Chapter IV: Games of Pure Chance Generated by Gambler's Ruin with Unlimited Credit: The General case

We will now consider the general case where there is an **arbitrary** set of non-zero integers, and an **arbitrary** probability distribution on them, that we will call the **die** (or *spinner*), and at each round, the random walker walks (forward or backwards, as the case may be) according to the outcome of the die. He starts at 0, and the game ends as soon as he reaches a positive location, i.e. as soon as its location is ≥ 1. We will later treat the more general case where the goal is to reach a location that is $\geq m$, for any positive integer m.

The engine driving our algorithms is the powerful **Buchberger algorithm**, that finds *Gröbner bases*, and that is implemented in Maple and all the other major computer algebra systems.

It is convenient to separate the set of allowed steps into the set of positive steps, that we will call U, and the set if negative steps, $-D$, so D is a set of positive integers. For example if the set of allowed steps is $\{-2,-5,1,3,4\}$, then $U=\{1,3,4\}$ and $D=\{2,5\}$.

Let us now state precisely the input and the desired output for our algorithms.

Main Algorithm

Input:

Two sets of positive integers D and U, corresponding to allowed steps $-d$ (where $d \in D$) and u (where $u \in U$) in the 1D lattice, or equivalently, $(1,-d)$ $(d \in D)$ and $(1,u)$, $u \in U$, on the two-dimensional lattice, and an assignment of probabilities $\{p_d : d \in D\}$, $\{p_u : u \in U\}$, such that $\sum_{d \in D} p_d + \sum_{u \in U} p_u = 1$ with the meaning that the random walker walks d units backward if the die landed on $d \in D$ and moves u units forward if it landed on $u \in U$. The walker starts at location 0 and ends as soon as he reaches a strictly positive location. In addition, we input two **symbol**s (variables), t and f.

Output: A polynomial $P(f,t)$ of two variables, such that

$$P(f(t),t) \equiv 0 \quad ,$$

holds, where $f(t)$ is the probability generating function of the random variable: 'number of rounds until reaching a strictly positive location for the first time', obeying the above random walk.

Our algorithm guarantees that such a polynomial $P(f,t)$ **always** exists.

As in Chapter 3, we will first do the corresponding *enumerative combinatorics* version, and later use it to our probability purposes. We will use the powerful algorithm described in Bryan Ek's brilliant PhD thesis [Ek1], and also covered in [Ek2].

We will sometimes think of the 'gambling history' listing the outcomes, getting a **dynamic word** in the alphabet $U \cup (-D)$, i.e. a 1D path, but sometimes as a static entity, its graph where $d \in D$

corresponds to the down step $(1, -d)$ and $u \in U$ corresponds to an up-step $(1, u)$. So our problem is equivalent to counting such graphs whose atomic steps are as above, that start at the origin, and except for the **endpoint** that **must** be above the x-axis, is **weakly below** the x-axis.

As before for any word $w = w_1 \ldots w_n$, where w_i are integers, let $Weight(w) = \prod_{i=1}^{n} z_{w_i}$. For example $Weight(1, 2, -1, -3) = z_1\, z_2\, z_{-1}\, z_{-3}$. For any set of words S, its *weight-enumerator* is the sum of weights of all its members. If S is infinite (as is the case here) it is a **formal power series** in the set of variables

$\{z_{-d}\,;\, d \in D\} \cup \{z_u\,;\, u \in U\}$.

Bryan Ek's Algorithm for the Enumeration Problem

The algorithm described in [Ek1][Ek2] does the following.

Input: Finite sets of positive integers D and U. This gives rise to the **alphabet** $U \cup \{-d\,:\, d \in D\}$.

Output: A polynomial $P(f; \{z_u, z_{-d}\})$ of $1 + |U| + |D|$, variables such that

$$P(f(\{z_u, z_{-d}\})\,;\{z_u, z_{-d}\}) \equiv 0 \quad ,$$

holds, where $f(\{z_u, z_{-d}\})$ is the weight-enumerator of all words in the alphabet S whose sum is 0 and all whose partial sums are non-positive.

