The Mathematica® Journal

Betting Two Patterns against Each Other

Jan Vrbik

We present a technique for computing the probability that a specific pattern of successes and failures is generated randomly before another such pattern, thus winning the corresponding game. The program we build for this purpose finds the mean and standard deviation of the number of trials needed to complete one round of such a game. It can be used to maximize the probability of winning a game by choosing the best possible pattern, and also by adjusting the probability of a success. Finally, we verify our theoretical results by a Monte Carlo simulation.

■ Generating a Single Pattern

We consider an experiment (such as flipping a coin or rolling a die) in which each trial results in either a success S (obtaining a head, a six, etc.) or a failure F, with probability p and $q = 1 - p$, respectively. The trials are independently repeated until a specific pattern (such as FSFSF) is generated for the first time.

□ Starting from Scratch

Following [1], we introduce g_n to be the probability that the pattern has been completed for the first time in the n^{th} trial. Similarly, u_n is the probability of completing the pattern in the n^{th} trial, but this may now be its k^{th} occurrence, where k is any positive integer (we also set $g_0 = 0$ and $u_0 = 1$). Note that the g_n probabilities must add up to 1, whereas the u_n probabilities have an infinite sum, since $u_n \to u_\infty > 0$. In the definition of u_n, it is important to stipulate that consecutive occurrences of the pattern are not allowed to overlap—once the pattern is generated, none of its parts can be used to help build its next occurrence. Thus, for example, the sequence FSFSFSFSF contains only one completion of FSFSF, not three.

To find the probabilities u_n, we assume that n trials of the experiment have just been completed. The probability of the last five of these trials resulting in FSFSF (it is easier to explain the procedure using an example) can be expanded using the formula of total probability as follows:

$$p^2\, q^3 = u_n + u_{n-2}\, p\, q + u_{n-4}\, p^2\, q^2. \tag{1}$$

Here, the left-hand side is the simple probability of generating FSFSF; the right-hand side partitions the sample space according to where (during the last five trials) the corresponding pattern has been actually completed (in terms of our example, this could have happened at the last trial, but also either two or four trials earlier).

To make sure not to miss any such possibility, one should slide the pattern past itself, obtaining a term for each perfect match, thus:

FSFSF	FSFSF■	FSFSF■■	FSFSF■■■	FSFSF■■■■
FSFSF	■FSFSF	■■FSFSF	■■■FSFSF	■■■■FSFSF
yes	no	yes	no	yes

Multiplying (1) by s^n and summing over n, from 5 to ∞ (the equation is incorrect when $n \le 4$), one obtains

$$\frac{p^2\, q^3\, s^5}{1-s} = (U(s)-1)\left(1 + p\, q\, s^2 + p^2\, q^2\, s^4\right),$$

where $U(s)$ is the generating function of the u_n sequence. We had to subtract 1 from $U(s)$ to account for the missing $u_0 = 1$ term ($u_1 = u_2 = u_3 = u_4 = 0$).

From [1], we know that the probability generating function of the g_n sequence is given by

$$G\,(s) = \frac{U(s)-1}{U(s)} = \frac{1}{1+\frac{1}{U(s)-1}} = \frac{1}{1+(1-s)\,Q(s)}, \tag{2}$$

where

$$Q\,(s) = \frac{1 + p\, q\, s^2 + p^2\, q^2\, s^4}{p^2\, q^3\, s^5}.$$

The numerator of $Q(s)$ is obtained from the right-hand side of (1) after replacing u_{n-i} by s^i, while the denominator equals the left-hand side of (1) multiplied by s^N (where N is the total length of the pattern).

The computation of $G(s)$ is done by the following program.

```
PGF[b_] := Module[{n, a = conv[b], h = 0}, n = Length[a];
  Do[
    h = h + If[Drop[a, i - 1] == Drop[a, 1 - i],
        poly[Drop[a, n + 1 - i]], 0], {i, n}];            1
                                               ───────────────────── ]
                                                              h
                                               1 + (1 - s) ───────
                                                           poly[a]
```

The first line converts the argument from a string, say "FSFSF", to a binary list {0, 1, 0, 1, 0}, sets `n` to be the corresponding length, and initializes `h` to 0. The second line slides the list past itself by `i - 1` symbols (where `i` ranges from `1` to `n`) and, whenever a full match results, adds the product of the corresponding powers of `p` and `q` to `h`. The last line converts the resulting `h` to $G(s)$.

