[VA] SRC #012a - Then and Now: Probability Threaded Mode | Linear Mode

5th October, 2022, 22:38 **Post: #1**

Valentin Albillo
Senior Member

Posts: 958
Joined: Feb 2015
Warning Level: 0%

[VA] SRC #012a - Then and Now: Probability

.
Hi, all,

After a 7-month hiatus here's my brand-new *multi-part* ***SRC #012 - Then an Now***, where I'll convincingly demonstrate that some advanced vintage *HP* calcs which were great problem-solvers back **THEN** in the 80's (some 40 years ago !), are **NOW** still perfectly capable of solving recently-proposed ***non-trivial*** problems intended to be tackled using modern *2020-era* personal computers, not ancient pocket calcs.

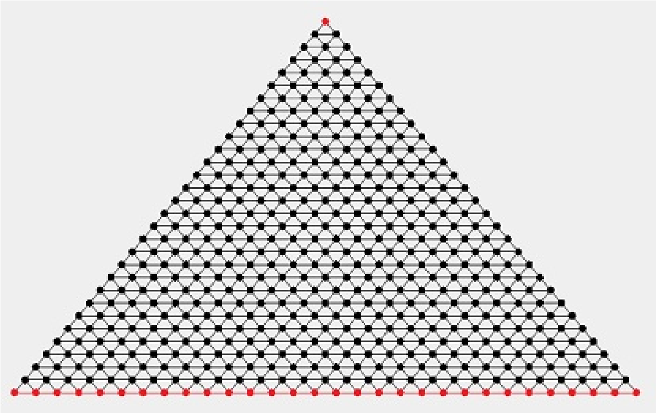
To that effect, in the following weeks I'll be proposing a number of such ***hard*** problems for you to try and solve using **EXCLUSIVELY VINTAGE HP CALCULATORS** (physical or virtual,) coding in either **RPN**, **RPL** or **HP-71B** language **AND NOTHING ELSE**: NO CAS/XCAS, MATHEMATICA, EXCEL, C/C++, PYTHON, etc. Besides, **you must post actual code**, not just **LENGTHY THEORY** SESSIONS/EXPOSITIONS (***A. C.***, *I'm looking at you !* 🙂).

Finally, **NO CODE PANELS** at all, just post your **RPN/RPL/71B** code *as-is* or formatted however your prefer. Please consider that I`m taking the trouble to use a lot of time and effort to carefully format and solve these problems for your entertainment and potential benefit so *please be fair to me and respect those simple rules*: only vintage *HP* calcs, only *RPN/RPL/71B* code, no math sessions/expositions, no *CODE* panels. **That's it !**

'Nuff said, let's begin with one of the easiest problems from the lot, namely:

Problem 1: Probability

A man starts at the top of an equilateral triangular grid having ***R*** rows of points and then takes random steps from point to point. Write a program to compute the probability ***P*** that after ***S*** such steps he ends up in the *bottom row*, and run it for the case ***R*** = *30 rows* and ***S*** = *60 steps, see the figure below*



Once you've found that probability you'll find it very easy to answer any number of additional questions, e.g. What's the probability he ends up in any edge ? In any corner ? In the first 7 rows ? What's the point which has the highest probability ? The lowest ? As a quick check, adding up the probabilities for all the points should return ***1*** ... after all, he must end up on *some* point or another ! 🙂.

You should strive for *10-12 correct digits* (give or take a few *ulps*) depending on whether you're using a 10- or 12-digit *HP* model, and of course the faster the running time, the better. If desired, you can check the correctness of your code by running the simpler ***R*** = *5 rows*, ***S*** = *4 steps* case, which should return a probability ***P*** = *23/288*.

If I see enough interest, in a few days I'll post my own original solution for the **HP-71B**, which is a short program capable of quickly solving the generic problem for *any* number of rows and steps. I'll also comment on accurate results and possible optimizations, as well as on some other related probabilities and statistics, and once everything is said and done I'll post the next *Problem 2*.

Let's see your very own clever solutions **AND** remember the above rules, please.

V.

PM WWW FIND EDIT QUOTE REPORT

6th October, 2022, 05:18 **Post: #2**

ttw
Member

Posts: 264
Joined: Jun 2014

RE: [VA] SRC #012a - Then and Now: Probability

I am assuming that the probabilities are equal for all branches from any point but the bottom row. (The probabilities fall into various classes depending on the location of the points.) I also assume that the bottom row is an absorbing barrier; the red lines on the border have probability zero (or are non-existent.)

The point being that it's not a copy of a Quincunx.

EMAIL PM FIND QUOTE REPORT

6th October, 2022, 11:58 **Post: #3**

ijabbott
Senior Member

Posts: 1,228
Joined: Jul 2015

RE: [VA] SRC #012a - Then and Now: Probability

> **ttw Wrote:** (6th October, 2022 05:18)
> I am assuming that the probabilities are equal for all branches from any point but the bottom row. (The probabilities fall into various classes depending on the location of the points.) I also assume that the bottom row is an absorbing barrier; the red lines on the border have probability zero (or are non-existent.)
>
> The point being that it's not a copy of a Quincunx.

I do not think there is any significance to the red lines, in which case it would be have been better to only colour the points. I would assume that all edges emanating from a point have an equal probability of being traversed on the next step from the that point.

EMAIL PM FIND QUOTE REPORT

6th October, 2022, 12:49 **Post: #4**

ijabbott
Senior Member

Posts: 1,228
Joined: Jul 2015

RE: [VA] SRC #012a - Then and Now: Probability

I wonder if it would be easier to work it out starting at the top and reaching the bottom, or vice versa?

EMAIL PM FIND QUOTE REPORT

6th October, 2022, 15:31 **Post: #5**

Valentin Albillo
Senior Member

Posts: 958
Joined: Feb 2015
Warning Level: 0%

RE: [VA] SRC #012a - Then and Now: Probability

.
Hi, **ttw**,

> **ttw Wrote:** (6th October, 2022 05:18)
> I am assuming that **the probabilities are equal for all branches from any point** [...]

Correct.

> **Quote:**
> [...] **but** the bottom row.

Nope. The bottom row is like any other row, nothing special about it. I could've asked the probability for the penultimate row or any arbitrary row instead, even the top "row" (the single starting point at the top) for that matter.

> **Quote:**
> I also assume that **the bottom row is an absorbing barrier**; the red lines on the border have probability zero (or are non-existent.)

Wrong assumption, the bottom row doesn't absorb anything and the red color is just cosmetic, to highlight the bottom row, it means nothing.

Thanks for your interest and let's see your ***RPN/RPL/71B*** code ! You can do it ! 🙂

Regards.
V.

PM WWW FIND EDIT QUOTE REPORT

8th October, 2022, 18:57 (This post was last modified: 8th October, 2022 20:24 by C.Ret.) **Post: #6**

05
PP

C.Ret
Member

Posts: 223
Joined: Dec 2013

RE: [VA] SRC #012a - Then and Now: Probability

Hi there,

Just three small questions that I ask everyone to confirm that I have understood the problem and not read it too quickly - I have this bad habit -.

```
            7
           / \
          1 — 2
         / \ / \
        6 — A — 3
       / \ / \ / \
      . — 5 — 4 — .
     / \ / \ / \ / \
    . — . — 8 — . — .
```

From point A, a step can only lead the man only to points 1 2 3 4 5 and 6. To go to points 7 and 8, you need at least two steps by passing through one intermediate points?

My program finds that on a triangle of R=5 rows, the probability that the man is on the last row after S=4 steps is P ≃ 0.07961.
Is this a correct value?

On this same triangle R=5, my program finds a probability P ≃ 0.1766 that the man is on the last line after S =7 steps. Does this seem possible to you?

EDIT:
I upgraded my code to avoid rounding errors as much as possible.
Now I find P = 1656 / 20736 for R=5 and S=4 and P = 6329160 / 35831808 for R=5 and S=7.

PM FIND QUOTE REPORT

9th October, 2022, 03:29 **Post: #7**

Valentin Albillo
Senior Member

Posts: 958
Joined: Feb 2015
Warning Level: 0%

RE: [VA] SRC #012a - Then and Now: Probability

.
Hi, **C.Ret**,

> **C.Ret Wrote:** (8th October, 2022 18:57)

My program finds that on a triangle of R=5 rows, the probability that the man is on the last row after S=4 steps is P ~ **0.07961**. Is this a correct value?

Well, in my OP I said:

> *"If desired, you can check the correctness of your code by running the simpler R = 5 rows, S = 4 steps case, which should return a probability P = **23/288**"*

As **23/288** evaluates to **0.079861...** then yes, your result is correct, except that you omitted the **8**, a simple typo.

> **Quote:**
> EDIT:
> I upgraded my code to avoid rounding errors as much as possible. Now I find **P = 1656 / 20736** for R=5 and S=4 [...]

But **1656/20736** immediately simplifies to **23/288**, which is the correct result, as stated in my OP. Why would you give your result as a fraction not reduced to its lowest terms ? 🙂

Thanks for your interest and regards.
V.

PM WWW FIND EDIT QUOTE REPORT

9th October, 2022, 06:57 (This post was last modified: 9th October, 2022 07:18 by C.Ret.) **Post: #8**

C.Ret
Member

Posts: 223
Joined: Dec 2013

RE: [VA] SRC #012a - Then and Now: Probability

> **Valentin Albillo Wrote:** (9th October, 2022 03:29)
> As **23/288** evaluates to **0.079861...** then yes, your result is correct, except that you omitted the **8**, a simple typo.
>
> > **Quote:**
> > EDIT:
> > I upgraded my code to avoid rounding errors as much as possible. Now I find **P = 1656 / 20736** for R=5 and S=4 [...]
>
> But **1656/20736** immediately simplifies to **23/288**, which is the correct result, as stated in my OP. Why would you give your result as a fraction not reduced to its lowest terms ?

Thanks Valentin.

I was very tired last night, which certainly explains the errors in copying, but also my lack of lucidity. I didn't even see that my fraction was reduced to 23/288.
My general condition explains this but does not excuse me.

Today, I wake up in great shape and I will go over my notes and resume my program. I will post it here and explain how it works. The major problem now is its efficiency because in the current version, on my poor HP-71B, it needs no less than 1'27" for (R,S)=(5, 4) and 2'32" for (R, S)=(5, 7).

And of all the authorized machines I own, this HP-71B is by far the fastest.

I expect it to take over 4 hours to calculate (R,S)=(50,60).

I recently got a MATH module, I hope to find the algorithmic way to take advantage of it. For the moment, the calculation is done using arrays and numerous nested loops. It looks very much like a matrix product.

I'm going to have a good breakfast and finish some work in the garden to prepare it for the arrival of winter while thinking about it.

I will post this evening the fruit of my developments, trying to do it before falling asleep and being unable to reread myself...

Best regards.
C.Ret

PM FIND QUOTE REPORT

9th October, 2022, 18:45 **Post: #9**

PeterP
Member

Posts: 172
Joined: Jul 2015

RE: [VA] SRC #012a - Then and Now: Probability

Valentin, while it feels out of my reach, it is nonetheless very intriguing as your puzzles always are. Please apologize my question as a burden on your time: we are looking for the probability that we are in the last row after S steps, correct? Not the probability that we can reach the last row in at most S steps?

For S = R - 1 as in the example that is identical, but for S > R one can reach the last row and then step away from it again.

PM FIND QUOTE REPORT

9th October, 2022, 23:00 **Post: #10**

PeterP
Member

Posts: 172
Joined: Jul 2015

RE: [VA] SRC #012a - Then and Now: Probability

I am afraid I am missing something in Valentin's description and I was hoping someone from the community can help me find my error.

I can only find 144 possible 4-step paths in the triangle with 5 rows. Rather than, as the answer/hint in the OP points out 288 (probability of success = good outcomes / all possible outcomes)

Each dot can reach each neighboring dot in 1 step. Each dot has either 2 (the corners of the pyramid), 4 (the middle part of the edges), or 6 (the inner part of the pyramid) neighbors.

I have now gone through the manual process of writing out all possible paths by naming the dots in the Pyramide from 1 to 15 (ie the starting point is dot 1, the second row is dots 2 and 3, the third row is dots 4, 5, and 6, and so forth)

That manual process yields the same outcome as my code = 144 possible 4-step paths, of which 16 end up in the last row. (How often you hit each dot in the last row seems to be the binomial triangle row equivalent to the number of steps, so 1-4-6-4-1 or a total of 16 times)

Clearly I am missing something fundamentally here. Maybe some kind soul can give me a pointer on what I am missing?

PM FIND QUOTE REPORT

10th October, 2022, 00:56 **Post: #11**

Valentin Albillo
Senior Member

Posts: 958
Joined: Feb 2015
Warning Level: 0%

RE: [VA] SRC #012a - Then and Now: Probability

.
Hi, **PeterP**, long time no see !

> **PeterP Wrote:** (9th October, 2022 18:45)
>
> Valentin, while **it feels out of my reach**, [...]

Not at all, you're being far too humble, I've seen you solve challenges ten times as difficult. This one's pretty easy, and the algorithm is the same for the huge 30-row grid as for a much smaller one.

> **Quote:**
>
> we are looking for the probability that we are in the last row after S steps, **correct?** Not the probability that we can reach the last row in at most S steps? For S = R - 1 as in the example that is identical, but for S > R one can reach the last row and then **step away** from it again.

Correct. You perform ***S*** random steps from the initial position, then check whether you're in the last row or not and that's it; the man can step away, step towards, or dance a *cha-cha-cha* for that matter.

Thanks for your interest, looking forward to your code and results, and best regards.
V.

PM WWW FIND EDIT QUOTE REPORT

10th October, 2022, 01:19 **Post: #12**

Valentin Albillo
Senior Member

Posts: 958
Joined: Feb 2015
Warning Level: 0%

RE: [VA] SRC #012a - Then and Now: Probability

> **C.Ret Wrote:** (9th October, 2022 06:57)
>
> > **Valentin Albillo Wrote:** (9th October, 2022 03:29)
> >
> > *As **23/288** evaluates to **0.079861...** then yes, your result is correct, except that you omitted the **8**, a simple typo. But **1656/20736** immediately simplifies to **23/288**, which is the correct result, as stated in my OP. Why would you give your result as a fraction not reduced to its lowest terms ?*
>
> Thanks Valentin.

My pleasure.

> **Quote:**
> Today, I wake up in great shape and I will go over my notes and resume my program. **I will post it here and explain how it works**.

That's great ! Please do !

> **Quote:**
> The major problem now is its efficiency because in the current version, on my poor HP-71B, [...] **I expect it to take over 4 hours to calculate (R,S)=(50,60)**.

I estimate that my own, non-optimized original solution would solve the *(30,60)* case *{not (**50**,60), yet another typo}* in less than *30 min*. when running on a physical **HP-71B**. Some pretty obvious optimization would make it run in half the time.

> **Quote:**
> **I will post this evening the fruit of my developments**, trying to do it before falling asleep and being unable to reread myself...

Ok, good luck ! :)

Thanks and best regards.
V.

PM WWW FIND EDIT QUOTE REPORT

10th October, 2022, 11:50 (This post was last modified: 10th October, 2022 11:51 by pier4r.) **Post: #13**

pier4r
Senior Member

Posts: 2,224
Joined: Nov 2014

RE: [VA] SRC #012a - Then and Now: Probability

Nice Problems! Do you come across those on your own, due to your work or tinkering, or do you see those (at least in part) in other places? They are really "tasty"!

One observation:

> **Valentin Albillo Wrote:** (5th October, 2022 22:38)
>
> Finally, **NO CODE PANELS** at all, just post your **RPN/RPL/71B** code *as-is* or formatted however your prefer. Please consider that I`m taking the trouble to use a lot of time and effort to carefully format and solve these problems for your entertainment and potential benefit so *please be fair to me and respect those simple rules*: only vintage *HP* calcs, only *RPN/RPL/71B* code, no math sessions/expositions, no *CODE* panels. **That's it !**

While I agree that given the effort you took it would be only fair to follow your requests, I was thinking that limiting explorations may take away some fun for some people (you mentioned a couple of users too). Further one thing leads to another if the discussion is lively and thus things can be interesting also with "less pure" discussions. Therefore - asking mostly the mods here - would it be possible if the community (not necessarily Valentin, as he already put a lot of effort in the #1 post) opens an extra thread to put there all the math discussion and the non-HP-calc code ?

In other words having two threads, one "pure" and the other for all the other discussions. This to compromise on limiting the explorations and the sharing.

10th October, 2022, 14:31 **Post: #14**

rprosperi
Super Moderator

Posts: 5,642
Joined: Dec 2013

RE: [VA] SRC #012a - Then and Now: Probability

> **pier4r Wrote:** (10th October, 2022 11:50)
>
> [snip]
> Therefore - asking mostly the mods here - would it be possible if the community (not necessarily Valentin, as he already put a lot of effort in the #1 post) opens an extra thread to put there all the math discussion and the non-HP-calc code ?
>
> In other words having two threads, one "pure" and the other for all the other discussions. This to compromise on limiting the explorations and the sharing.

Sure, go ahead, no harm in encouraging related discussions in a parallel thread, it lets folks participate in the problem without violating Valentin's requested 'rules', which I believe were put in place to allow easy capture into PDF documents, which remain problem-focused in the style he prefers, to preserve for subsequent publication, likely on his excellent site. As you say, it's only reasonable that folks respect the rules given the significant effort Valentin puts into creating, documenting and monitoring these great articles.

10th October, 2022, 23:43 **Post: #15**

Vincent Weber
Member

Posts: 288
Joined: May 2015

RE: [VA] SRC #012a - Then and Now: Probability

Hi Valentin,

Very interesting problem, many thanks!

I have a solution that is somewhat working (albeit not optimised at all), using a HP vintage calculator that is... The HP 27S, with its powerful formula-based language (using LET and GET functions for intermediate results). Before I post it, does it qualify? The calculator is vintage, but the formula language is not explicitely stated in your RPN/RPL/Basic list, so I'm in doubt...

Best regards,

Vincent

11th October, 2022, 00:26 **Post: #16**

Valentin Albillo
Senior Member

Posts: 958
Joined: Feb 2015
Warning Level: 0%

RE: [VA] SRC #012a - Then and Now: Probability

.
Hi, **Vincent**, glad to hear from you,

> **Vincent Weber Wrote:**
>
> **Very interesting problem**, many thanks!

You're welcome, thanks to you for your kind words.

> **Quote:**
>
> I have a solution that is somewhat working (albeit not optimised at all), using a HP vintage calculator that is... The HP 27S, with its powerful formula-based language (using LET and GET functions for intermediate results). Before I post it, **does it qualify?**

Of course it does. I didn't consider formula-based languages (do they have a name ?) because I didn't think they could tackle this kind of challenge but if you have a working solution, no matter how raw, I think it's pretty interesting so please *post it* and, if possible, explain in detail its workings.

Thanks and best regards.
V.

PM WWW FIND EDIT QUOTE REPORT

11th October, 2022, 01:33 **Post: #17**

Vincent Weber
Member

Posts: 288
Joined: May 2015

RE: [VA] SRC #012a - Then and Now: Probability

Many thanks, Valentin.

Here is my formula. Although I used Plus42, I made sure it does not use any of the Plus42 extensions, so that it can work on a vanilla 17B,27S or 19BII.

```
P=Σ(T:1:N:1:0×(L(J:1)+L(K:1)+Σ(I:1: S:1:L(Q:IF(J=1:2:IF(K=1 OR
K=J:4:6)))+L(Z:INT(RAN#×Q))+IF(Z=0:L(J:J+1):IF(Z=1:L(J:J+1)+L(K:K+1):IF(Z=2:IF(K=1:L(K:2):L(K:K-
1)):IF(Z=3:IF(K=1:L(J:J-1):IF(K=J:L(J:J-1)+L(K:K-1):L(K:K+1))):IF(Z=4:L(J:J-1)+L(K:K-1):L(J:J-
1))))))))+IF(J=R:1:0))÷N
```

This works by taking enough samples (N, 10.000 for instance) of scenarios defined by random numbers, like a Monte-Carlo simulation. I initially considered brute force, e. g. trying every single possible scenario, but if you ignore the edge cases you have something like 6 possible moves ^ S possibilities, which is astronomical if S=60, not feasible...

Now the weakness of my code is the random numbers generation. The 27S language only has RAN# (pseudo-random numbers between 0 and 1), has no SEED function, and I am well aware that using INT(RAND#*6) as a proxy for dice rolling is grossly wrong, biased towards lower numbers... I just don't see what else to do.

With S=4 and R=5, with N big enough (e.g. 1 million, which takes literally minutes, even on high performance Plus42 - I don't dare to imagine how much time it would take on a real machine), I get mediocre results that tend to be somewhat close to 23/288.

With S=30 and R=60 I get almost every time a probability of 0. Strange! It does not go very deep down the rows, numbers in the range of 10-20.

The algorithm is simple: J is the row number (from 1 to R), K is the position within the row (from 1 to J, as row J has J positions). If we are at the edges (K=1 or K=J), special treatment arises: only 4 possible moves (even only 2 at the top of the pyramid), so only 4 integer choices for the random numbers: 0 for down-left, 1 for down-right, 2 for horizontal-left (or right if we are at the left edge). The general case adds 3 more choices: 3 for horizontal-right, 4 for up-left, 5 for up-right. J and K are updated accordingly, until S moves are done. Then if J=R 1 is sumed, otherwise 0. In the end P is the number of successful scenarios, just divide by N to get the probability.

This is still an alpha version, I will try to improve it, especially on the random number generation...

Cheers,

Vincent

EMAIL PM FIND QUOTE REPORT

11th October, 2022, 11:56 **Post: #18**

Fernando del Rey
Junior Member

Posts: 19
Joined: Dec 2013

RE: [VA] SRC #012a - Then and Now: Probability

Thanks Valentin, as always, nice and interesting challenge!

I have written a still very raw little program for the plain vanilla 71B (about 25 lines of code), which is giving me the following result for the case of a 30-row grid and after 60 steps.

9.55109846817E-6

As the number of arithmetic operations (divisions and sums) is quite large, I fear that rounding errors may have crept up enough to significantly affect the result.

Using Emu71/Win in my PC, execution time is about 6 seconds for the 30-rowns/60-steps case. I have not yet tried it in the real 71, but I'm afraid execution time may be too long to be practical. I'll give it a try next.

