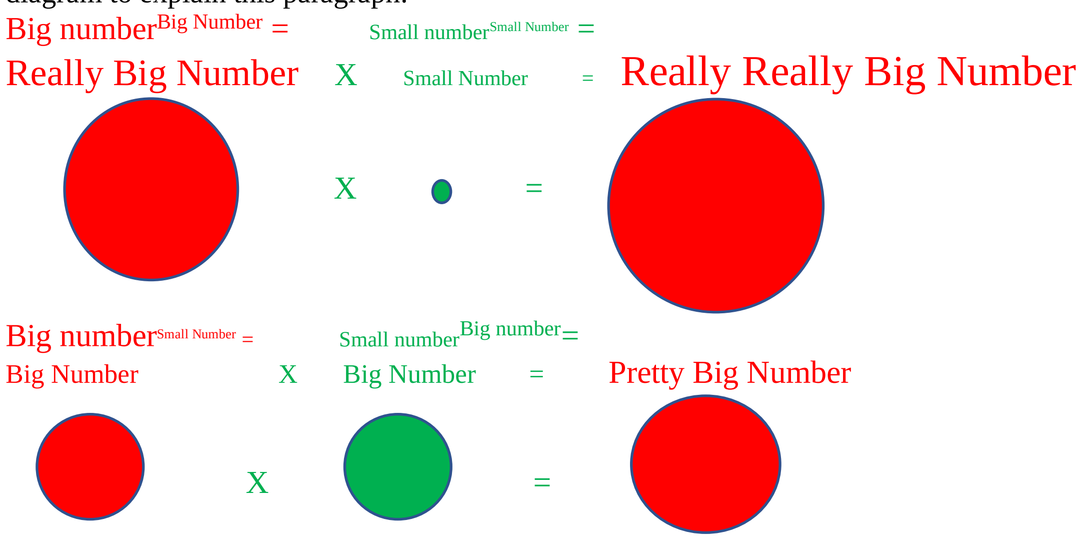
Adam Frenkel

My algorithm sorts the lists and then takes the next item off each list for its next a_i and b_i. It then calculates a^b. It continues doing this, multiplying the previous results together, till there no more longs in the lists.

Intuitively this algorithm works because as long as you always pair the two biggest remaining longs from the list you'll always get the largest number possible for that round. Also, if you wouldn't pair the largest two remaining numbers, rather you would pair the largest number from list A with a smaller number from List B, and later you would pair the largest number in B with another number from list A, you would end up having a smaller total. This is because *exponents grow exponentially* therefore the bigger numbers always benefit more by being paired with a larger number than a smaller number would. Here's a diagram to explain this paragraph:



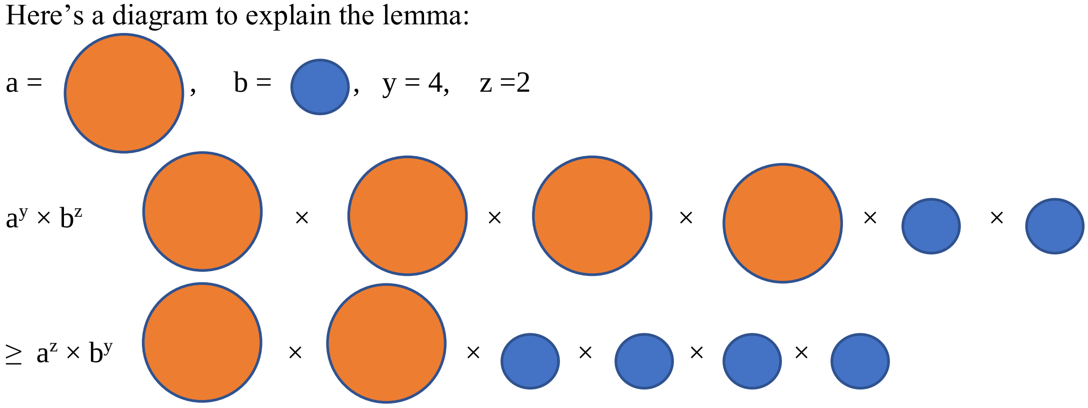
Lemma: Given four numbers, a, b, y, z, where $a \geq b$ and $y \geq z$. $a^y \times b^z \geq a^z \times b^y$.

Direct Proof to the Lemma: An exponent (3^4) is just notation which, by definition, means multiply the base times itself as many times as the power says to ($3 \times 3 \times 3 \times 3$).

-Thus, the equation $a^y \times b^z$, really means multiply a (the larger number) more times than b (the smaller number). By basic math, multiplying a larger number (a) more times (y) and a smaller number (b) less times (z) will yield a larger result than multiplying a smaller number (b) more times (y) and a larger number (a) less times (z).

Thus, $a^y \times b^z \geq a^z \times b^y$. QED

Here's a diagram to explain the lemma:



(Note throughout the following proof whenever I write terms signifying greatness, for example larger, more, higher, I really mean greater than or equal, for example if I write a is larger than b, I mean $a \geq b$.)

Theorem: My Algorithm maximizes the payout of the given formula.

Proof By Induction:

Base Case: After the first step my result (MR_1) is $\geq$ the optimal result after step 1 (OR_1).
-My algorithm takes the largest two longs from the lists (a and y), therefore MR_1 is the highest possible result. This is because if we were to take any other long (c) from another list instead of one of the longs (a) it would be a smaller number ($c \leq a$), and an exponent with a smaller base or power (or both) is smaller than an exponent with a higher base or power (or both), thus, $MR_1 \geq OR_1$.
Inductive Step: Assuming $MR_i \geq OR_i$.
-At the next step, i +1, my algorithm will take the largest long off each list and pair them.
-(Considering that multiplication is associative the exact order doesn't matter, so for simplicity let's assume that the optimal algorithm will take the largest items off one of the lists at this point, as it makes no difference at which stage that actually happens.)
-Since my algorithm always takes the next largest element off each list the elements of each list are paired in descending order.
-Meaning the highest number of one list is paired with the highest number of the other list, then the second highest is paired with the second highest etc. (Meaning, given a, b, y, z, where $a \geq b$ and $y \geq z$, then $a^y \times b^z$ etc.). Thus, at i + 1, the two largest remaining pairs will be paired (b^z).
-The only way for the optimal algorithm to get ahead of my algorithm would be if it had a larger number from a previous step (or from a later step, but for simplicity I'm assuming it's also pairing in descending order as mentioned above) to pair with the current largest number of this list (b^y). Since at this step it would increase the payout more than my algorithm.
-But, that would mean that at a previous step the largest number on one list wasn't paired with the largest of the other. In other words, where $a \geq b$ and $y \geq z$, $a^z \times b^y$.
- By the lemma, that would mean the optimal algorithm would produce a smaller payout than my algorithm, even though at this step it happens to be increase more.
-Thus, $MR_{i+1} \geq OR_{i+1}$ QED