The algorithm guarantees that such a polynomial $P(f; \{z_u, z_{-d}\})$ always exists.

We use the same approach as in Chapter 3, but now we need the computer to 'do the thinking', and we humans do the 'meta-thinking', teaching it how to do the 'research'.

Let's abbreviate our desired weight-enumerator $f(\{z_u, z_{-d}\})$ by $W_{0,0}$, and let P_{00} be the actual set of paths weight-enumerated by it. In other words, the set of paths starting and ending on the x-axis, where each step is either $(1, u)$ $(u \in U)$ or $(1, -d)$, $(d \in D)$ and that lie **weakly** below the x-axis.

As we will soon see, we will be forced to introduce more general quantities. Let $W_{a,b}$ be the weight-enumerator of the set of paths $P_{a,b}$, that start at the horizontal line $y = -a$, end at the horizontal line $y = -b$, and always stay weakly-below the x-axis.

Setting up a system of Non-Linear Equations

The case $(a, b) = (0, 0)$

Let's look at an arbitrary member, w, of $P_{0,0}$. It may be the **empty path**, but otherwise, let w_1 be the **longest** prefix whose sum is 0, then we can write

$$w = w_1\, w_2 \quad ,$$

where $w_1 \in W_{0,0}$, and w_2 is also in $W_{0,0}$ but with the additional property that except for the endpoints, lies **strictly** below the x-axis. Let's call this subset $\overline{W_{0,0}}$. Obviously, the first step of w_2 must be a down step, $-d$, for some $d \in D$, and the last step must be an up-step, u, for some $u \in U$. For such a path (alias word), we can write (note that $\bar{d} = -d$)

$$w_2 = \bar{d}\, w_3\, u \quad ,$$

where w_3 is a path that starts at the horizontal line $y = -d$ and ends at the horizontal $y = -u$ but that is **strictly** below the $x - axis$. Such paths are 'isomorphic' to paths that start at $y = -(d-1)$ and end at $y = -(u-1)$ and stay weakly below the x-axis, in other words paths that belong to $W_{d-1,u-1}$.

So our desired quantity, $W_{0,0}$, satisfies the *one* non-linear equation

$$W_{0,0} = 1 + \sum_{d \in D} \sum_{u \in U} z_{-d}\, W_{d-1,u-1}\, z_u \quad .$$

Alas, now we have to handle all the '*uninvited guests*', the $W_{a,b}$ with $(a,b) \neq (0,0)$ that showed up.

We already handled the case $(a,b) = (0,0)$, we have to address three more cases.

The case $a > 0$ and $b > 0$

If such a path, w, is **strictly** below the x-axis then it is 'isomorphic' to a member of $P_{a-1,b-1}$. Otherwise, sooner or later, it would meet the x-axis for the **first time**. Let w_1 be the sub-path leading to that event.

We can write

$$w = w_1\, w_2 \quad ,$$

where w_1 is a path from $y = -a$ to $y = 0$ that, except for the last point, lies **strictly** below the x-axis, let's call that set $\overline{W_{-a,0}}$.

On the other hand w_2 is a member of $W_{0,b}$. Conversely, every two such paths $w_1 \in \overline{P_{a,0}}$ and $w_2 \in P_{0,b}$, when joined is a member of $W_{a,b}$ that touches the x-axis. Every path in $\overline{W_{-a,0}}$ must obviously end with $u \in U$, and the path obtained by removing the last step belongs to $\overline{W_{-a,-u}}$, that is 'isomorphic' to $W_{a-1,u-1}$. We thus have the equation

$$W_{a,b} = W_{a-1,b-1} + \left(\sum_{u \in U} W_{a-1,u-1}\, z_u \right) W_{0,b} \quad .$$

The case $a > 0$ and $b = 0$

The above discussion is also applicable to the case $b = 0$, except that the first term on the right, $W_{a-1,b-1}$, disappears . So we have

$$W_{a,0} = \left(\sum_{u \in U} W_{a-1,u-1}\, z_u \right) W_{0,0} \quad .$$

The case $a = 0$ and $b > 0$

We have $P_{0,b} = P_{00}\,\overline{P_{0,b}}$, so similarly

$$W_{0,b} \;=\; W_{0,0}\left(\sum_{d\in D} z_{-d}W_{d-1,b-1}\right) \quad .$$

Symbolic Dynamical Programming

Since our primary interest is , for now, $W_{0,0}$, the other quantities $W_{a,b}$ with $(a,b) \neq (0,0)$ are only auxiliary unknowns, that we are not interested in for their own sake, but that would hopefully enable us to find $W_{0,0}$.

