## ANALYSIS OF HEURISTIC COMBINATIONS

Some keywords and terms:

- MRV – Minimum remaining values heuristic: The solver will choose a cell that has the smallest domain when deciding on what cell to assign values to next. This is to reduce the number of nodes.
- MD – Maximum degree heuristic: The solver will choose a cell with the largest degrees, or the largest unassigned neighbors. This is usually used as a tie-breaker for the MRV heuristic. This is also to reduce the number of nodes.
- LCV – Least constraining value heuristic: The solver will choose a value that constrains the least number of unassigned neighbors. This is to avoid failure as much as possible.
- ACP – Arc consistency 3 pre-processor: The solver will run an ac3 check on the puzzle before solving it. An ac3 check will ensure that each cell's values have at least one value in each of their unassigned neighbors that is consistent, or legal.
- MAC – Maintenance of Arc consistency 3: The solver will run an ac3 check on the assigned cell's neighbors after an assignment.
- FC – Forward check: The solver will check an assigned cell's neighbors after assignment, and ensure that those cells are consistent with the assignment.
- Puzzle parameters – m, n, p, q: m is the number of clues, or already solved cells in the puzzle. n is the size of the puzzle. p and q are the height and width of the blocks, respectively.

In this part, I ran the solver on the same 10 random puzzles with the parameters: m = 17, n = 9, p = 3, q = 3. The solver is set to timeout after 300s. Each iteration of the test had the solver using a different combination of heuristic methods. First, the solver ran with only constraint propagation. Next, I tested the solver on individual heuristic methods. Then, the solver ran with a few different combinations, and finally with all of the methods. For each iteration, certain statistics were recorded: average number of nodes, dead ends, average time, % completed, etc.

| Heuristics Combination | Avg. # Nodes | Avg. # Dead Ends | Avg. Total Time | % Complete |
|---|---|---|---|---|
| None | 262 090 | 262 032 | 39.2957s | 100% |
| MRV only | 56 169 | 56 111 | 15.0611s | 100% |
| MD only | 1 937 753 | 1 937 717 | 300s | 0% |
| LCV only | 4 186 | 4 128 | 0.9162s | 100% |
| ACP only | 257 130 | 257 072 | 38.4658s | 100% |
| MAC only | 135 996 | 135 939 | 26.4882s | 100% |
| FC only | 262 090 | 262 032 | 28.5434s | 100% |
| MRV, MD, LCV | 67 012 | 66 951 | 30.0306s | 90% |
| ACP, MAC, FC | 130 624 | 130 566 | 25.2502s | 100% |
| MRV, LCV | 37 986 | 37 928 | 12.8338s | 100% |
| MRV, LCV, ACP, MAC, FC | 37 986 | 37 928 | 15.2296s | 100% |

| | | | | |
|---|---|---|---|---|
| All | 60 007 | 59 945 | 30.0495s | 90% |

From this table alone, LCV only has the best performance on the 10 puzzles. Do note that typically, the lower the number of nodes the solver has to expand and explore, the less time it takes to solve. LCV only has the smallest average nodes, 4 186, and thus has the smallest time. This may certainly just be a product of a small sample size, and may not scale well with larger puzzles. Also note, MD only has the highest time, with an average of 300s. This is due to timeout. With only MD, the solver is unable to solve these puzzles within 5 minutes.

Compared to the no heuristics option, ACP only has similar times. ACP only has less nodes than the baseline, but it only saved about 1s. This 1s saved can be attributed to differences in CPU usage and background programs running. How many nodes ACP prunes is most likely not enough to overcome its algorithm time overhead. MAC only, FC only, and ACP, MAC, FC all have similar times.

The solver saved a significant 18s by disabling MD when also using MRV and LCV. This and MD only's 300s average time leads me to believe that the MD implementation is simply too inefficient to be worth it. On a similar not, the solver saved about 15s when disabling only MD when compared to using all of the heuristics.