I do have an idea how to improve the program for execution speed, but for the 30/60 case at hand the reduction in execution time would be about 25% only. Not a great deal.

Before posting my code or going further, please let me know if the result I am obtaining is not too far away from the correct figure.

Thanks again for your efforts to keep us all entertained!

EMAIL PM FIND QUOTE REPORT

11th October, 2022, 12:22 **Post: #19**

Fernando del Rey
Junior Member

Posts: 19
Joined: Dec 2013

RE: [VA] SRC #012a - Then and Now: Probability

Ups!

I was a bit too quick posting my result!

I found a silly error in my code. After correcting the error, the result I get for the 30/60 case is:

9.51234350207E-6

EMAIL PM FIND QUOTE REPORT

11th October, 2022, 13:00 **Post: #20**

Valentin Albillo
Senior Member

Posts: 958
Joined: Feb 2015
Warning Level: 0%

RE: [VA] SRC #012a - Then and Now: Probability

.
Hi, **Fernando**,

Can't reply at length right now but I'll answer your question, namely:

> **Fernando del Rey Wrote:** (11th October, 2022 12:22)
>
> I found a silly error in my code. After correcting the error, the result I get for the 30/60 case is:
> **9.51234350207E-6**

Fully correct to the 12 digits you provide. Now go on and eventually post your code and, if at all possible, some comments on its inner workings.

Thanks for your interest, kind words and above all, your results and forthcoming code !

Best regards.
V.

PM WWW FIND EDIT QUOTE REPORT

11th October, 2022, 19:56 (This post was last modified: 11th October, 2022 22:26 by Fernando del Rey.) **Post: #21**

Fernando del Rey
Junior Member

Posts: 19
Joined: Dec 2013

RE: [VA] SRC #012a - Then and Now: Probability

Here's the raw code to get quick results using Emu71/Win, with no attemps to optimize execution speed or to minimize memory usage. I have ideas for how to do both, but I wanted just to test the algorithm and get quick results.

The idea of the algorith is simple. Wherever the man starts, in our case at position (1,1) in the grid, he has a certain probability to move to the adjacent cells.

If the man is located at any if the 3 corner cells of the grid, he has a 1/2 probability to move to any of their two adjacent cells.

If he is located in a border cell, he has a 1/4 probability to move to each of the 4 adjacent cells.

If he is located at any of the remaining inner cells, he has a 1/6 probability to move to each of its 6 adjacent cells.

The program calculates a probability matrix after each step (matrix B) from the results of the probability matrix resulting from the previous step (matrix A).

After some trivial initilization (lines 10 to 50), the program iterates M times (M number of steps) in the big loop ranging from lines 60 to 240.

Line 70 copies matrix B to A and clears matrix B to start the calculation of a new probability matrix at each step.

Lines of code 80, 90 and 100 calculate the probability propagation of the corner cells.

Lines 120 and 130 calculate the probability propagation of the left border cells. Lines 140 and 150 calculate the propagation of the right border cells. Lines 160 and 170 calculate the propagation of the lower border cells.

Lines 190 to 230 calculate the probability propagation of the inner cells.

Finally, to obtain the probabilty of the man finishing at the lower line of the grid, we just need to add the probabilities of all cells in the lower line of the grid (line 250).

If we would like to know the probability of the man finishing at any other location of the grid after M steps, we just need to look at the final probability map in matrix B.

Also, if we want to know what happens if the man starts at an (X,Y) location different from the upper corner, we just need to change line 50 to B(X,Y)=1.

Please note the the program listing is a manual transcription from the code in Emu71/Win. I have tried to be careful to avoid any errors, but I cannot be 100% sure. I don't know how a to do a copy/paste of the program code in Emu71/Win. If anyone could guide me how to do it, I'd be most grateful.

In a physical 71B, this program will take about 108 minutes to resolve the 30/60 case. As mentioned earlier, it can surely be optimized for speed, but I was looking for simplicity and code clarity just to test the algorithm, not speed or memory usage optimization.

You may have noticed that I am a complete newbe with the 71B. I had to do a quick read of the manual to be able to produce this code as I have practically no experience with the 71B. But I must admit it has been real fun!

Here's the code:

```
10 DESTROY ALL @ OPTION BASE 1 @ STD
20 INPUT "Grid Size? ";N
30 INPUT "# Steps? ";M
40 REAL A(N,N), B(N,N), F
50 B(1,1)=1
60 FOR K=1 TO M
70 FOR I=1 TO N @ FOR J=1 TO I @ A(I,J)=B(I,J) @ B(I,J)=0 @ NEXT J @ NEXT I
80 F=A(1,1)/2 @ B(2,1)=F @ B(2,2)=F
90 F=A(N,1)/2 @ B(N-1,1)=F @ B(N,2)=F
100 F=A(N,N)/2 @ B(N-1,N-1)=F @ B(N,N-1)=F
110 FOR I=2 TO N-1
120 F=A(I,1)/4 @ B(I-1,1)=B(I-1,1)+F
130 B(I,2)=B(I,2)+F @ B(I+1,1)=B(I+1,1)+F @ B(I+1,2)=B(I+1,2)+F
140 F=A(I,I)/4 @ B(I-1,I-1)=B(I-1,I-1)+F
150 B(I,I-1)=B(I,I-1)+F @ B(I+1,I)=B(I+1,I)+F @ B(I+1,I+1)=B(I+1,I+1)+F
160 F=A(N,I)/4 @ B(N,I-1)=B(N,I-1)+F
170 B(N-1,I-1)=B(N-1,I-1)+F @ B(N-1,I)=B(N-1,I)+F @ B(N,I+1)=B(N,I+1)+F
180 NEXT I
190 FOR I=3 TO N-1 @ FOR J=2 TO I-1
200 F=A(I,J)/6 @ B(I-1,J-1)=B(I-1,J-1)+F @ B(I-1,J)=B(I-1,J)+F
210 B(I,J-1)=B(I,J-1)+F @ B(I,J+1)=B(I,J+1)+F
220 B(I+1,J)=B(I+1,J)+F @ B(I+1,J+1)=B(I+1,J+1)+F
230 NEXT J @ NEXT I
240 NEXT K
250 F=0 @ FOR I=1 TO N @ F=F+B(N,I) @ NEXT I @ DISP "Pr.=";F
```


11th October, 2022, 20:51 (This post was last modified: 13th October, 2022 09:55 by J-F Garnier.) **Post: #22**

J-F Garnier
Senior Member
Posts: 790
Joined: Dec 2013

RE: [VA] SRC #012a - Then and Now: Probability

I just finished to debug my code.
Valentin's hint to check that the sum of the probability of all cells should be 1 was very helpful to discover several coding errors :-)

The principle of my code is, I believe, similar to Fernando's solution, but the triangle cells are computed in a different order, row by row.

Also I took advantage of the symmetry of the starting condition.
Note that I used the fact that the HP BASIC doesn't enter a FOR loop if the final index is smaller than the starting index.
This is not true in all BASIC, if needed enable back lines 160 and 220.
Note also that I used some HP-71 Math ROM matrix statements (and for the fun, one statement of the Math 2B version :-)

My code is longer, partially because I avoided multi-statement lines for clarity.
The result is very slightly different: **9.51234350205E-6** [updated, was 9.51234350213E-6 in the previous version]

Here is my code [updated: lines 90, 140 and 240 replaced by lines 91, 141 and 241]:

```
 10 OPTION BASE 1
 20 N=30 ! rows
 30 S=60 ! steps
 40 DIM A(N,N),B(N,N)
 50 ! set up A
 60 MAT A=ZER @ MAT B=ZER
 70 A(1,1)=1
 80 FOR X=1 TO S
 90 ! B(1,1)=(A(2,1)+A(2,2))/4 ! row 1
 91   B(1,1)=A(2,1)/4+A(2,2)/4 ! row 1
100   B(2,1)=A(1,1)/2+A(2,2)/4+A(3,1)/4+A(3,2)/6
120   B(2,2)=B(2,1) ! row 2
130   B(3,1)=A(2,1)/4+A(3,2)/6+A(4,1)/4+A(4,2)/6
140 ! B(3,2)=(A(2,1)+A(2,2)+A(3,1)+A(3,3))/4+(A(4,2)+A(4,3))/6
141   B(3,2)=A(2,1)/4+A(2,2)/4+A(3,1)/4+A(3,3)/4+A(4,2)/6+A(4,3)/6
150   B(3,3)=B(3,1) ! row 3
160   ! IF N<6 THEN 310
170   ! rows 4..N-2
180   FOR I=4 TO N-2
190     B(I,1)=A(I-1,1)/4+A(I,2)/6+A(I+1,1)/4+A(I+1,2)/6
210     B(I,2)=A(I-1,1)/4+A(I-1,2)/6+A(I,1)/4+A(I,3)/6+A(I+1,2)/6+A(I+1,3)/6
220     ! IF I<5 THEN 270
230     FOR J=3 TO INT((I+1)/2)
240 !     B(I,J)=(A(I-1,J-1)+A(I-1,J)+A(I,J-1)+A(I,J+1)+A(I+1,J)+A(I+1,J+1))/6
241       B(I,J)=A(I-1,J-1)/6+A(I-1,J)/6+A(I,J-1)/6+A(I,J+1)/6+A(I+1,J)/6+A(I+1,J+1)/6
250       B(I,I+1-J)=B(I,J)
260     NEXT J
270     B(I,I-1)=B(I,2)
280     B(I,I)=B(I,1)
290   NEXT I
310   ! row N-1
320   B(N-1,1)=A(N-2,1)/4+A(N-1,2)/6+A(N,1)/2+A(N,2)/4
330   B(N-1,2)=A(N-2,1)/4+A(N-2,2)/6+A(N-1,1)/4+A(N-1,3)/6+A(N,2)/4+A(N,3)/4
340   FOR J=3 TO INT(N/2)
350     B(N-1,J)=A(N-2,J-1)/6+A(N-2,J)/6+A(N-1,J-1)/6+A(N-1,J+1)/6+A(N,J)/4+A(N,J+1)/4
360     B(N-1,N-J)=B(N-1,J)
370   NEXT J
380   B(N-1,N-1)=B(N-1,1)
390   B(N-1,N-2)=B(N-1,2)
400   ! row N
410   B(N,1)=A(N-1,1)/4+A(N,2)/4
420   B(N,2)=A(N,1)/2+A(N,3)/4+A(N-1,1)/4+A(N-1,2)/6
430   FOR J=3 TO INT((N+1)/2)
440     B(N,J)=A(N,J-1)/4+A(N,J+1)/4+A(N-1,J-1)/6+A(N-1,J)/6
450     B(N,N+1-J)=B(N,J)
460   NEXT J
470   B(N,N-1)=B(N,2)
480   B(N,N)=B(N,1)
490   !
500   MAT A=B
550 NEXT X
555 ! output result
560 DIM C(N)
570 MAT C=RSUM(A)
580 DISP "PROB.=";C(N)
```

```
>RUN
PROB.= 9.512343502105-6
```

Now, I would be curious to see RPN/RPL solutions :-)

J-F

EMAIL PM WWW FIND QUOTE REPORT

11th October, 2022, 21:07 (This post was last modified: 11th October, 2022 22:34 by rawi.) **Post: #23**

rawi
Member

Posts: 132
Joined: Nov 2019

RE: [VA] SRC #012a - Then and Now: Probability

Very nice and complicated problem. Thank you very much, Valentin.

Since I was not able to find the exact solution (like Fernando was, chapeau!) I did what statisticians do if brain is not enough: They replace brain by computer power and make simulations of the proplem. So I did with the DM42, which is not really vintage but vintage in programming. So I hope it is within the rules.
I did 250.000 simulations (no typo, it took about 9 hours with power supply attached) and got in total 2 cases where the man ended at the bottom line of the triangle with a triangle size of 30 and 60 movements. This is 8E-6 which is not bad compared to the exact solution of 9.5 E-6. Here is the code:

```
00 { 203-Byte Prgm }
01 LBL „VA“
02 „ROWS?“
03 PROMPT
04 STO 00
05 „MOVES?“
06 PROMPT
07 STO 01
08 „SIMUL?“
09 PROMPT
10 STO 04
11 „SEED?“
12 PROMPT
13 SEED
14 0
15 STO 05
16 STO 06
17 LBL 00
18 RCL 01
19 1000
20 /
21 STO11
22 1
23 STO 02
24 STO 03
25 STO+06
26 LBL 07
27 0
28 X<>F
29 RCL 03
30 1
31 X=Y?
32 SF 02
33 X=Y?
34 SF 03
35 X<>Y
36 RCL 02
37 X=Y?
38 SF 06
39 X=Y?
40 SF01
41 RCL 00
42 X=Y?
43 SF 04
44 X=Y?
45 SF 05
46 1
47 ENTER
48 ENTER
49 ENTER
50 FS? 01
51 +
52 FS? 02
```

```
+
FS? 03
+
FS? 04
+
FS? 05
+
FS? 06
+
+/-
7
+
1/X
RAN
X<>Y
/
IP
1
+
STO 08
0
STO 09
STO 10
LBL 08
1
STO+ 09
FS? IND 09
GTO 08
STO+ 10
RCL 10
RCL 08
X>Y?
GTO 08
1
XEQ IND 09
ISG 11
GTO 07
RCL 00
RCL 02
X=Y?
XEQ 09
RCL 06
RCL 04
X>Y?
GTO 00
RCL 05
RCL 04
/
RTN
LBL 09
1
STO+ 05
RTN
LBL 01
STO- 02
RTN
LBL 02
STO- 02
STO- 03
RTN
LBL 03
STO- 03
RTN
LBL 04
STO+ 02
RTN
LBL 05
STO+ 02
STO+ 03
RTN
LBL 06
STO+ 03
```

125 RTN
126 END

Some explanations:
Number: Register, L+Number: Line in program
ROWS = Size of the triangle, e.g. 30
MOVES = Steps through the triangle, e.g. 60
SIMUL = Number of Simulations, e.g. 10000
SEED = Starting value of random number generator
LBL 00: Start of simulation
01 Number of moves
02 Actual row position in the triangle
03 Actual column position in the triangle
06 Count of simulation
LBL 07: Start of random walk through triangle, starting at the top (row 1, column 1)
L28: Deleting flags; flags are used to denote
those directions that are blocked starting with FLAG 1 for the upper left direction and then FlAGS 2 to 6 in clockwise direction.
L46 – L64: Getting the number of directions the man can go (depending on position)
L66: Random number for simulated walk
08: Random walk; 1: first possible, starting with upper left direction, 2-6 clockwise
LBL 08: Getting the direction of the random walk
L77-80: Skipping walks that are not allowed because of actual position
09: Direction 1-6 with starting 1 at upper left and 2-6 in a clockwise direction
L90-93: Add 1 to Reg 05 if man ends at bottom
LBL 09: Counting number of cases when random walk ends at the button of the triangle
LBL 1 to LBL 6: Changing position in triangle.
Row from top to bottom, columns from right to left

12th October, 2022, 00:19 **Post: #24**

Vincent Weber
Member

Posts: 288
Joined: May 2015

RE: [VA] SRC #012a - Then and Now: Probability

Hi Valentin,

I renounced Monte-Carlo simulations and managed to get a direct solving which I find quite elegant. I managed to find correct results.

But I have a dilmena: I did so by using recursion, and the vanilla 17B/27S calculators don't have user defined functions, let alone recursive ones. I had to use Plus42, which extends the HP solver with such functions.

Can I still post the code, or is it too much cheating, which I would perfectly understand?

Best regards

12th October, 2022, 00:44 **Post: #25**

Valentin Albillo
Senior Member

Posts: 958
Joined: Feb 2015
Warning Level: 0%

RE: [VA] SRC #012a - Then and Now: Probability

.
Hi, **Vincent**,

> **Vincent Weber Wrote:** (12th October, 2022 00:19)
>
> Hi Valentin,
>
> I renounced Monte-Carlo simulations and managed to get a direct solving which I find quite elegant. I managed to find correct results.
>
> But I have a dilmena: I did so by using recursion, and the vanilla 17B/27S calculators don't have user defined functions, let alone recursive ones. I had to use Plus42, which extends the HP solver with such functions.
>
> **Can I still post the code**, or is it too much cheating, which I would perfectly understand?

Sorry, **Vincent**, but using ***Plus42*** with extensions not available in vintage *HP* calcs completely defeats the purpose of this ***SRC #012***, which is to demonstrate that vintage *HP* calcs (or at least *"very old"*) can still solve nowadays recently proposed non-trivial problems intended to be solved in modern PCs.

Using *Plus42* extensions or *Python* or *Matlab* or *C++* may be pretty satisfying and elegant but demonstrates nothing of the sort and has no place here.

I'd suggest you post your surely-very-interesting solution to a parallel thread, as suggested by ***pier4r*** (post #13 above) and *Super Moderator* ***rprosperi*** (post #14 above) for posts like yours that would not fit my stated rules.

Would you, please ?

Thanks a lot for asking and best regards.
V.

PM WWW FIND EDIT QUOTE REPORT

12th October, 2022, 00:48 **Post: #26**

Vincent Weber
Member

Posts: 288
Joined: May 2015

RE: [VA] SRC #012a - Then and Now: Probability

Hi Valentin,

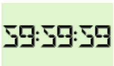
I fully understand and respect that, in fact I was expecting this answer

Before posting my code in another thread, let me try to remember how to convert a recursive algorithm into an iterative one

Best regards

EMAIL PM FIND QUOTE REPORT

12th October, 2022, 09:56 **Post: #27**



Werner
Senior Member

Posts: 767
Joined: Dec 2013

RE: [VA] SRC #012a - Then and Now: Probability

> **Vincent Weber Wrote:** (12th October, 2022 00:48)
>
> Hi Valentin,
>
> I fully understand and respect that, in fact I was expecting this answer
>
> Before posting my code in another thread, let me try to remember how to convert a recursive algorithm into an iterative one
>
> Best regards

I'd like to see the recursive one ;-)
Werner
I have a solution much along the lines of the '71 programs above, but for the 42S - but the memory requirements are too high for the 42S ;-) and Free42 is Not Allowed ;-)
(on the other hand, 32K 42S's exist..)

EMAIL PM FIND QUOTE REPORT

12th October, 2022, 14:10 **Post: #28**

PeterP
Member

Posts: 172
Joined: Jul 2015

RE: [VA] SRC #012a - Then and Now: Probability

Hi Valentin,

I was not able to find a solution that can work on a standard HP41 so I worked on a very limited edge case of steps equal to rows minus one, like the example you provided (btw - it was the provision of that example that allowed little ol' me to engage and learn, perfectly chosen, thank you!)

My code does deliver the correct result for R = 5, but I dont have a good way (especially right now on a plane and my work computer has no simulators installed...) to check if it is correct for R = 30, S=29. (It comes out to 1.311095094 e-13). And given this is such a specific edge case of your wonderful problem I assume posting code and explanation here would not be in your spirit anyway.

However I wanted to thank you for a couple hours of respite from powerpoints with wonderful recreational math and programming and thinking, its been a loon time since I had this pleasure.

And, as always, I learned a lot in the process, which is the most enjoyable part for me.

PM FIND QUOTE REPORT

13th October, 2022, 09:45 **Post: #29**

Werner
Senior Member

Posts: 767
Joined: Dec 2013

RE: [VA] SRC #012a - Then and Now: Probability

We're going about this the wrong way, I'm sure, and as usual, Valentin has given us a few clues.
The first being that he has a simple program that can calculate the desired answer for any number of rows and steps, quickly. So our (exponential) approach of summing up all point probabilities in each step is correct, but not feasible on our vintage calculators, for lack of speed or memory, or both.
The second being the test case for R=5 and S=4, with the number of steps just enough to reach the last row. Now, this probability is a lot easier to calculate as each successive row probability only depends on the previous row, and there's no need to keep the whole triangle.
And, the third that the sum of probabilities over the triangle necessarily has to be 1.
So, we have to ask ourselves: how does the last-row probability for R rows and R-1 steps change if we take an extra step?
The new last-row probability is the sum of
- the probability of staying in the last row, which is easy, as it is half of the total probability of being in the row at step R-1 and
- the probability of coming down from row R-1
and here I'm stuck, as that is not easy.. or I'm missing the obvious.
Coming down from the edge has a probability of 1/2 and coming down from any middle point has a probability of 1/3, but you'd still need to know all the point probabilities as well.

Hope it gave someone else an idea..
Cheers, Werner

EMAIL PM FIND QUOTE REPORT

13th October, 2022, 10:14 (This post was last modified: 13th October, 2022 11:13 by J-F Garnier.) **Post: #30**

J-F Garnier
Senior Member

Posts: 790
Joined: Dec 2013

RE: [VA] SRC #012a - Then and Now: Probability

More comments on Fernando's solution and mine.

The basic principle is the same iterative method, starting from the probability matrix at step N, it calculates the probability at step N+1.
Fernando scanned the *previous matrix* cells (step N-1) and distributes the probabilities to the *next step matrix* (N), with several statements like:
```
F=A(I,I)/6 @ B(I+1,I)=B(I+1,I)+F ...
```
On my side, I scanned the *next matrix* cells (step N) and computes the cell probability (in one go) from the *previous cells* (step N-1) with something like:
```
B(I,J)=(A(I-1,J-1)+A(I-1,J)+A(I,J-1)+A(I,J+1)+A(I+1,J)+A(I+1,J+1))/6
```

Both methods are equivalent, but Fernando's code is shorter.
However, I would have thought that my method was more accurate since most of the cell probabilities (the inner cells) are computed in one go, with only one division by 6.

But it's the contrary: Fernando's result is exact to 12 places, whereas my result had an "error" of 6 ULP.
To figure out why my answer was less accurate, I tried to mimic Fernando's calculation by not factoring the divisor term but distribute it to each cell term, that is replacing:
```
B(I,J)=(A(I-1,J-1)+A(I-1,J)+A(I,J-1)+A(I,J+1)+A(I+1,J)+A(I+1,J+1))/6
```
by
```
B(I,J)=A(I-1,J-1)/6+A(I-1,J)/6+A(I,J-1)/6+A(I,J+1)/6+A(I+1,J)/6+A(I+1,J+1)/6
```
In this way our methods sum exactly the same terms, but in a different order.

My result is now **9.51234350205E-6** with a difference of only 2 ULP (instead of 6) from the "exact" value.

I updated my code above accordingly.

Now, the question is: why is it better to distribute the divisions to each term, instead of factoring it?
Is it better just by chance?