We start out with the equation for $W_{0,0}$ that introduces $|D||U|$ new quantities,

$$\{W_{d-1,u-1} \,:\, d \in D\,,\, u \in U\} \quad .$$

For each new equation that we set-up, we may get brand new quantities, $W_{a,b}$, not yet encountered, but also some of which that already showed up before. For each new quantity, we set up a new equation. A priori, it is conceivable that we would have *infinite regress*, getting an infinite set of non-linear equations for an infinite set of unknowns. Luckily, this does not happen! Sooner or later there are no more new 'uninvited guests', and we are left with a **finite** set of non-linear (in fact quadratic) equations with the **same** number of unknowns, enabling us by **elimination** (using, in our case the Buchberger algorithm in Maple) to get **one** (usually, very complicated!) equation in the one unknown, $W_{0,0}$. We can do the same thing for any of the other $W_{a,b}$ and for that matter any linear combination of $W_{a,b}$, calling that linear combination Z, introducing one more equation and one more unknown, and **eliminating** Z, getting an algebraic equation satisfied by Z.

Straight Enumeration

Suppose that you are interested in the actual *enumerating sequence*, i.e. given a set of non-negative integers, S, you want to have an "explicit" expression for $a(n)$ the number of 1D walks, starting at 0 using the steps of S, ending at 0, and always staying weakly to the left of the origin. Equivalently, given a set of integers S, our 'alphabet', $a(n)$ is the number of words of length n whose sum is 0 and all whose partial sums are non-positive.

By **specializing** $z_u = t$, $(u \in U)$; $z_d = t$, $(d \in D)$ we have our next conceptual theorem.

Theorem 5: For an **arbitrary** set of integers S, let $a_S(n)$ be the number of sequences of length n, whose entries are drawn from S with the property that the sum is 0 and all its partial sums are non-positive. Then the ordinary generating function, in the variable t,

$$f(t) \;:=\; \sum_{n=0}^{\infty} a_S(n)t^n \quad ,$$

is an algebraic formal power series, i.e. there exists a two-variable polynomial P, with **integer coefficients**, such that

$$P(\, f(t)\, ,\, t\,) \,\equiv\, 0 \quad .$$

Furthermore, there exists an algorithm for finding this polynomial $P(f,t)$.

As a corollary, we know that $f(t)$ is $D-finite$, and hence $a(n)$ is P-recursive, and we have

Theorem 5': For an **arbitrary** set of integers S, let $a_S(n)$ be the number of sequences of length n, whose entries are drawn from S with the property the sum is 0, and that all the partial sums are non-positive. The sequence $a_S(n)$ is **P-recursive**, i.e. there exists a positive integer L and polynomials $p_i(n)$ in n, such that

$$\sum_{i=0}^{L} p_i(n) a_S(n-i) \,\equiv\, 0 \quad .$$

Furthermore, there exists an algorithm for finding this linear recurrence.

Rigorous Experimental Mathematics

Now that we have the **theoretical guarantee** that the polynomial $P(f(t),t)$ and the recurrence **exists**, it may be more efficient, to crank out, using (the usual, not symbolic) **dynamical programming**, sufficiently many terms and then fit them with a recurrence. The Maple commands `listtoalgeq(l,y(x))` and `listtorec(l,a(n))` do just that. These two useful commands are part of the versatile Maple package `gfun` written by Bruno Salvy and Paul Zimmerman [SZ].

The Weight-Enumerator of 1D walks that start at 0 end at strictly positive location but otherwise stay weakly to the left of 0

Equivalently, in the two-dimensional version, our walks start at the origin, end above the x-axis, and except for the end-point are weakly below the x-axis.

