Report on

“Offline 2: Solving Latin Square”

Course Code: CSE 318
Course Title: Artificial Intelligence Sessional

Submitted By:

Md. Azizur Rahman Anik
Student ID: 1805115

Date of Submission:

09-01-2023

<u>Value Order Heuristic:</u> Least-Constraining-Value

After a variable has been selected using a Variable-Order-Heuristic, to decide on the order in which to examine its values, **"Least-Constraining-Value"** heuristic has been chosen. It prefers the value that rules out the fewest choices for the neighboring variables (**in this problem, the neighboring variables are the ones that are either in same row or same column**) in the constraint graph. As it tries to leave the **maximum flexibility for subsequent variable assignments**, it is more likely to go through the path which will provide a solution faster. As we need only one solution in this assignment, this heuristic provides better performance than choosing the values from the domain of a variable randomly or serially. If we need to enumerate all solutions rather than just fine one, then value ordering would be irrelevant.

Data Table: The * marker denotes the solution has not come within 30 minutes.

Problem	Solver	VAH	#Node	#BT	Time(ms)
d-10-01	BT	VAH1	58	0	16
	BT	VAH2	416178375	150607121	813578 =13.56 mins
	BT	VAH3	62	1	15
	BT	VAH4	64	2	0
	BT	VAH5	47450637	14132066	51880
	FC	VAH1	58	0	0
	FC	VAH2	6947264	1831610	32250
	FC	VAH3	61	1	0
	FC	VAH4	62	2	16
	FC	VAH5	23952	5492	109
d-10-06	BT	VAH1	245	18	0
	BT	VAH2	240377219	86763283	379593= 6.33 mins
	BT	VAH3	58	0	0
	BT	VAH4	59	1	0
	BT	VAH5	52301642	15024497	48591
	FC	VAH1	227	18	0
	FC	VAH2	5266806	1412938	18132
	FC	**VAH3**	**58**	**0**	**0**
	FC	**VAH4**	**58**	**0**	**0**
	FC	VAH5	911503	249150	2702
d-10-07	**BT**	**VAH1**	**58**	**0**	**0**
	BT	VAH2	17948366	6560063	34221
	BT	VAH3	104	5	0
	BT	VAH4	59	1	0
	BT	VAH5	79306908	22432707	89826
	FC	**VAH1**	**58**	**0**	**0**
	FC	VAH2	336402	86607	1236
	FC	VAH3	99	5	0

	FC	**VAH4**	**58**	**0**	**0**
	FC	VAH5	11385	2885	31
d-10-08	**BT**	**VAH1**	**73**	**1**	**0**
	BT	VAH2	*	*	*
	BT	VAH3	388	27	15
	BT	VAH4	795	89	0
	BT	VAH5	32235958	9206314	32308
	FC	**VAH1**	**72**	**1**	**0**
	FC	VAH2	75343876	20485700	270199 =4.5mins
	FC	VAH3	361	27	0
	FC	VAH4	706	83	16
	FC	VAH5	663129	175800	1442
d-10-09	BT	VAH1	3275	490	31
	BT	VAH2	896016	296740	1276
	BT	VAH3	70	3	0
	BT	**VAH4**	**67**	**4**	**0**
	BT	VAH5	400728277	118466648	402598 =6.7 mins
	FC	VAH1	2785	490	16
	FC	VAH2	34063	8476	110
	FC	VAH3	67	3	0
	FC	**VAH4**	**63**	**1**	**0**
	FC	VAH5	55911080	15271021	132063 =2.2 mins
d-15-01	**BT**	**VAH1**	**136196**	**12274**	**453**
	BT	VAH2	*	*	*
	BT	VAH3	710729	66597	2015
	BT	VAH4	1672762	191924	5421
	BT	VAH5	*	*	*
	FC	**VAH1**	**123922**	**12274**	**281**
	FC	VAH2	*	*	*
	FC	VAH3	644132	66597	1952
	FC	VAH4	1480838	188764	5186

	FC	VAH5	*	*	*

Analysis:

To apply the heuristics given, **a modified Backtrack** is used where the modification is: When a value of a variable is successfully assigned, then the domain of its neighboring variables have been updated.

The difference between the Forward Checking and the modified Backtrack is: in forward checking, after updating the domain of neighboring variables, if any domain becomes empty, then immediately backtracked from that node, whereas no such decision is made based on the size of the domain in the modified backtrack.

Reason to use a Modified Backtrack: If a pure backtrack solver is used, then the heuristics will not get the updated domain. Therefore it will work on the initial domain every time and this will not be able to provide the power of the heuristics.

Observations: From the data table, it is clear that the Forward checking scheme performs better than Backtrack for all the test cases as it detects failure earlier and reduces the number of nodes in the search tree by pruning larger parts of the tree earlier. Therefore the number of nodes are less in forward checking than that in backtracking.

Performance Comparison among the 5 Variable-Order-Heuristics: The "Minimum Remaining Values (MRV)", that is, VAH1 heuristic seems to be the best from the data table as it gives the best performance (considering number of nodes, backtracks and time to find the solution) in most of the test cases. The reason is, it picks a variable that is most likely to cause a failure soon, thereby pruning the search tree- avoiding pointless searches through other variables.

After VAH1, the VAH4 gives better performance, and then VAH3. **VAH4** tries to combine heuristic VAH1 and VAH2 by minimizing the ratio. It gives similar performance to VAH3 in some test cases.

The **VAH3** is a extended version of VAH1, where the tie is broken using the **degree heuristic (selecting the variable that is involved in the largest number of constraints on other unassigned variables).** Therefore some extra work has to be done to get the degree of the variables that make a tie.

The **VAH2** heuristic is mainly used to break tie of VAH1 heuristic. It alone does not provide much improvement in performance which is evident from the data table.

Finally the **VAH5** heuristic picks variables randomly and its performance changes in different runs but on average, it provides poor performance than VAH1, VAH3 and VAH4.

Conclusion:

Forward checking combining with VAH1 seems the best scheme. The reason is, it picks a variable that is most likely to cause a failure soon and checking the size of the domain of neighboring variables, it detects failure early, thereby pruning the search tree- avoiding pointless searches through other variables.

Strength or Weakness of a solver: It is clear that the strength of Forward checking is more than that of Backtracking. Considering the heuristics, if the data instance is as such that the tie breaking scenario does not occur, then heuristic VAH1 gives better performance than VAH3 and VAH4 as there's no overhead of tie breaking in VAH1. On the other hand, if there are tie breaking scenarios, then VAH3

and VAH4 performs better than VAH1. VAH2 has a poor performance in every data instance. Finally, as VAH5 is a random selection of variables, its performance varies in different runs.