Backtracking | Set 8 (Solving Cryptarithmetic Puzzles)

Newspapers and magazines often have crypt-arithmetic puzzles of the form:

```
  SEND
+ MORE
--------
 MONEY
--------
```

The goal here is to assign each letter a digit from 0 to 9 so that the arithmetic works out correctly. The rules are that all occurrences of a letter must be assigned the same digit, and no digit can be assigned to more than one letter.

- First, create a list of all the characters that need assigning to pass to Solve
- If all characters are assigned, return true if puzzle is solved, false otherwise
- Otherwise, consider the first unassigned character
- for (every possible choice among the digits not in use)

  make that choice and then recursively try to assign the rest of the characters
  if recursion sucessful, return true
  if !successful, unmake assignment and try another digit

- If all digits have been tried and nothing worked, return false to trigger backtracking

```
/* ExhaustiveSolve
* ---------------
* This is the "not-very-smart" version of cryptarithmetic solver. It t
* the puzzle itself (with the 3 strings for the two addends and sum) a
```


```
* string of letters as yet unassigned. If no more letters to assign
* then we've hit a base-case, if the current letter-to-digit mapping s
* the puzzle, we're done, otherwise we return false to trigger backtra
* If we have letters to assign, we take the first letter from that lis
* try assigning it the digits from 0 to 9 and then recursively working
* through solving puzzle from here. If we manage to make a good assign
* that works, we've succeeded, else we need to unassign that choice an
* another digit. This version is easy to write, since it uses a simple
* approach (quite similar to permutations if you think about it) but i
* not so smart because it doesn't take into account the structure of t
* puzzle constraints (for example, once the two digits for the addends
* been assigned, there is no reason to try anything other than the cor
* digit for the sum) yet it tries a lot of useless combos regardless
*/
bool ExhaustiveSolve(puzzleT puzzle, string lettersToAssign)
{
    if (lettersToAssign.empty()) // no more choices to make
        return PuzzleSolved(puzzle); // checks arithmetic to see if wo
    for (int digit = 0; digit <= 9; digit++)   // try all digits
    {
        if (AssignLetterToDigit(lettersToAssign[0], digit))
        {
            if (ExhaustiveSolve(puzzle, lettersToAssign.substr(1)))
                return true;
            UnassignLetterFromDigit(lettersToAssign[0], digit);
        }
    }
    return false;  // nothing worked, need to backtrack
}
```

The algorithm above actually has a lot in common with the permutations algorithm, it pretty much just creates all arrangements of the mapping from characters to digits and tries each until one works or all have been successfully tried. For a large puzzle, this could take a while.
A smarter algorithm could take into account the structure of the puzzle and avoid going down dead-end paths. For example, if we assign the characters starting from the ones place and moving to the left, at each stage, we can verify the correctness of what we have so far before we continue onwards. This definitely complicates the code but leads to a tremendous improvement in efficiency, making it much more feasible to solve large puzzles.

Below pseudocode in this case has more special cases, but the same general design

- Start by examining the rightmost digit of the topmost row, with a carry of 0
- If we are beyond the leftmost digit of the puzzle, return true if no carry, false otherwise

Popular Posts

- If we are currently trying to assign a char in one of the addends
  If char already assigned, just recur on row beneath this one, adding value into sum
  If not assigned, then
    - for (every possible choice among the digits not in use)
      make that choice and then on row beneath this one, if successful, return true
      if !successful, unmake assignment and try another digit
    - return false if no assignment worked to trigger backtracking
- Else if trying to assign a char in the sum
- If char assigned & matches correct,
  recur on next column to the left with carry, if success return true,
- If char assigned & doesn't match, return false
- If char unassigned & correct digit already used, return false
- If char unassigned & correct digit unused,
  assign it and recur on next column to left with carry, if success return true
- return false to trigger backtracking

Source:
http://see.stanford.edu/materials/icspacs106b/H19-RecBacktrackExamples.pdf

Related Tpoics:

- Tail Recursion
- Find if two rectangles overlap
- Analysis of Algorithm | Set 4 (Solving Recurrences)
- Print all possible paths from top left to bottom right of a mXn matrix
- Generate all unique partitions of an integer
- Russian Peasant Multiplication
- Closest Pair of Points | O(nlogn) Implementation
- Find the maximum distance covered using n bikes

Writing code in comment? Please use **ideone.com** and share the link here.

Recent Comments

Abhi You live US or India?

Google (Mountain View) interview · 8 minutes ago

Aman Hi, Why arent we checking for conditions...

Write a C program to Delete a Tree. · 47 minutes ago

kzs please provide solution for the problem...

2 Comments GeeksforGeeks

Sort by Newest

Join the discussion…

Pavi • 2 months ago

I couldn't find out where to raise this issue, hence landing here. Sorry! for that. I want to suggest a small change in your site in the way the ad page opens in s Request you to change your site's code so that URLs when clicked (especially or window and not on the same page. Thanks! if you consider this.

2

rajeev • 2 months ago

great, could you provide complete implementation?

Subscribe Add Disqus to your site

Backtracking | Set 2 (Rat in a Maze) · 51 minutes ago

Sanjay Agarwal bool tree::Root_to_leaf_path_given_sum(tree...

Root to leaf path sum equal to a given number · 1 hour ago

GOPI GOPINATH @admin Highlight this sentence "We can easily...

Count trailing zeroes in factorial of a number · 1 hour ago

newCoder3006 If the array contains negative numbers also. We...

Find subarray with given sum · 1 hour ago