1. Problem 5.6 in the book

Solve the cryptarithmetic problem in Figure 5.2 by hand, using backtracking, forward checking, and the MRV and least-constraining-value heuristics. (10)

Most students didn’t use MRV and LCV correctly but full grade is given when it is roughly correct.

+ Variables: \( \{T, W, O, F, U, R, X1, X2, X3\} \)
+ Domain for \( \{T, F\} = \{1, 2, 3, 4, 5, 6, 7, 8, 9\} \) since no leading zeros are allowed.
+ Domain for \( \{W, O, U, R\} = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\} \)
+ Domain for \( \{X1, X2, X3\} = \{0, 1\} \) since they represent carries.
+ Constraints:
  1. \( \text{Alldiff}(T, W, O, F, U, R) \)
  2. \( O + O = R + 10 \times X1 \)
  3. \( X1 + W + W = U + 10 \times X2 \)
  4. \( X2 + T + T = O + 10 \times X3 \)
  5. \( X3 = F \)

+ Start backtracking

(1)

<table>
<thead>
<tr>
<th>Current Assignment</th>
<th>T</th>
<th>W</th>
<th>O</th>
<th>F</th>
<th>U</th>
<th>R</th>
<th>X1</th>
<th>X2</th>
<th>X3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Remaining Values</td>
<td>1~9</td>
<td>0~9</td>
<td>0~9</td>
<td>1~9</td>
<td>0~9</td>
<td>0~9</td>
<td>0,1</td>
<td>0,1</td>
<td>0,1</td>
</tr>
</tbody>
</table>

Choose X1 by MRV.

If \( X1 = 0 \), O would be \( 0~4 \) by Constraint (2) and U would be an even number by Constraint (3). Otherwise, O would be \( 5~9 \) and U would be an odd number.

Choose the value for X1 in \( 0, 1 \) randomly since 0 and 1 have same LCV and both of them would survive forward checking.

Then R would be an even number by forward checking Constraint (2).

(2)

<table>
<thead>
<tr>
<th>Current Assignment</th>
<th>X1 = 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Remaining Values</td>
<td>T</td>
</tr>
<tr>
<td></td>
<td>1~9</td>
</tr>
</tbody>
</table>

Choose X2 by MRV.

If \( X2 = 0 \), W would be \( 0~4 \) by Constraint (3) and O would be \( 0, 2, 4 \) by Constraint (4). Otherwise, W would be \( 5~9 \) and O would be \( 1, 3 \).

Choose \( X2 = 0 \) by LCV.

(3)

<table>
<thead>
<tr>
<th>Current Assignment</th>
<th>X1 = 0, X2 = 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Remaining Values</td>
<td>T</td>
</tr>
<tr>
<td></td>
<td>1~9</td>
</tr>
</tbody>
</table>
Then T would be (5,6,7) and F would be 1 by forward checking Constraints (4) and (5).

(4)

<table>
<thead>
<tr>
<th>Current Assignment</th>
<th>X1 = 0, X2 = 0, X3 = 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Remaining Values</td>
<td>T  W   O   F   U   R   X1   X2   X3</td>
</tr>
<tr>
<td></td>
<td>5,6,7 0~4 0,2,4 1 0,2,4,6,8 0,2,4,6,8</td>
</tr>
</tbody>
</table>

Choose F by MRV.
Choose F = 1 since it would survive forward checking.
Then 1 is removed from the domain of W.

(5)

<table>
<thead>
<tr>
<th>Current Assignment</th>
<th>X1 = 0, X2 = 0, X3 = 1, F = 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Remaining Values</td>
<td>T  W   O   F   U   R   X1   X2   X3</td>
</tr>
<tr>
<td></td>
<td>5,6,7 0,2,3,4 0,2,4,6,8 0,2,4,6,8</td>
</tr>
</tbody>
</table>

Choose T by MRV.
If T = 5, O would be 0, if T = 6, O would be 2, and if T = 7, O would be 4.
Choose the value for T in (5,6,7) randomly since they have same LCV and would survive forward checking.

(6)

<table>
<thead>
<tr>
<th>Current Assignment</th>
<th>X1 = 0, X2 = 0, X3 = 1, F = 1, T = 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Remaining Values</td>
<td>T  W   O   F   U   R   X1   X2   X3</td>
</tr>
<tr>
<td></td>
<td>0,2,3,4 0 0,2,4,6,8 0,2,4,6,8</td>
</tr>
</tbody>
</table>

Choose O by MRV.
If O = 0, R would be 0 and it conflicts with Constraint (1).
Backtrack to (5) because there is no consistent value for O.

(7)

<table>
<thead>
<tr>
<th>Current Assignment</th>
<th>X1 = 0, X2 = 0, X3 = 1, F = 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Remaining Values</td>
<td>T  W   O   F   U   R   X1   X2   X3</td>
</tr>
<tr>
<td></td>
<td>6,7 0,2,3,4 0,2,4,6,8 0,2,4,6,8</td>
</tr>
</tbody>
</table>

Choose T by MRV and T = 6.
Then O would be 2 by forward checking Constraint (4) and we remove 6 in the domains of U and R by forward checking Constraint (1).

