Jumbled String

This problem seems much more complex than the previous two…

I first noted that both a and d must be triangular numbers. This is because each added 0 (1) increases the 00 subsequences by the number of already existing 0s (1s).

Next, I constructed a short python program to brute force the reverse of the problem: finding a, b, c, and d given a string. Some of the output from this program is shown below:

```
0 0 0 0 : 0 : 0
0 0 0 0 : 0 : 1
1 0 0 0 : 1 : 00
0 1 0 0 : 1 : 01
0 0 1 0 : 1 : 10
0 0 0 1 : 1 : 11
3 0 0 0 : 3 : 000
1 2 0 0 : 3 : 001
1 1 1 0 : 3 : 010
0 2 0 1 : 3 : 011
1 0 2 0 : 3 : 100
0 1 1 1 : 3 : 101
0 0 2 1 : 3 : 110
0 0 0 3 : 3 : 111
6 0 0 0 : 6 : 0000
3 3 0 0 : 6 : 0001
3 2 1 0 : 6 : 0010
1 4 0 1 : 6 : 0011
3 1 2 0 : 6 : 0100
1 3 1 1 : 6 : 0101
1 2 2 1 : 6 : 0110
0 3 0 3 : 6 : 0111
3 0 3 0 : 6 : 1000
1 2 2 1 : 6 : 1001
1 1 3 1 : 6 : 1010
0 2 1 3 : 6 : 1011
1 0 4 1 : 6 : 1100
0 1 2 3 : 6 : 1101
0 0 3 3 : 6 : 1110
0 0 0 6 : 6 : 1111
10 0 0 0 : 10 : 00000
6 4 0 0 : 10 : 00001
6 3 1 0 : 10 : 00010
3 6 0 1 : 10 : 00011
6 2 2 0 : 10 : 00100
3 5 1 1 : 10 : 00101
3 4 2 1 : 10 : 00110
1 6 0 3 : 10 : 00111
6 1 3 0 : 10 : 01000
3 4 2 1 : 10 : 01001
3 3 3 1 : 10 : 01010
1 5 1 3 : 10 : 01011
3 2 4 1 : 10 : 01100
1 4 2 3 : 10 : 01101
1 3 3 3 : 10 : 01110
```

The first four terms here are a, b, c, and d, the fifth term is their sum $a+b+c+d$, and the final term is the string that they produce.

I spent a (very) long time looking for patterns in the output, and found that:

- a and d must be triangular numbers.
- $a+b+c+d$ must be a triangular number.
- If $b=0$ and $c=0$, then only of a or d can >0.
- If a is the Ath triangular number, then:
  - If $A=0$, then the number of 0s n_0 is 1 iff $b>0$ or $c>0$, otherwise $n_0=0$.
  - If $A>0$, then $n_0=A$.
- If d is the Dth triangular number, then:
  - If $D=0$, then the number of 1s n_1 is 1 iff $b>0$ or $c>0$, otherwise $n_1=0$.
  - If $D>0$, then $n_1=D$.
- The product of the numbers of 0s and 1s $n_0 n_1$ must be equal to $b+c$.

Reasoning some more and looking at the larger outputs from the python program, if a, b, c, and d are valid, then there is some valid result string that matches the regex `1*0*1?0*1*`, i.e. some number of 0s sandwiched by some number of 1s, with a single 1 optionally in the middle of the 0s. The number of 0s and 1s in each segment is determined as a function of c, since c is what specificies the number of 10s.