INDIAN INSTITUTE OF TECHNOLOGY KHARAGPUR

Stamp / Signature of the Invigilator

EXAMINATION (Mid Semester)										**SEMESTER (Autumn)**			
Roll Number										**Section**		**Name**	
Subject Number	C	S	6	0	0	4	5	**Subject Name**		***Artificial Intelligence***			
Department / Center of the Student											**Additional sheets**		

Important Instructions and Guidelines for Students

1. You must occupy your seat as per the Examination Schedule/Sitting Plan.
2. Do not keep mobile phones or any similar electronic gadgets with you even in the switched off mode.
3. Loose papers, class notes, books or any such materials must not be in your possession, even if they are irrelevant to the subject you are taking examination.
4. Data book, codes, graph papers, relevant standard tables/charts or any other materials are allowed only when instructed by the paper-setter.
5. Use of instrument box, pencil box and non-programmable calculator is allowed during the examination. However, exchange of these items or any other papers (including question papers) is not permitted.
6. Write on both sides of the answer script and do not tear off any page. **Use last page(s) of the answer script for rough work.** Report to the invigilator if the answer script has torn or distorted page(s).
7. It is your responsibility to ensure that you have signed the Attendance Sheet. Keep your Admit Card/Identity Card on the desk for checking by the invigilator.
8. You may leave the examination hall for wash room or for drinking water for a very short period. Record your absence from the Examination Hall in the register provided. Smoking and the consumption of any kind of beverages are strictly prohibited inside the Examination Hall.
9. Do not leave the Examination Hall without submitting your answer script to the invigilator. **In any case, you are not allowed to take away the answer script with you.** After the completion of the examination, do not leave the seat until the invigilators collect all the answer scripts.
10. During the examination, either inside or outside the Examination Hall, gathering information from any kind of sources or exchanging information with others or any such attempt will be treated as '**unfair means**'. Do not adopt unfair means and do not indulge in unseemly behavior.

Violation of any of the above instructions may lead to severe punishment.

Signature of the Student

To be filled in by the examiner

Question Number	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**Total**
Marks Obtained											

Marks obtained (in words)	**Signature of the Examiner**	**Signature of the Scrutineer**

CS60045 Artificial Intelligence, Autumn 2023–2024

Mid-Semester Test

25–September–2023 | 02:00pm–04:00pm | Maximum marks: 60

[*Write your answers in the question paper itself. Be brief and precise. Answer all questions.*]

1. [Heuristic search]

(a) For a graph-search problem where every transition has a positive cost, prove/disprove each of the following three assertions. Supply proper justifications.

(i) *The path returned by uniform-cost search (Dijkstra's algorithm) may change if we add a positive constant C to the cost of every transition.* **(2)**

Solution *True.* Consider that there are two paths from the start state (S) to the goal (G), denoted as $S \to A \to G$ and $S \to G$. Let $cost(S,A) = 1$, $cost(A,G) = 1$, and $cost(S,G) = 3$. So the optimal path is through A. Now, if we add 2 to each of the costs, the optimal path is directly from S to G. Since the uniform-cost search finds the optimal path, the path output by the algorithm will change.

(ii) *Suppose that A* search produces an optimal solution for a state-space graph. Then, the heuristic used in the search must be admissible.* **(1)**

Solution *False.* Consider a graph with start S, goal G, and another node A. The graph has the three edges $S \to G$, $S \to A$ and $A \to G$, each of cost 1. Take $h(S) = 1$, $h(A) = 2$, and $h(G) = 0$.

(iii) *If $h_1(n)$ and $h_2(n)$ are two admissible A* heuristics for a given state-space search instance, then the average $h_{avg}(n) = \frac{1}{2}\Big(h_1(n) + h_2(n)\Big)$ (for all nodes n) must also be admissible.* **(2)**

Solution *True*. Let $h^*(n)$ be the actual shortest distance from a node n to the goal. We know that $h_1(n) \leqslant h^*(n)$ and $h_2(n) \leqslant h^*(n)$, so $h_{avg}(n) = \frac{1}{2}\Big(h_1(n) + h_2(n)\Big) \leqslant \frac{1}{2}h^*(n) + \frac{1}{2}h^*(n) = h^*(n)$.