(8)

<table>
<thead>
<tr>
<th>Current Assignment</th>
<th>X1 = 0, X2 = 0, X3 = 1, F = 1, T = 6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Remaining Values</td>
<td>T  W   O   F   U   R   X1   X2   X3</td>
</tr>
<tr>
<td></td>
<td>0,2,3,4 2 0,2,4,8 0,2,4,8</td>
</tr>
</tbody>
</table>

Choose O by MRV and O = 2.
Then R would be 4 and we remove 2 in the domains of W and U by forward checking Constraints (1) and (2)

(9)

<table>
<thead>
<tr>
<th>Current Assignment</th>
<th>X1 = 0, X2 = 0, X3 = 1, F = 1, T = 6, O = 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Remaining Values</td>
<td>T  W   O   F   U   R   X1   X2   X3</td>
</tr>
<tr>
<td></td>
<td>0,3,4 0,4,8 4</td>
</tr>
</tbody>
</table>

Choose R by MRV and R = 4.
Then 4 is removed from the domains of W and U.
Choose \( U \) by MRV.

If \( U = 0 \), \( W \) would be 0 by Constraint (3) and otherwise, there is no consistent value for \( W \).

Choose \( U = 0 \), and then 0 is removed from the domain of \( W \) by Constraint (1).

Choose \( W \) but there is no consistent value for \( W \).

Thus backtrack to (9).

Choose \( T \) by MRV and \( T = 7 \).

Then, by forward checking, \( O = 4 \).

Choose \( O \) by MRV and \( O = 4 \).

Then \( R \) would be 8 and 4 is removed from the domains of \( W \) and \( U \).

Choose \( R \) by MRV and \( R = 8 \).

Then 8 is removed from the domain of \( U \) by Constraint (1).
Choose \( W \) by MRV.
If \( W = 0 \), \( U \) would be 0, if \( W = 3 \), \( U \) would be 6, and if \( W = 2 \), \( U \) would have no consistent value.
Choose \( W = 0 \).
Then 0 is removed from the domain of \( U \).

\[(17)\]

Choose \( U \) but there is no consistent value for \( U \).
Thus backtrack to (16)

\[(18)\]

Choose \( W \) by MRV.
Choose \( W = 3 \) since it would survive forward checking.
Then \( U \) would be 6 by Constraint (3).

\[(19)\]

Choose \( U = 6 \).

\[(20)\]

Since \( 734 + 734 = 1468 \), the assignment is consistent and the algorithm finishes.
2. Three employees in a small company, Alice, Rob and Ian must contact their respective customers as quickly as possible. The company has one telephone, one fax and one computer (for email), with independent lines. Alice needs to contact two customers: one who only has a telephone and the second one has both a telephone and a fax. Rob must contact a customer who has also has a telephone and a fax. Ian’s customer can be reached by fax or by computer. Suppose that Alice contacts her customers one after the other while, during the same time, Rob and Ian are communicating with their customers. (20) There are various solutions for the problem because the problem is not clear. Thus full grade is given when you correctly solve based on own assumptions.

a. Model this problem as a CSP. (Choose the communications as variables and the communication media as values.) State the variables, values, and constraints. (5)

b. Draw the constraint graph of this CSP. (3)

+ Variables: \{A1, A2, R, I\}
  A1: Alice’s 1st customer
  A2: Alice’s 2nd customer
  R: Rob’s customer
  I: Ian’s customer
+ Domain for A1 = \{T\}
+ Domain for \{A2, R\} = \{T, F\}
+ Domain for I = \{F, C\}
  T: telephone
  F: fax
  C: computer
+ Constraints:
  (1) Alldiff(A1, R, I)
  (2) Alldiff(A2, R, I)

+ Constraint Graph

\[
\begin{align*}
A1 & \rightarrow A2 & R & \rightarrow I \\
R & \rightarrow A1 & A2 & \rightarrow I
\end{align*}
\]

c. Apply arc-consistency, and describe which values are eliminated at each step. Enumerate the possible solutions. (5)
+ AC-3 algorithm

(1)

<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Domain</td>
<td>A1</td>
</tr>
<tr>
<td></td>
<td>T</td>
</tr>
</tbody>
</table>

check (A1, R) : consistent
check (R, A1) : inconsistent
remove T from the domain of R
push (A2, R) (I, R) (A1, R) : (A2, R) (I, R) already exist.

(2)

<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Domain</td>
<td>A1</td>
</tr>
<tr>
<td></td>
<td>T</td>
</tr>
</tbody>
</table>

check (A1, I) : consistent
check (I, A1) : consistent
check (A2, R) : inconsistent,
remove F from the domain of A2
push (I, A2) (R, A2) : both of them already exist.