The program uses `conv` to convert a string to a binary list and `poly` to convert a binary list to the probability of generating that string, further multiplied by `s`$^{\texttt{n}}$.

```
conv[a_] := Map[If[# == "S", 1, 0] &, Characters[a]]
```

```
poly[a_] := Module[{n = Length[a], i = Total[a]}, p^i (1 - p)^(n-i) s^n]
```

Here is an example. The result of `PGF["FSFSF"]` returns the result quoted in [2].

```
conv["FSFSF"]
```

```
{0, 1, 0, 1, 0}
```

```
poly[{0, 1, 0, 1, 0}]
```

$$(1-p)^3\, p^2\, s^5$$

```
PGF["FSFSF"]
```

$$\frac{1}{1+\frac{(1-s)\,\left(1+(1-p)\,p\,s^2+(1-p)^2\,p^2\,s^4\right)}{(1-p)^3\,p^2\,s^5}}$$

Based on the probability generating function $G(s)$, we can easily find the corresponding mean of the number of trials required to generate the pattern for the first time (see [3]) by

$$\mu = G'(1) = Q(1),$$

and the variance by

$$\sigma^2 = G''(1) + \mu - \mu^2 = 2\,Q'(1) + \mu^2 + \mu.$$

In the case of SFSFS, one gets $\mu = 42$ and $\sigma = 37.87$ for $p = \frac{1}{2}$, and $\mu = 72.05$ and $\sigma = 67.72$ for $p = \frac{1}{6}$.

□ Head-Start Distributions

In the next section, we will also need the conditional version of the distribution of the number of trials to obtain a specific pattern of length N, given that the first k of its symbols have already been generated (where $0 \le k \le N$). Let $g_n^{(k)}$ be the corresponding conditional probability, namely that exactly n extra trials will be needed to complete the pattern (see [4]). Based on what happens in the next trial (after the first k symbols are already there), and using the formula of total probability, one can derive the following set of equations for ${g_n}^{(k)}$ (again, using the FSFSF pattern as our example):

$$\begin{aligned}
{g_n}^{(0)} &= p\,{g_{n-1}}^{(0)} + q\,{g_{n-1}}^{(1)},\\
{g_n}^{(1)} &= p\,{g_{n-1}}^{(2)} + q\,{g_{n-1}}^{(1)},\\
{g_n}^{(2)} &= p\,{g_{n-1}}^{(0)} + q\,{g_{n-1}}^{(3)},\\
{g_n}^{(3)} &= p\,{g_{n-1}}^{(4)} + q\,{g_{n-1}}^{(1)},\\
{g_n}^{(4)} &= p\,{g_{n-1}}^{(0)} + q\,{g_{n-1}}^{(5)}.
\end{aligned} \tag{3}$$

The only nontrivial task to set these up is to establish the subscripts of both g_{n-1} on the right-hand side of each equation. This is done as follows: to the existing k symbols we append first an extra S (to find the first subscript) and then an extra F (to find the second one). Then, for each of the two strings thus created, we must figure out how many of its symbols (at most) can be used to build the whole pattern. Clearly, when appending the "correct" symbol, the answer is $k+1$, for the "incorrect" one, it can be anywhere from 0 to k. The following table may help:

+S	+F
F + S	F + F
FS + S	FS + F
FSF + S	FSF + F
FSFS + S	FSFS + F

Note that ${g_n}^{(0)}$ is the old g_n and that ${g_0}^{(5)} = 1$ (the full pattern has already been completed—no extra trials are needed).