J-F

EMAIL PM WWW FIND QUOTE REPORT

13th October, 2022, 10:28 **Post: #31**

Werner
Senior Member

Posts: 767
Joined: Dec 2013

RE: [VA] SRC #012a - Then and Now: Probability

Hi J-F,
Can't say why it seems better in your case to do the divisions straight away, but Fernando's code calculates the edges first, and those are the smallest numbers.
It's like summing a row of numbers: if you sum them from small to large, the result will be more accurate.
Cheers, Werner

EMAIL PM FIND QUOTE REPORT

13th October, 2022, 14:04 **Post: #32**

Fernando del Rey
Junior Member

Posts: 19
Joined: Dec 2013

RE: [VA] SRC #012a - Then and Now: Probability

I must admit it's pure chance that my method seems to be more accurate than J-F's. I started by summing the corners first and borders next as it seemed easier to program, but I was not considering that those cells would have smaller values and thus give a more accurate result when adding values as commented by Werner.

In my first code I had introduced a silly error in one of the corner cells (I forgot to divide it by 2 in the propagation). I was checking that the addition of all values in the probability matrix B was close to 1, and I noticed that it was a bit higher than 1, but initially I thought the difference was coming from the accumulation of rounding errors.

Later, I discovered the error in the code, and after correcting it I got a value of 1 accurate to 2 ULP, which gave me confidence that the code was now correct.
I have some ideas to improve the program for speed and memory usage. For speed, you can consider that if you are in iteration M all cells below row M+1 will be zero, and you can save the divisions and additions of zeros.

For memory usage, as the matrices A and B are not used to the right of their diagonal, you could work with a single matrix of size Nx(N+1) using the left side of the diagonal for holding results of iteration M and the right side of the diagonal to calculate results of iteration M+1. But I'm afraid it would make the code much less readable.

And you could also consider the symmetry of the solution if the man is starting at cell (1,1), calculating only half of the grid. But then the algorithm would not be valid for a starting position which is not located in the central column of the grid, which is therefore not symmetrical.

In the end, getting the result of the 30/60 case in less than two hours on a plain 71B (with no Math ROM), might still have been considered acceptable at the times of the 71B.

But I am convinced that Valentin will show us a much cleverer and faster method to arrive at the correct solution 🙂

EMAIL PM FIND QUOTE REPORT

13th October, 2022, 23:44 (This post was last modified: 13th October, 2022 23:54 by C.Ret.) **Post: #33**

C.Ret
Member

Posts: 223
Joined: Dec 2013

RE: [VA] SRC #012a - Then and Now: Probability

> **J-F Garnier Wrote:** (13th October, 2022 10:14)
> I updated my code above accordingly.

I get exactly the same value. But it's not surprising, even if our codes don't look alike, I actually do the calculation in the same direction as Jean-François.

It takes 5708.4 seconds (nearly 2 hours) to display the result using this compact code that is more readable by using line breaks and indents to show all nested loops:

```
10   DESTROY ALL @ OPTION BASE 1 @ DELAY 0
   @ INPUT "[VA]SRC012a R,S=";R,S @ T0=TIME
20   REAL T,W(R,R),P(R,R),Q(R,R) @ INTEGER A,B,I,J,K,M
30   DISP "Init"
   @ FOR I=1 TO R
```

```
    @     FOR  J=1  TO  I
    @        W(I,J)=2*(3-(J=1)-(I=J)-(I=R))
    @     NEXT  J
    @  NEXT  I
40  DISP  "Comp"
    @  Q(1,1)=1  @  M=2
    @  FOR  K=1  TO  S
    @     FOR  I=1  TO  M
50           FOR  J=1  TO  CEIL(I/2)
    @           P(I,J)=0
60              FOR  A=MAX(1,I-1)  TO  MIN(I+1,M)
    @              FOR  B=MAX(1,J-(A<=I))  TO  MIN(J+(I<=A),A)
70                   IF  A<>I  OR  J<>B  THEN  P(I,J)=P(I,J)+Q(A,B)/W(A,B)
80                 NEXT  B
    @           NEXT  A
    @           P(I,1+I-J)=P(I,J)
    @        NEXT  J
    @     NEXT  I
    @     IF  M<R  THEN  M=M+1
90        DISP  K;TIME-T0
    @     INVERSE  0,131*K*M/S/R
    @     MAT  Q=P
    @  NEXT  K
100  T=0
    @  FOR  J=1  TO  R  @  T=T+P(R,J)  @  NEXT  J  @  DISP  TIME-T0;R;STR$(S);T
```

To save time, I use the symmetry of the problem by calculating only the right half of each row of the equilateral triangle. In the I-th row, columns J and 1+I-J are symmetrical positions with equal probability P(I,J) = P(I,1+I-J).

Also, after M steps, the man goes further than the M+1 row. This leaves all the probabilities of the following rows at zero, which limits calculations and saves a little time.

My code uses two instructions from specific modules.
* The first is the INVERSE instruction of the JPC ROM module which show the progress of the calculation by inverting the display. Suffice to say that it does not speed things up and that we can easily do without it. It's just a gadget to make the user wait.
* The second is a MAT Q=P instruction which allows you to 'quickly' copy the newly calculated probabilities from table P into the table Q. I was hoping for a smarter use of my new module. but, for the moment, I have not found a way to perform the calculation more directly. By tha way,

Note the power of the HP-71B, while other systems do not have enough registers to memorize all the points of the triangle, the HP-71B allows, for simplicity, to use half of two matrices.
Moreover, I use a third W array to store the weights of each point. So much memory!

But I am not satisfied with this version. I'm like Werner, I think we're on the wrong track and there's a smarter way that avoids calculating all the iterations and probabilities of every point in the triangle.

So maybe my MATH module can be better exploited. Besides, I discover with horror that my 1A version does not have an RSUM command or any other means of obtaining the sum of the probabilities of the last row in one instruction.

Perhaps calculating iteratively, the probabilities of each point of the triangle will help us for the following questions?

PM FIND QUOTE REPORT

14th October, 2022, 01:27 **Post: #34**

Valentin Albillo
Senior Member

Posts: 958
Joined: Feb 2015
Warning Level: 0%

RE: [VA] SRC #012a - Then and Now: Probability

.
Hi, all,

A few comments before I post my original solution in a few days ...

> **Vincent Weber Wrote:**

> Before posting my code in another thread, let me try to remember how to convert a recursive algorithm into an iterative one

It will be quite a sight for sure, please do.

> **PeterP Wrote:**
> [...] like the example you provided (btw - it was the provision of that example that allowed little ol' me to engage and learn, perfectly chosen, thank you!)

I'm glad it was so helpful to you. I believe that including a 'toy' version of a problem helps immensely to detect and iron out bugs in one's program, as well as eficiency issues, e.g. if the toy case takes too long then the real McCoy will be hopeless so one has better improve the algorithm instead.

> **PeterP Wrote:**
> I dont have a good way (especially right now on a plane and my work computer has no simulators installed…) to check if it is correct for R = 30, S=29. (It comes out to **1.311095094 e-13**).

Looks pretty good for the 10-digit *41C*, I get **1.31109509664e-13** in the 12-digit *71B*.

> **PeterP Wrote:**
> And given this is such a specific edge case of your wonderful problem I assume posting code and explanation here would not be in your spirit anyway.

On the contrary, go ahead with the *41C* code, the more the merrier.

> **PeterP Wrote:**
> However I wanted to thank you for a couple hours of respite from powerpoints with wonderful recreational math and programming and thinking, its been a loon time since I had this pleasure. And, as always, I learned a lot in the process, which is the most enjoyable part for me.

Wow, what can I say ... many thanks and I'm extremely glad you enjoyed it and even learned while dealing with it, that's the idea !

> **Fernando del Rey Wrote:**
> But I am convinced that Valentin will show us a much cleverer and faster method to arrive at the correct solution.

You're such a good friend, **Fernando**, but let's not overhype my abilities lest disillusionment ensues, ok ?

As a general remark, I'm somewhat mystified that no *RPL* solutions have been posted or even discussed so far. I know that writing *RPL* code adds an enormous layer of sheer incomprehensibleness to the task but still ...

Best regards.
V.

PM WWW FIND EDIT QUOTE REPORT

14th October, 2022, 03:07 (This post was last modified: 14th October, 2022 05:22 by Albert Chan.) **Post: #35**

Albert Chan
Senior Member

Posts: 2,142
Joined: Jul 2018

RE: [VA] SRC #012a - Then and Now: Probability

> **J-F Garnier Wrote:** (13th October, 2022 10:14)
> `B(I,J)=A(I-1,J-1)/6+A(I-1,J)/6+A(I,J-1)/6+A(I,J+1)/6+A(I+1,J)/6+A(I+1,J+1)/6`

When we divide by 6, we almost always get an error of 1/3 ULP
It may be better to go for scaled probability (by 6^S), then unscale it.

```
10 OPTION BASE 1 ! JF scaled P version
20 N=30 ! rows
30 S=60 ! steps
40 T0=TIME @ DIM A(N,N),B(N,N)
50 ! set up A
60 MAT A=ZER @ MAT B=ZER
70 A(1,1)=1
```

```
80 FOR X=1 TO S
90 B(1,1)=(A(2,1)+A(2,2))*1.5 ! row 1
100 B(2,1)=A(3,2) + (A(2,2)+A(3,1))*1.5 + A(1,1)*3
120 B(2,2)=B(2,1) ! row 2
130 B(3,1)=(A(3,2)+A(4,2)) + (A(2,1)+A(4,1))*1.5
140 B(3,2)=(A(4,2)+A(4,3)) + (A(2,1)+A(2,2)+A(3,1)+A(3,3))*1.5
150 B(3,3)=B(3,1) ! row 3
160 ! IF N<6 THEN 310
170 ! rows 4..N-2
180 FOR I=4 TO N-2
190 B(I,1)=(A(I,2)+A(I+1,2)) + (A(I-1,1)+A(I+1,1))*1.5
210 B(I,2)=(A(I-1,2)+A(I,3)+A(I+1,2)+A(I+1,3)) + (A(I-1,1)+A(I,1))*1.5
220 ! IF I<5 THEN 270
230 FOR J=3 TO INT((I+1)/2)
240 B(I,J)=(A(I-1,J-1)+A(I-1,J)+A(I,J-1)+A(I,J+1)+A(I+1,J)+A(I+1,J+1))
250 B(I,I+1-J)=B(I,J)
260 NEXT J
270 B(I,I-1)=B(I,2)
280 B(I,I)=B(I,1)
290 NEXT I
310 ! row N-1
320 B(N-1,1)=A(N-1,2) + (A(N-2,1)+A(N,2))*1.5 + A(N,1)*3
330 B(N-1,2)=(A(N-1,3)+A(N-2,2)) + (A(N-2,1)+A(N-1,1)+A(N,2)+A(N,3))*1.5
340 FOR J=3 TO INT(N/2)
350 B(N-1,J)=(A(N-2,J-1)+A(N-2,J)+A(N-1,J-1)+A(N-1,J+1)) + (A(N,J)+A(N,J+1))*1.5
360 B(N-1,N-J)=B(N-1,J)
370 NEXT J
380 B(N-1,N-1)=B(N-1,1)
390 B(N-1,N-2)=B(N-1,2)
400 ! row N
410 B(N,1)=(A(N-1,1)+A(N,2))*1.5
420 B(N,2)=A(N-1,2) + (A(N,3)+A(N-1,1))*1.5 + A(N,1)*3
430 FOR J=3 TO INT((N+1)/2)
440 B(N,J)=(A(N-1,J-1)+A(N-1,J)) + (A(N,J-1)+A(N,J+1))*1.5
450 B(N,N+1-J)=B(N,J)
460 NEXT J
470 B(N,N-1)=B(N,2)
480 B(N,N)=B(N,1)
490 !
500 MAT A=B
550 NEXT X
555 ! output result
560 DIM C(N)
570 MAT C=RSUM(A)
580 DISP "PROB.="; C(N)/6^S, TIME-T0

>RUN
prob.= 9.51234350205E-6          69.54
```

Since most vertices can go 6 ways, scale by 6 also speed up calculations
Compare to original version, scaling speed up = 101.77/69.54 - 1 ≈ 46%

Trivia: if S goes infinite, eventually we reach an equilibrium

P(corners) : P(edges) : P(rest) = 1 : 2 : 3

3 + 3*(n-2) + (n-2)*(n-3)/2 = n*(n+1)/2

[3, 3*(n-2), (n-2)*(n-3)/2] * [1,2,3] = 3*n*(n-1)/2

P(bottom row, s=inf) = [2, n-2, 0] * [1,2,3] / (3*n*(n-1)/2) = 4/(3*n)

```
>N=5 @ S=100 @ RUN 40
prob.= .266666666668          2.68

>4/(3*N)
 .266666666667
```

14th October, 2022, 09:22 **Post: #36**

J-F Garnier
Senior Member

Posts: 790
Joined: Dec 2013

RE: [VA] SRC #012a - Then and Now: Probability

> **C.Ret Wrote:** (13th October, 2022 23:44)
>
> > **J-F Garnier Wrote:** (13th October, 2022 10:14)
> >
> > I updated my code above accordingly.
>
> I get exactly the same value. But it's not surprising, even if our codes don't look alike, I actually do the calculation in the same direction as Jean-François.
>
> It takes 5708.4 seconds (nearly 2 hours) to display the result using this compact code that is more readable by using line breaks and indents to show all nested loops:
> [...]

Your code is remarkably compact, at the expense of lower speed due to the two inner loops to explore the neighbouring points.
Thanks for formatting your code in a legible way, multi-statement line codes are usually very hard to read!

> **Quote:**
>
> Note the power of the HP-71B, while other systems do not have enough registers to memorize all the points of the triangle, the HP-71B allows, for simplicity, to use half of two matrices.
> Moreover, I use a third W array to store the weights of each point. So much memory!

Yes, the BASIC language is really much conformable to use for these kinds of problems.

> **Quote:**
>
> Besides, I discover with horror that my 1A version does not have an RSUM command or any other means of obtaining the sum of the probabilities of the last row in one instruction.

You may consider using the enhanced MATH 2B 🙂

J-F

EMAIL PM WWW FIND QUOTE REPORT

14th October, 2022, 09:35 **Post: #37**

Vincent Weber
Member

Posts: 288
Joined: May 2015

RE: [VA] SRC #012a - Then and Now: Probability

> **Valentin Albillo Wrote:** (14th October, 2022 01:27)
>
> > **Vincent Weber Wrote:**
> >
> > Before posting my code in another thread, let me try to remember how to convert a recursive algorithm into an iterative one
>
> [...]
> As a general remark, I'm somewhat mystified that no *RPL* solutions have been posted or even discussed so far. I know that writing *RPL* code adds an enormous layer of sheer incomprehensibleness to the task but still ... 🙂

Hi Valentin,

Unfortunately I have to renounce making an interative solution for the vanilla 17B/27S. I realize that this would require full indirect addressing of registers. By full I mean read and write. But the HP solver only allows reading values of a list indirectly (with ITEM[I]), not writing them (the L(ITEM[I] :X) construct, while straightforward, is illegal). Why HP didn't allow this, which would have brought their powerful and elegant solver language to Turing-compatibility level to solve any kind of problems, is beyond me...

Of course Plus42 allows this construct, but is out of the competition, so I renounce this.

But I might try to implement my recursive algorithm in RPL and see how it performs 🙂

Best regards

EMAIL PM FIND QUOTE REPORT

14th October, 2022, 12:24 (This post was last modified: 14th October, 2022 12:42 by J-F Garnier.) **Post: #38**

J-F Garnier
Senior Member

Posts: 790
Joined: Dec 2013

RE: [VA] SRC #012a - Then and Now: Probability

During my search for another (better) solution, I considered counting the different paths and attempting to deduce the probability from the path tree. For the case 5 rows/4 steps, I found the same numbers of paths as Peter, with 16 paths ending at the bottom row over a total of 144. And here I was puzzled because the probability 16/144=0.111111... is significantly different from Valentin's hint 23/288=0.07986... confirmed by the results of Fernando's program or mine.

So to remove any doubt, I attempted to use the Monte-Carlo statistical approach to confirm the probability in this particular case, since 5 rows/4 steps is still manageable in this way.
Here is my program:

```
 10 OPTION BASE 0
 20 DIM G(15,6)
 30 ! the nodes are coded from P=1 to 15:
 40 !        1
 50 !       2 3
 60 !      4 5 6
 70 !     7 8 9 10
 80 !   11 12 13 14 15
 90 ! matrix G(,) codes the graph:
100 ! G(P,0) codes the number of neighbours of node P
110 ! G(P,1..6) code the neighbouring nodes, up to 6
120 ! the zeros are just fillers.
130 DATA 0,0,0,0,0,0,0
140 DATA 2,2,3,0,0,0,0
150 DATA 4,1,3,4,5,0,0
160 DATA 4,1,2,5,6,0,0
170 DATA 4,2,5,7,8,0,0
180 DATA 6,2,3,4,6,8,9
190 DATA 4,3,5,9,10,0,0
200 DATA 4,4,8,11,12,0,0
210 DATA 6,4,5,7,9,12,13
220 DATA 6,5,6,8,10,13,14
230 DATA 4,6,9,14,15,0,0
240 DATA 2,7,12,0,0,0,0
250 DATA 4,7,8,11,13,0,0
260 DATA 4,8,9,12,14,0,0
270 DATA 4,9,10,13,15,0,0
280 DATA 2,10,14,0,0,0,0
290 READ G(,)
300 !
310 K=0
320 M=10000 ! number of trials
330 FOR I=1 TO M
340   P=1 ! start at node 1
350   FOR J=1 TO 4
360      N=G(P,0) ! number of neighbours
370      X=INT(RND*N+1) ! select one randomly 1..N
380      P=G(P,X) ! move to this node
390   NEXT J
400   IF P>=11 THEN K=K+1 ! count if at last row
410 NEXT I
430 DISP K/M
```

and the result is ... indeed around 0.08 and not 0.11, confirming the value 23/288 given by Valentin and the validity of the published programs above :-)

Now, I think I understood why 16/144 is NOT the probability of ending in the last row: there are indeed 16 paths over 144 that end in the last row, but all paths are not equally probable, we can't just do 16/144.
However, I thought that my little program was worth to be published here.

J-F

14th October, 2022, 20:16 (This post was last modified: 15th October, 2022 23:23 by Albert Chan.) **Post: #39**

Albert Chan
Senior Member

Posts: 2,142
Joined: Jul 2018

RE: [VA] SRC #012a - Then and Now: Probability

J-F Garnier Wrote: (14th October, 2022 12:24)

> ```
> 30 ! the nodes are coded from P=1 to 15:
> 40 !         1
> 50 !        2 3
> 60 !       4 5 6
> 70 !      7 8 9 10
> 80 !   11 12 13 14 15
> ```
>
> there are indeed 16 paths over 144 that end in the last row, but all paths are not equally probable,
> we can't just do 16/144.

Yes. all paths are not equally probable.
Above triangle, from 1 to bottom row in 4 steps, all steps must go down a row.

In 2 steps:
P(1 to 4) = 1/2*1/4 = 1/8
P(1 to 5) = 1/4

In 2 steps:
P(4 to bottom) = 1/4*1/2 + 1/4*1/3 = 5/24
P(5 to bottom) = 1/3*1/3 = 1/9

In 4 steps:
P(1 to 4 to bottom) = 1/8*5/24 = 5/192 ≈ 2.604% = P(1 to 6 to bottom)
P(1 to 5 to bottom) = 1/4*1/9 = 1/36 ≈ 2.778% , slightly more probable.

→ P(1 to bottom) = 2*5/192 + 1/36 = 23/288 ≈ 7.986%

Or, we can simply list all paths, then sum the probabilities.

```
Paths           1/Probability
1 2 4  7 11     2*4*4*4 = 128
1 2 4  7 12     2*4*4*4 = 128
1 2 4  8 12     2*4*4*6 = 192
1 2 4  8 13     2*4*4*6 = 192
1 2 5  8 12     2*4*6*6 = 288
1 2 5  8 13     2*4*6*6 = 288
1 2 5  9 13     2*4*6*6 = 288
1 2 5  9 14     2*4*6*6 = 288
1 3 5  8 12     2*4*6*6 = 288
1 3 5  8 13     2*4*6*6 = 288
1 3 5  9 13     2*4*6*6 = 288
1 3 5  9 14     2*4*6*6 = 288
1 3 6  9 13     2*4*4*6 = 192
1 3 6  9 14     2*4*4*6 = 192
1 3 6 10 14     2*4*4*4 = 128
1 3 6 10 15     2*4*4*4 = 128
```

P(1 to bottom) = 4/128 + 4/192 + 8/288 = 23/288

EMAIL PM FIND QUOTE REPORT

15th October, 2022, 20:28 (This post was last modified: 15th October, 2022 21:49 by Albert Chan.) **Post: #40**

Albert Chan
Senior Member

Posts: 2,142
Joined: Jul 2018

RE: [VA] SRC #012a - Then and Now: Probability

C.Ret Wrote: (13th October, 2022 23:44)

> IF A<>I OR J<>B THEN P(I,J) = P(I,J) + Q(A,B)/W(A,B)

1. Most vertices can go 6 ways, same Q(A,B)/W(A,B) calculated upto 6 times.

It is more efficient to calculate weighted Q first.
Bonus: MAT Q=P line is not needed anymore.

2. we can remove IF THEN statement, NOT(A<>I OR J<>B) ≡ (A==I AND B==J)

3. it is faster to use regular variable, sum it, then assign to P(I,J)

Combined 1,2,3: quoted line is simply T = T+Q(A,B), with T initially set to -Q(I,J)

4. it may be more accurate to go for scaled probability (by 6^S), then unscale it.