Since every such walk must obviously end with an up-step, $u \in U$, and the endpoint could be either at $y = 1, y = 2, \dots, y = u$, the desired weight-enumerator, let's call it Z, using the $W_{a,b}$ above is

$$Z \,=\, \sum_{u \in U} \left(\sum_{u'=0}^{u-1} W_{0,u'} \right) z_u \quad .$$

This is our quantity Z mentioned above, and thanks for the Buchberger algorithm, we can eliminate everything except Z, and get a **single** polynomial equation in Z.

From Enumeration to Probability

Having gotten a polynomial equation satisfied by the formal power series in the set of $|D| + |U|$ variables $\{z_u \,:\, u \in U\} \cup \{z_{-d} \,:\, d \in D\}$ that enumerates the above set of words, we **plug-in** $z_u \,=\, p_u\, t$, $z_{-d} \,=\, p_d\, t$, getting an equation of the form

$$P(f(t),t) \,\equiv\, 0 \quad ,$$

satisfied by the probability generating function, $f(t)$, for the random variable 'duration until reaching a positive amount for the first time' if you start at 0 dollars, at each time step (round) you win u dollars with probability p_u , if $u \in U$, and lose d dollars with probability p_d, if $d \in D$.

If the expected gain of a single step

$$\sum_{u \in U} p_u \, u \, - \, \sum_{d \in D} p_d \, d \quad ,$$

is positive, then sooner or latter the game ends.

So far $f(t)$ was a **formal power series**, but when you plug-in $t = 1$, you get a (numerical) **convergent** power series that must sum to 1, so $f(1) = 1$. It follows, that 1 is one of the roots of the **numerical** equation , in $f(1)$.

$$P(f(1), 1) \, = \, 0 \quad ,$$

This implies our next **conceptual** result, Theorem 6.

Theorem 6: Let $\mathcal{P}$ be any 'die' (with **any** number of faces, **any** loading, and **any** assignments of non-zero integers to its faces), where, at each step, you win or lose according to the outcome. Assume that the expected gain of a single round is positive. Let X be the random variable 'number of rounds until reaching a positive amount for the first time'.

The probability generating function of X,

$$f(t) \, = \, \sum_{k=0}^{\infty} Prob(X = k) \, t^k \quad ,$$

is an **algebraic** formal power series, in other words, there exists a two-variable polynomial, $P(f, t)$, such that

$$P(f(t), t) \equiv 0 \quad .$$

Furthermore, there is an algorithm for finding the two-variable polynomial $P(f, t)$.

As a corollary, we know that $f(t)$ is $D - finite$, and hence $Prob(X = k)$ is P-recursive, in k.

Theorem 6': For an **arbitrary** 'die' as in Theorem 6, and X defined there, the sequence $a(k) = Prob(X = k)$ is **P-recursive**, i.e. there exists a positive integer L and polynomials $p_i(k)$ in k, such that

$$\sum_{i=0}^{L} p_i(k) \, a(k - i) \, \equiv \, 0 \quad .$$

Furthermore, there exists an algorithm for finding this linear recurrence.

What about expectation?, using *implicit differentiation*, we get

$$P_f(f, t) \cdot f'(t) + P_t(f, t) \equiv 0 \quad .$$

Eliminating $f = f(t)$, from the two equations with three variables $\{f, f', t\}$ (f' is short for $f'(t)$)

$$P_f(f,t) \cdot f' + P_t(f,t) \, = \, 0 \quad , \quad P(f,t) \, = \, 0 \quad .$$

we (or rather our computers) get a polynomial equation of the form $Q(f'(t), t) \, = \, 0$, and plugging-in $t = 1$, ($f'(1)$ is a numerical convergent series), we get that the expected duration is one of the roots of the **numerical** equation

$$Q(f'(1), 1) \, = \quad .$$

By repeated implicit differentiation, and elimination of $(t\frac{d}{dt})^2 f(t)$, and then $(t\frac{d}{dt})^3 f(t)$, etc., we get algebraic equations for as many moments as desired, and hence for the variance, and higher moments about the mean. All this is implemented in procedure `Momk(N,P,fk,k)` in the Maple package `VGPileGames.txt`. Readers who wish to see more details are more than welcome to examine the Maple source code.