(b) If $h_1(n)$, $h_2(n)$, and $h_3(n)$ are all admissible A* heuristics for a given state-space search instance, then which of the following are also guaranteed to be admissible heuristics? Supply proper justifications. **(4)**

(1) $h_1(n) + h_2(n) + h_3(n)$

(2) $\frac{1}{6}h_1(n) + \frac{1}{3}h_2(n) + \frac{1}{2}h_3(n)$

(3) $\min\left[h_1(n), h_2(n), h_3(n)\right]$

(4) $\max\left[h_1(n), h_2(n), h_3(n)\right]$

(5) $\frac{1}{3}h_1(n) + \frac{1}{3}h_2(n) + \frac{1}{3}h_3(n)$

(6) $h_1(n) \cdot h_2(n) \cdot h_3(n)$

(7) $\min\left[h_1(n), h_2(n) + h_3(n)\right]$

(8) $\max\left[h_1(n), h_2(n) + h_3(n)\right]$

Solution The admissible heuristics are highlighted in GREEN, and the non-admissible ones are shaded in RED.

(1) $h_1(n) + h_2(n) + h_3(n)$

(2) $\frac{1}{6}h_1(n) + \frac{1}{3}h_2(n) + \frac{1}{2}h_3(n)$

(3) $\min\left[h_1(n), h_2(n), h_3(n)\right]$

(4) $\max\left[h_1(n), h_2(n), h_3(n)\right]$

(5) $\frac{1}{3}h_1(n) + \frac{1}{3}h_2(n) + \frac{1}{3}h_3(n)$

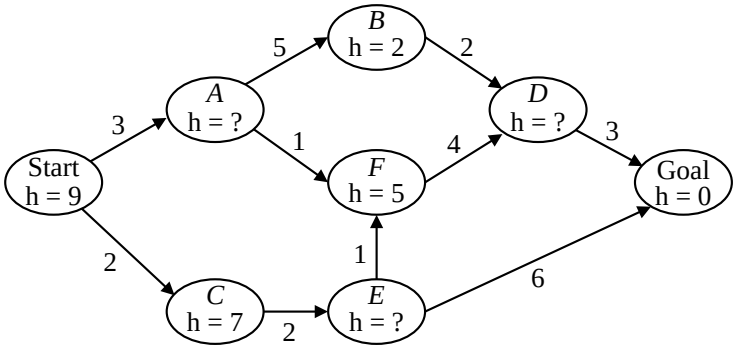
(6) $h_1(n) \cdot h_2(n) \cdot h_3(n)$

(7) $\min\left[h_1(n), h_2(n) + h_3(n)\right]$

(8) $\max\left[h_1(n), h_2(n) + h_3(n)\right]$

In order to guarantee that a function of admissible heuristics is still admissible, the expression must be less than or equal to the maximum of the heuristics. Sums and products do not satisfy these, so (1), (6), and (8) all immediately fail. (3), (4), and (7) all work because the maximum of admissible heuristics is still admissible, as is the minimum of an admissible heuristic and anything else. (5) is the average of the heuristics, so it must be less than the maximum, and is thus admissible. Lastly, (2) is a weighted average, and is thus also less than the maximum, and is thus admissible.

(c) In the state-space graph shown in the adjacent figure, the states A, D, and E are missing a heuristic value. Determine the possible range for each missing heuristic value so that the heuristic is consistent. If no consistent heuristic is possible, write so. Show all your calculations and/or explanations.



(6)

Solution A consistent heuristic must satisfy $h(s) \leqslant c(s,s') + h(s')$ for all edges (s,s'). For the search graph above, this means the following. We use S for the start node, and G for the goal node.

For $s = A$:
$h(S) \leqslant c(S,A) + h(A) \Rightarrow 6 \leqslant h(A)$
$h(A) \leqslant c(A,B) + h(B) \Rightarrow h(A) \leqslant 7$
$h(A) \leqslant c(A,F) + h(F) \Rightarrow h(A) \leqslant 6$ to be consistent

For $s = D$:
$h(D) \leqslant c(D,G) + h(G) \Rightarrow h(D) \leqslant 3$
$h(B) \leqslant c(B,D) + h(D) \Rightarrow 0 \leqslant h(D)$
$h(F) \leqslant c(F,D) + h(D) \Rightarrow 1 \leqslant h(D)$ to be consistent

For $s = E$:
$h(E) \leqslant c(E,G) + h(G) \Rightarrow h(E) \leqslant 6$
$h(C) \leqslant c(C,E) + h(E) \Rightarrow 5 \leqslant h(E)$
$h(E) \leqslant c(E,F) + h(F) \Rightarrow h(E) \leqslant 6$ to be consistent

Therefore, the following table gives the range for each missing heuristic value so that the heuristic is consistent.