Multiplying each equation in (3) by s^n and summing over n from 1 to ∞, we obtain the following set of linear equations for the corresponding probability generating functions $G^{(0)}(s), \ldots, G^{(4)}(s)$:

$$\begin{aligned}
G^{(0)}(s) &= s\,p\,G^{(0)}(s) + s\,q\,G^{(1)}(s),\\
G^{(1)}(s) &= s\,p\,G^{(2)}(s) + s\,q\,G^{(1)}(s),\\
G^{(2)}(s) &= s\,p\,G^{(0)}(s) + s\,q\,G^{(3)}(s),\\
G^{(3)}(s) &= s\,p\,G^{(4)}(s) + s\,q\,G^{(1)}(s),\\
G^{(4)}(s) &= s\,p\,G^{(0)}(s) + s\,q,
\end{aligned} \tag{4}$$

since $G^{(5)}(s) = 1$ (as ${g_0}^{(5)} = 1$ and ${g_n}^{(5)} = 0$ for $n > 1$). After solving these, one can verify (as a way of checking) that $G^{(0)}(s)$ agrees with the old $G(s)$.

Again, the whole task can be delegated to the following program.

```
Gset[b_] := Module[{a = conv[b], eq = {}, n}, n = Length[a];
Do[
   eq = Append[eq,
     G_i == s p G_ovlp[Append[Take[a,i],1],a] +
       s (1 - p) G_ovlp[Append[Take[a,i],0],a]], {i, 0, n - 1}];
Solve[eq /. G_n -> 1, Table[G_i, {i, 0, n - 1}]][[1]] // Simplify]
```

The first line is self-explanatory (compare it with the first line of PGF). The second and third lines build the corresponding set of n equations for the unknown $G^{(i)}(s)$—now denoted G_i—and the last line solves them.

The critical ingredient is the following program for establishing how deeply one string can penetrate another, while fully matching the overlapping symbols.

```
ovlp[a_, b_] := Module[{n = Min[Length[a], Length[b]], A},
   A = Take[a, -n]; While[Take[A, -n] != Take[b, n], n = n - 1];
  n]
```

The first line initializes n to its largest potential value (the length of the shorter string). The second line matches the last n elements of the first string to the first n elements of the second string. If a perfect match is found, n is returned; otherwise, n is reduced by 1 and the process is repeated until a perfect match is achieved (when no perfect match ever results, n returns the value of 0). Thus, for example, the following command returns the value of 2.

```
ovlp[{0, 0, 1}, {0, 1, 0, 1}]
```

```
2
```

This illustrates the use of Gset.

```
D[G_3 /. Gset["FSFSF"], s] /. {s -> 1, p -> 1 / 2}
```

```
32
```

It tells us that, with the FSF head start, generating the FSFSF pattern takes 32 extra flips, on average.

This is the number of flips needed on average for $p = \frac{1}{6}$.

```
D[G_3 /. Gset["FSFSF"], s] /. {s -> 1, p -> 1 / 6.}
```

```
62.208
```

Competing Patterns

Suppose that each of two players selects a specific pattern and bets that, in a series of independent Bernoulli trials, the pattern appears sooner than the pattern chosen by his opponent, as investigated in [5]. We want to compute the probability of winning this game for each of the two players, the game's expected duration in terms of the number of trials, and the corresponding standard deviation.

To achieve this, we assume that n trials of the experiment have just been completed, and define the following:

- $x_{1,n}$ to be the probability that the first of the two patterns has been completed at the n^{th} trial for the first time without being preceded by the second pattern (and thus winning the game); we take $x_{1,0}$ to be equal to 0
- $x_{2,n}$, similarly, to be the probability of the second pattern winning the game at the n^{th} trial

We also need the $g_{1,n}$, the probability of the first pattern being completed for the first time at the n^{th} trial—ignoring the second pattern (this is the old g_n) and $g_{2,n}$ (vice versa), and the following modification of these.

