Use Mathematica to Solve the Sum and Product Puzzle

Chuanlong Du

Department of Statistics, Iowa State University

April 30, 2013

Outline

1. Introduction of the Sum and Product Puzzle
2. Solve the puzzle using Mathematica
3. Some discussion about the problem
4. Why Mathematica is a good choice for such problems
5. Questions

A Popular Version of the Sum and Product Puzzle

Two numbers (not necessarily unique) between 2 and 99 are chosen. The *sum* of them is told to *Sam* and the *product* of them is told to *Peter*.

Sam: "Now *I don't know what the 2 numbers are*, but I'm sure *you don't know either*."

Peter: "I have to thank you for the information, because *I did have no idea of what the 2 numbers are*, but *now I already know*."

Sam: "*Now the same here*."

Question: what are the two numbers?

Solve the Puzzle as a Sophomore

1. solve the Sum and Product Puzzle logically is too hard (also called the Impossible Puzzle), so I decide to write a program to solve the Puzzle
2. a sophomore (almost 8 years ago) who knew C, MATLAB and *Mathematica*
3. only took me around an hour to write the code even I was a newbie in Mathematica

Some Notation for Solving the Puzzle

- R: range of the 2 numbers, which is $\{2,3,\ldots,99\}$ in this case
- x_0, y_0: a (the) solution to the Puzzle

Mathematical Information Hidden in Words

"Now *I don't know what the 2 numbers are*, but I'm pretty sure *you don't know either*."

1. $\exists$ multiple pairs of $x, y \in R$ such that $x + y = x_0 + y_0$
2. for each pair of $x, y \in R$ such that $x + y = x_0 + y_0$, $\exists$ multiple pairs of $x', y' \in R$ such that $x' \times y' = x \times y$

Let's called the above conditions set I.

Mathematical Information Hidden in Words

"I have to thank you for the information, because *I did have no idea of what the 2 numbers are*, but *now I already know*."

1. $\exists$ multiple pairs of $x, y \in R$ such that $x \times y = x_0 \times y_0$ (already in conditions set I)
2. among all pairs $x, y \in R$ such that $x \times y = x_0 \times y_0$, $\exists$ an unique pair satisfying conditions set I

Let's called the above conditions set II.

Mathematical Information Hidden in Words

"*Now the same here*."

1. $\exists$ multiple pairs of $x, y \in R$ such that $x + y = x_0 + y_0$ (already in conditions set I)
2. among all pairs $x, y \in R$ such that $x + y = x_0 + y_0$, $\exists$ an unique pair satisfying conditions set II

Let's called the above conditions set III.

Algorithm to Solve the Sum and Product Puzzle

A/The solution (pair of x_0 and y_0) must satisfies conditions set I, II and III at the same time.

1. construct all possible combinations of $x, y \in R$
2. select pairs (among all possible pairs) that satisfy conditions set I, II and III at the same time

Code for the Sum and Product Puzzle

```
TwoAddends[s_Integer, range_List] := Module[{lower, upper},
   lower = range[[1]];
   upper = range[[2]];
   Table[{i, s - i}, {i, Max[lower, s - upper], Min[upper, s - lower, s/2]}]
];
```

Code for the Sum and Product Puzzle

```
TwoFactors[p_Integer, range_List] := Module[{lower, upper, div, n},
   lower = range[[1]];
   upper = range[[2]];
   div = Select[Divisors[p], # >= Max[lower, p/upper] && # <= Min[upper, p/lower, Sqrt[p]] &];
   Map[{#, p/#} &, div]
];
```

Code for the Sum and Product Puzzle

```
S1[pair_List, range_List] := Module[{s, candidates},
   s = Total[pair];
   candidates = TwoAddends[s, range];
   Length[candidates] > 1 && Apply[And, Length[TwoFactors[Times @@ #, range]] > 1 & /@ candidates]
];
```

Code for the Sum and Product Puzzle

```
P1[pair_List, range_List] := Module[{p, candidates},
   p = Times @@ pair;
   candidates = TwoFactors[p, range];
   Length[candidates] > 1 && Total[Boole[S1[#, range] & /@ candidates]] == 1
];
```

Code for the Sum and Product Puzzle

```
S2[pair_List, range_List] := Module[{s, candidates},
   s = Total[pair];
   candidates = TwoAddends[s, range];
   Length[candidates] > 1 && Total[Boole[P1[#, range] & /@ candidates]] == 1
];
```

Code for the Sum and Product Puzzle

```
SumProductPuzzle[range_List] := Module[{lower, upper, candidates},
   lower = range[[1]];
   upper = range[[2]];
   candidates = Flatten[Table[{i, j}, {i, lower, upper}, {j, i, upper}], 1];
   Select[candidates, S1[#, range] && P1[#, range] && S2[#, range] &]
]

SumProductPuzzle[{2, 99}]
{{4, 13}}
```

Some Discussions about the Sum and Product Puzzle

Use the following code to do computation in parallel

```
DistributeDefinitions[TwoAddends, TwoFactors, S1, P1, S2, SumProductPuzzle]
rr = Table[ParallelSubmit[{i}, SumProductPuzzle[{2, i}]], {i, 61, 600}]
WaitAll[rr]
```

1. the solution depends on the range (can have no, unique or multiple answers)
2. an unique solution exists for [2, 62] to [2, 500+]

Why Mathematica is Good for This Kind of Problems

1. support vector/list operations (Apply, Map and so on)
2. lots of useful built-in list manipulation functions (Table, Flatten, Select and so on)
3. lots of useful built-in math (number theory related) functions (Divisors and so on)
4. easy parallel computing

More things I Did Using Mathematica

http://dclong.github.io/en/blog/categories/mathematica/

Questions

Any Question?