State	Range for $h(s)$
A	$6 \leqslant h(A) \leqslant 6$
D	$1 \leqslant h(D) \leqslant 3$
E	$5 \leqslant h(E) \leqslant 6$

2. [Adversarial search]

Two professors Maxitra and Minijit play a two-player game. At all points of time, the game maintains two piles containing m and n marbles with $m \leqslant n$. Here, m and n are **positive integer variables** that are redefined by every move. The moves alternate between MAX and MIN. MAX makes the first move. In each move, a player collects **all** the marbles from one of the two piles. He then splits the other pile into two non-empty piles of marbles. If there are options to continue the game, one of the available options must be chosen. The game ends when no further move can be made.

(a) What would be the numbers of marbles in the two piles, in the end-game situation. Just write your answer. No need to explain. **(1)**

Solution $m = 1$ and $n = 1$.

Look at an end-game situation. Let P be the player who fails to make the next move, and Q the other player. Let c_P and c_Q be the counts of marbles collected so far by P and Q, respectively. If $c_P \leqslant c_Q$, then P loses (that is, Q wins). If $c_P > c_Q$, then the game ends in a draw.

(b) The goal is to suggest the/a best move for MAX at the beginning of the game. This is to be posed as an adversarial-search problem. Clearly mention what each state in the search space should consist of. No need for an explanation. **(2)**

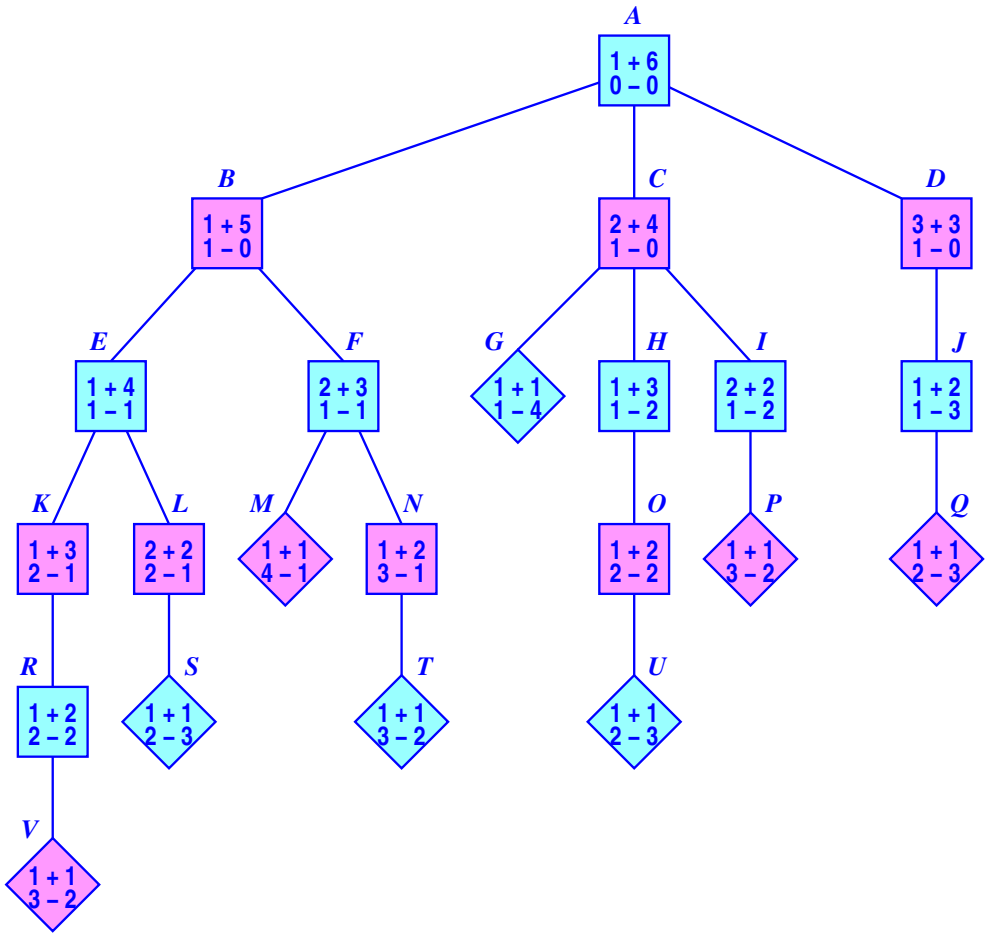
Solution Each state consists of the following four items.