This brings us to the next 'conceptual' result.

Theorem 7: Consider **any** finite set of non-zero integers, and any probability distribution on them with positive expectation, where, at each step, you win or lose according to the outcome. Assume that the expected gain of a single round is positive. Let X be the random variable 'number of rounds until reaching a positive amount for the first time.' Then the expectation, variance, and any higher moments of X are **algebraic numbers**, whose minimal polynomial can be explicitly computed.

If the expected gain of a single round, $\sum_{u \in U} p_u \, u \, - \, \sum_{d \in D} p_d \, d$, is 0, then the expectation and higher moments are infinite. If it is negative, then the probability of exiting with a positive amount is less than 1, and $f(1)$ is one of the roots of the numerical equation $P(f(1), 1) = 0$, that is an explicit algebraic number. Then one has to talk about the 'conditional duration', and replace $f(t)$ by $f(t)/f(1)$, and Theorems 6 and 7 are still valid.

We will only present here one case. Quite a few similar propositions can be found in the web-page of this article, and readers are welcome to generate many more using the command

```
Paper(N,P,k,K1,K2,f,eps);
```

in the Maple package `VGPileGames.txt` also available from there.

Proposition 16: Consider a 1D random walk with a set of steps $\{-1,-2,1,2\}$ where $Pr(-1) = \frac{1}{4}$, $Pr(-2) = \frac{1}{8}$, $Pr(1) = \frac{1}{4}$, $Pr(2) = \frac{3}{8}$, that starts at 0. Let X be the random variable:

'number of steps until reaching a strictly positive location for the first time'.

The probability generating function of X

$$f(t) \, = \, \sum_{k=0}^{\infty} Prob(X = k) \, t^k \quad ,$$

is a formal power series that satisfies the algebraic equation

$$t^3 f^6 + \left(6\,t^3 - 8\,t^2\right) f^5 + \left(19\,t^3 - 48\,t^2\right) f^4 + \left(84\,t^3 - 80\,t^2 + 128\,t\right) f^3$$

$$+ \left(71\,t^3 - 608\,t^2 + 320\,t\right) f^2 + \left(262\,t^3 - 360\,t^2 + 768\,t - 512\right) f + 69\,t^3 - 432\,t^2 + 320\,t \;=\; 0 \quad .$$

The expectation, f_1, is one of the roots of the cubic equation

$$f_1{}^3 - 12\,f_1{}^2 + 16\,f_1 + 32 \;=\; 0 \quad ,$$

whose floating-point rendition is $2.9653919099833889\ldots$.

The second moment, f_2, is one of the roots of the cubic equation

$$101\,f_2{}^3 - 14140\,f_2{}^2 + 367216\,f_2 - 273824 \;=\; 0 \quad .$$

whose floating-point rendition is $33.31799734943726426\ldots$. It follows that the variance is $24.5244481696423327\ldots$, and hence the standard-deviation is $4.9522164905870515\ldots$.

The two-player version

Suppose two players take turns rolling the $\mathcal{P}$ die, and the one who is the first to reach a positive amount is declared the winner. Recall that the probability of the first player winning the game is $(1+s)/2$, where

$$s = \sum_{n=1}^{\infty} Prob(X=k)^2 \quad ,$$

it follows from Theorem 6' that s is a **holonomic constant**.

Playing until reaching at least m dollars for the first time

If the set of up-steps, U, consists of only one dollar, i.e. if $U = \{1\}$, then the probability generating function for the random variable "number of rounds it takes until reaching at least m dollars for the first time" is simply $f(t)^m$, and everything goes through, and the probability of the first player winning is holonomic in m. In the more general case, it is also true, but a bit more complicated, and we did not implemented it yet. The probability generating function for the 'first time of reaching $\geq m$' case can be shown to satisfy a linear recurrence equation in m whose coefficients are what we called $\{W_{a,b}\}$ above. After differentiating with respect to t, we can get recurrences for the expectation, and higher moments.