With above optimizations, we have:

```
10 DESTROY ALL @ OPTION BASE 1 @ INPUT "[VA]SRC012A R,S= ";R,S @ T0=TIME
20 REAL T,W(R,R),P(R,R),Q(R,R) @ INTEGER A,B,I,J,K,M
30 FOR I=1 TO R @ FOR J=1 TO I @ W(I,J)=3/(3-(J=1)-(I=J)-(I=R)) @ NEXT J @ NEXT I
40 MAT P=ZER @ P(1,1)=1 @ M=2
50 FOR K=1 TO S
60 FOR I=1 TO M @ FOR J=1 TO I @ Q(I,J)=P(I,J)*W(I,J) @ NEXT J @ NEXT I
80 FOR I=1 TO M
90 FOR J=1 TO CEIL(I/2) @ T=-Q(I,J)
100 FOR A=MAX(1,I-1) TO MIN(I+1,M)
110 FOR B=MAX(1,J-(A<=I)) TO MIN(J+(I<=A),A)
120 T=T+Q(A,B)
130 NEXT B
140 NEXT A
150 P(I,J)=T @ P(I,1+I-J)=T
160 NEXT J
170 NEXT I
180 IF M<R THEN M=M+1
190 DISP K;TIME-T0
200 NEXT K
210 T=0 @ FOR J=1 TO R @ T=T+P(R,J) @ NEXT J
220 DISP TIME-T0;R;S;T/6^S
```

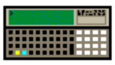
```
>RUN
[VA]SRC012A R,S= 5,4
1     .04
2     .08
3     .19
4     .36
.4         5 4       7.98611111111E-2
>T, 6^S              ! P = 23/288
103.5     1296

>RUN
[VA]SRC012A R,S= 30,60
...
173.47     30 60     9.51234350207E-6
```

Compare to original version, speed up = 245/173.47 - 1 ≈ 41%

EMAIL PM FIND QUOTE REPORT

16th October, 2022, 10:32 **Post: #41**



J-F Garnier
Senior Member

Posts: 790
Joined: Dec 2013

RE: [VA] SRC #012a - Then and Now: Probability

> **Albert Chan Wrote:** (15th October, 2022 20:28)
>
> With above optimizations, we have:
> [...]
> `20 REAL T,W(R,R),P(R,R),Q(R,R) @` **`INTEGER A,B,I,J,K,M`**

In Series 70 BASIC, there is no speed gain to use INTEGER variables; on the contrary it adds some overhead to convert INTEGER to REAL before evaluating the expression and convert the result back to INTEGER. This also applies to FOR .. NEXT loops and matrix indexes.

J-F

EMAIL PM WWW FIND QUOTE REPORT

16th October, 2022, 11:20 **Post: #42**

Albert Chan
Senior Member

Posts: 2,142
Joined: Jul 2018

RE: [VA] SRC #012a - Then and Now: Probability

Here is optimized C.Ret version, without "neighbor hunting" 2 inner loops.
I also remove INTEGER declaration, for some speedup (J-F Garnier, thanks for the tip!)

```
10 DESTROY ALL @ INPUT "[VA]SRC012A R,S= ";R,S @ T0=TIME
20 OPTION BASE 1 @ REAL W(R,R),P(R,R)
30 OPTION BASE 0 @ REAL Q(R+1,CEIL(R/2)+1)
40 FOR I=1 TO R @ FOR J=1 TO I @ W(I,J)=3/(3-(J=1)-(I=J)-(I=R)) @ NEXT J @ NEXT I
50 P(1,1)=1 @ M=2
60 FOR K=1 TO S
70 FOR I=1 TO M @ FOR J=1 TO CEIL(I/2)+1 @ Q(I,J)=P(I,J)*W(I,J) @ NEXT J @ NEXT I
80 FOR I=1 TO M
90 FOR J=1 TO CEIL(I/2)
100 P(I,J) = Q(I-1,J-1)+Q(I-1,J) + Q(I,J-1)+Q(I,J+1) + Q(I+1,J)+Q(I+1,J+1)
110 P(I,1+I-J)=P(I,J)
120 NEXT J
130 NEXT I
140 IF M<R THEN M=M+1
150 DISP K;TIME-T0
160 NEXT K
170 T=0 @ FOR J=1 TO R @ T=T+P(R,J) @ NEXT J
180 DISP TIME-T0;R;S;T/6^S
```

>RUN
[VA]SRC012A R,S= 30,60
...
67.71   30 60   9.51234350207E-6

From this thread so far, this is the most elegant code, and the fastest!

EMAIL PM FIND QUOTE REPORT

16th October, 2022, 12:25 **Post: #43**

Vincent Weber
Member

Posts: 288
Joined: May 2015

RE: [VA] SRC #012a - Then and Now: Probability

I'm very impressed by the HP-71B (which I discovered too late) elegance and capabilities.

And all this without even user-defined functions or recursion. The only advanced concept in use is 2-dimensional arrays, which makes me think that this could be ported to any SHARP pocket above the 1211, the HP-41 with advantage module (or even without, emulating matrices with single arrays), or the HP-42S.

Except that... Memory requirements are huge. The needed registers seem in the range of 2.5*R^2 at bare minimum. For R=30 you need something like 18Kb if each register takes 8 bytes, plus extra variables, plus program stack... So you need something like 20Kb RAM. This rules out the 41, the 42, the SHARP pc-1261... You need a 32K machine basically!

Best regards

EMAIL PM FIND QUOTE REPORT

16th October, 2022, 16:06 (This post was last modified: 16th October, 2022 16:26 by C.Ret.) **Post: #44**

C.Ret
Member

Posts: 223
Joined: Dec 2013

RE: [VA] SRC #012a - Then and Now: Probability HP-28S

> **Vincent Weber Wrote:** (16th October, 2022 12:25)
>
> Except that... Memory requirements are huge. The needed registers seem in the range of 2.5*R^2 at bare minimum. For R=30 you need something like 18Kb if each register takes 8 bytes, plus extra variables, plus program stack... So you need something like 20Kb RAM.

For the **HP-41**, I am currently thinking about a version where the among of registers would depend only on the number of steps.

It is based on the idea given by **Jean-François** and illustrated by **Chan** for the case (R,S)=(5,4).
However, exploring all the paths of S STEPS is likely to take an infinite amount of time...
Especially where S is greater than R !
On the other hand, this solution requires only S=60 well-used registers (sign, integer part and fractional cleverly exploited) and a few others for counters, coordinates, multiplicative factor and final probability...

So far, my buggy prototypes have only worked on reduced grids and for ridiculously small path's lengths. I am not very confident in the practical feasibility.
The real & practical solution would be to find a more direct and efficient calculation than listing all the paths in depth.

> **J-F Garnier Wrote:** (16th October, 2022 10:32)
>
> In Series 70 BASIC, there is no speed gain to use INTEGER variables; on the contrary it adds some overhead to convert INTEGER to REAL before evaluating the expression and convert the result back to INTEGER. This also applies to FOR .. NEXT loops and matrix indexes.

This was one of the questions I asked myself while writing my code. I didn't take the time to measure the gain or loss of time with all REAL variables.
I imagine that variables of the SHORT type do not offer any gain in speed either.

Thanks to **Jean-François** who answers and confirms my doubts without me having to time anything.

> **Albert Chan Wrote:** (15th October, 2022 20:28)
>
> Compare to original version, speed up = 245/173.47 - 1 ≈ 41%

Thanks to **Chan** who optimize my code. The time savings are far from trivial or insignificant. I like the new versions

By reading his post, I had already modified the code in my HP-71B in order to make the rounding errors disappear. But I hadn't thought of merging weight and probability in the matrix Q(,) = W(,) .* P(,). Too bad that the MATH module has no instruction making the dot-product. Is there a pretext for a third version of the MATH module?

I love the last version that smashes! thank you **Chan**!

This made me want to transcribe this algorithm for my **HP-28S**.

I get for the problem (R,S)=(30,60) exactly *P =9.51234350207E-6* after a very long time of 3 hours 12 min 48sec. It must be said that this version has the two internal loops **FOR a** and **FOR b**.

Now that **Chan** gave the idea, I will remake my RPN code to avoid the hunting for neighbors. The next version will therefore use matrices of dimension (R+1)x(R+1), the HP-28S having enough memory (32 kb).

I transcribed this version below. Will it inspired someone or will anyone, as CHAN already does, found some elegant and effective way to improve it?

The equilateral triangle of probabilities P is stored in the form of a one-dimension vector of length $d = \frac{r^2+r}{2}$. It is not memorized in a register but remains throughout the calculation in the stack.
Similarly, the weight coefficients, which are calculated once at the start of the computation, stay in the stack in the form of a vector of identical length.
On the other hand, the size of the vector Q is reduced (at least at the beginning) and depends on the progress variable *m* in order to save some effort.

When designing, I intended to calculate vector *Q* using the ***DPrdct*** instruction. But I realized that this instruction is not a native instruction of the HP-28S but is part of the personal programs that I usually use on this machine. So, I had to add the code of this program within the code. I took the opportunity to limit the size of Q and therefore reduce the calculations somewhat. But not the execution time, the HP-28S not being particularly quick to execute my codes.

Similarly, a ***Tind*** instruction is used to calculate the index of the triangular position (I,J) with J<=I. This is an instruction that I use elsewhere. It's very short, but I leave it in the code below which makes it 'a much more easy to read'.
As it's RPL, you have to understand 'a little less unreadable'.

SRC12a:
```
« OVER DUP Tind 2 → r s d m
    « 1 r FOR i
              1 i FOR j
                      3 DUP j 1 == - i j == - i r == - /
              NEXT
        NEXT
        { d } →ARRY
%% That's W
        DUP 0 CON 1 1 PUT                                                             %%
That's P = MAT ZER & P(1,1)=1
        1 s FOR k
```

```
                 k  1  DISP
                 DUP2  m  m  Tind  →  W  P  n                                        %%
Embedded  version  of  personal  Dot.Product  code
                      «  1  n  FOR  q
%%  size  limited  to  m  row  since  next  rows  all  zero
                                 W  q  GET  P  q  GET  *
                         NEXT
                         {  n  }  →ARRY  »
%%  That's  Q  =  W  .*  P
                 1  m  FOR  i
                         1  i  2  /  CEIL  FOR  j
                                 DUP  i  j  Tind  GET  NEG                           %%
Init  t  =-  Q(i,j)
                                 1  i  1  -  MAX  1  i  +  m  MIN  FOR  a
                                         1  j  a  i  <=  -  MAX  j  i  a  <=  +  a  MIN  FOR  b
                                                 OVER  a  b  Tind  GET  +            %%
Add   t  +=  Q(a,b)
                                         NEXT
                                 NEXT

ROT                                                              %%  stack  W  t  Q  P  order
                                 i  j                 Tind  3  PICK  PUT
                                 i  1  i  +  j  -  Tind       ROT  PUT

SWAP                                                             %%  back  initial  stack  order
                         NEXT
                 NEXT

DROP                                                                         %%  Drop  Q
                 m  DUP  r  <  +  'm'  STO                                           %%
Increase  m  (  STO+  doesn't  work  with  local  variables  )
         NEXT
         SWAP  0  CON
         r  1  Tind
%%  W  =  0  except  for  last  row  where  W  =  1
         DO
                 1  PUTI
         UNTIL  46  FS?  END                                                         %%
Flag  46  set  when  PUTI  reach  end  of  vector
         DOT  6  s  ^  /
         CLMF  1430  .4  BEEP  »                                                     %%
Restore  stack  display  and  bip
»
```

Tind:

```
%%  Personal  Triangular  Indice
«  OVER  SQ  ROT  -  2  /  +  »
%%     Tind(i,j)  =  j+(i²-i)/2
```

Feel free to comment or ask any questions that you deem useful.

Best regards.
C.Ret

PM FIND QUOTE REPORT

16th October, 2022, 16:10 (This post was last modified: 16th October, 2022 16:46 by J-F Garnier.) **Post: #45**

J-F Garnier
Senior Member

Posts: 790
Joined: Dec 2013

RE: [VA] SRC #012a - Then and Now: Probability

> **Albert Chan Wrote:** (16th October, 2022 11:20)
>
> Here is optimized C.Ret version, without "neighbor hunting" 2 inner loops.
>
> >RUN
> [VA]SRC012A R,S= 30,60
> ...
> 67.71   30 60   9.51234350207E-6

> From this thread so far, this is the most elegant code, and the fastest!

And perfectly exact too, down to the last place! Well maybe also by a little bit of chance.

Also thanks to the slightly optimized memory required for Q(R+1,CEIL(R/2)+1), it fits in a 24k-RAM HP-75 !

Here are the few simple changes to run your code on the HP-75:

```
7 OPTION BASE 0 @ REAL W(30,30),P(30,30),Q(31,16)
10 INPUT "[VA]SRC012A R,S= "; R,S @ T0=TIME
20 REDIM W(R,R),P(R,R)
30 REDIM Q(R+1,CEIL(R/2)+1)
35 MAT P=ZER @ MAT Q=ZER @ MAT W=ZER ! vars must be initialized
```

The rest is unchanged.

HP-75 result is: **9.51234350244E-6**

Slightly OT: why is the HP-75 result different, and less accurate (I like to investigate these kind of questions) ?
After all, it's all about additions, multiplications and divisions, and the HP-75 and HP-71 share the same algorithms and 12-digit (15 internally) accuracy.

Well, *almost* the same algorithms.
The Saturn machine algorithms (from the HP-71) introduce a very small improvement, known as the "round-to-even" or "banker's rounding" rule.
When an internal 15-digit result is rounded to the user 12-digit form, and it ends exactly with ...500, then it is rounded to the closest even value, instead of being rounded upward.
It can be demonstrated for instance with 1/(2^18) = 3.81469726562(500) e-6
HP-71: 3.81469726562e-6 (round to even)
HP-75: 3.81469726563e-6 (round up)

This difference occurs in the first steps of the calculation, and introduces a small bias that is enough to bring a different and less accurate answer at the end.

J-F

EMAIL PM WWW FIND QUOTE REPORT

16th October, 2022, 17:07 (This post was last modified: 16th October, 2022 17:24 by Albert Chan.) **Post: #46**

Albert Chan
Senior Member

Posts: 2,142
Joined: Jul 2018

RE: [VA] SRC #012a - Then and Now: Probability

> **J-F Garnier Wrote:** (16th October, 2022 16:10)
> HP-75 result is: **9.51234350244E-6**
> ...
> The Saturn (from the HP-71) introduces a very small improvement, known as the "round-to-even"

My code of scaling by P by 6^S take account of round-to-even rule.
For machine that round-away-from-zero, I would scale by gcd(2,4,6) = 12 instead.
Or, scale by 1.2 for decimal machine, to keep denominator (1.2^S) from overflow, even with big S

< 40 FOR I=1 TO R @ FOR J=1 TO C @ W(I,J)=3/(3-(J=1)-(I=J)-(I=R)) @ NEXT J @ NEXT I
> 40 FOR I=1 TO R @ FOR J=1 TO C @ W(I,J)=**0.6**/(3-(J=1)-(I=J)-(I=R)) @ NEXT J @ NEXT I

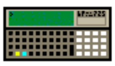
< 180 DISP TIME-T0;R;S;T/6^S
> 180 DISP TIME-T0;R;S;T/1.2^S

With above changes, on emu71, I get: (don't worry about error of few ULP's)

>RUN
[VA]SRC012A R,S= 30,60
...
70.6 30 60 9.51234350204E-6

EMAIL PM FIND QUOTE REPORT

16th October, 2022, 17:53 **Post: #47**



J-F Garnier
Senior Member

Posts: 790
Joined: Dec 2013

RE: [VA] SRC #012a - Then and Now: Probability

> **Albert Chan Wrote:** (16th October, 2022 17:07)
>
> > **J-F Garnier Wrote:** (16th October, 2022 16:10)
> >
> > HP-75 result is: **9.51234350244E-6**
>
> For machine that round-away-from-zero, I would scale by gcd(2,4,6) = 12 instead.
> Or, scale by 1.2 for decimal machine, to keep denominator (1.2^S) from overflow, even with big S
>
> < 40 FOR I=1 TO R @ FOR J=1 TO C @ W(I,J)=3/(3-(J=1)-(I=J)-(I=R)) @ NEXT J @ NEXT I
> > 40 FOR I=1 TO R @ FOR J=1 TO C @ W(I,J)=**0.6**/(3-(J=1)-(I=J)-(I=R)) @ NEXT J @ NEXT I
>
> < 180 DISP TIME-T0;R;S;T/6^S
> > 180 DISP TIME-T0;R;S;T/1.2^S
>
> With above changes, on emu71, I get: [..]
> 9.51234350204E-6

With these changes, I now get on the HP-75 (actually emu75):
9.51234350227E-6
which is indeed better, still not at the level of the best HP-71 performance but closer to my first HP-71 result (9.51234350213E-6).

J-F

EMAIL PM WWW FIND QUOTE REPORT

16th October, 2022, 19:28 (This post was last modified: 16th October, 2022 20:08 by C.Ret.) **Post: #48**

C.Ret
Member

Posts: 223
Joined: Dec 2013

RE: [VA] SRC #012a - Then and Now: Probability

The latest version for HP-71B proposed by **Chan** knocks the beast out.

The probability P(r=30 s=60) = 9.51234350207E-6 is displayed in just 31’27.4” with the last version.

As explained by **Chan**, since the same values P(i,j)/W(i,j) were used six times for the vast majority of points; the execution time is divided accordingly now that they are precomputed in the new matrix Q.

Moreover, as **Jean-françois** pointed out, a significant amount of memory is saved by using only the columns located in the right part of the triangle.

In this new version, the BASIC program is 402 bytes.
For (R,S)=(30,60), it uses 18855 bytes of data (18.4 kB) mainly for the three matrices W, P and Q.

I can't resist the pleasure of sharing this new version with you. Do not hesitate to dissect it:

```
10 DEF FNL(X)=1+CEIL(X/2)                                      @@ User function
for easy column limit indice computation
20 DESTROY ALL
 @ DELAY 0
 @ INPUT"[VA]SRC012a R,S=";R,S @ T0=TIME
30 OPTION BASE 1 @ DIM W(R,R),P(R,R)                           @@ MAT W and P
ranging ( 1..R    , 1..   R    )   2x7.04 ko
 @ OPTION BASE 0 @ DIM Q(1+R,FNL(R))                           @@ MAT      Q
ranging ( 0..R+1     , 1..1+R/2 )   4.12 ko
 @ P(1,1)=1 @ M=2
40 FOR I=1 TO R
 @    FOR J=1 TO I
 @       W(I,J)=3/(3-(J=1)-(I=J)-(I=R))                        @@ W(I,J) is
either 0 or 1 or 1.5 or 3
 @    NEXT J
 @ NEXT I
50 FOR K=1 TO S
 @    FOR I=1 TO M
 @       FOR J=1 TO FNL(I)
 @          Q(I,J)=P(I,j)*W(I,J)                               @@ no division,
spare time, preserve precision
 @       NEXT J
```

```
  @     NEXT  I
60     FOR  I=1  TO  M
  @        FOR  J=1  TO  CEIL(I/2)
70            P(I,J)=Q(I-1,J-1)+Q(I-1,J)+Q(I,J-1)+Q(I,J+1)+Q(1+I,J)+Q(1+I,1+J)     @@  no  neighbors
hunting,  all  I±1  or  J±1  in  Q's  subscripts  range
  @           P(I,1+I-J)=RES                                                     @@  RES  is  previous
computed  arithmetic  sum  stored  in  P(I,J)
80        NEXT  J
  @     NEXT  I
  @     M=M+(M<R)                                                                @@  slow  but  faster
than  the  IF  THEN  statment  which  need  an  extra  line!
  @     DISP  K;TIME-T0
  @  NEXT  K
90  T=0  @  FOR  J=1  TO  R  @  T=T+P(R,J)  @  NEXT  J
  @  DISP  TIME-J;R;S;T/6^S
  @  BEEP
```

PM FIND QUOTE REPORT

16th October, 2022, 19:29 **Post: #49**

Albert Chan
Senior Member

Posts: 2,142
Joined: Jul 2018

RE: [VA] SRC #012a - Then and Now: Probability

We can cut memory required more than half, by using symmetry.
Removed W, we reduced memory to 2 arrays, P and Q:

Let C = ceil(R/2)

P array elements =   R * C
Q array elements = (R+2)*(C+2)

Below code had P and Q with same dimensions, to take advantage of fast MAT Q=P (*)

Total memory (elements) = (R+2)*(C+2) * 2 < (R+3.5)^2

```
10 DESTROY ALL @ INPUT "[VA]SRC012A R,S= ";R,S @ T0=TIME
20 C=CEIL(R/2) @ OPTION BASE 0 @ REAL P(R+1,C+1),Q(R+1,C+1)
30 P(1,1)=1 @ M=2
40 FOR K=1 TO S
50 MAT Q=P @ Q(1,1)=Q(1,1)*3 ! BUILD Q
60 FOR I=2 TO M-(M=R) @ Q(I,1)=Q(I,1)*1.5 @ J=CEIL(I/2) @ Q(I,J+1)=Q(I,I-J) @ NEXT I
70 IF M<>R THEN 100
80 FOR J=2 TO C @ Q(R,J)=Q(R,J)*1.5 @ NEXT J
90 Q(R,C+1)=Q(R,R-C) @ Q(R,1)=Q(R,1)*3
100 FOR I=1 TO M @ FOR J=1 TO CEIL(I/2) ! BUILD P
110 P(I,J)=Q(I+1,J)+Q(I+1,J+1)+Q(I,J-1)+Q(I,J+1)+Q(I-1,J-1)+Q(I-1,J)
120 NEXT J @ NEXT I
130 IF M<R THEN M=M+1
140 DISP K;TIME-T0
150 NEXT K
160 T=0 @ FOR J=1 TO R-C @ T=T+P(R,J) @ NEXT J @ T=2*T+(2*C-R)*P(R,C)
170 DISP TIME-T0;R;S;T/6^S
```

>RUN
[VA]SRC012A R,S= 30,60
...
49.11   30 60   9.51234350203E-6

Code run even faster! 5× speed, against C.Ret orignial code (245 sec)

(*) I don't really need a copy, swapping name of array (P,Q) is enough.
Too bad ... VARSWAP don't work for arrays.

>VARSWAP P, Q
ERR:Data Type

EMAIL PM FIND QUOTE REPORT

16th October, 2022, 19:52 **Post: #50**

Valentin Albillo
Senior Member

Posts: 958
Joined: Feb 2015
Warning Level: 0%

RE: [VA] SRC #012a - Then and Now: Probability

Hi, all,

Thank you very much for your interest in my ***SRC #012a***, and in particular thanks to **Vincent Weber**, **Fernando del Rey**, **J-F Garnier**, **C.Ret**, **Werner**, **Albert Chan**, **PeterP**, **ijabbot**, **ttw**, **rprosperi**, **rawi**, and **pier4r** for your solutions and/or comments, much appreciated and *very* fine efforts throughout.