1. The current number m of marbles in the first pile.
2. The current number n of marbles in the second pile.
3. The number c_{MAX} of marbles collected by MAX so far.
4. The number c_{MIN} of marbles collected by MIN so far.

In the rest of this exercise, we assume that the game begins with $m = 1$ and $n = 6$. Note again that MAX moves first.

(c) On the next page, draw the complete game tree. Clearly mark which nodes are MAX nodes, which are MIN nodes, and which are the end-game nodes. Against every node in the tree, show all the information maintained in that state (see Part (b)). Name the nodes $A, B, C, D, \ldots$ in the breadth-first fashion (left-to-right ordering in each level). Note that you should maintain $m \leqslant n$ always. (**Hint:** Your game tree would contain 22 nodes.) **(5)**

Solution In each node of the tree, we show the marble counts in the two piles as $m+n$. Also the counts c_{MAX} and c_{MIN} of marbles collected so far by MAX and MIN are shown as $c_{MAX} - c_{MIN}$. The MAX nodes are shaded by cyan, and the MIN nodes are shaded by magenta. The end-game (terminal) nodes are shown as diamonds. Other nodes are shown as squares.

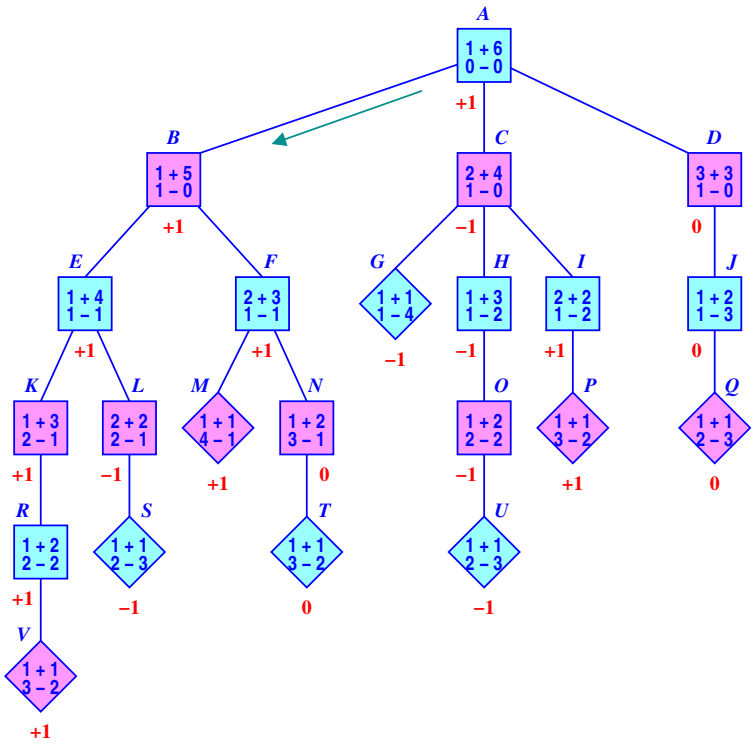


(d) Give a utility value of $+1$, -1, or 0 to an end-game node according as whether MAX wins, MAX loses, or the game ends in a draw, at that node. Compute the minimax values at all the nodes in the game tree. Do not redraw the game tree. Do not show the minimax values in the tree drawn in Part (c). Instead, show the calculations at each node (like $Minimax(W) = \max(Minimax(X), Minimax(Y), Minimax(Z)) = \cdots$). **(5)**

Solution Let us compute the minimax values of the nodes in the sequence $V, U, T, S, \ldots, C, B, A$.