Let $\hat{g}_{1,n}$ be the conditional probability that the first pattern requires an extra n trials to be generated for the first time, given that the second pattern has just occurred (this time, the first pattern is allowed to "get help" from any portion of the second pattern). It is obvious that these probabilities equal one of the ${g_n}^{(k)}$ of the previous section (constructed for the first pattern), where k is determined by sliding the second pattern past the first one until the longest perfect match is found. Thus, for example, when the first pattern is FSFSF and the second one is FFFFSF, k is equal to 3; we can get the answer by typing `ovlp[{0, 0, 0, 0, 1, 0}, {0, 1, 0, 1, 0}]`.

Similarly (in the same vice versa manner) we define $\hat{g}_{2,n}$.

Probability of Winning

Partitioning the sample space (of n trials) according to the trial at which the second pattern first occurred (including the possibility that it has not occurred yet—the last event of this partition), the formula of total probability yields

$$g_{1,n} = x_{2,0}\,\hat{g}_{1,n} + x_{2,1}\,\hat{g}_{1,n-1} + x_{2,2}\,\hat{g}_{1,n-2} + \ldots + x_{2,n}\,\hat{g}_{1,0} + x_{1,n}.$$

The left-hand side is the simple probability of the first pattern having been completed for the first time at trial n. On the right-hand side, this probability is broken down according to the first occurrence of the second pattern at trial $0, 1, 2, \ldots, n$. When the second pattern is completed at trial k, it wins, explaining the $x_{2,k}$ factor. This factor is multiplied by the conditional probability of completing the first pattern for the first time after the occurrence of the second pattern, in exactly $n-k$ trials, that is, $\hat{g}_{1,n-k}$. The last term accounts for the remaining possibility of the second pattern not having occurred yet and thus letting the first pattern win, which has probability $x_{1,n}$.

Multiplying the previous equation by s^n and summing over n from 0 to ∞ results in

$$G_1(s) = X_2(s)\,\hat{G}_1(s) + X_1(s).$$

To understand why, recall that

$$\begin{aligned}Y(s)\,Z(s) = {} & y_0\,z_0 + (y_0\,z_1 + y_1\,z_0)\,s + \\ & (y_0\,z_2 + y_1\,z_1 + y_2\,z_0)\,s^2 + (y_0\,z_3 + y_1\,z_2 + y_2\,z_1 + y_3\,z_0)\,s^3 + \ldots.\end{aligned}$$

The vice versa argument similarly yields

$$G_2(s) = X_1(s)\,\hat{G}_2(s) + X_2(s).$$

These two linear equations for $X_1(s)$ and $X_2(s)$ can be easily solved.

```
Solve[{G₁ == X₂ Ĝ₁ + X₁, G₂ == X₁ Ĝ₂ + X₂}, {X₁, X₂}][[1]]
```

$$\left\{X_1 \to -\frac{G_1 - G_2\,\hat{G}_1}{-1 + \hat{G}_1\,\hat{G}_2},\; X_2 \to -\frac{G_2 - G_1\,\hat{G}_2}{-1 + \hat{G}_1\,\hat{G}_2}\right\}$$

$$\left\{X_1 \to -\frac{G_1 - G_2\,\hat{G}_1}{-1 + \hat{G}_1\,\hat{G}_2},\; X_2 \to -\frac{G_2 - G_1\,\hat{G}_2}{-1 + \hat{G}_1\,\hat{G}_2}\right\}$$

The probability that the first pattern wins the game is clearly given by $x_{1,2} + x_{1,2} + x_{1,3} + \ldots = X_1(1)$. Unfortunately, substituting $s = 1$ into the right-hand sides of

$$\begin{aligned}X_1(s) &= \frac{G_1(s) - \hat{G}_1(s)\,G_2(s)}{1 - \hat{G}_1(s)\,\hat{G}_2(s)}, \\ X_2(s) &= \frac{G_2(s) - \hat{G}_2(s)\,G_1(s)}{1 - \hat{G}_1(s)\,\hat{G}_2(s)}\end{aligned} \tag{5}$$

results in $\frac{0}{0}$, so we use l'Hôpital's rule to find the answer,

$$X_1(1) \equiv P_1 = \frac{\hat{\mu}_1 + \mu_2 - \mu_1}{\hat{\mu}_1 + \hat{\mu}_2},$$

where μ denotes the mean corresponding to $G(s)$, that is, $\mu = G'(1)$.

