# Count of Matches for a Highly Ambiguous Regular Expression

*Mike French*

2022-11-18

**Abstract**

We evaluate the number of matches of the regular expression `(a?)^n (a*)^n` for the string `a^n`. We show the total count is the dot product of two vectors taken from a row and a diagonal of Pascal's Triangle. The counts are given for n=1..9.

**Problem Statement**

Let the exponential meta-operator '`^`' mean repetition. So '`a^4`' for a string means repeating the '`a`' character 4 times `'aaaa'`, and '`(a?)^4`' for a regex means '`(a?a?a?a?)`'. We will consider a regex of the form '`(a?)^n (a*)^n`' matching a string of '`a^n`', which is a highly ambiguous exaggeration of the example given in [Cox].

**Definitions**

Consider the match counts for each operator in the regular expression:

- Optional quantifier *zero or one* '`?`' matches 0 or 1 characters.
  The first half of the expression '`?^n`' matches a sequence of *n* binary digits
- Star quantifier *zero or more* '`*`' matches *0..n* characters.
  The second half '`?^n`' matches a sequence of *n* numbers in the range *0..n*.

Count the ways to get a specific partial sum of matches *m=0..n* for each half of the expression. Assemble these counts into a vector of *n+1* values over index *m=0..n*:

- $M_{?n}[m]$ ways for '`?^n`' to match *m* characters.
- $M_{*n}[m]$ ways for '`*^n`' to match *m* characters

For a successful match, the two counts for each half of the expression must add up to *n*:
if the second matches *m*, the first must have matched *n-m.*
*S*o the total count is the pairwise multiplication of the M vectors:

Total count : $M_n = \sum_{m=0..n} ( M_{?n}[n-m] \times M_{*n}[m] )$

**? Quantifiers**

The count value $M_{?n}[m]$ is:

- The number of ways to get *n* optional matches accepting a total of *m* characters.
- The count of n-digit binary numbers that have *m* bits set (1s).
- The number of ways of choosing *m* from *n*, which is the binomial coefficient *nCm*.

The table of binomial coefficients is just Pascal's Triangle with the recurrence relation:

```
nCm = (n-1)C(m-1) + (n-1)Cm
```

The vector of counts $M_{?n}[m]$ is *nCm* for *m=0..n,* which is just the $n^{th}$ diagonal in Pascal's Triangle.

The diagonal vector is symmetrical because: `nCm = nC(n-m)` and so $M_{?n}[m] = M_{?n}[n-m]$ which means we can reverse the $M_{?n}$ vector and justify the dot product formulation:

Total count: $M_n = \sum_{m=0..n} ( M_{?n}[m] \times M_{*n}[m] ) = M_{?n} \bullet M_{*n}$

*** Quantifiers**

The count value $M_{*n}[m]$ is:

- The number of ways to get *n* zero-or-more matches accepting a total of *m* characters.
- *Sum of digits* problem: the ways *n* numbers in the range *0..n* can have sum of *m.*

Construct a recurrence relation for sum of digits problem. Each set of *n-1* numbers with a sum in the range *0..m* can be brought up to sum *m* by adding the number *n-m:*

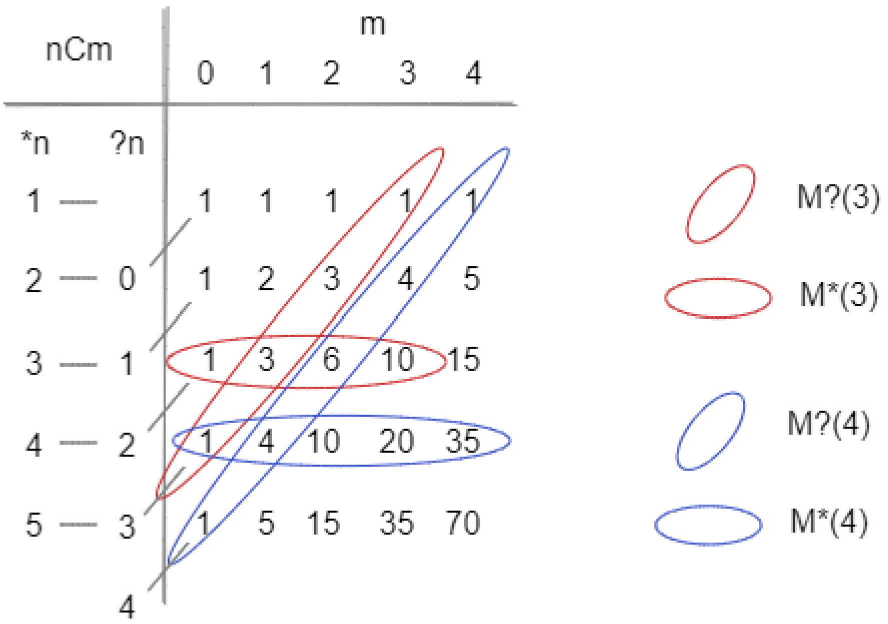
$$M_{*n}[m] \;=\; \sum\nolimits_{m=0..n} \;\; M_{*n-1}[m] \;=\; M_{*n}[m-1] \;+\; M_{*n-1}[m]$$

Each entry is the sum of values in the row above (*n-1*) up to and including the same column (*m*), and hence also the sum of the terms to the left and above. The recurrence is grounded in 1:

$$\forall_{m=0,n} \;\; M_{*1}[m] \;=\; 1, \qquad \forall_{n>1} \;\; M_{*n}[0] \;=\; 1$$

This is just Pascal's Triangle, but indexed by 1-based $n^{th}$ row, rather than the 0-based $n^{th}$ diagonal.

**Vectors in Pascal's Triangle**




**Formula**

$$M_n \;=\; \sum\nolimits_{m=0..n} \;\; nCm \;\times\; (n+m-1)Cm$$

**Specific Examples**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| M(1) = | [1,1] | * | [1,1] | = | 1+1 | = | 2 |
| M(2) = | [1,2,1] | * | [1,2,3] | = | 1+4+3 | = | 8 |
| M(3) = | [1,3,3,1] | * | [1,3,6,10] | = | 1+9+18+10 | = | 38 |
| M(4) = | [1,4,6,4,1] | * | [1,4,10,20,35] | = | 1+16+60+80+35 | = | 192 |
| M(5) = | [1,5,10,10,5,1] | * | [1,5,15,35,70,126] | = | 1+25+150+350+350+126 | = | 1,002 |
| M(6) = | [1,6,15,20,15,6,1] | * | [1,6,21,56,126,252,462] | = | 1+36+315+1120+1890+1512+462 | = | 5,336 |

| *n* | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| M*n* | 2 | 8 | 38 | 192 | 1,002 | 5,336 | 28,814 | 157,184 | 864,146 |

**References**


[Cox] "*Regular Expression Matching Can Be Simple And Fast*", Russ Cox, January 2007 [web].