In my OP I mentioned *"takes **random** steps"* with the idea of misleading people into thinking that *Monte Carlo* simulations would be the way to go, but as some of you discovered upon attempting the feat, even a large numer of simulations would get just *one* correct digit for the toy case, let alone the *(30, 60)* case, as demonstrated by **J-F Garnier**'s awesome little piece of code.

Getting the 10-12 digits asked for required a fully *deterministic* method and there are at least two ways to proceed: *(a)* for each point, identify its neighbors and take the due part from them, or *(b)* for each location, identify its neighbors and distribute the point's probability among them. It's a zero-sum game and the total probability is the same at the beginning (**P=1** for the top location, *0* for all the rest), after every step and thus at the end.

I tried both approaches *(a)* and *(b)* above and both worked alright but approach *(b)*, distributing the probability among the neighbors, proved to be faster so that's what my original solution uses. Though tempting and perfectly adequate for the *(30, 60)* case as originally stated, *I don't use* ***symmetry*** for the reasons given in the fourth **Note** below, mainly because *it would lose generality*, which I wanted to preserve so that I could also solve *arbitrary* starting positions (non-symmetric, one or more as long as their initial peobabilities added up to exactly ***1***).

So, this is my original solution, a *13-line*, *683-byte* program. For convenience, it uses two matrix keywords from the *Math ROM*, namely `MAT..ZER` and `MAT=`, easily replaced by trivial loops if the *Math ROM*'s not available):

PROBLEM1 (683 bytes)

```
10 DESTROY ALL @ OPTION BASE 1 @ INPUT "Rows,Steps=";M,N @ DIM A(M,M),B(M,M) @ SETTIME 0
15 A(1,1)=1 @ W=M-1 @ FOR I=1 TO N @ MAT B=ZER
20 P=A(1,1)/2 @ IF P THEN B(2,1)=B(2,1)+P @ B(2,2)=B(2,2)+P
25 P=A(M,1)/2 @ IF P THEN B(W,1)=B(W,1)+P @ B(M,2)=B(M,2)+P
30 P=A(M,M)/2 @ IF P THEN B(W,W)=B(W,W)+P @ B(M,W)=B(M,W)+P
35 FOR X=2 TO W @ U=X-1 @ V=X+1 @ P=A(X,1)/4 @ Q=A(X,X)/4 @ R=A(M,X)/4
40 IF P THEN B(U,1)=B(U,1)+P @ B(V,1)=B(V,1)+P @ B(X,2)=B(X,2)+P @ B(V,2)=B(V,2)+P
45 IF Q THEN B(U,U)=B(U,U)+Q @ B(V,V)=B(V,V)+Q @ B(X,U)=B(X,U)+Q @ B(V,X)=B(V,X)+Q
50 IF R THEN B(M,U)=B(M,U)+R @ B(M,V)=B(M,V)+R @ B(W,U)=B(W,U)+R @ B(W,X)=B(W,X)+R
55 NEXT X @ FOR X=3 TO W @ U=X-1 @ V=X+1 @ FOR Y=2 TO U @ R=Y-1 @ S=Y+1 @ P=A(X,Y)/6
60 IF P THEN B(U,R)=B(U,R)+P @ B(U,Y)=B(U,Y)+P @ B(X,R)=B(X,R)+P
65 IF P THEN B(X,S)=B(X,S)+P @ B(V,Y)=B(V,Y)+P @ B(V,S)=B(V,S)+P
70 NEXT Y @ NEXT X @ MAT A=B @ NEXT I @ P=0 @ FOR Y=1 TO M @ P=P+A(M,Y) @ NEXT Y @ DISP P;TIME
```

```
>STD @ RUN
     Rows,Steps= 5,4 [ENDLINE] -> 7.98611111112E-2  (=23/288)  .03"
```

After running the program, we can display the just-built *probability matrix* (which contains the probability that the man is in each of the *5*(5+1)/2 = 15 points* in the grid after he's walked the 4 steps in this toy case, which is small enough that we can get it in *exact rational form* (requires the *JPC ROM*) right from the command line:

```
>FOR I=1 TO 5 @ FOR J=1 TO I @ FRAC$(A(I,J));" "; @ NEXT J @@ NEXT I

                                  11/96
                            49/384     49/384
                   113/1152     101/576     113/1152
             35/1152       17/288     17/288       35/1152
      1/128        23/1152      7/288      23/1152       1/128
```

Now for the big *(30, 60)* case:

```
>RUN
     Rows,Steps= 30,60 [ENDLINE] -> 9.51234350207E-6   27.46"
```

We can now use the *probability matrix* right from the command line to answer all kinds of questions, e.g.:

(a) check the matrix correctness by adding up the probabilities for all points:

```
>S=0 @ FOR I=1 TO M @ FOR J=1 TO I @ S=S+A(I,J) @ NEXT J @ NEXT I @ S

      1.00000000002
```

which gives **1** as it should, after all the man must be in one of the 465 points.

(b) probability that he ends in the left border:

```
>L=0 @ FOR I=1 TO M @ L=L+A(I,1) @ NEXT I @ L

      .117372130231
```

which is *12338.9289091* times more probable than ending in the bottom row.

(c) probability that he ends in <u>any</u> border as compared to ending inside the grid:

```
>D=0 @ FOR I=1 TO M @ D=D+A(M,I) @ NEXT I @ D

      9.51234350207E-6 (down edge = bottom row)

>R=L @ L+R+D

      .234753772806
```

so he ends in a border *23.48%* of the time and thus *76.52%* of the time he ends inside the grid, so he's *3.26 times* more likely to end inside the grid than in a border.

(d) probability that he ends in each of the corners or in any of them:

```
>A(1,1);A(M,1);A(M,M) @ A(1,1)+A(M,1)+A(M,M)

      8.30321536116E-3   2.21000524869E-8   2.21000524869E-8 (symmetric)

      8.30325956126E-3
```

(e) probability that he ends up in any of the first N rows:

```
>FOR N=1 TO M @ P=0 @ FOR I=1 TO N @ FOR J=1 TO I @ P=P+A(I,J) @ NEXT J @ NEXT I @ N;P @ NEXT N

      N      P
     ---------------------------------------------
      1   0.00830      top corner
      2   0.04100      top two rows
            ...
      5   0.25643      top 5 rows
            ...
      7   0.44905      top 7 rows, less than 50%
      8   0.54414      top 8 rows, more than 50%
            ...
     10   0.71182      top 10 rows, 71%
     15   0.94283      top 15 rows, 94%
     20   0.99430      top 20 rows, 99%
     25   0.99972      top 25 rows, almost sure
     29   0.99999      top 29 rows, 99.999% certain
     30   1.00000      all 30 rows, 100% certain.
```

Notes:

- All runtimes are for a virtual **HP-71B** *(go71b)* running on a mid-range *Samsung Galaxy Tab A 6* tablet *(Android)*. A physical *HP-71B* should take *~58 min* for the *(30, 60)* case.

- The *(30, 60)* case needs *14,533 bytes of RAM* to run. If the matrices are dimensioned as `SHORT` then it requires only *8,245* bytes of RAM but it runs *slower* and gives only ***5*** correct digits save 2 *ulp* (`P=0.0000095121`), instead of ***12*** correct digits.

- The program checks whether a location's probability is currently *zero*, in which case it skips it when distributing the probability among its neighbors. This saves about *20%* running time for the *(30,60)* case.

- Another pretty obvious optimization would be to take advantage of the *symmetry*, as the man's starting location is at the top corner; this would cut running time and required RAM roughly in half but ***it loses generality***, i.e.: arbitrary non-symmetrically placed starting positions (one or more) would not run at all, and it would complicate the coding somewhat; also, for a *one-time* program which already runs suitably fast it's really a moot point.

- It's a real pity that the *HP-71B BASIC* dialect doesn't include the **+=** , etc., operators, as it would make my program that much shorter and faster, i.e. line

```
50 IF R THEN B(M,U)=B(M,U)+R @ B(M,V)=B(M,V)+R @ B(W,U)=B(W,U)+R @ B(W,X)=B(W,X)+R
```

  would become

```
50 IF R THEN B(M,U)+=R @ B(M,V)+=R @ B(W,U)+=R @ B(W,X)+=R
```

  and so on. Perhaps **J-F Garnier** could do something about it.

- The *exact value* is *(465 points updated per step * 60 steps = ~ 200,000 arithmetic operations in all)*:

  $$\frac{37222007778846266183855309067888660226890969631735228955 29}{391302183008102676318141068027642364938466415279908207464546304}$$

  = **9.51234350207**433209735313473753833163507673100717 37**e-6** *(50 digits)*

Frankly, I thought that some accomplished *RPL* programmers would use the multiprecision capabilities of their advanced *RPL* models to compute the exact rational solution above but no such luck ...

Thanks to all of you for your interest, solutions and comments. A number of you produced correct solutions for this *Problem 1* but it's the *easiest* of the lot and I wonder how many of you will achieve a perfect ***6 for 6*** score by the time all six parts of ***SRC #012*** are over ...

See you in *Problem 2* featuring soon.
V.

[VA] SRC #012a - Then and Now: Probability

17th October, 2022, 02:13 **Post: #51**

Albert Chan
Senior Member

Posts: 2,142
Joined: Jul 2018

RE: [VA] SRC #012a - Then and Now: Probability

> **Valentin Albillo Wrote:** (16th October, 2022 19:52)
>
> Another pretty obvious optimization would be to take advantage of the symmetry, as the man's starting location is at the top corner; this would cut running time and required RAM roughly in half but it loses generality, i.e.: arbitrary non-symmetrically placed starting positions (one or more) would not run at all, and it would complicate the coding somewhat; also, for a one-time program which already runs suitably fast it's really a moot point.

Here is my code, removed symmetry for generality.

For other starting positions, edit LINE 30, with M = first empty row.
Tips: we can *still* use symmetry, by rotation for the smallest starting M

```
10 DESTROY ALL @ INPUT "[VA]SRC012A R,S= ";R,S @ SETTIME 0
20 OPTION BASE 0 @ T=R+1 @ REAL P(T,T),Q(T,T)
30 P(1,1)=1 @ M=2
40 FOR K=1 TO S
50 MAT Q=P @ Q(1,1)=Q(1,1)*3 ! BUILD Q
60 FOR I=2 TO M-(M=R) @ Q(I,1)=Q(I,1)*1.5 @ Q(I,I)=Q(I,I)*1.5 @ NEXT I
70 IF M<>R THEN 100
80 FOR J=2 TO R-1 @ Q(R,J)=Q(R,J)*1.5 @ NEXT J
90 Q(R,1)=Q(R,1)*3 @ Q(R,R)=Q(R,R)*3
100 FOR I=1 TO M @ FOR J=1 TO I ! BUILD P
110 P(I,J)=Q(I+1,J)+Q(I+1,J+1)+Q(I,J-1)+Q(I,J+1)+Q(I-1,J-1)+Q(I-1,J)
120 NEXT J @ NEXT I
130 M=M+(M<R)
140 DISP K;TIME
150 NEXT K
160 T=0 @ FOR J=1 TO R @ T=T+P(R,J) @ NEXT J
170 DISP TIME;R;S;T/6^S
```

```
>RUN
[VA]SRC012A R,S= 30,60
...
87.6     30 60     9.51234350205E-6
```

As expected, without symmetry, speed almost cut in half.

EMAIL PM FIND QUOTE REPORT

17th October, 2022, 03:15 (This post was last modified: 17th October, 2022 03:18 by Xorand.) **Post: #52**

Xorand
Member

Posts: 76
Joined: Feb 2015

RE: [VA] SRC #012a - Then and Now: Probability

> **Vincent Weber Wrote:** (16th October, 2022 12:25)
>
> The only advanced concept in use is 2-dimensional arrays, which makes me think that this could be ported to any SHARP pocket above the 1211...
>
> Except that... Memory requirements are huge. The needed registers seem in the range of 2.5*R^2 at bare minimum.

> For R=30 you need something like 18Kb if each register takes 8 bytes, plus extra variables, plus program stack... So you need something like 20Kb RAM. This rules out the 41, the 42, the SHARP pc-1261... You need a 32K machine basically!

I ported Fernando del Rey's BASIC program to my Sharp PC-1500 with 8K memory module (10k total storage). It ran the size 5, step 6 case easily. I ran a 10/20 case (took just a couple minutes - didn't time it). When I try the suggested size 30, 60 steps case, the computer unsurprisingly gives an out of memory error. Trial and error shows that I could go up to a grid size of 23.

EMAIL PM FIND QUOTE REPORT

17th October, 2022, 09:42 **Post: #53**

Vincent Weber
Member

Posts: 288
Joined: May 2015

RE: [VA] SRC #012a - Then and Now: Probability

> **Xorand Wrote:** (17th October, 2022 03:15)
>
> > **Vincent Weber Wrote:** (16th October, 2022 12:25)
> > The only advanced concept in use is 2-dimensional arrays, which makes me think that this could be ported to any SHARP pocket above the 1211...
> >
> > Except that... Memory requirements are huge. The needed registers seem in the range of 2.5*R^2 at bare minimum. For R=30 you need something like 18Kb if each register takes 8 bytes, plus extra variables, plus program stack... So you need something like 20Kb RAM. This rules out the 41, the 42, the SHARP pc-1261... You need a 32K machine basically!
>
> I ported Fernando del Rey's BASIC program to my Sharp PC-1500 with 8K memory module (10k total storage). It ran the size 5, step 6 case easily. I ran a 10/20 case (took just a couple minutes - didn't time it). When I try the suggested size 30, 60 steps case, the computer unsurprisingly gives an out of memory error. Trial and error shows that I could go up to a grid size of 23.

Excellent, thanks !

EMAIL PM FIND QUOTE REPORT

17th October, 2022, 15:38 (This post was last modified: 17th October, 2022 15:38 by Dave Britten.) **Post: #54**

Dave Britten
Senior Member

Posts: 2,222
Joined: Dec 2013

RE: [VA] SRC #012a - Then and Now: Probability

> **Valentin Albillo Wrote:** (16th October, 2022 19:52)
> So, this is my original solution, a *13-line*, *683-byte* program. For convenience, it uses two matrix keywords from the *Math ROM*, namely `MAT..ZER` and `MAT=`, easily replaced by trivial loops if the *Math ROM*'s not available):[font=Courier]

Darn, I was hoping your mention of "which is a **short** program capable of **quickly** solving the generic problem" meant you had a clever multinomial approach that could solve the problem in mere seconds on an HP 67. 😊

Still, it's a nice elegant solution, even if matrix iteration is the only feasible way to do it. I have a direct adaptation of your program running on my Sharp PC-1403H now. We'll see how long it takes for the large case - it's been going for about 40 minutes...

PM WWW FIND QUOTE REPORT

17th October, 2022, 17:45 **Post: #55**

PeterP
Member

Posts: 172
Joined: Jul 2015

RE: [VA] SRC #012a - Then and Now: Probability

Here is my code for the HP41, which does not have enough memory for the matrix approach. It only works for S = N-1 and uses that one only has to worry about the paths that increase R in every step, as all other paths do not reach the last row in time. It then uses that the number of ways of reaching the last row is the binomial tree (from each node there are two ways to reach the next row, with the exception of the edges).
As such, its rather trivial.

Inspired by Albert and others I was then trying to figure out if there is a way to scale the result for S=N-1 to S=N, S=N+1, etc given that it trends asymptotically to a fixed value. However, I was not able to do so and now it has closed, so apologies for posting my code only now.

The code uses that only steps that increase the row can work. It also uses the symmetry between left and right.

When walking down the left edge, starting at some point in row y, the first step then always has a probability of 1/4 to go towards the middle. Afterwards all probabilities to go downwards are 2 * 1/6 (one to the left and one to the right) or 1/3. This goes on for the remaining rows x to the last row.
The first y = R-3, with R being the total number of rows. (The step from row 1 to 2 is just 1/2 to go to the left, then you have 1/4 to go right down towards the middle, row 3, from which the 1/6 probabilities to go down left and right to row 4 start)
x = (R-3) - y
For each of these possible vertices the probabilities reach final row are (1/4)^x * (1/3)^y
At the end we have to take care of the symmetry and the first step.

STO 00......R-3
STO 01......y
STO 02...... probability of ending in last row
STO 03...... 1/3 (for speed, number entry in 41 is very slow)
STO 04...... 1/4 (for speed, number entry in 41 is very slow)

Enter into X the number of rows, press R/S

```
LBL "SRC12A"
3
-
STO 00 ! R-3
STO 01 ! y
LastX
1/x
STO 03 ! (1/3)
X<>Y
y^x
STO 02 ! prob P
4
1/x
STO 04 ! (1/4)
LBL 00 ! loop over y = (R-3) to 0
RCL 03
RCL 01
DECx
x<0?
GTO 01 ! -> we are done
STO 01
y^x
RCL 04
RCL 00
RCL 01
-
y^x
*
ST+ 02
GTO 00
LBL 01
RCL 04
RCL 00
y^x
ST+ 02
RCL 04
ST* 02
View 02
STOP
end
```

PM FIND QUOTE REPORT

17th October, 2022, 17:49 (This post was last modified: 19th October, 2022 23:53 by Albert Chan.) **Post: #56**

Albert Chan
Senior Member

Posts: 2,142
Joined: Jul 2018

RE: [VA] SRC #012a - Then and Now: Probability

Vincent Weber Wrote: (16th October, 2022 12:25)

> Memory requirements are huge. The needed registers seem in the range of 2.5*R^2 at bare minimum. For R=30 you need something like 18Kb if each register takes 8 bytes, plus extra variables, plus program stack... So you need something like 20Kb RAM. This rules out the 41, the 42, the SHARP pc-1261... You need a 32K machine basically!

I was thinking of minimum memory requirement, without code of checking corners, edges.
Also, code that would work with non-symmetrical starting conditions. (no symmetry trick)

In other words, this would still hold, collecting probabilities from hexagon vertices.
Q = weighted probability of previous P

P(I,J) = Q(I-1,J-1) + Q(I-1,J) + Q(I,J-1) + Q(I,J+1) + Q(I+1,J) + Q(I+1,J+1)

Current implementations treated (P, Q) as square matrix, with lots of 0's on the right.
But, if we flatten it, say into (p,q) of 1 dimension, all we need is 1 zero between rows.

Q(I,J) ≡ q(I*(I+1)/2 + J)

Q(1,1) = q(2)
Q(2,1), Q(2,2) = q(4), q(5)
Q(3,1), Q(3,2), Q(3,3) = q(7), q(8), q(9)
Q(4,1), Q(4,2), Q(4,3), Q(4,4) = q(11), q(12), q(13), q(14)
...

Note that q of triangular numbers are outside the triangle, with probability of 0.

Example, to get next iteration of P(3,2) = p(3*4/2+2 = 8)

p(8) = q(4) + q(5) + q(7) + q(9) + q(12) + q(13)

In general, for P(I,J) = p(x = I*(I+1)/2 + J)

p(x) = q(x-I-1) + q(x-I) + q(x-1) + q(x+1) + q(x+I+1) + q(x+I+2)

Q(0,0) .. Q(R+1,R+1) ≡ q(0) .. q((R+1)*(R+4)/2)

Total q elements = (R+1)*(R+4)/2 + 1= (R+2)*(R+3)/2

Technically p does not need as many spaces.
But for convenience, we like to keep both arrays with same dimensions.

Total array elements needed = (R+2)*(R+3) = floor((R+2.5)^2)

With theory out of the way, here is the flattened array version.

```
10 DESTROY ALL @ INPUT "[VA]SRC012A R,S= ";R,S @ SETTIME 0
20 OPTION BASE 0 @ T=(R+1)*(R+4)/2 @ REAL P(T),Q(T)
30 P(2)=1 @ M=2
40 FOR K=1 TO S
50 MAT Q=P @ Q(2)=Q(2)*3 @ T=3 ! BUILD Q
60 FOR I=2 TO M-(M=R) @ X=T+1 @ Q(X)=Q(X)*1.5 @ X=T+I @ Q(X)=Q(X)*1.5 @ T=X+1 @ NEXT I
70 IF M<>R THEN 100
80 FOR X=T+2 TO T+R-1 @ Q(X)=Q(X)*1.5 @ NEXT X
90 T=T+1 @ Q(T)=Q(T)*3 @ Q(X)=Q(X)*3
100 T=1 @ FOR I=1 TO M @ FOR X=T+1 TO T+I ! BUILD P
110 P(X)=Q(X-I-1)+Q(X-I)+Q(X-1)+Q(X+1)+Q(X+I+1)+Q(X+I+2)
120 NEXT X @ T=X @ NEXT I
130 M=M+(M<R)
140 DISP K;TIME
150 NEXT K
160 T=0 @ K=R*(R+1)/2 @ FOR X=K+1 TO K+R @ T=T+P(X) @ NEXT X
170 DISP TIME;R;S;T/6^S
```

On line 90, I use the quirk the last X is outside loop limits. Q(X) == Q(T+R)
Same idea on line 120, "T=X" same as "T=T+I+1", next triangular number

>RUN
[VA]SRC012A R,S= 30,60
...
83.1 30 60 9.51234350205E-6

```
>RUN
[VA]SRC012A R,S= 5,4
...
.31        5 4        7.98611111111E-2

>MAT P = (6^(-S)) * P ! scaled to have sum(P) = 1.0
>MAT DISP P ! note the single zero gap, between "row"
0
0
.114583333333
0
.127604166667
.127604166667
0
9.80902777778E-2
.175347222222
9.80902777778E-2
0
0
3.03819444445E-2
5.90277777778E-2
5.90277777778E-2
3.03819444445E-2
0
.0078125
1.99652777778E-2
2.43055555556E-2
1.99652777778E-2
.0078125
0
0
0
0
0
0
0
```

EMAIL PM FIND QUOTE REPORT

17th October, 2022, 19:12 **Post: #57**

Dave Britten
Senior Member

Posts: 2,222
Joined: Dec 2013

RE: [VA] SRC #012a - Then and Now: Probability

Sharp PC-1403H version of Valentin's 71B program (I didn't really have to change much, aside from minor syntactic/keyword differences):

```
10 CLEAR:INPUT "ROWS?";M:INPUT "STEPS?";N:DIM A(M,M):DIM B(M,M)
15 A(1,1)=1:W=M-1:FOR I=1 TO N:GOSUB 100
20 P=A(1,1)/2:IF P LET B(2,1)=B(2,1)+P:B(2,2)=B(2,2)+P
25 P=A(M,1)/2:IF P LET B(W,1)=B(W,1)+P:B(M,2)=B(M,2)+P
30 P=A(M,M)/2:IF P LET B(W,W)=B(W,W)+P:B(M,W)=B(M,W)+P
35 FOR X=2 TO W:U=X-1:V=X+1:P=A(X,1)/4:Q=A(X,X)/4:R=A(M,X)/4
40 IF P LET B(U,1)=B(U,1)+P:B(V,1)=B(V,1)+P:B(X,2)=B(X,2)+P:B(V,2)=B(V,2)+P
45 IF Q LET B(U,U)=B(U,U)+Q:B(V,V)=B(V,V)+Q:B(X,U)=B(X,U)+Q:B(V,X)=B(V,X)+Q
50 IF R LET B(M,U)=B(M,U)+R:B(M,V)=B(M,V)+R:B(W,U)=B(W,U)+R:B(W,X)=B(W,X)+R
55 NEXT X:FOR X=3 TO W:U=X-1:V=X+1:FOR Y=2 TO U:R=Y-1:S=Y+1:P=A(X,Y)/6
60 IF P LET B(U,R)=B(U,R)+P:B(U,Y)=B(U,Y)+P:B(X,R)=B(X,R)+P
65 IF P LET B(X,S)=B(X,S)+P:B(V,Y)=B(V,Y)+P:B(V,S)=B(V,S)+P
70 NEXT Y:NEXT X:GOSUB 200:NEXT I:P=0:FOR Y=1 TO M:P=P+A(M,Y):NEXT Y:BEEP 3: PRINT P:END
100 FOR J=1 TO M:FOR K=1 TO J:B(J,K)=0:NEXT K:NEXT J:RETURN
200 FOR J=1 TO M:FOR K=1 TO J:A(J,K)=B(J,K):NEXT K:NEXT J:RETURN
300 "A"S=0:FOR I=1 TO M:FOR J=1 TO I:S=S+A(I,J):NEXT J:NEXT I:PRINT S:END
```

This model takes about 4 hours to run for the 30, 60 case, and uses about 15 KB RAM for the program + data. After the program finishes, you can press DEF A to calculate the total probability for the whole map (should be extremely close to 1). One could easily tack on some more lines with user-key labels for performing other calculations on the finished matrix (probability of ending in the starting position, on any edge, at specific coordinates, etc.).

