Decision Making - ex 3

Filippo Brajucha, Youssef Hanna

October 2023

1 nQueens

1.1 Table

		30	**35**	**45**	**50**
input order	min value	**1.588.827**	-	-	-
	random value	9	10	**6**	42
min domain size	min value	15	21	**6**	123
	random value	1	**0**	1	10
domWdeg	min value	15	21	**6**	123
	random value	1	**0**	1	10

Table 1: nQueens problem with n = 30, 35, 45, 50

1.2 Comment

https://www.minizinc.org/doc-2.7.6/en/mzn_search.html
We can observe that the number of failures with the "input_order - min_value" is extremely high compared to other ones. It is in the order of the 10^6, other datas are in the order of 10^2; this is the only case where the system cannot perform the research for `n = 35, 45, 50`.
Generally we can observe that the random model is always better in terms of failures for this problem.

2 Poster Placement

2.1 Table

		19x19		20x20	
		Fails	Time	Fails	Time
input order	**min value**	**1.315.598**	**11s 35ms**	26.063.823	3m 12s
	random value	-	-	-	-
min domain size	**min value**	239.954	1s 796ms	**1.873**	**244ms**
	random value	2.929.153	19s 172ms	5.797.312	35s 987ms
domWdeg	**min value**	236.024	1s 820ms	**1.873**	**244ms**
	random value	2.929.030	19s 30ms	5.797.456	35s 957ms

Table 2: Poster Placement using 19x19.dzn and 20x20.dzn (unsorted)

2.2 Table with sorted rectangles

	19x19		**20x20**	
	Fails	Time	Fails	Time
min value	29.871	562ms	16.631	479ms
random value	-	-	-	-

Table 3: Poster Placement using 19x19.dzn and 20x20.dzn (sorted)

2.3 Comment

3 Quasigroup

3.1 Table

		default	**domWdeg** - random	**domWdeg** + Luby
qc30-03	Fails	-	234.522	234.522
	Time	-	**15s 569ms**	19s 109ms
qc30-05	Fails	-	36.866	36.866
	Time	-	2s 909ms	**2s 820ms**
qc30-08	Fails	324	324	324
	Time	**373ms**	394ms	583ms
qc30-12	Fails	470	470	470
	Time	409ms	399ms	**396ms**
qc30-19	Fails	2.192	2.192	2.192
	Time	513ms	**500ms**	574ms

Table 4: Quasigroup problem resolution with using qc30-03.dzn, qc30-05.dzn, qc30-08.dzn, qc30-12.dzn and qc30-19.dzn

3.2 Comment

rifai tutti i tempi

4 Questions

1. When are random decisions (not) useful? Why?
2. Are dynamic heuristics always better than static heuristics? Why?
3. Is programming search and/or restarting always a good idea? Why?

4.1 Answers

1. Measured data varies greatly depending on the nature of the problem; the only scenario where resolving with random heuristic is the best solution is with the nQueens problem.
   In the poster placement problem, it's always the worst solution, differing by orders of magnitude both in the number of errors (6 million errors compared to around 2 thousand) and in resolution times (several seconds compared to about 300ms). In the situation with ordered posters, it times out.
   Using the random heuristic with the quasigroup problem, that result to be the best solution 2 times out of 5. Due to this we can think that the input order is crucial in the poster placement problem.
   Random solutions, therefore, are not suitable in situations where maintaining an order in data usage is necessary. In fact, in both the quasigroup and nQueens problems, the choice of starting point isn't important since finding a solution requires starting from any point.
   The scenario where it becomes most apparent that the random heuristic is the worst is solving the nQueens problem when input order is specified. Using the input order heuristic means that each element, in order, is assigned the minimum value from the domain. The issue arises when it's necessary to find a solution; the solver encounters a situation where all elements do not satisfy the constraints. Consequently, it has to perform numerous backtracking operations to ensure that all points on the grid adhere to the constraints. $perchedomwdeg\grave{e}ugualeamindomain?condellesemplificazioniperch\grave{e}\grave{e}costantesiottie$
2.
3.