On the Calculation of a Perfect Gem Equivalent Value for Arbitrary Rune Combinations in Diablo 2: Resurrected

May 11, 2022

By: Osvaldo Enriquez

Email: ozzy@d2charsifood.com

Abstract

The game Diablo 2: Resurrected (D2R) is an online action role playing game with a vibrant user driven economy. However, it does not necessarily offer a good liquid currency to facilitate trading within the game. There is gold in the game, but it is generally so easily obtained that is holds little to no value amongst the user base. Thus, most users tend to use runes as a form of currency.

Unfortunately, valuing runes vs each other is not a straight forward task. In fact it is so inconvenient to use runes as currency that a significant portion of the trading economy has resorted to a website controlled digital currency called forum gold [1].

Runes can be combined to create higher runes, which acts as a hard cap on the relative value of runes vs one another. Though in reality the actual value of runes is driven both by their usefulness in runewords, and their relative scarcity through ladder seasons [2].

The goal of this paper is to calculate a "perfect gem equivalent" for every individual rune so that combinations of runes can be objectively compared to one another.

We will define an initial set of common rune trades based off of my own personal market observations. This initial set will be used to calculate a best-fit model where each value is assigned a scalar pgem equivalent.

Furthermore, we will demonstrate how the calculations can be expanded to include an arbitrary number of rune trades to calculate a best-fit model.

d2CharsiFood.com uses real market trade data to constantly refine and re-calculate the best-fit pgem equivalent model.

Runes

There are 33 total runes in D2R. Runes can be combined to form higher runes. The following table lists the possible runes, and the rune combination to make them:

	Rune	Cube Recipe
1	El	N/A
2	Eld	3x El
3	Tir	3x Eld
4	Nef	3x Tir
5	Eth	3x Nef
6	Ith	3x Eth
7	Tal	3x Ith
8	Ral	3x Tal
9	Ort	3x Ral
10	Thul	3x Ort
11	Amn	3x Thul + chipped Topaz
12	Sol	3x Amn + chipped Amethyst
13	Shael	3x Sol + chipped Sapphire
14	Dol	3x Shael + chipped Ruby
15	Hel	3x Dol + chipped Emerald
16	Io	3x Hel + chipped Diamond
17	Lum	3x Io + flawed Topaz
18	Ko	3x Lum + flawed Amethyst
19	Fal	3x Ko + flawed Sapphire
20	Lem	3x Fal + flawed Ruby
21	Pul	3x Lem + flawed Emerald
22	Um	2x Pul + flawed Diamond
23	Mal	2x Um + Topaz
24	Ist	2x Mal + Amethyst
25	Gul	2x Ist + Sapphire
26	Vex	2x Gul + Ruby
27	Ohm	2x Vex + Emerald
28	Lo	2x Ohm + Diamond
29	Sur	2x Lo + flawless Topaz
30	Ber	2x Sur + flawless Amethyst
31	Jah	2x Ber + flawless Sapphire
32	Cham	2x Jah + flawless Ruby
33	Zod	2x Cham + flawless Emerald

Example

This is a small example to demonstrate the calculations.

Example Initial Trade Sample Set

The following equations demonstrate some of the most common rune trades:

(1) $2xLem = Pul$
(2) $Lem + Pul = Um$
(3) $2xLem + Pul = Um$
(4) $40x\ Perfect\ Gems = Pul$

Notice that equations (2) and (3) are already mathematically contradictory unless Lem = 0. But Lem cannot = 0 in any reasonable pgem equivalent model.

We can rewrite those 4 equations as a matrix equation:

(5) $$\begin{bmatrix} 0 & 2 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 2 & 1 & 0 \\ 40 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} pgems \\ Lem \\ Pul \\ Um \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} pgems \\ Lem \\ Pul \\ Um \end{bmatrix}$$

Now we can subtract to get all the unknowns on the same side.

(6) $$\begin{bmatrix} 0 & 2 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 2 & 1 & 0 \\ 40 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} pgems \\ Lem \\ Pul \\ Um \end{bmatrix} - \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} pgems \\ Lem \\ Pul \\ Um \end{bmatrix} = 0$$

(7) $$\begin{bmatrix} 0 & 2 & -1 & 0 \\ 0 & 1 & 1 & -1 \\ 0 & 2 & 1 & -1 \\ 40 & 0 & -1 & 0 \end{bmatrix} \begin{bmatrix} pgems \\ Lem \\ Pul \\ Um \end{bmatrix} = 0$$

Now we have a matrix equation in the form of Ax = 0. Since we know equations (2) and (3) are contradictory, we know that there is no real solution for x (except x = 0). However, we can use Singular Value Decomposition [3] to find the best-fit, non-zero, solution for x.