I believe this model has some machine-code matrix routines that can be invoked from within BASIC. I'll have to see if those can be used to speed up any of the matrix copying.

PM WWW FIND QUOTE REPORT

17th October, 2022, 20:01 **Post: #58**

Valentin Albillo
Senior Member

Posts: 958
Joined: Feb 2015
Warning Level: 0%

RE: [VA] SRC #012a - Then and Now: Probability

> **Dave Britten Wrote:**
> Darn, I was hoping your mention of "which is a **short** program capable of **quickly** solving the generic problem" meant you had a clever multinomial approach that could solve the problem in mere seconds on an HP 67.

Sarcastic much, are we ?

Sorry to **disillusion** you, but quoting myself:

> **Valentin Albillo Wrote:**
> [...] **let's not overhype** my abilities lest **disillusionment** ensues, ok ?

Too bad you didn't read it.

> **Dave Britten Wrote:**
> **Sharp PC-1403H version of Valentin's 71B program** (I didn't really have to change much, aside from minor syntactic/keyword differences) [...] This model takes about **4 hours** to run for the 30, 60 case [...]

Thanks for taking the trouble to key in my program into the **PC-1403H**, appreciated. That model is *slow* as molasses and the fact that you had to use `FOR..NEXT` loops to clear and copy matrices at each of the 60 steps slows down the program even further.

I would be curious to know how long does it take Mr. **Chan**'s fastest non-symmetric (general) program to run the *(30, 60)* case in: (a) The **1403H**, and (b) a physical **HP-71B**, if you'd obligue.

V.

PM WWW FIND EDIT QUOTE REPORT

17th October, 2022, 20:04 **Post: #59**

Xorand
Member

Posts: 76
Joined: Feb 2015

RE: [VA] SRC #012a - Then and Now: Probability

Fernando del Rey's program on my Sharp PC-1500 yielded the following various results. Originally, it would support a size of 23, but by adding a couple statements to grab the start and end times it dropped to 22 max.

Code:

```
Size  Step  Result           Elapsed
5     6     1.529224538E-01  00:00:28
10    12    4.841708817E-03  00:03:32
10    15    1.376503457E-02  00:04:13
10    20    3.34319009E-02   00:05:34
15    20    3.3016796293E-04 00:13:03
15    30    4.414569829E-03  00:19:33
20    30    4.948015608E-05  00:35:20
20    40    5.702504313E-04  00:47:04
22    44    2.51225541E-04   01:02:59
```

EMAIL PM FIND QUOTE REPORT

17th October, 2022, 20:06 **Post: #60**

Vincent Weber
Member

Posts: 288
Joined: May 2015

RE: [VA] SRC #012a - Then and Now: Probability

Hi Valentin,

I'm surprised to read that the PC-1403H is slow.. Is the PC-1475 faster ? Or the PC-1350/60 ?

Best regards,

Vincent

EMAIL PM FIND QUOTE REPORT

17th October, 2022, 20:29 **Post: #61**

Dave Britten
Senior Member

Posts: 2,222
Joined: Dec 2013

RE: [VA] SRC #012a - Then and Now: Probability

> **Valentin Albillo Wrote:** (17th October, 2022 20:01)
>
> > **Dave Britten Wrote:**
> > **Sharp PC-1403H version of Valentin's 71B program** (I didn't really have to change much, aside from minor syntactic/keyword differences) [...] This model takes about **4 hours** to run for the 30, 60 case [...]
>
> Thanks for taking the trouble to key in my program into the **PC-1403H**, appreciated. That model is *slow* as molasses and the fact that you had to use **FOR..NEXT** loops to clear and copy matrices at each of the 60 steps slows down the program even further.
>
> I would be curious to know how long does it take Mr. **Chan**'s fastest non-symmetric (general) program to run the *(30, 60)* case in: (a) The **1403H**, and (b) a physical **HP-71B**, if you'd oblige.
>
> **V.**

The results are in: it took around 4 hours to run the 30, 60 scenario on the pure-BASIC 1403H program. But it looks like I can speed this up by using the ROM-based matrix routines to handle copying/swapping/zeroing the matrices. Current estimates with smaller parameters suggests I might trim 1/3 off the runtime. We'll see! It's only been going about 20 minutes now.

I'll have to take a look at Albert's program and see if that might be usable (and possibly faster) here.

PM WWW FIND QUOTE REPORT

17th October, 2022, 22:16 **Post: #62**

Albert Chan
Senior Member

Posts: 2,142
Joined: Jul 2018

RE: [VA] SRC #012a - Then and Now: Probability

> **Dave Britten Wrote:** (17th October, 2022 20:29)
>
> I'll have to take a look at Albert's program and see if that might be usable (and possibly faster) here.

My flattened array code use little memory, and fast (fastest so far, without using symmetry)

The code use **MAT Q=P**, but it is not necessary.
All it need is to swap the name of 2 arrays. Sadly, VARSWAP P, Q does not work.

Here is a version that simulate swapping of array name.

Array P of x ≡ A(P, x)
Array Q of x ≡ A(Q, X)

Note: P, Q are now numbers 0 or 1, P+Q=1, not the array itself.

```
10 DESTROY ALL @ INPUT "[VA]SRC012A R,S= ";R,S @ SETTIME 0
20 OPTION BASE 0 @ REAL A(1,(R+1)*(R+4)/2) @ P=0
30 A(P,2)=1 @ M=2
40 FOR K=1 TO S
50 Q=P @ P=1-P @ A(Q,2)=A(Q,2)*3 @ T=3 ! BUILD Q
60 FOR I=2 TO M-(M=R) @ X=T+1 @ A(Q,X)=A(Q,X)*1.5 @ X=T+I @ A(Q,X)=A(Q,X)*1.5 @ T=X+1 @ NEXT I
70 IF M<>R THEN 100
80 FOR X=T+2 TO T+R-1 @ A(Q,X)=A(Q,X)*1.5 @ NEXT X
```

```
90 T=T+1 @ A(Q,T)=A(Q,T)*3 @ A(Q,X)=A(Q,X)*3
100 T=1 @ FOR I=1 TO M @ FOR X=T+1 TO T+I ! BUILD P
110 A(P,X)=A(Q,X-I-1)+A(Q,X-I)+A(Q,X-1)+A(Q,X+1)+A(Q,X+I+1)+A(Q,X+I+2)
120 NEXT X @ T=X @ NEXT I
130 M=M+(M<R)
140 DISP K;TIME
150 NEXT K
160 T=0 @ K=R*(R+1)/2 @ FOR X=K+1 TO K+R @ T=T+A(P,X) @ NEXT X
170 DISP TIME;R;S;T/6^S
```

>RUN
[VA]SRC012A R,S= 30,60
...
91.38 30 60 9.51234350205E-6

Not as good as original flattened array version (83.1 sec), but not too bad.

EMAIL PM FIND QUOTE REPORT

17th October, 2022, 22:38 **Post: #63**

Valentin Albillo
Senior Member

Posts: 958
Joined: Feb 2015
Warning Level: 0%

RE: [VA] SRC #012a - Then and Now: Probability

> **Albert Chan Wrote:** (17th October, 2022 22:16)
>
> Here is a version that simulate swapping of array name [...]
> >RUN
> [VA]SRC012A R,S= 30,60
> ...
> **91.38** 30 60 9.51234350205E-6

Just one question: the quoted time (**91.38 seconds**) applies when your program is run on what machine ? A *physical* **HP-71B** ? If not, what's the time for a *physical* **HP-71B** ?

Thanks and regards.
V

PM WWW FIND EDIT QUOTE REPORT

17th October, 2022, 23:19 **Post: #64**

Albert Chan
Senior Member

Posts: 2,142
Joined: Jul 2018

RE: [VA] SRC #012a - Then and Now: Probability

> **Valentin Albillo Wrote:** (17th October, 2022 22:38)
>
> > **Albert Chan Wrote:** (17th October, 2022 22:16)
> >
> > Here is a version that simulate swapping of array name [...]
> > >RUN
> > [VA]SRC012A R,S= 30,60
> > ...
> > **91.38** 30 60 9.51234350205E-6
>
> Just one question: the quoted time (**91.38 seconds**) applies when your program is run on what machine ?

It was **Emu71/DOS**, running under DOSBOX, at cycles = max 50%, on my laptop

Running [VA] code on same condition, it clocked at **126.92 seconds.**
Fastest so far is my flattened array code, clocked at **83.10 seconds.**

All 3 codes can handle non-symmetrical starting conditions.

EMAIL PM FIND QUOTE REPORT

17th October, 2022, 23:36 **Post: #65**

Albert Chan
Senior Member

Posts: 2,142
Joined: Jul 2018

RE: [VA] SRC #012a - Then and Now: Probability

When I time MAT P=ZER vs. MAT P=(0), I see no difference.

Is it the same on an actual HP71B ?

EMAIL PM FIND QUOTE REPORT

18th October, 2022, 00:49 **Post: #66**

Fernando del Rey
Junior Member

Posts: 19
Joined: Dec 2013

RE: [VA] SRC #012a - Then and Now: Probability

In a physical HP-71B I got the following timing for AC's program in post #62

2464.74 30 60 9.51234350205E-6

Thus, a little over 41 minutes.

EMAIL PM FIND QUOTE REPORT

18th October, 2022, 10:00 **Post: #67**

J-F Garnier
Senior Member

Posts: 790
Joined: Dec 2013

RE: [VA] SRC #012a - Then and Now: Probability

> **Dave Britten Wrote:** (17th October, 2022 20:29)
>
> The results are in: it took around 4 hours to run the 30, 60 scenario on the pure-BASIC 1403H program.

What is the numeric result? I'm curious to compare the accuracy too, is the 1403H a 12-digit machine and does it manage the "round-to -even" rule? I gave a simple test to check it above.

> **Albert Chan Wrote:** (17th October, 2022 22:16)
>
> Here is a version that simulate swapping of array name.
> Array P of x ≡ A(P, x)
> Array Q of x ≡ A(Q, X)
> Note: P, Q are now numbers 0 or 1, P+Q=1, not the array itself.
> [..]
>
> `20 OPTION BASE 0 @ REAL A(1,(R+1)*(R+4)/2) @ P=0`

I don't see the benefit of this version, you are still using the same amount of memory, and the access to the array A is slower which is not compensated by the gain of the MAT copy.

> **Albert Chan Wrote:** (17th October, 2022 23:36)
>
> When I time MAT P=ZER vs. MAT P=(0), I see no difference.

There is a difference, MAT P=ZER is about 2x faster:

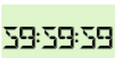
```
10 DIM A(4096)
20 T=TIME @ MAT A=ZER @ T=TIME-T @ DISP T
20 T=TIME @ MAT A=(0) @ T=TIME-T @ DISP T
>RUN
.38
.76
```

(physical HP-71B)
However, if you target the smallest code, MAT A=(0) is one byte shorter :-)

J-F

EMAIL PM WWW FIND QUOTE REPORT

18th October, 2022, 10:19 **Post: #68**



Werner
Senior Member

Posts: 767
Joined: Dec 2013

RE: [VA] SRC #012a - Then and Now: Probability

> **J-F Garnier Wrote:** (18th October, 2022 10:00)

> **Albert Chan Wrote:** (17th October, 2022 22:16)
>
> > Here is a version that simulate swapping of array name.
> > Array P of x ≡ A(P, x)
> > Array Q of x ≡ A(Q, X)
> > Note: P, Q are now numbers 0 or 1, P+Q=1, not the array itself.
> > [..]
> >
> > ```
> > 20 OPTION BASE 0 @ REAL A(1,(R+1)*(R+4)/2) @ P=0
> > ```
>
> I don't see the benefit of this version, you are still using the same amount of memory, and the access to the array A is slower which is not compensated by the gain of the MAT copy.

No, it uses only half the memory, roughly?
Werner

EMAIL PM FIND QUOTE REPORT

18th October, 2022, 11:08 **Post: #69**

J-F Garnier
Senior Member

Posts: 790
Joined: Dec 2013

RE: [VA] SRC #012a - Then and Now: Probability

> **Valentin Albillo Wrote:** (16th October, 2022 19:52)
>
> So, this is my original solution, a *13-line*, *683-byte* program.
> [..]
> Now for the big *(30, 60)* case:
> >RUN
>
> Rows,Steps= 30,60 [ENDLINE] -> 9.51234350207E-6 **27.46"**

> **Valentin Albillo Wrote:** (17th October, 2022 22:38)
>
> > **Albert Chan Wrote:** (17th October, 2022 22:16)
> >
> > Here is a version that simulate swapping of array name [...]
> > >RUN
> > [VA]SRC012A R,S= 30,60
> > ...
> > **91.38** 30 60 9.51234350205E-6
>
> Just one question: the quoted time (**91.38 seconds**) applies when your program is run on what machine ? A *physical* **HP-71B** ? If not, what's the time for a *physical* **HP-71B** ?

With the various machines and platforms running HP-71 code (Android, Windows, go71, Emu71/Win, Emu71/DOS w/ or w/o DOSBox), it's difficult to compare execution times.
Downloading and running each code on a physical HP-71 is not convenient, and not everybody has the means to do it - although they should :-)

So I propose the use Valentin's solution as the reference.
Here are the results for my solution:
VA's reference solution: 91"
my solution : 63" = **0.69 in Valentin's units** :-)
Note I used the symmetry of the problem.

J-F

EMAIL PM WWW FIND QUOTE REPORT

18th October, 2022, 13:16 **Post: #70**

Albert Chan
Senior Member

Posts: 2,142
Joined: Jul 2018

RE: [VA] SRC #012a - Then and Now: Probability

> **J-F Garnier Wrote:** (18th October, 2022 10:00)
>
> > **Albert Chan Wrote:** (17th October, 2022 22:16)

> Here is a version that simulate swapping of array name.
> Array P of x ≡ A(P, x)
> Array Q of x ≡ A(Q, X)
> Note: P, Q are now numbers 0 or 1, P+Q=1, not the array itself.
> [..]
>
> `20 OPTION BASE 0 @ REAL A(1,(R+1)*(R+4)/2) @ P=0`

I don't see the benefit of this version, you are still using the same amount of memory, and the access to the array A is slower which is not compensated by the gain of the MAT copy.

That's why it is called *simulate swapping of array name.*
True variable name swapping should be almost cost free.

It is a proof of concept, showing MAT Q=P is not needed.
The code is useful for machine that use slow FOR-NEXT for MAT copy.
Simulated array name swapping removed the slow FOR-NEXT loops.

Perhaps add **MAT SWAP** to HP Article VA044 - HP-71B Math Pac 2 Comments and Proposals.pdf ?

Flattened A array of 1 dimension, array access cost almost matched removal of MAT COPY
However, for optimized code, it may be hard to deduce where A is pointing to.
That's why I posted the slower A(P, x) version, instead of faster A(P + x)

Anyway, this was my flattened A version.

Note: we reduced array elements required by 1, but still safe.
Note: Build Q part, to simplify code, T is triangular number minus one, plus Q

```
10 DESTROY ALL @ INPUT "[VA]SRC012A R,S= ";R,S @ SETTIME 0
20 OPTION BASE 0 @ P=0 @ Q=(R+1)*(R+4)/2 @ REAL A(2*Q)
30 A(2)=1 @ M=2
40 FOR K=1 TO S
50 VARSWAP P,Q @ T=Q+2 @ A(T)=A(T)*3 ! BUILD Q
60 FOR I=1 TO M-(M=R)-1 @ T=T+2 @ A(T)=A(T)*1.5 @ T=T+I @ A(T)=A(T)*1.5 @ NEXT I
70 IF M<>R THEN 100
80 FOR X=T+3 TO T+R @ A(X)=A(X)*1.5 @ NEXT X
90 T=T+2 @ A(T)=A(T)*3 @ A(X)=A(X)*3
100 T=Q+1 @ Y=P-Q @ FOR I=1 TO M @ FOR X=T+1 TO T+I ! BUILD P
110 A(X+Y)=A(X-I-1)+A(X-I)+A(X-1)+A(X+1)+A(X+I+1)+A(X+I+2)
120 NEXT X @ T=X @ NEXT I
130 M=M+(M<R)
140 DISP K;TIME
150 NEXT K
160 T=0 @ K=P+R*(R+1)/2 @ FOR X=K+1 TO K+R @ T=T+A(X) @ NEXT X
170 DISP TIME;R;S;T/6^S
```

>RUN
[VA]SRC012A R,S= 30,60
...
84.49 30 60 9.51234350205E-6

Very close to fastest MAT Q=P version, clocked at 83.1 sec.

EMAIL PM FIND QUOTE REPORT

18th October, 2022, 14:25 **Post: #71**

Dave Britten
Senior Member

Posts: 2,222
Joined: Dec 2013

RE: [VA] SRC #012a - Then and Now: Probability

> **J-F Garnier Wrote:** (18th October, 2022 10:00)
>
> > **Dave Britten Wrote:** (17th October, 2022 20:29)
> >
> > The results are in: it took around 4 hours to run the 30, 60 scenario on the pure-BASIC 1403H program.
>
> What is the numeric result? I'm curious to compare the accuracy too, is the 1403H a 12-digit machine and does it manage the "round-to -even" rule? I gave a simple test to check it above.

I get 9.512343561E-06 for the 30, 60 bottom-row case, and the full probability map sums up to 1.000 000 009. I believe most Sharps are 10-digit machines.

Here's an amended version of the program that uses two of the ROM routines for matrix handling. Now it only takes a little over 3 hours to run the full 30, 60. :)

```
10 CLEAR:INPUT "ROWS?";M:INPUT "STEPS?";N:DIM Y(M,M):DIM X(M,M)
15 Y(1,1)=1:W=M-1:FOR I=1 TO N:X=0:CALL 26153
20 P=Y(1,1)/2:IF P LET X(2,1)=X(2,1)+P:X(2,2)=X(2,2)+P
25 P=Y(M,1)/2:IF P LET X(W,1)=X(W,1)+P:X(M,2)=X(M,2)+P
30 P=Y(M,M)/2:IF P LET X(W,W)=X(W,W)+P:X(M,W)=X(M,W)+P
35 FOR X=2 TO W:U=X-1:V=X+1:P=Y(X,1)/4:Q=Y(X,X)/4:R=Y(M,X)/4
40 IF P LET X(U,1)=X(U,1)+P:X(V,1)=X(V,1)+P:X(X,2)=X(X,2)+P:X(V,2)=X(V,2)+P
45 IF Q LET X(U,U)=X(U,U)+Q:X(V,V)=X(V,V)+Q:X(X,U)=X(X,U)+Q:X(V,X)=X(V,X)+Q
50 IF R LET X(M,U)=X(M,U)+R:X(M,V)=X(M,V)+R:X(W,U)=X(W,U)+R:X(W,X)=X(W,X)+R
55 NEXT X:FOR X=3 TO W:U=X-1:V=X+1:FOR Y=2 TO U:R=Y-1:S=Y+1:P=Y(X,Y)/6
60 IF P LET X(U,R)=X(U,R)+P:X(U,Y)=X(U,Y)+P:X(X,R)=X(X,R)+P
65 IF P LET X(X,S)=X(X,S)+P:X(V,Y)=X(V,Y)+P:X(V,S)=X(V,S)+P
70 NEXT Y:NEXT X:CALL 26163:NEXT I:P=0:FOR Y=1 TO M:P=P+Y(M,Y):NEXT Y:BEEP 3:PRINT P:END
300 "A"S=0:FOR I=1 TO M:FOR J=1 TO I:S=S+Y(I,J):NEXT J:NEXT I:PRINT S:END
```

The list of matrix routine addresses can be found in message 51 of this thread. These are the two I used:

26153 - Multiply array X() by scalar variable X and store the result back in X() (in this case, I multiply by 0 to zero out the array)
26163 - Swap X() and Y() arrays

PM WWW FIND QUOTE REPORT

18th October, 2022, 17:38 (This post was last modified: 18th October, 2022 17:55 by J-F Garnier.) **Post: #72**

J-F Garnier
Senior Member

Posts: 790
Joined: Dec 2013

RE: [VA] SRC #012a - Then and Now: Probability

> **Fernando del Rey Wrote:** (18th October, 2022 00:49)
>
> In a physical HP-71B I got the following timing for AC's program in post #62
>
> **2464.74** 30 60 9.51234350205E-6
>
> Thus, a little over 41 minutes.