Similarly

$$X_2(1) \equiv P_2 = \frac{\hat{\mu}_2 + \mu_1 - \mu_2}{\hat{\mu}_1 + \hat{\mu}_2}.$$

One can verify that $P_1 + P_2 = 1$.

When the two patterns are “incompatible” (no matching overlaps, for example, in a run of successes played against a run of failures), the two formulas reduce to

$$P_1 = \frac{\mu_2}{\mu_1 + \mu_2},$$
$$P_2 = \frac{\mu_1}{\mu_1 + \mu_2},$$

since all capped quantities are now equal to their uncapped counterparts (getting “help” from the other pattern is the same as starting from scratch).

Example: playing the SFFSS pattern against FFSSF.

We first get, by the technique of the previous section,

$$\mu_1 = \frac{1 + p^2\, q^2}{p^3\, q^2},$$
$$\hat{\mu}_1 = \mu_1{}^{(2)} = \frac{1 - p^3\, q}{p^3\, q^2},$$
$$\mu_2 = \frac{1 + p^2\, q^2}{p^2\, q^3},$$
$$\hat{\mu}_2 = \mu_2{}^{(4)} = \frac{1 + p\, q^2}{p\, q^3}.$$

The probability of SFFSS winning over FFSSF is thus

$$\frac{p\left(1 - p\, q^3\right)}{p^2 + q}.$$

When $p = \frac{1}{2}$, this yields a rather surprising value of 62.5%.

More easily, the same result can be found with the help of the following program.

```
game[A_, B_] := Module[
  {a = conv[A], b = conv[B], k1, k2, Ga, cGa, Gb, cGb, h},
  k1 = ovlp[b, a];
  k2 = ovlp[a, b];
  h = Gset[A]; Ga = G0 /. h; cGa = Gk1 /. h;
  h = Gset[B]; Gb = G0 /. h; cGb = Gk2 /. h;
        Ga - cGa Gb
  Limit[───────────, s → 1] // Simplify
        1 - cGa cGb
 ]
```

```
game["SFFSS", "FFSSF"] /.p → 0.5
```

```
0.625
```

Selecting the Optimal Pattern

The program `game` can also help us find the pattern that maximizes our chances of beating an opponent, assuming that we know the pattern that the opponent chose, and given that both patterns must be of the same length. The easiest way of doing this is to go over all possibilities and see which one results in the highest probability of winning.

In this manner, we find that when $p = \frac{1}{2}$, the best chance of beating SFFSS is to select FSFFS (that way, our probability of winning is 65.38%), and similarly to beat FFSSF we should select FFFSS (yielding a $\frac{2}{3}$ chance of winning). It appears that we should always drop the last symbol of the opponent's choice, and attach the opposite symbol in front. But this is only a conjecture which needs to be investigated more systematically (and is certainly not true for other values of p).

Similarly, when adding the condition that the two patterns must contain the same number of symbols of each type (for example, SFFFS and FSFFS), we may like to know which value of p gives the first pattern the highest chance of beating the second one (and vice versa). This can be done by the following two commands.

```
FindMaximum[{game["SFFFS", "FFSSF"], 0 < p < 1}, {p, .5}]
```

```
{0.548612, {p → 0.667978}}
```

```
FindMaximum[{game["FFSSF", "SFFFS"], 0 < p < 1}, {p, .5}]
```

```
{0.515401, {p → 0.131633}}
```

Have fun exploring some of the other possibilities!