(3)

<table>
<thead>
<tr>
<th>Queue</th>
<th>(R, A2) (A2, I) (I, A2) (R, I) (I, R) (A1, R)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Domain</td>
<td>A1</td>
</tr>
<tr>
<td></td>
<td>T</td>
</tr>
</tbody>
</table>
check (R, A2) : consistent  
check (A2, I) : consistent  
check (I, A2) : consistent  
check (R, I) : consistent   
check (I, R) : inconsistent,  
remove F from the domain of I  
push (A1, I) (A2, I) (R, I) : both of them already exist.

(4)

<table>
<thead>
<tr>
<th>Queue</th>
<th>(A1, R) (A1, I) (A2, I) (R, I)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Domain</td>
<td>A1</td>
</tr>
<tr>
<td></td>
<td>T</td>
</tr>
</tbody>
</table>
check (A1, R) : consistent  
check (A1, I) : consistent  
check (A2, I) : consistent  
check (R, I) : consistent

(5)

<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Domain</td>
<td>A1</td>
</tr>
<tr>
<td></td>
<td>T, C</td>
</tr>
</tbody>
</table>
Since queue is empty, the algorithm finishes.

d. Suppose that the first customer Alice needs to communicate with has also a computer. What happens when we try to apply arc-consistency again? (2)

+ AC-3 algorithm

<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Domain</td>
<td>A1</td>
</tr>
<tr>
<td></td>
<td>T, C</td>
</tr>
</tbody>
</table>
check (A1, R) : consistent  
check (R, A1) : consistent  
check (A1, I) : consistent  
check (I, A1) : consistent  
check (A2, R) : consistent  
check (R, A2) : consistent  
check (A2, I) : consistent  
check (I, A2) : consistent  
check (R, I) : consistent  
check (I, R) : consistent  
Since queue is empty, the algorithm finishes.
All arcs in the queue are consistent and the domains of all variables have not been reduced at all.

e. How to solve this new problem? Propose two solutions. (5)
We can solve this new problem
+ using backtracking method with forward checking, MRV, and LCV, and
+ using k-consistency where k is greater than 2.
3. Problem 6.1 in the book

This problem exercises the basic concepts of game playing, using tic-tac-toe (noughts and crosses) as an example. We define $X_n$ as the number of rows, columns, or diagonals with exactly $n$ X’s and no O’s. Similarly, $O_n$ is the number of rows, columns, or diagonals with just $n$ O’s. The utility function assigns +1 to any position with $X_3 = 1$ and −1 to any position with $O_3 = 1$. All other terminal positions have utility 0. For nonterminal positions, we use a linear evaluation function defined as

$$
Eval(s) = 3X_2(s) + X_1(s) - (3O_2(s) + O_1(s)).
$$

(20)

a. Approximately how many possible games of tic-tac-toe are there? (5)

-2 is given when there is no explanation about the solution.

+ Simplistically, there are 362,880 (ie. 9!) ways of placing Xs and Os on the board, without regard to winning combinations.

+ When winning combinations are considered, there are 255,168 possible games. Assuming that X makes the first move every time:

131,184 finished games are won by (X)
1,440 are won by (X) after 5 moves
47,952 are won by (X) after 7 moves
81,792 are won by (X) after 9 moves
77,904 finished games are won by (O)
5,328 are won by (O) after 6 moves
72,576 are won by (O) after 8 moves
46,080 finished games are drawn

+ Ignoring the sequence of Xs and Os, and after eliminating symmetrical outcomes (ie. rotations and/or reflections of other outcomes), there are only 138 unique outcomes. Assuming once again that X makes the first move every time:

  91 unique outcomes are won by (X)
  21 won by (X) after 5 moves
  58 won by (X) after 7 moves
  12 won by (X) after 9 moves

44 unique outcomes are won by (O)
21 won by (O) after 6 moves
23 won by (O) after 8 moves
3 unique outcomes are drawn

from Wikipedia

b. Show the whole game tree starting from an empty board down to depth 2 (i.e., one X and O on the board), taking symmetry into account. (4)

c. Mark on your tree the evaluations of all the positions at depth 2. (3)

d. Using the minimax algorithm, mark on your tree the backed-up values for the positions at depth 1 and 0, and use those values to choose the best starting move. (4)

e. Circle the nodes at depth 2 that would not be evaluated if alpha-beta pruning were applied, assuming the nodes are generated in the optimal order for alpha-beta pruning. (4)

#3-1=2 : the evaluations

1 : the backed-up value
the order of alpha-beta pruning
4. The attempts to build an intelligent computer player for the game of “Go” has not produced good results compared to other games such as chess or checkers. Do some research and write an insightful paragraph (~250 words) about this. Your research can be about any of the following:

a. Why “Go” is harder than chess or other two-player games
b. An algorithm for “Go” that has produced promising results
c. A novel or clever heuristic or technique that shows promising results
d. Any other interesting topic about computational models of “Go”

You can look at the following survey paper as a starting point from which to find other papers. You must have at least one other paper as your reference. You must explicitly state your references. (20)

15 is given when the solution is only for one of 4 sub-problems in the above.
18 is given when the solution is for more than two of 4 sub-problems in the above.
20 is given when you did precise research.