To restore a little the HP-75 *prestige*, the same post #62 program (with minor changes°) runs on the 75 as:

1353.469 30 60 9.51234350246E-6

About 23 minutes, or about 2x faster than the HP-71.
Of course the price of the 75, at the time, was also in the *prestige* class.

° Changes:

```
5 OPTION BASE 0 @ REAL A(1,527)
10 INPUT "[VA]SRC012A R,S= "; R,S @ T0=TIME
20 REDIM A(1,(R+1)*(R+4)/2) @ P=0
25 MAT A=ZER ! init vars
170 DISP TIME-T0;R;S;T/6^S
```
(I hate to change the clock setting with SETTIME)

J-F

EMAIL PM WWW FIND QUOTE REPORT

18th October, 2022, 19:15 **Post: #73**

Valentin Albillo
Senior Member

Posts: 958
Joined: Feb 2015
Warning Level: 0%

RE: [VA] SRC #012a - Then and Now: Probability

Hi, **J-F**,

> **J-F Garnier Wrote:**
>
> So I propose the use Valentin's solution as the reference [...]
> **VA's reference solution: 91"**
> my solution : 63" = **0.69 in Valentin's units** :-)
> Note **I used the symmetry** of the problem.

I don't get it. You say my *"reference solution"* runs in **91"**, but ... *on which hardware/software !?*

As far as I can tell, so far my solution runs in these times:

```
3496.49" (58'16.49") on an actual, physical HP-71B (Fernando del Rey dixit)
  27.46" on my 4-yo Samsung Galaxy Tab A 6 tablet (Android)
 ~ 3 hr  on a Sharp PC-1403H (Dave Britten dixit)
 126.92" on Emu71/DOS, running under DOSBOX, unknown hardware (A.Chan dixit)
```

so from where did you take those **91"** you adscribe to my solution ?

As an additional comment, I don't think that merely comparing runtimes among different solutions is meaningful even for the exact same hardware/software, because *other factors* also play a big role, among them the following come to mind:

- ***general program*** (works for any starting positions because it doesn't rely on symmetry) vs. ***program particularized*** for *symmetric* starting positions (rely on symmetry to halve memory requirements and speed up the computation but won't work at all on non-symmetric starting positions.)

- ***didactic program*** whose inner workings are readily understandable to any newcomer upon seeing the problem's definition and the listing vs. highly-optimized but *overwhelmingly* ***unfathomable*** code to most people upon looking at the listing.

- ability to ***answer the additional questions very easily*** from the command line once the program has been run vs. ***very dificult/impossible*** to answer those additional questions from the command line due to very intrincate element addressing.

As you can see, without taking those factors and others into accound the comparisons are mostly ***meaningless***.

Thanks for your continued interest in this problem and best regards.
V.

19th October, 2022, 23:55 **Post: #74**

Albert Chan
Senior Member

Posts: 2,142
Joined: Jul 2018

RE: [VA] SRC #012a - Then and Now: Probability

> **Albert Chan Wrote:** (18th October, 2022 13:16)
>
> Anyway, this was my flattened A version.
>
> Note: we reduced array elements required by 1, but still safe.
> Note: Build Q part, to simplify code, T is triangular number minus one, plus Q

We might as well reduce memory use to its absolute minimum, remove 1 more element.
For consistency, Build P part also defined T as triangular number minus one, plus Q
In other words, A(T) is triangle Q right edge, A(T+2) is Q left edge.

```
10 DESTROY ALL @ INPUT "[VA]SRC012A R,S= ";R,S @ SETTIME 0
20 OPTION BASE 0 @ P=0 @ Q=(R+1)*(R+4)/2-1 @ REAL A(2*Q+1)
30 A(2)=1 @ M=2
40 FOR K=1 TO S
50 VARSWAP P,Q @ T=Q+2 @ A(T)=A(T)*3 ! BUILD Q
60 FOR I=1 TO M-(M=R)-1 @ T=T+2 @ A(T)=A(T)*1.5 @ T=T+I @ A(T)=A(T)*1.5 @ NEXT I
70 IF M<>R THEN 100
80 FOR X=T+3 TO T+R @ A(X)=A(X)*1.5 @ NEXT X
90 T=T+2 @ A(T)=A(T)*3 @ A(X)=A(X)*3
100 T=Q @ Y=P-Q+1 @ FOR I=1 TO M @ FOR X=T+1 TO T+I ! BUILD P
110 A(X+Y)=A(X-I)+A(X-I+1)+A(X)+A(X+2)+A(X+I+2)+A(X+I+3)
120 NEXT X @ T=X @ NEXT I
130 M=M+(M<R)
140 DISP K;TIME
150 NEXT K
```

```
160 T=0 @ K=P+R*(R+1)/2 @ FOR X=K+1 TO K+R @ T=T+A(X) @ NEXT X
170 DISP TIME;R;S;T/6^S
```

>RUN
[VA]SRC012A R,S= 5,4
...
82.79 30 60 9.51234350205E-6

This version is currently the fastest, but only by a hair.

I like MAT Q=P version better; code is more clear.

Quote:

Flattened A array of 1 dimension, array access cost almost matched removal of MAT COPY
However, for optimized code, it may be hard to deduce where A is pointing to.

EMAIL PM FIND QUOTE REPORT

19th October, 2022, 23:56 (This post was last modified: 20th October, 2022 00:05 by Albert Chan.) **Post: #75**

Albert Chan
Senior Member

Posts: 2,142
Joined: Jul 2018

RE: [VA] SRC #012a - Then and Now: Probability

Running Emu71/Dos under DosBox is slow, with bad timing info.
Running it in WinXP is faster, with more consistent timing.

This despite my Toshiba laptop is supposed to be 10x faster than my 22 years old Dell ... go figures.

HP71B codes so far, running in Dell Optiplex GX110 866MHz, with 512M Ram
All codes adjusted to return only P(R=30,S=60), and time needed.

Timings from best of 3
Symmetry of Yes meant starting condition must be symmetric.

Post	Member	Time(s)	Symmetry	Array elements
21	Fernando del Rey	34.29	No	2*R^2
22	J-F Garnier	12.91	Yes	2*R^2
33	C.Ret	34.94	Yes	3*R^2
35	Albert Chan	9.79	Yes	2*R^2
40	Albert Chan	26.09	Yes	3*R^2
42	Albert Chan	9.79	Yes	2*R^2 + (R+2)*(ceil(R/2)+2)
48	C.Ret	9.79	Yes	2*R^2 + (R+2)*(ceil(R/2)+2)
49	Albert Chan	6.72	Yes	2*(R+2)*(ceil(R/2)+2)
50	Valentin Albillo	17.82	No	2*R^2
51	Albert Chan	12.14	No	2*(R+2)^2
56	Albert Chan	11.42	No	(R+2)*(R+3)
62	Albert Chan	12.79	No	(R+2)*(R+3)
70	Albert Chan	11.86	No	(R+2)*(R+3) - 1
74	Albert Chan	11.32	No	(R+2)*(R+3) - 2

EMAIL PM FIND QUOTE REPORT

20th October, 2022, 20:57 **Post: #76**

Albert Chan
Senior Member

Posts: 2,142
Joined: Jul 2018

RE: [VA] SRC #012a - Then and Now: Probability

Fernando del Rey Wrote: (13th October, 2022 14:04)

And you could also consider the symmetry of the solution if the man is starting at cell (1,1), calculating only half of the grid. But then the algorithm would not be valid for a starting position which is not located in the central column of the grid, which is therefore not symmetrical.

If the goal is row probability, symmetric solution can still work.

Say, D is starting distribution, D' is its mirror image, P_i is i-th row probability
Note: sum of probability distribution = 1.0

Symmetry: $P_i(D) = P_i(D') = P_i(D + D')$

Example, below 3 initial conditions produce same row probability.

A(1,1)=1/2 @ A(3,2)=1/6 @ A(3,1)=1/3 ! asymmetric distribution
A(1,1)=1/2 @ A(3,2)=1/6 @ A(3,3)=1/3 ! mirror image
A(1,1)=1/2 @ A(3,2)=1/6 @ A(3,1)=1/6 @ A(3,3)=1/6 ! symmetric distribution

For row probability, we can transform to symmetric distribution, then process only half the grid.

EMAIL PM FIND QUOTE REPORT

20th October, 2022, 23:39 **Post: #77**

Gjermund Skailand
Member

Posts: 52
Joined: Dec 2013

RE: [VA] SRC #012a - Then and Now: Probability

This has been a very interesting thread.
This is a sys-rpl version for HP50g of CReth SRC12a.
On an actual HP50g the calculation time for the 30 60 problem is 5 min 21sec.
p= 9.51234350207E-6

```
"
!RPL
!NO CODE
!JAZZ
::
CK2NOLASTWD
CK&DISPATCH2
#11
:: COERCE2
'
CODE
GOSBVL POP2# RSTK=C SAVE
B=A.A A*A.A A-B.A ASRB.A
C=RSTK A+C.A
GOSBVL PUSH#ALOOP
ENDCODE
3PICK DUP 3PICK EVAL SWAP 2 SWAPOVER
{{ r s d m Tind ii }}
r #1+_ONE_DO (i)
INDEX @ #1+_ONE_DO (j)
3
DUP INDEX@ #1<> ?SKIP #1-
JINDEX@ INDEX@ #<> ?SKIP #1-
JINDEX@ r #<> ?SKIP #1-
UNCOERCE %/
LOOP (j)
LOOP (i)
d UNCOERCE ONE{}N FPTR2 ^XEQ>ARRY
DUP %0 xCON
%1 BINT1 PUTREALEL
s #1+_ONE_DO (k)
INDEX@ #>$ BIGDISPROW1
2DUP m m 2GETEVAL DUP4UNROLL TOTEMPOB (n)
#1+_ONE_DO (q)
SWAP INDEX@ PULLREALEL
ROT INDEX@ PULLREALEL
ROT %* 4UNROLL
LOOP (q)
2DROP
UNCOERCE ONE{}N x>ARRY
m #1+_ONE_DO (i)
INDEX@ #2 #/ #+ #1+_ONE_DO (j)
JINDEX@ INDEX@ 2GETEVAL
PULLREALEL
%CHS
JINDEX@ TOTEMPOB !ii
1 JINDEX@ #1- #MAX
JINDEX@ #1+ m #MIN
#1+ SWAP DO (a)
1 JINDEX@ INDEX@ ii #> ?SKIP #1- MAX
JINDEX@ ii INDEX@ #> ?SKIP #1+ INDEX@ MIN
```

```
#1+SWAP DO (b)
SWAP JINDEX@ INDEX@ 2GETEVAL PULLREALEL
ROT %+
LOOP (b)
LOOP (a)
ROT
JINDEX@ INDEX@ 2GETEVAL
3PICKSWAP PUTREALEL
JINDEX@ #1+ INDEX@ #-
2GETEVAL
ROTSWAP
PUTREALEL
SWAP
LOOP (j)
LOOP (i)
DROP
m DUP r #>=_ ?SKIP #1+ !m
LOOP (k)
SWAP %0 xCON
r 1 2GETEVAL
UNCOERCE
xDO %1 xPUTI xUNTIL
% -64 xFS?
xENDDO
xDROP
xDOT
%6
s UNCOERCE
x^
x/
ABND
;
;
@
"
```

I hope I got it without typing errors.
The small code object "Tind" reduces calculation time with 29%, from about 450sec to 321.
I lost the userRPL version, but it was many times slower.
br Gjermund

EMAIL PM FIND QUOTE REPORT

21st October, 2022, 07:27 **Post: #78**

Werner
Senior Member

Posts: 767
Joined: Dec 2013

RE: [VA] SRC #012a - Then and Now: Probability

> **Albert Chan Wrote:** (16th October, 2022 19:29)
>
> (*) I don't really need a copy, swapping name of array (P,Q) is enough.
> Too bad ... VARSWAP don't work for arrays.
>
> >VARSWAP P, Q
> ERR:Data Type

As far as I know (which is not a lot, admittedly), VARSWAP swaps the *values* of the variables, not their names.
Cheers, Werner

EMAIL PM FIND QUOTE REPORT

21st October, 2022, 18:10 **Post: #79**

Albert Chan
Senior Member

Posts: 2,142
Joined: Jul 2018

RE: [VA] SRC #012a - Then and Now: Probability

> **PeterP Wrote:** (12th October, 2022 14:10)
>
> My code does deliver the correct result for R = 5, but I dont have a good way (especially right now on a plane and my work computer has no simulators installed...) to check if it is correct for R = 30, S=29. (It comes out to 1.311095094 e-13).

For S = R-1, we can treat triangle as without bottom edge (and the 2 corners).

First step from top corner, it gives equal probability to left or right side.
We can thus skip first iteration, simplified the problem without top corner.

Problem now is relatively simple, with only inside (6 ways) and edge (4 ways)
Only edge probability can "leak" to the inside; inside probabilities never "gets out".

Work out the geometric progression (not shown), with p=1/6, q=1/4, we have:

P(R, S=R-1) = ((2p)^(R-3) - q^(R-3)) / (2*p-q) * (2*p*q) + 2*q^(R-2)

(2*p*q) / (2*p-q) = 1 / (1/q-1/(2*p)) = 1 / (4-3) = 1. It simplified to:

P(R, S=R-1) = 3^(3-R) - 2^(5-2*R)

Example:

P(1,0) = 9 - 8 = 1
P(2,1) = 3 - 2 = 1
P(3,2) = 1 - 1/2 = 1/2
P(4,3) = 1/3 - 1/8 = 5/24
P(5,4) = 1/9 - 1/32 = 23/288
P(6,5) = 1/27 - 1/128 = 101/3456
...
P(30,29) = 1/3^27 - 1/2^55 ≈ 1.31109509664e-13

EMAIL PM FIND QUOTE REPORT

21st October, 2022, 19:38 **Post: #80**

pier4r
Senior Member

Posts: 2,224
Joined: Nov 2014

RE: [VA] SRC #012a - Then and Now: Probability

> **Albert Chan Wrote:** (19th October, 2022 23:56)
> This despite my Toshiba laptop is supposed to be 10x faster than my 22 years old Dell ... go figures.

semi-OT.

Because emulating a system may be done in a not too efficient way and a lot is lost on the way. For example Saturn code is emulated in the 50g, but if you take the emulation away and you write code with tools that are more optimized for the machine (newRPL for example), the speed difference is impressive.

So yes one could execute things on very powerful systems that are simply wasting a lot due to inefficiencies.

Good to know that the Dos emulator is not that great. Would be interesting if you try compatibility options offered by windows itself or virtual machines via microsoft virtual PC.

Anyway as valentin said, in the big picture execution time is only partially meanigful.

EMAIL PM FIND QUOTE REPORT

21st October, 2022, 19:52 **Post: #81**

Albert Chan
Senior Member

Posts: 2,142
Joined: Jul 2018

RE: [VA] SRC #012a - Then and Now: Probability

Here, we do P(R, S=R-1) by hand, to discover its patterns.
Even rows are previous row probabilities, scaling to 1 way, edge /4, inside /6.

```
1/2     1/2 --> P(2,1) = 1
1/8     1/8
1/8     1/4     1/8 --> P(3,2) = 1/2
1/32    1/24    1/32
1/32    7/96    7/96    1/32 --> P(4,3) = 5/24
1/128   7/576   7/576   1/128
1/128   23/1152 7/288   23/1152   1/128 --> P(5,4) = 23/288
```

We can also get P(5,4), directly from scaled P(4,3):

P(5,4) = 2*(1/128 + 7/576 + 7/576 + 1/128) = 4*(1/128 + 7/576) = 23/288

> **Albert Chan Wrote:** (21st October, 2022 18:10)
>
> **P(R, S=R-1) = 3^(3-R) - 2^(5-2*R)**

We are now ready to proof above, by induction
Assume formula is correct, we split it to two types, to get P(R+1, S=R):

P(R, S=R-1) = P(edges) + P(inside)

3^(3-R) - 2^(5-2*R) = 2^(4-2*R) + (3^(-R+3) - 3*2^(4-2*R))

P(R+1, S=R)
= 2 * sum(scaled to 1 way of P(R, S=R-1)
= 2 * (2^(4-2*R)**/4** + (3^(-R+3) - 3*2^(4-2*R))**/6**)
= 9*3^(-R) - 8*2^(-2*R)
= 3^(3-(R+1)) - 2^(5-2*(R+1)) **QED**

EMAIL PM FIND QUOTE REPORT

22nd October, 2022, 00:00 **Post: #82**

PeterP
Member

Posts: 172
Joined: Jul 2015

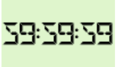
RE: [VA] SRC #012a - Then and Now: Probability

> **Albert Chan Wrote:** (21st October, 2022 18:10)
>
> > **PeterP Wrote:** (12th October, 2022 14:10)
> >
> > My code does deliver the correct result for R = 5, but I dont have a good way (especially right now on a plane and my work computer has no simulators installed...) to check if it is correct for R = 30, S=29. (It comes out to 1.311095094 e-13).
>
> For S = R-1, we can treat triangle as without bottom edge (and the 2 corners).
>
> First step from top corner, it gives equal probability to left or right side.
> We can thus skip first iteration, simplified the problem without top corner.
>
> Problem now is relatively simple, with only inside (6 ways) and edge (4 ways)
> Only edge probability can "leak" to the inside; inside probabilities never "gets out".
>
> Work out the geometric progression (not shown), with p=1/6, q=1/4, we have:
>
> P(R, S=R-1) = ((2p)^(R-3) - q^(R-3)) / (2*p-q) * (2*p*q) + 2*q^(R-2)
>
> (2*p*q) / (2*p-q) = 1 / (1/q-1/(2*p)) = 1 / (4-3) = 1. It simplified to:
>
> **P(R, S=R-1) = 3^(3-R) - 2^(5-2*R)**
>
> Example:
>
> P(1,0) = 9 - 8 = 1
> P(2,1) = 3 - 2 = 1
> P(3,2) = 1 - 1/2 = 1/2
> P(4,3) = 1/3 - 1/8 = 5/24
> P(5,4) = 1/9 - 1/32 = 23/288
> P(6,5) = 1/27 - 1/128 = 101/3456
> ...
> P(30,29) = 1/3^27 - 1/2^55 ≈ 1.31109509664e-13

Very neat Albert! It converts the summation into a formula for the sum as its a geometric progression (which I did not recognize). You clearly did not need a computer and could have proven the result to be correct on an airplane with just a simple calculator, your pen and pencil :-) Thank you for sharing.

PM FIND QUOTE REPORT

27th October, 2022, 16:07 **Post: #83**



Werner
Senior Member

Posts: 767
Joined: Dec 2013

RE: [VA] SRC #012a - Then and Now: Probability

> **Valentin Albillo Wrote:** (5th October, 2022 22:38)
>
> using **EXCLUSIVELY VINTAGE HP CALCULATORS** (physical or virtual,) coding in either **RPN**, **RPL** or **HP-71B** language **AND NOTHING ELSE**

Well, to make Valentin's wishes come true, here's an entry that will solve the 30/60 problem on a real 42S, the only vintage RPN calculator able to do it.
To make it fit the 42S' memory, I have taken Albert Chan's flattened code to the extreme: you don't need a full P *and* Q, they can largely overlap, all you need is an extra buffer row at the end.
The memory requirements are then (R+4)/2 x (R+3), and I define REGS as such.
When P is calculated it is shifted down a full row with regard to Q, in rows 2..(R+4)/2, and we move it one row up by deleting the first row and adding a new empty row at the end (which, incidentally, you can't do with INSR).

here's the code. Not much time has been spent in trying to improve it, just to make it work ;-)
Estimate of real 42S running time: 3h05m

I use VARMENU "TRW" to set R and S, EXIT the menu and do XEQ "TRW"

```
00 { 325-Byte Prgm }
01▸LBL "TRW"
02 MVAR "R"
03 MVAR "S"
04 4
05 RCL+ "R"
06 2
07 STO "M"
08 ÷
09 3
10 RCL+ "R"
11 CLV "REGS"
12 DIM "REGS"
13 1
14 STO 02
15 RCL "S"
16 STO "K"
17 EDITN "REGS"
18 GROW

19▸LBL 20
@ ---------------------------------
@ P-> Q, adjust corners and edges
@ ---------------------------------
20 3
21 STO× 02 @ top
22 RCL "M"
23 RCL "R"
24 X=Y?
25 DSE ST Y
26 SIGN
27 -
28 1E3
29 STO+ ST Y
30 ÷ @ I=1..M-1-(M=R)
31 2
32▸LBL 02 @ left and right edges
33 2
34 +
35 RCL+ ST Y
36 1.5
37 STO× IND ST Y
38 STO× IND ST L
39 R↓
40 IP
41 ISG ST Y
42 GTO 02
43 RCL "M"
44 RCL "R"
```

```
45 X>Y?
46 GTO 00
47 RCL ST Z
48 ENTER
49 ENTER
50 RCL+ "R"
51 1E3
52 ÷
53 +
54 3
55 +
56 1.5
57▸LBL 03 @ bottom edge
58 STO× IND ST Y
59 ISG ST Y
60 GTO 03
61 R^
62 2
63 +
64 3
65 STO× IND ST Y
66 STO× IND ST T
67▸LBL 00
@ ---------------------------------------------------------
@ Q->P
@ P(X) := Q(X-1)+Q(X+1)+Q(X-I-1)+Q(X-I)+Q(X+I+1)+Q(X+I+2)
@ and P(X) is just Q(X+R+3)
@ ---------------------------------------------------------
@ find I,J of P(M,M) in the (R+4)/2 x (R+3) matrix
@ qmm = Reg(M*(M+1)/2 + M)
@ pmm = Reg(qmm + R+3)
@ J = pmm MOD (R+3) + 1
@ I = (pmm + 1 - J)/(R+3) + 1
68 RCL "M"
69 STO "I"
70 ENTER
71 XEQ 99 @ qmm
72 RCL ST X
73 3
74 RCL+ "R"
75 +
76 RCL ST X
77 LASTX @ R+3 pmm+1 pmm+1 qmm
78 MOD
79 STO- ST Y
80 X<>Y
81 LASTX
82 STO+ ST Y
83 ÷
84 X<>Y
85 1
86 +
87 STOIJ
88 R^
89 RCL- "M"
90 LASTX
91 2
92 +
93 RCL+ "M"
94 LASTX

95▸LBL 04
96 RCL "I"
97 STO "J"
98 DSE ST Y
99▸LBL 05
100 CLX
101 RCL IND ST T
102 RCL+ IND ST Z
103 RCL+ IND ST Y
104 DSE ST T
```

```
105 DSE ST Z
106 DSE ST Y
107 DSE ST Y
108 RCL+ IND ST T
109 RCL+ IND ST Z
110 RCL+ IND ST Y
111 ISG ST Y
112▸LBL 00
113 ←
114 DSE "J"
115 GTO 05
116 DSE ST Z
117 DSE ST Z
118 CLX
119 ←
120 R↓
121 DSE "I"
122 GTO 04

123 I-
124 DELR @ we are at 1,1 now
125 CLX
126 ←
127 → @ GROW mode causes an extra row now
128 RCL "M"
129 RCL "R"
130 X>Y?
131 ISG "M"
132▸LBL 00
133 DSE "K"
134 GTO 20
135 RCLEL
136 EXITALL
137 RCL "R"
138 ENTER
139 ENTER
140 XEQ 99
141 0
142▸LBL 06
143 RCL+ IND ST Y
144 DSE ST Y
145 DSE ST Z
146 GTO 06
147 6
148 RCL "S"
149 Y^X
150 ÷
151 RTN
152▸LBL 99
153 ENTER
154 X^2
155 +
156 2
157 ÷
158 +
159 END
```

Cheers, Werner

EMAIL PM FIND QUOTE REPORT

3rd November, 2022, 00:56 **Post: #84**

Valentin Albillo
Senior Member

Posts: 958
Joined: Feb 2015
Warning Level: 0%

RE: [VA] SRC #012a - Then and Now: Probability

Hi, all,

After 4 weeks to the day, it seems this ***SRC #012a*** has run its course, so time for a few ***additional comments*** and a few ***new results***.