Level	Minimax values	Comments (if any)
5	$Minimax(V) = +1$	[Terminal node]
4	$Minimax(U) = -1$	[Terminal node]
	$Minimax(T) = 0$	[Terminal node]
	$Minimax(S) = -1$	[Terminal node]
	$Minimax(R) = \max(Minimax(V)) = +1$	
3	$Minimax(Q) = 0$	[Terminal node]
	$Minimax(P) = +1$	[Terminal node]
	$Minimax(O) = \min(Minimax(U)) = -1$	
	$Minimax(N) = \min(Minimax(T)) = 0$	
	$Minimax(M) = +1$	[Terminal node]
	$Minimax(L) = \min(Minimax(S)) = -1$	
	$Minimax(K) = \min(Minimax(R)) = +1$	
2	$Minimax(J) = \max(Minimax(Q)) = 0$	
	$Minimax(I) = \max(Minimax(P)) = +1$	
	$Minimax(H) = \max(Minimax(O)) = -1$	
	$Minimax(G) = -1$	[Terminal node]
	$Minimax(F) = \max(Minimax(M), Minimax(N)) = +1$	
	$Minimax(E) = \max(Minimax(K), Minimax(L)) = +1$	
1	$Minimax(D) = \min(Minimax(J)) = 0$	
	$Minimax(C) = \min(Minimax(G), Minimax(H), Minimax(I)) = -1$	
	$Minimax(B) = \min(Minimax(E), Minimax(F)) = +1$	
0	$Minimax(A) = \max(Minimax(B), Minimax(C), Minimax(D)) = +1$	

Although not asked, here is the game tree with these minimax values.



(e) Use Part (d) to express, in a single sentence, the/a best move for MAX at the beginning of the game. **(2)**

Solution Take the first pile, and split the second pile into two piles containing 1 and 5 marbles, respectively.

3. [Local search]

Let $A = (a_0, a_1, a_2, \ldots, a_{n-1})$ be an array, where each a_i is an integer in the range $0, 1, 2, \ldots, n-1$. Construct the array $C = (c_0, c_1, c_2, \ldots, c_{n-1})$, where for each $i = 0, 1, 2, \ldots, n-1$, the array element c_i stores the number of times i appears in the array A. We call A **special** is $c_i = a_i$ for all $i = 0, 1, 2, \ldots, n-1$.

(a) Special arrays do exist. For instance, take $n = 4$, and the array $A = (1, 2, 1, 0)$. The numbers of times $0, 1, 2, 3$ appear in A are all equal to a_0, a_1, a_2, a_3, respectively. Find another special array for $n = 4$. Do not run any AI-specific algorithm. Use your natural intelligence. Write only the final answer. There is no need to explain how you came up with this example. **(2)**

Solution $(2, 0, 2, 0)$

Given n, we want to find a special array of size n. We want to solve this problem using a greedy local-search algorithm (similar to that used for the n-queens problem).

(b) How do you define a state for this search problem? **(2)**

Solution A state can be any array of size n with each entry coming from $\{0, 1, 2, \ldots, n-1\}$. You may additionally demand that the sum of the elements of the array must be n. But this restriction makes the rest of the endeavor quite difficult. Of course, you may come up with a nice way to live in the restricted world, but please explain things clearly later.

(c) Define a badness function $b()$ on the set of states such that for each state σ, its badness $b(\sigma)$ would meaningfully measure how far σ is from the/a goal state. If σ is a goal state, you should have $b(\sigma) = 0$. **(3)**

Solution Given an array A standing for a state, compute the counts array C as explained above. You may take several badness functions. Here are a few possibilities.

- $b(A)$ is the count of $i \in \{0, 1, 2, \ldots, n-1\}$, for which $c_i \neq a_i$.
- $b(A) = \sum_{i=0}^{n-1} |a_i - c_i|$.
- $b(A) = \sum_{i=0}^{n-1} (a_i - c_i)^2$.

(d) Propose a greedy local-search algorithm to find a special array of a given size n. You should use your state-space formulation of Part (b) and your badness function of Part (c). Write a pseudocode of your algorithm. **(7)**

Solution The following algorithm is a straightforward adaptation of the greedy local-search algorithm.