□ Game’s Duration

Clearly

$$H\,(s) \equiv X_1\,(s) + X_2\,(s) = \frac{G_1(s) + G_2(s) - \hat{G}_1(s)\,G_2(s) - \hat{G}_2(s)\,G_1(s)}{1 - \hat{G}_1(s)\,\hat{G}_2(s)}$$

is the probability generating function of the number of trials required to finish the game. To find the corresponding expected value $H'\,(1)$, we need to differentiate

$$H(s)\left(1 - \hat{G}_1(s)\,\hat{G}_2(s)\right) = G_1(s) + G_2\,(s) - \hat{G}_1(s)\,G_2(s) - \hat{G}_2(s)\,G_1(s), \tag{6}$$

twice, substituting $s = 1$ at the end. This yields the following result.

```
D[H[s] (1 - Ĝ₁[s] Ĝ₂[s]) - G₁[s] - G₂[s] + Ĝ₁[s] G₂[s] + Ĝ₂[s] G₁[s],
    {s, 2}] /. {q_''[s] → q''[1], q_'[s] → μq, q_[s] → 1} //
  Simplify;
Solve[% == 0, μH][[1]]
```

$$\left\{\mu_{\mathrm{H}} \to \frac{\mu_{\mathrm{G}_2}\,\mu_{\hat{\mathrm{G}}_1} + \mu_{\mathrm{G}_1}\,\mu_{\hat{\mathrm{G}}_2} - \mu_{\hat{\mathrm{G}}_1}\,\mu_{\hat{\mathrm{G}}_2}}{\mu_{\hat{\mathrm{G}}_1} + \mu_{\hat{\mathrm{G}}_2}}\right\}$$

This implies that (divide both the numerator and denominator by $\mu_{\hat{G}_1}\,\mu_{\hat{G}_2} \equiv \hat{\mu}_1\,\hat{\mu}_2$):

$$H'\,(1) \equiv M = \frac{\frac{\mu_1}{\hat{\mu}_1} + \frac{\mu_2}{\hat{\mu}_2} - 1}{\frac{1}{\hat{\mu}_1} + \frac{1}{\hat{\mu}_2}}. \tag{7}$$

For the two patterns of the previous example, this equals

$$\frac{1}{p^2 + q}\left(1 + p^2\,q^2 + \frac{1}{p^2\,q^2}\right),$$

which evaluates to 22.75 when $p = \frac{1}{2}$.

The same result can be obtained by modifying the last line of `game`.

```
game2[A_, B_] := Module[
  {a = conv[A], b = conv[B], k1, k2, Ga, cGa, Gb, cGb, h},
  k1 = ovlp[b, a];
  k2 = ovlp[a, b];
  h = Gset[A]; Ga = G0 /. h; cGa = Gk1 /. h;
  h = Gset[B]; Gb = G0 /. h; cGb = Gk2 /. h;
       Ga + Gb - cGa Gb - Ga cGb
  h = ─────────────────────────── ;
              1 - cGa cGb
  Limit[D[h, s], s → 1] // Simplify
 ]
```

```
game2["SFFSS", "FFSSF"] /.p → 0.5
```

```
22.75
```

In the “incompatible” case (no possible overlap between the two patterns), (7) simplifies to

$$M = \frac{1}{\frac{1}{\mu_1} + \frac{1}{\mu_2}},$$

which is half of the harmonic mean of μ_1 and μ_2.

Similarly, by differentiating (6) three times, we can find $H''(1)$; this can be easily converted to the following variance of the number of trials:

$$\frac{\hat{\mu}_2}{\hat{\mu}_1 + \hat{\mu}_2}\left(\sigma_1{}^2 - P_2\,\hat{\sigma}_1{}^2\right) + \frac{\hat{\mu}_1}{\hat{\mu}_1 + \hat{\mu}_2}\left(\sigma_2{}^2 - P_1\,\hat{\sigma}_2{}^2\right) - \hat{\mu}_1\,\hat{\mu}_2\,P_1\,P_2, \tag{8}$$

where $\sigma_i{}^2$ and $\hat{\sigma}_i{}^2$ are the respective variances of the $G_i(s)$ and $\hat{G}_i(s)$ distributions ($i = 1, 2$).