The next few analyses will be only using the MRV + LCV heuristics and the MRV, LCV, ACP, MAC, FC heuristics only.

## ESTIMATION OF THE HARDEST R FOR 9X9 PUZZLES

The Hardest R is a value that denotes how many clues a puzzle will have so that the solver will have the hardest time solving it. R can be calculated as $R = m/n^2$ . For each m, 15 random 9x9 puzzles were generated, and the average number of nodes, average dead ends, average time, % complete, and % solvable were recorded. Completed puzzles are puzzles in which the solver has found a solution to or determined unsolvable. Solved puzzles are puzzles in which the solver has found a solution to.

With only MRV and LCV.

| m | R | Avg. # Nodes | Avg. # Dead Ends | Avg. Time | % Complete | % Solved |
|---|---|---|---|---|---|---|
| 4 | 0.0494 | 77 | 0 | 0.02262s | 100% | 100% |
| 8 | 0.0988 | 83 946 | 83 875 | 20.0193s | 93.3% | 93.3% |
| 12 | 0.148 | 69 | 0 | 0.0194s | 100% | 100% |
| 16 | 0.198 | 915 | 854 | 0.2251s | 100% | 93.3% |
| 17 | 0.210 | 61 | 1 | 0.0169s | 100% | 93.3% |
| 18 | 0.222 | 108 | 62 | 0.0437s | 100% | 73.3% |
| 19 | 0.235 | 3 718 | 3 656 | 0.9212s | 100% | 100% |
| 20 | 0.247 | 83 051 | 83 008 | 20.159s | 93.3% | 67.7% |
| 21 | 0.259 | 6 918 | 6 866 | 1.571s | 100% | 86.7% |
| 22 | 0.272 | 11 241 | 11 198 | 2.8444s | 100% | 73.3% |
| 24 | 0.296 | 108 140 | 108 112 | 25.9309s | 93.3% | 26.7% |
| 28 | 0.346 | 123 | 119 | 0.0292s | 100% | 6.7% |

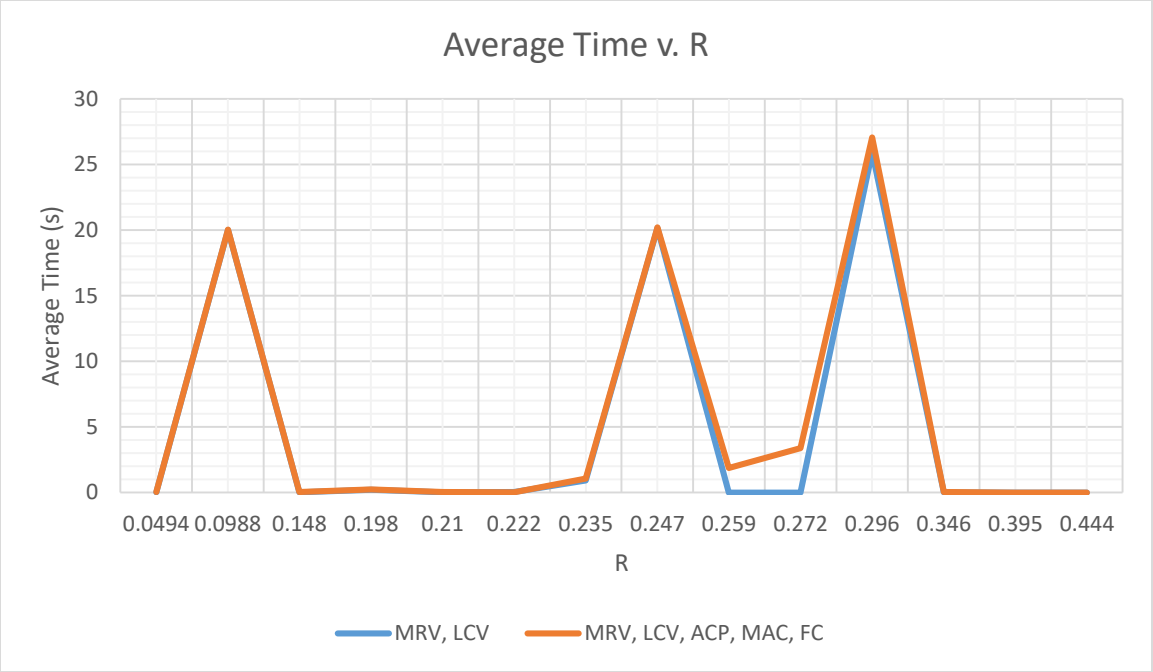
| | | | | | | |
|---|---|---|---|---|---|---|
| 32 | 0.395 | 1 | 1 | 0.0013s | 100% | 0% |
| 36 | 0.444 | 0 | 0 | 0.001s | 100% | 0% |