Generate an array A with each a_i randomly chosen from $\{0,1,2,\ldots,n-1\}$.
Repeat the following loop until broken explicitly:
    Compute $currentbadness = b(A)$.
    If $currentbadness = 0$, return A.
    Set $minbadness = +\infty$. [You may also set $minbadness = currentbadness$.]
    for $i = 0,1,2,\ldots,n-1$, repeat:
        for $j = 0,1,2,\ldots,n-1$ with $j \neq a_i$, repeat:
            Set $a_i = j$.
            Compute $newbadness = b(A)$.
            If $(newbadness < minbadness)$, then do:
                Record $minbadness := newbadness$.
                Also record the values of i and j as i^* and j^*.
            Restore A.
    If $(minbadness \geqslant currentbadness)$, return *failure*.
    Set $A[i^*] := j^*$.

(e) Suppose that your algorithm gets stuck in a shoulder or a local minimum. How can you still use your algorithm to improve your odds of finding a solution? Do not use a separate algorithm (like SA or GA). **(1)**

Solution Keep on making random restarts until you get your special array or feel too tired to stay awake.

Note: This puzzle is usually posed for $n = 10$ with A viewed as a 10-digit integer in the decimal notation. Apply your natural intelligence to solve the decimal puzzle. Then, generalize to other values of n. Do this after the test. Here, your teachers need to review your understanding of local-search algorithms. Please give them a chance. BTW, what about $n = 6$?

4. [Constraint satisfaction problem (CSP)]

Consider the following cryptarithmetic problem, where each English letter symbolizes a fixed decimal digit (distinct from one another). Here, the two summands are 3-digit numbers, and the sum is a 4-digit number.

$$\begin{array}{cccc} & \mathbf{T} & \mathbf{W} & \mathbf{O} \\ + & \mathbf{T} & \mathbf{W} & \mathbf{O} \\ \hline \mathbf{F} & \mathbf{O} & \mathbf{U} & \mathbf{R} \end{array}$$

Your task is to formulate and solve this problem using CSP techniques. Answer the following parts.

(a) Express all the constraints for this problem mathematically. **(4)**

Solution Here, F, O, U, R, T, W are the variables that we intend to solve for. We introduce three additional variables C_1, C_2, C_3 to hold the carry-out values. The constraints are given below.

$$\begin{aligned} O + O &= R + 10 \times C_1 \\ C_1 + W + W &= U + 10 \times C_2 \\ C_2 + T + T &= O + 10 \times C_3 \\ C_3 &= F \\ \text{ALL\_DIFFERENT}(F, O, U, R, T, W) & \end{aligned}$$

(b) Write the domain of each of the variables defined in your constraints. **(3)**

Solution

$$\begin{aligned} \mathscr{D}(F), \mathscr{D}(C_3) &= \{1\} \\ \mathscr{D}(C_1), \mathscr{D}(C_2) &= \{0, 1\} \\ \mathscr{D}(T) &= \{1, 2, 3, 4, 5, 6, 7, 8, 9\} \\ \mathscr{D}(W), \mathscr{D}(O), \mathscr{D}(U), \mathscr{D}(R) &= \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\} \end{aligned}$$

(c) Solve this cryptarithmetic problem by hand, using (chronological) Backtracking (BKT), Forward Checking (FWC), and the Minimum Remaining Values (MRV) and the Least Constraining Value (LCV) heuristics. Clearly show your effort (progress) step-wise by mentioning, in each step (iteration), how the variable values are selected, and the domains of the variables are updated by forward checking. **(8)**

Solution A trace leading to a solution is given below. Some state expansions were skipped when the MRV heuristic would have chosen a state with only a single value in its domain.