Running SVD on equation (7) and then normalizing so pgems = 1 yields:

$$\begin{bmatrix} pgems \\ Lem \\ Pul \\ Um \end{bmatrix} = \begin{bmatrix} 1 \\ 18.5 \\ 40 \\ 68.7 \end{bmatrix}$$

Python Code

```
import numpy as np

# define the A matrix
A = np.matrix([
    [0, 2, -1, 0],
    [0, 1, 1, -1],
    [0, 2, 1, -1],
    [40, 0, -1, 0],
])

# run SVD
u, s, vh = np.linalg.svd(A)

# transpose the V matrix, cus Python is weird...
vh = np.transpose(vh)

# grab the last column from the v matrix
best_vals = vh[:, -1]
# grab the raw value for pgems
pgem_val = best_vals[0, 0]
# divide all values by pgem value to normalize pgems to 1
best_vals = best_vals / pgem_val
print(best_vals)
```

Initial Trades Sample Set

The following table lists the initial trade sample set used to train the initial model:

1	2x Lum	1x Ko
2	1x Lum + 1x Ko	1x Fal
3	2x Lum + 1x Ko	1x Fal
4	2x Ko	1x Fal
5	1x Lum + 2x Ko	1x Fal
6	2x Lum + 2x Ko	1x Fal
7	1x Ko + 1x Fal	1x Lem
8	2x Ko + 1x Fal	1x Lem
9	2x Fal	1x Lem
10	1x Ko + 2x Fal	1x Lem
11	2x Ko + 2x Fal	1x Lem
12	1x Fal + 1x Lem	1x Pul
13	2x Fal + 1x Lem	1x Pul
14	2x Lem	1x Pul
15	1x Fal + 2x Lem	1x Pul
16	2x Fal + 2x Lem	1x Pul
17	1x Lem + 1x Pul	1x Um
18	2x Lem + 1x Pul	1x Um
19	1x Fal + 1x Lem + 1x Pul	1x Um
20	2x Fal + 1x Lem + 1x Pul	1x Um
21	1x Pul + 1x Um	1x Mal
22	1x Lem + 1x Pul + 1x Um	1x Mal
23	2x Lem + 1x Pul + 1x Um	1x Mal
24	1x Um + 1x Mal	1x Ist
25	1x Pul + 1x Um + 1x Mal	1x Ist
26	1x Lem + 1x Pul + 1x Um + 1x Mal	1x Ist
27	2x Lem + 1x Pul + 1x Um + 1x Mal	1x Ist
28	1x Mal + 1x Ist	1x Gul
29	1x Um + 1x Mal + 1x Ist	1x Gul
30	1x Pul + 1x Um + 1x Mal + 1x Ist	1x Gul
31	1x Ist + 1x Gul	1x Vex
32	1x Mal + 1x Ist + 1x Gul	1x Vex
33	1x Gul + 1x Vex	1x Ohm
34	1x Ist + 1x Gul + 1x Vex	1x Ohm
35	1x Vex + 1x Ohm	1x Lo
36	1x Gul + 1x Ohm	1x Lo
37	1x Ist + 1x Ohm	1x Lo
38	1x Ohm	1x Lo
39	1x Ohm	1x Sur
40	1x Lo	1x Sur
41	1x Ist + 1x Ohm	1x Sur
42	1x Ist + 1x Lo	1x Sur
43	1x Ohm + 1x Sur	1x Ber

44	1x Lo + 1x Sur	1x Ber
45	1x Vex + 1x Sur	1x Ber
46	1x Ohm + 1x Sur	1x Jah
47	1x Lo + 1x Sur	1x Jah
48	1x Vex + 1x Sur	1x Jah
49	1x Ber	1x Jah
50	1x Cham	1x Zod
51	1x Ohm	1x Cham
52	1x Ohm	1x Zod
53	1x Lo	1x Cham
54	1x Lo	1x Zod

The initial pgem equivalent values differ for each region (hardcore/softcore, ladder/non-ladder). For this example, we will continue with 40x pgems = Pul.

Following the same model as the above example, we get:

	Rune	Pgem Equivalent
	Pgem	1
17	Lum	1.9
18	Ko	3.27
19	Fal	8.24
20	Lem	17.2
21	Pul	40
22	Um	73.51
23	Mal	156.20
24	Ist	275.11
25	Gul	491.62
26	Vex	988.67
27	Ohm	1582.8
28	Lo	1829.34
29	Sur	1939.98
30	Ber	3457.08
31	Jah	3457.08
32	Cham	1744.03
33	Zod	1744.03

References

1. d2jsp.org forum gold faq: https://forums.d2jsp.org/info.php?p=35
2. Thread on reddit explaining ladder seasons: https://www.reddit.com/r/Diablo/comments/1mblsc/can_anyone_explain_me_what_ladder_is/
3. Python Numpy svd method: https://numpy.org/doc/stable/reference/generated/numpy.linalg.svd.html