The aditional comments

> **Gjermund Skailand Wrote:**
>
> This has been a very interesting thread. This is a **sys-rpl** version for **HP50g** of CReth SRC12a. On an actual *HP50g* the calculation time for the 30 60 problem is *5 min 21sec*. p= *9.51234350207E-6*

Thank you for your appreciation. Your *SysRPL* version looks *amazing*, kinda assembly language, producing the correct 12-digit result at least *10x faster* than a physical **HP-71B**, which is truly *awesome*.

Can someone please confirm that the listing is correct and will produce the stated result in the stated time ?

> **Albert Chan Wrote:**
>
> For **S = R-1**, we can treat triangle as without bottom edge (and the 2 corners) [...] It simplified to:
>
> $$P(R, S=R-1) = 3^{(3-R)} - 2^{(5-2*R)}$$

Very nice exact *symbolic* result for that particular case, **Albert Chan**, congratulations !

Normally this could be construed as going against my stated rules but as you previously posted *tons* of actual **HP-71B** code, I'm not complaining.

> **Werner Wrote:**
>
> Well, to make Valentin's wishes come true, here's an entry that will solve the 30/60 problem on a real 42S, the only vintage RPN calculator able to do it. [...] Estimate of real 42S running time: 3h05m.

Thank you very much, **Werner**, for taking my wishes into consideration and producing such a fine **HP-42S** solution. Your running time estimation on a physical *HP-42S* seems to be *3x slower* than my solution running on a physical **HP-71B** but, as you say, optimization would probably reduce the timing considerably and some compromises had to be made to fit it into the available *RAM*.

Nevertheless, running your program in **Free42** on my *Samsung* tablet takes just *4.5 seconds*, while still within the rules.

The new results

As I've stated oftentimes, one of my main goals is to get people who like vintage *HP* calculators to not consider them as obsolete gadgets only fit for collecting or nostalgia, with no real place in the real world, but as still useful devices which can indeed be used to solve modern problems and best of all, to improve one's sleuthing and programming abilities while attempting the solution, in the way of *"Experimental Mathematics" (EM for short)*; quoting from *Wikipedia*:

> *"Experimental mathematics is an approach to mathematics in which computation is used to investigate mathematical objects and identify properties and patterns."*

In what follows, I'll describe my own *EM* approach to this *Problem 1*, always using my program (as listed in *Post #50*) to do the sleuthing. First of all we get this assorted data, in `SCI 6` for easier typing:

Rows	Steps	Probability	Rows	Steps	Probability
5	4	7.986111e-2	**20**	20	5.006369e-8
	40	*2.666*558e-1		200	5.204890e-2
	50	*2.6666*60e-1		1000	*6.666*566e-2
	60	*2.66666*6e-1		2000	***6.666667e-2***
	70	***2.666667e-1***			
10	10	1.317953e-3	**30**	30	1.289121e-12
	100	*1.*317596e-1		60	9.512344e-6
	1000	***1.333333e-1***		120	1.694782e-3
				240	1.531651e-2
				300	2.225664e-2
				480	3.539453e-2
				960	***4.3***68177e-2

and it clearly seems that there's a *limit* for the value of the probability ***P*** as the number of steps increases, which is ***P_lim(5)*** = *2.666667e-1* = ***4/15***, ***P_lim(10)*** = ***2/15***, ***P_lim(20)*** = ***1/15*** and though not so fast ***P_lim(30)*** seems to be converging to *4.444444e-2* = ***2/45*** , so recognizing the obvious pattern we might then conjecture that in the limit we have, for ***N*** rows:

P_lim(N) = 4/(3*N)

Now we can check additional cases (say *N = 7, 13, 22, ...* rows) to see if the conjectured formula *holds*, and if it does we can attempt to find a *symbolical* proof for it, ***A. Chan***-style !

So far this applies to the probability of being in the *bottom row* at the end of the walk, but what about the probability in the limit of being in a particular, *single* location as the number of steps grows indefinitely ? Adding this line to my program will display the resulting *probability matrix* which gives the probability for each and every grid point, using the **FRAC$** keyword from the *JPC ROM* to output *exact* rational results:

```
75  FOR I=1 TO M @ FOR J=1 TO I @ DISP FRAC$(A(I,J),5);" "; @ NEXT J @ DISP @ NEXT I
```

Running the program for ***5*** *rows* and a sufficiently large number of steps *(S=**100**)*, we get in **SCI 6**:

```
>RUN
        Rows,Steps=5,100 -> 2.666667e-1

                    1/30
                 1/15 1/15
              1/15   1/10   1/15
           1/15   1/10   1/10   1/15
        1/30   1/15   1/15   1/15   1/30
```

and we see that ***P(corners)*** = ***1/30***, ***P(edges)*** = *1/15* = ***2/30*** and ***P(inner)*** = *1/10* = ***3/30***, so we conjecture that the ratios are

P(corners) : P(edges) : P(inner) = ***1 : 2 : 3***

which ***A. Chan*** also discovered and posted here. As a check, if we run my program for *10 rows*, we'll get ***P(corners)***, ***P(edges)***, ***P(inner)*** = ***1/135***, ***2/135***, *1/45* = ***3/135***, further confirming the **1 : 2 : 3** ratios and allowing us to conjecture a formula for the probabilities for the general ***N***-rows case (which will be discussed and obtained next.)

Now you may be wondering if there's some way to ***automatically*** get the exact *probability matrix* in the limit for a given number ***N*** of rows (for reasonable ***N*** and running times,) and indeed there is a simple procedure we might try out. First create a copy of my program and edit these *five* lines to be as follows:

```
10  DESTROY ALL @ OPTION BASE 1 @ INPUT "Rows=";M @ DIM A(M,M),B(M,M)
15  MAT A=(2/(M*(M+1))) @ W=M-1 @ K=1E-6 @ FOR I=1 TO INF @ MAT B=ZER

70  NEXT Y @ NEXT X @ MAT A=A-B @ DIM A(M*M) @ DISP I;CNORM(A) @ IF RES<K THEN 80
75  DIM A(M,M) @ MAT A=B @ NEXT I
80  FOR I=1 TO M @ FOR J=1 TO I @ DISP FRAC$(B(I,J),6);" "; @ NEXT J @ DISP @ NEXT I
```

When run, the program asks for the number of rows ***N*** and then initializes the matrix probabilities to be the *same* for all grid locations at the very beginning (and of course adding up to ***1***) since the limit probability matrix after infinite steps clearly does *not* depend on the starting position(s) and initially assuming uniformly distributed probabilities greatly speeds up the convergence and accuracy.

Once the initialization is over the program then computes the probability matrix for *steps 1, 2, 3, ...*, comparing each matrix with the previous one. When the difference is less than a hardcoded *tolerance (K=1E-6 at line 15)* the process is over and the limit probability matrix is output in *exact rational* form.

> **Note**: if the rational matrix displayed doesn't look correct (the probabilities don't comply with the *1:2:3* ratios) you can fine-tune the *tolerance* (say `K=1E-7` or smaller, the running time will possibly increase) and/or the *accuracy* in the conversion to rational form (change the parameter ***6*** in **FRAC$** to some other value, say ***7***) and run the program again.

While it runs, the program will display the *step number* and the current *difference* after each step so you can see it converging to zero, and once it meets the tolerance (will take a long while for large enough ***N***) it will display the limit probability matrix in exact rational form. Let's run it for ***N***=***15*** *rows* in **FIX 6**:

```
>FIX 6 @ RUN

      Rows=15

      Step        Difference
      ----------------------
      1.000000     0.991667
      2.000000     0.043056
      3.000000     0.020833
      4.000000     0.012770
      ...
```

```
128.000000   0.000001
129.000000   0.000001

Limit Probability matrix:

1/315
2/315   2/315
2/315   1/105   2/135
    ...
2/315   1/105 ...   1/105   2/315
1/315   2/315 ... ... ...   2/315   1/315
```

and we see that it took ***130 steps*** (not infinite !) to achieve the specified tolerance. The displayed rational *limit probability matrix* is nevertheless *exact*.

Now that we have a working program, we can create a simplified version which just checks the difference of the probability for *only* the *top corner* location *(1,1)* at successive steps, simply comparing *A(1,1)* vs. *B(1,1)* instead of considering the whole matrices, and once the tolerance is met we simply output *B(1,1)*, the top corner's probability. If we run this simpler and faster program for various numbers of rows, we get:

N rows	5	7	9	10	11	12	13	14	15
P(1,1)	1/30	1/63	1/108	1/135	1/165	1/198	1/234	1/273	1/315

and a little experimentation quickly reveals a *pattern* for the *denominators*, e.g.:

P(N=**9**) -> 108 = **9***__24__/2,   P(N=**10**) -> 135 = **10*****27**/2,   P(N=**11**) -> 165 = **11*****30**/2

so we conjecture that the probability in the limit for the top corner of an ***N***-row grid is:

P = 2/(3*N*(N-1))

and taking into account the previously stablished ***1:2:3*** ratio we finally get

P(corners) = 2/(3*N*(N-1)),   P(edges) = 2*P(corners),   P(inner) = 3*P(corners)

and finally we can create a *much simpler* and *faster* program *(5 lines or less in all)* which will accept ***N*** and non-iteratively proceed to immediately display the corresponding limit probability matrix. Checking it for the ***N=30*** *rows* case we get these probabilities:

P(corners) = ***1/1305***,   ***P(edges)*** = ***2/1305*** and   ***P(inner)*** = *3/1305* = ***1/435***

which our final program computes and uses to fill up and output the *exact* probability matrix for the ***30***-row grid in very little time. Doing the same for a ***100,000***-row grid would be equally fast.

Well, I hope this has provided a good example of how you can use your vintage *HP* calc to do some sleuthing and get nice *symbolic* results in the spirit of *Experimental Mathematics*. Once you get the numeric results, conjecturing the symbolic ones and afterwards attempting to prove them is that much easier. :)

Regards.
V.

PM   WWW   FIND   EDIT   QUOTE   REPORT

3rd November, 2022, 13:19   **Post: #85**

Albert Chan
Senior Member

Posts: 2,142
Joined: Jul 2018

RE: [VA] SRC #012a - Then and Now: Probability

> **Valentin Albillo Wrote:** (3rd November, 2022 00:56)
>
> Running the program for ***5*** *rows* and a sufficiently large number of steps *(S=**100**)*, we get in SCI 6:
>
> ```
> >RUN
>        Rows,Steps=5,100 -> 2.666667e-1
>
>                 1/30
>              1/15 1/15
>           1/15   1/10   1/15
>        1/15   1/10   1/10   1/15
>     1/30   1/15   1/15   1/15   1/30
> ```

and we see that ***P(corners)*** = ***1/30***, ***P(edges)*** = *1/15* = ***2/30*** and ***P(inner)*** = *1/10* = ***3/30***, so we conjecture that the ratios are

> ***P(corners) : P(edges) : P(inner)*** = ***1 : 2 : 3***

which ***A. Chan*** also discovered and posted here.

Werner Wrote: (27th October, 2022 16:07)

@ P-> Q, adjust corners and edges
...
@ Q->P
@ P(X) := Q(X-1)+Q(X+1)+Q(X-I-1)+Q(X-I)+Q(X+I+1)+Q(X+I+2)
...

Asymptotic P ratios can be easily derived from the code.
If Q entries are all the same, nextP = P

Say, all entries in Q = k (for above example, k = 1/60), Q -> P:

P(corners) : P(edges) : P(inner) = 2k : 4k : 6k = 1 : 2 : 3

EMAIL PM FIND QUOTE REPORT

4th November, 2022, 17:31 (This post was last modified: 4th November, 2022 17:38 by DavidM.) **Post: #86**

DavidM
Senior Member

Posts: 918
Joined: Dec 2013

RE: [VA] SRC #012a - Then and Now: Probability

Valentin Albillo Wrote: (3rd November, 2022 00:56)

Gjermund Skailand Wrote:
This has been a very interesting thread. This is a **sys-rpl** version for **HP50g** of CReth SRC12a. On an actual *HP50g* the calculation time for the 30 60 problem is *5 min 21sec*. p= *9.51234350207E-6*

Thank you for your appreciation. Your *SysRPL* version looks *amazing*, kinda assembly language, producing the correct 12-digit result at least *10x faster* than a physical **HP-71B**, which is truly *awesome*.

Can someone please confirm that the listing is correct and will produce the stated result in the stated time ?

There were a few typos in Gjermund's posted SysRPL code. I've made some corrections to his version below to produce a program which appears to give the proper results, so hopefully I haven't introduced any new bugs in the process. I can confirm that his code produces the correct real result for 30-60 input in about **317 seconds** on my physical 50g.

His version is really a hybrid mix of SysRPL, Saturn**+** (the plus is important here), and even embedded UserRPL.

The main "get" and "put" commands for arrays in SysRPL require that the index given for the element in question is expressed as a single binary integer. So "x,y" must be converted to a single integer representing the linear index of the element as if the array were actually a vector.

In particular, PULLREALEL and PUTREALEL are the commands which do the dirty work of recalling and storing the array elements. Given that this program (along with the others, of course) depends very heavily on array indexing, anything that can speed up the conversion of x-y coordinates to a single vector index will have a substantial impact on the runtime.

Gjermund has a single subroutine for this conversion ("Tind"), which I suspect he originally wrote in standard SysRPL. Given that it is a critical routine, re-coding that in Saturn assembly made good sense. Not only is it running at Saturn speed, though, he also used a "Saturn+" opcode for squaring a 5-nibble integer. This allows squaring the value in 1 assembly step instead of having to call a subroutine for that purpose. That opcode is only available on the ARM-based RPL calcs, but speeds up the conversion of x,y coordinates into a single index noticeably.

Combine the above with the usual speed increase of SysRPL over UserRPL and you've got a significantly faster program than the original UserRPL version that C.Ret posted.

Here's Gjermund's program with the typos corrected:

```
!RPL
!NO CODE
!JAZZ
::
```

```
  CK2NOLASTWD
  CK&DISPATCH2
  #11
  :: COERCE2
    '
    CODE
      GOSBVL POP2# RSTK=C SAVE
      B=A.A A*A.A A-B.A ASRB.A
      C=RSTK A+C.A
      GOSBVL PUSH#ALOOP
    ENDCODE
    3PICK DUP 3PICK EVAL SWAP 2 SWAPOVER
    {{  r s d m Tind ii }}
    r #1+_ONE_DO (i)
*     INDEX @ #1+_ONE_DO (j)
      INDEX@ #1+_ONE_DO (j)               ( ***CORRECTION*** )
        3
        DUP INDEX@ #1<> ?SKIP #1-
        JINDEX@ INDEX@ #<> ?SKIP #1-
        JINDEX@ r #<> ?SKIP #1-
*       UNCOERCE %/
        UNCOERCE2 %/                      ( ***CORRECTION*** )
    LOOP (j)
  LOOP (i)
  d UNCOERCE ONE{}N FPTR2 ^XEQ>ARRY
  DUP %0 xCON
  %1 BINT1 PUTREALEL
  s #1+_ONE_DO (k)
    INDEX@ #>$ BIGDISPROW1
    2DUP m m 2GETEVAL DUP4UNROLL TOTEMPOB (n)
    #1+_ONE_DO (q)
      SWAP INDEX@ PULLREALEL
      ROT INDEX@ PULLREALEL
      ROT %* 4UNROLL
    LOOP (q)
    2DROP
    UNCOERCE ONE{}N x>ARRY
    m #1+_ONE_DO (i)
      INDEX@  #2 #/ #+ #1+_ONE_DO (j)
        JINDEX@ INDEX@ 2GETEVAL
        PULLREALEL
        %CHS
        JINDEX@ TOTEMPOB !ii
        1 JINDEX@ #1- #MAX
        JINDEX@ #1+ m #MIN
        #1+ SWAP DO (a)
*         1 JINDEX@ INDEX@ ii #> ?SKIP #1- MAX
          1 JINDEX@ INDEX@ ii #> ?SKIP #1- #MAX         ( ***CORRECTION*** )
*         JINDEX@ ii  INDEX@ #> ?SKIP #1+ INDEX@ MIN
          JINDEX@ ii  INDEX@ #> ?SKIP #1+ INDEX@ #MIN   ( ***CORRECTION*** )
          #1+SWAP DO (b)
            SWAP JINDEX@ INDEX@ 2GETEVAL PULLREALEL
            ROT %+
          LOOP (b)
        LOOP (a)
        ROT
        JINDEX@ INDEX@ 2GETEVAL
        3PICKSWAP PUTREALEL
*       JINDEX@ #1+ INDEX@ #-
        JINDEX@ DUP #1+ INDEX@ #-                        ( ***CORRECTION*** )
        2GETEVAL
        ROTSWAP
        PUTREALEL
        SWAP
      LOOP (j)
     LOOP (i)
     DROP
     m DUP r #>=_ ?SKIP #1+ !m
    LOOP (k)
    SWAP %0 xCON
    r 1 2GETEVAL
```

```
      UNCOERCE
      xDO %1 xPUTI xUNTIL
      % -64 xFS?
      xENDDO
      xDROP
      xDOT
      %6
      s UNCOERCE
      x^
      x/
      ABND
    ;
  ;
  @
```

PM FIND QUOTE REPORT

5th November, 2022, 00:51 **Post: #87**

Valentin Albillo
Senior Member

Posts: 958
Joined: Feb 2015
Warning Level: 0%

RE: [VA] SRC #012a - Then and Now: Probability

.
Hi, **DavidM**,

> **DavidM Wrote:**
>
> > **Valentin Albillo Wrote:**
> >
> > Can someone please confirm that the listing is correct and will produce the stated result in the stated time ?
>
> There were a few typos in Gjermund's posted SysRPL code. I've made some corrections to his version below to produce a program which appears to give the proper results, so hopefully I haven't introduced any new bugs in the process. I can confirm that his code produces the correct real result for 30-60 input in about **317 seconds** on my physical 50g.

As I said, truly impressive, about *10x* faster than a physical **HP-71B** running my original *BASIC* program. I guess that only an assembly language version of it would be able to approach the *50g's* performance, perhaps ***J-F Garnier*** would obligue, it's certainly within his outstanding capabilities ... :)

> **Quote:**
>
> Combine the above with the usual speed increase of SysRPL over UserRPL and you've got a **significantly faster program** than the original UserRPL version that C.Ret posted.

Indeed !

> **Quote:**
>
> Here's Gjermund's program **with the typos corrected**: [...]

Fantastic ! It exceeds my best expectations of having a *"certified"* version of the *SysRPL* program, both for correct listing and accurate running time, a real gem of reliability and efficiency.

Thank you very, very much, **DavidM**, for taking the trouble to fulfill my request, and for your really detailed explanations of its inner workings, never mind the corrections and improvements. Posts like yours is what adds the most value to these humble threads of mine and makes them everlasting contributions to the art of programming these wonderful vintage *HP* calcs, for the benefit of all of us. *Much, much appreciated !*

Best regards.
V.

PM WWW FIND EDIT QUOTE REPORT

17th February, 2023, 07:49 (This post was last modified: 17th February, 2023 07:50 by kostrse.) **Post: #88**

kostrse
Junior Member

Posts: 10
Joined: Mar 2021

RE: [VA] SRC #012a - Then and Now: Probability

> **DavidM Wrote:** (4th November, 2022 17:31)
>
> Here's Gjermund's program with the typos corrected:
>
> ```
> !RPL
> !NO CODE
> !JAZZ
> ::
>   CK2NOLASTWD
>   CK&DISPATCH2
>   #11
>   :: COERCE2
>     '
>     CODE
>       GOSBVL POP2# RSTK=C SAVE
>       B=A.A A*A.A A-B.A ASRB.A
>       C=RSTK A+C.A
>       GOSBVL PUSH#ALOOP
>     ENDCODE
>     ...
> @
> ```

Sorry, I'm trying to compile it on my HP-50G and it complains at the **GOSBVL** (Invalid Syntax). Is it a SysRPL or Assembly instruction or what?
What should I do to make it work on a stock calculator?

PM FIND QUOTE REPORT

18th February, 2023, 18:25 **Post: #89**

DavidM
Senior Member

Posts: 918
Joined: Dec 2013

RE: [VA] SRC #012a - Then and Now: Probability

I responded to your questions in a separate post in order to avoid hijacking Valentin's original thread with implementation-specific details.

PM FIND QUOTE REPORT

User(s) browsing this thread: Valentin Albillo*