Step	**State/Variable Valuation** $(F,O,U,R,T,W,C_1,C_2,C_3)$	**Assignment/Action** **(with Domain Update)**
0	$(-,-,-,-,-,-,-,-,-)$	Select F next via MRV.
1	$(1,-,-,-,-,-,-,-,-)$	Assign $F=1$. Remove 1 from all domains by FWC. $\mathscr{D}(C_3)=\{1\}$; $\mathscr{D}(C_1),\mathscr{D}(C_2)=\{0,1\}$; $\mathscr{D}(T)=\{2,3,4,5,6,7,8,9\}$; $\mathscr{D}(W),\mathscr{D}(O),\mathscr{D}(U),\mathscr{D}(R)=\{0,2,3,4,5,6,7,8,9\}$
2	$(1,-,-,-,-,-,-,-,1)$	Assign $C_3=1$. Remove $\{2,3,4\}$ from $\mathscr{D}(T)$ by FWC. Select C_2 using MRV. $\mathscr{D}(C_1),\mathscr{D}(C_2)=\{0,1\}$; $\mathscr{D}(T)=\{5,6,7,8,9\}$; $\mathscr{D}(W),\mathscr{D}(O),\mathscr{D}(U),\mathscr{D}(R)=\{0,2,3,4,5,6,7,8,9\}$
3	$(1,-,-,-,-,-,-,0,1)$	Assign $C_2=0$ using LCV. Remove $\{3,5,7,9\}$ from $\mathscr{D}(O)$ and $\{5,6,7,8,9\}$ from $\mathscr{D}(W)$ by FWC. Select C_1 using MRV. $\mathscr{D}(C_1)=\{0,1\}$; $\mathscr{D}(T)=\{5,6,7,8,9\}$; $\mathscr{D}(W)=\{0,2,3,4\}$; $\mathscr{D}(O)=\{0,2,4,6,8\}$; $\mathscr{D}(U),\mathscr{D}(R)=\{0,2,3,4,5,6,7,8,9\}$
4	$(1,-,-,-,-,-,0,0,1)$	Assign $C_1=0$ using LCV. Remove $\{6,8\}$ from $\mathscr{D}(O)$ and odds from $\mathscr{D}(U)$ by FWC. Select $\mathscr{D}(O)$ using MRV. $\mathscr{D}(T)=\{5,6,7,8,9\}$; $\mathscr{D}(W)=\{0,2,3,4\}$; $\mathscr{D}(U)=\{0,2,4,6,8\}$; $\mathscr{D}(O)=\{0,2,4\}$; $\mathscr{D}(R)=\{0,2,3,4,5,6,7,8,9\}$
5	$(1,2,-,-,-,-,0,0,1)$	Assign $O=2$ using LCV. Remove 2 from all domains. Set $\mathscr{D}(T)=\{6\}$ and $\mathscr{D}(R)=\{4\}$. Select W by MRV. $\mathscr{D}(T)=\{6\}$; $\mathscr{D}(O)=\{2\}$; $\mathscr{D}(R)=\{4\}$; $\mathscr{D}(U)=\{0,4,6,8\}$; $\mathscr{D}(W)=\{0,3,4\}$
6	$(1,2,-,4,7,-,0,0,1)$	Assign $W=3$ by LCV. No Solution! BKT to *Step 3*. $\mathscr{D}(U)=\{0,4,8\}$; $\mathscr{D}(W)=\{0\}$
7	$(1,-,-,-,-,-,1,0,1)$	Assign $C_1=1$. Remove $\{0,2,4\}$ from $\mathscr{D}(O)$ and evens from $\mathscr{D}(U)$ by FWC. Select $\mathscr{D}(O)$ using MRV. $\mathscr{D}(T)=\{5,6,7,8,9\}$; $\mathscr{D}(W)=\{0,2,3,4\}$; $\mathscr{D}(U)=\{3,5,7,9\}$; $\mathscr{D}(O)=\{6,8\}$; $\mathscr{D}(R)=\{0,2,3,4,5,6,7,8,9\}$
8	$(1,6,-,-,-,-,1,0,1)$	Assign $O=6$. Remove 6 from all domains. Set $\mathscr{D}(T)=8$ and $\mathscr{D}(R)=2$. Pick W next. $\mathscr{D}(T)=\{8\}$; $\mathscr{D}(W)=\{0,2,3,4\}$; $\mathscr{D}(U)=\{3,5,7,9\}$; $\mathscr{D}(O)=\{6\}$; $\mathscr{D}(R)=\{2\}$
9	$(1,6,7,2,8,3,0,0,1)$	Assign $W=3$ by LCV. Found Solution! $\mathscr{D}(U)=\{3,5,7,9\}$; $\mathscr{D}(W)=\{0,3,4\}$

This gives the following solution.

$$\begin{array}{cccc} & \mathbf{8} & \mathbf{3} & \mathbf{6} \\ + & \mathbf{8} & \mathbf{3} & \mathbf{6} \\ \hline \mathbf{1} & \mathbf{6} & \mathbf{7} & \mathbf{2} \end{array}$$