This brings us to the next theorem that we state without proof, and is not yet implemented in general.

Theorem 8: Consider **any** finite set of non-zero integers, and any probability distribution on them with positive expectation, where, at each step, you win or lose according to the outcome. Assume that the expected gain of a single round is positive. For any positive integer m, let X_m be the random variable 'number of rounds until reaching an amount that is $\geq m$ for the first time.' Then the probability generating function of X_m, let's call it $f_m(t)$ is an (constant) algebraic formal power series, and $f_m(t)$ satisfies a linear recurrence in m with coefficients that are algebraic formal power series.

Furthermore, $E[X_m]$ are **algebraic numbers**, that satisfy a linear recurrence equation with **constant** coefficients (but the constants featuring in the linear recurrence are, in general, *algebraic* numbers).

Keeping it Simple: Numerics Driven by Symbolics

For quite a few 'dice', our computers were able to find the **exact answer** for the question

What is the probability generating function for the random variable 'Number of rounds until reaching a positive amount for the first time .

Of course, except for the Catalan case, it is not **fully** explicit, but it is **as explicit as it gets**, the **exact** polynomial equation

$$P(f(t), t) \,=\, 0 \quad ,$$

satisfied by it, and that, in turn, enables us to find the **exact** values of the expectation, variance and higher moments, in terms of **explicit** algebraic numbers, i.e. numbers given by their minimal equation with integer coefficients.

But if the 'die', $\mathcal{P}$, gets larger, these algorithms are mainly of *theoretical* interest, i.e. for **conceptual computation**. To actually get **answers**, **very fast**, we recommend using the following simple-minded *symbolic-numeric* algorithm.

We can also do **simulation**, but these are very inexact. They are only useful (for our current project) as **sanity checks**, to make sure that we did not mess up.

Let $h(x)$ be the **probability generating function** of our die

$$h(x) \,=\, \sum_{u \in U} p_u \, x^u \,+\, \sum_{d \in D} p_d \, x^{-d} \quad .$$

For example, for the fair Catalan case $h(x) = \frac{1}{2}(x + x^{-1})$, for the Fuss-Catalan case considered in Chapter III, with 'one step forward, k steps backwards', $h(x) = px + (1-p)x^{-k}$, and for the more difficult case 'one step backwards, k steps forward' case, we have $h(x) = px^{-1} + (1-p)x^k$.

For any Laurent polynomial, define the operator: 'the positive part' as follows:

$$G[\sum_{i=c}^{d} a_i x^i)] \,=\, \sum_{i=1}^{d} a_i x^i \quad .$$

For example,

$$G[\frac{1}{10}x^{-3} + \frac{1}{20}x^{-2} + \frac{7}{20}x^{-} + \frac{1}{20} + \frac{1}{4}x + \frac{1}{4}x^2] = \frac{1}{4}x + \frac{1}{4}x^2 \quad .$$

The Symbolic-Numeric Algorithm to compute the first K terms of the probability generating function of our 'duration of the game' random variable

Input

- A die, $\mathcal{P}$, whose probability generating function is the Laurent polynomial $h(x)$.

- A positive integer K.

Output

The first K terms in the Maclaurin expansion of the probability generating function of the random variable: '*number of rounds until the player reaches a strictly positive amount for the first time*', let's call it $f_K(t)$.

Initialize: $F_0(x) := 1$, $f_0(t) = 0$.

For i from 1 to K do

$$A(x) := F_{i-1}(x)\, h(x) \quad ,$$

$$F_i(x) = A(x) - G[A(x)] \quad ,$$

$$f_i(t) = f_{i-1}(t) + G[A(x)]|_{x=1}\, t^i \quad .$$

Intuitively, $F_i(x)$ describes the scenarios that still did not make it to positivity by the i-th step. Multiplying by $h(x)$ is the 'roll of the die', $G[A(x)]$ describes the lucky scenarios that made it by the i-th round, and plugging in $x = 1$, gives the probability due to all the scenarios that made it exactly at the i-round for the first time.