With only MRV, LCV, ACP, MAC, and FC.

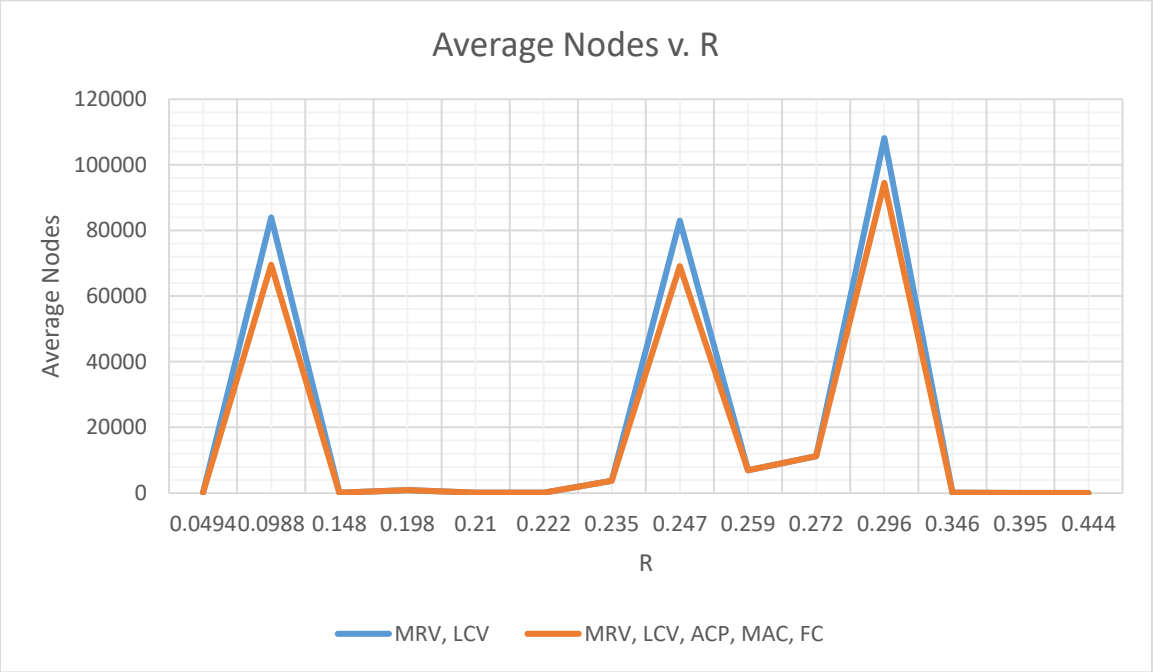
| m | R | Avg. # Nodes | Avg. # Dead Ends | Avg. Time | % Complete | % Solved |
|---|---|---|---|---|---|---|
| 4 | 0.0494 | 77 | 0 | 0.05144s | 100% | 100% |
| 8 | 0.0988 | 69 540 | 69 469 | 20.0436s | 93.3% | 93.3% |
| 12 | 0.148 | 69 | 0 | 0.0406s | 100% | 100% |
| 16 | 0.198 | 915 | 854 | 0.249s | 100% | 93.3% |
| 17 | 0.210 | 61 | 1 | 0.03505s | 100% | 93.3% |
| 18 | 0.222 | 108 | 62 | 0.0262s | 100% | 73.3% |
| 19 | 0.235 | 3 718 | 3 656 | 1.076s | 100% | 100% |
| 20 | 0.247 | 69 207 | 69 163 | 20.2036s | 93.3% | 66.7% |
| 21 | 0.259 | 6 918 | 6 866 | 1.876s | 100% | 86.7% |
| 22 | 0.272 | 11 241 | 11 198 | 3.3874s | 100% | 73.3% |
| 24 | 0.296 | 94 509 | 94 492 | 27.0683s | 93.3% | 26.7% |
| 28 | 0.346 | 122 | 118 | 0.0381s | 100% | 6.7% |
| 32 | 0.395 | 0 | 0 | 0.0029s | 100% | 0% |
| 36 | 0.444 | 0 | 0 | 0.0014s | 100% | 0% |

