2 - i_2| + \cdots + |a_k - i_k|$

The minimum value of this occurs, similarly to the case in 2 dimensions, when $a_i = \lfloor \frac{\ell}{2} \rfloor$ or $\lceil \frac{\ell}{2} \rceil$ $\forall i \in (1, 2, \ldots, k)$, and is equal to:

$$S_{min} = \begin{cases} \ell^{k-1} kn(n+1) & \text{if } \ell = 2n+1 \\ \ell^{k-1} kn^2 & \text{if } \ell = 2n \end{cases}$$

This is evident upon rewriting $S_{min} = \ell^{k-1} \sum_{j=1}^{k} (\sum_{i=1}^{l} |x_j - i|)$, whereupon similar manipulation as in the 2-dimensional case quickly leads us to the desired result.

Hence, ratio of mana cost to power:

$$\text{Ratio} = \begin{cases} \frac{kn(n+1)}{2n+1} & \text{if } \ell = 2n+1 \\ \frac{kn}{2} & \text{if } \ell = 2n \end{cases}$$

(b) Research Direction 2: Alternate Workbenches

Modifying the workbench into different shapes could significantly affect the methods used to move the runes across notches efficiently.

While changing the shape to be rectangular would keep most results similar [part 2) b) being a notable exception as creating those patterns becomes impossible for a rectangular workbench], having gaps in the middle of the workbench or modifying it into weirder shapes would change some results, especially those in parts 3 and 6, as they are heavily dependent on the orientation and structure of the board.