If $h'(1) > 0$, then $f(1) = 1$, and to see how good $f_K(t)$ approximates $f(t)$, plug-in $t = 1$. If this is very close to 1 (usually it is!), then you are safe.

Also, *frankly*, you are **not** immortal, and even if you are, it is good to **a priori** set a limit to the number of allowed rounds, and compute everything conditioned on finishing in $\leq K$ rounds.

The conditional expectation on finishing in $\leq K$ rounds is $f'_K(1)/f_K(1)$, the second moment is $(t\frac{d}{dt})^2 f_K(t)|_{t=1}/f_K(1)$ and the k-th moment is $(t\frac{d}{dt})^k f_K(t)|_{t=1}/f_K(1)$.

These give much faster, very accurate, approximations to the desired statistical quantities of our random variable.

Similarly, we can compute, very fast, the truncated Taylor series of the random variable 'first time of having an amount $\leq m$', for any desired m.

This is accomplished by procedure `Ngf(N,P,t,K)` in the Maple package `VGPileGames.txt`.

If you want to see many examples, look at the file

`http://sites.math.rutgers.edu/~zeilberg/tokhniot/oVGPileGames2.txt` .

Another route to get the **exact** value of the expectation, variance, etc., is to derive numerical approximations like we did, and use Maple's command `identify` or use the **Inverse Symbolic Calculator**,

`https://isc.carma.newcastle.edu.au/` .

If you get an algebraic number, then it is most likely the right one, since we know, from the 'conceptual part', that it **is** an algebraic number.

Using this 'experimental way', we discovered the following **lovely** proposition.

Proposition 17: Consider a 'two steps forward one step backwards random walk' starting at 0, with $Prob(-1) = Prob(2) = \frac{1}{2}$. The expected number of rounds until reaching a location $\geq m$ for the **first** time, equals, **exactly**

$$2\,m \,+\, (4 \,-\, 2\,\phi) \,+\, 2\,(F_{m+2}\,\phi - F_{m+3}) \quad ,$$

where F_m are the Fibonacci numbers and $\phi = \frac{1+\sqrt{5}}{2}$ is the Golden Ratio.

Note that this makes sense, since the last term $F_{m+2}\,\phi - F_{m+3}$ is exponentially small in m, so this is very close to $2\,m \,+\, 4 \,-\, 2\phi$, and since the expected gain of one round is $\frac{1}{2}$ a crude approximation to the expected duration until owning $\geq m$ dollars should be roughly $m/\frac{1}{2} \,=\, 2m$. Also note that when $m = 1$ we get $2+4-2\phi+2(F_3\,\phi \,-\, F_4) = 6-2\phi+2(2\,\phi \,-\, 3) \,=\, 2\phi$, in agreement with the result established in Chapter III.

What about a proof of Proposition 17?

Proposition 17 was discovered *experimentally*, but we **do** know how to prove it. Writing it up, though, will take time and effort that we are unwilling to spend. We will be glad to furnish a proof in return to a \$2000 donation to the *On-Line-Encyclopedia of Integer Sequences*.

Conclusion

In this article, using *Games of pure chance* as a **case study**, we preached the value of **computational diversity**. Purely numeric, numeric-symbolic, purely symbolic, and 'conceptual', as well as the **simulation**, that in our case plays a secondary role, as a **checker**. It is so easy to have bugs in your programs, or gaps in your reasoning, so it is still reassuring that you can confirm the numbers that you got are in the right ball-park. We also demonstrated a novel application of the **Buchberger algorithm**.

References

[Ek1] Bryan Ek, "Unimodal Polynomials and Lattice Walk Enumeration with Experimental Math-

ematics", PhD thesis, Rutgers University, May 2018. Available from
`http://sites.math.rutgers.edu/~zeilberg/Theses/BryanEkThesis.pdf` .

[Ek2] Bryan Ek, `Lattice Walk Enumeration`, 29 March, 2018, `https://arxiv.org/abs/1803.10920`.

[SZ] Bruno Salvy and Paul Zimmerman, *GFUN: a Maple package for the manipulation of generating and holonomic functions in one variable*, ACM Transactions on Mathematical Software **20**(1994), 163-177 .