Using the previous example with $p = \frac{1}{2}$, this variance is 336.1875 as $\mu_1 = \mu_2 = 34$, $\sigma_1{}^2 = \sigma_2{}^2 = 866$, $\hat{\mu}_1 = 30$, $\hat{\mu}_2 = 18$, $\hat{\sigma}_1{}^2 = 862$, $\hat{\sigma}_2{}^2 = 722$; the corresponding standard deviation is thus 18.34.

Again, the same computation can be achieved much more easily by modifying the last line of `game`. This then returns both the expected value and standard deviation of the game's duration.

```
game3[A_, B_] := Module[
   {a = conv[A], b = conv[B], k1, k2, Ga, cGa, Gb, cGb, h, m},
   k1 = ovlp[b, a];
   k2 = ovlp[a, b];
   h = Gset[A]; Ga = G0 /. h; cGa = Gk1 /. h;
   h = Gset[B]; Gb = G0 /. h; cGb = Gk2 /. h;
        Ga + Gb - cGa Gb - Ga cGb
   h = ------------------------- ;
              1 - cGa cGb
   m = Limit[D[h, s], s -> 1];
   {m, Sqrt[Limit[D[h, s, s], s -> 1] + m - m^2]} // Simplify
  ]
```

```
game3["SFFSS", "FFSSF"] /.p → 0.5
```

```
{22.75, 18.3354}
```

In the "incompatible" case, (8) reduces to a much simpler form,

$$P_1^2 \sigma_1^2 + P_2^2 \sigma_2^2 - M^2.$$

Monte Carlo Simulation

The correctness of all these formulas can be easily confirmed by the following program, whose four arguments are the two strings, the value of p, and the number of rounds of this game to be randomly generated.

```
MC[a_, b_, p_, n_] :=
  Module[{A = conv[a], B = conv[b], m1, m2, q, i = 0, j = 0,
    k, L = 0},
m1 = Length[A]; m2 = Length[B];
   Do[q = Table[-1, {Max[m1, m2]}]; k = 0;
While[Take[q, -m1] =!= A && Take[q, -m2] =!= B,
     q = Append[Drop[q, 1],
If[Random[] < p, 1, 0]]; k = k + 1];
    i = i + If[Take[q, -m1] == A, 1, 0];

j = j + k; L = L + k^2, {n}]; {i/n, j/n, Sqrt[L/n - j^2/n^2]} // N]
```

The program returns the resulting proportion of wins for the first string, the average length of the game, and the corresponding standard deviation.

This example is in good agreement with our theoretical results.

```
MC["SFFSS", "FFSSF", 1/2, 100 000]

{0.62212, 22.7245, 18.3041}
```

References

[1] W. Feller, *An Introduction to Probability Theory and Its Applications,* Vol. I, 3rd ed., New York: John Wiley & Sons, 1968.

[2] G. Blom and D. Thorburn, “How Many Random Digits Are Required Until Given Sequences Are Obtained?,” *Journal of Applied Probability*, **19**(3), 1982 pp. 518–531. www.jstor.org/pss/3213511.

[3] S. R. Li, “A Martingale Approach to the Study of Occurrence of Sequence Patterns in Repeated Experiments,” *Annals of Probability*, **8**(6), 1980 pp. 1171–1176. projecteuclid.org/DPubS?service=UI&version=1.0&verb=Display&handle=euclid.aop/1176994578.

[4] L. J. Guibas and A. M. Odlyzko, “String Overlaps, Pattern Matching, and Nontransitive Games,” *Journal of Combinatorial Theory A*, **30**(2), 1981 pp. 183–208. www.sciencedirect.com/science/article/pii/0097316581900054.

[5] E. Pegg Jr. “Coin Flips” from the Wolfram Demonstrations Project—A Wolfram Web Resource. www.demonstrations.wolfram.com/CoinFlips.

J. Vrbik, “Betting Two Patterns against Each Other,” *The Mathematica Journal*, 2011.
dx.doi.org/doi:10.3888/tmj.13–15.

About the Author

Jan Vrbik
Department of Mathematics, Brock University
500 Glenridge Ave., St. Catharines,
Ontario, Canada, L2S 3A1
jvrbik@brocku.ca