As you scan the two tables, you will realize that the MRV + LCV heuristics have faster average times than the other. This is primarily due to the algorithm overhead the ac3 and forward checking has. The MRV and LCV heuristics do most of the node pruning for these puzzles; thus, having extra heuristics doesn't reduce it much further if at all.

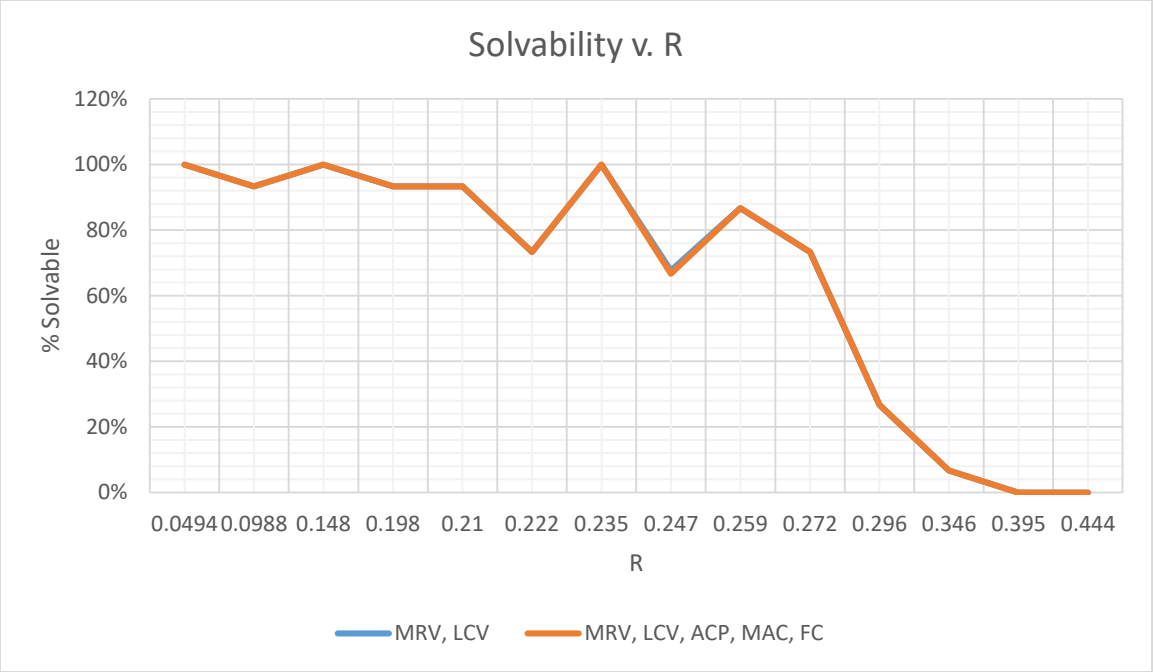
Starting at m = 4, the puzzles are fairly easy to solve. There are a few outliers that caused timeouts, but for the most part, m = 4, 8, 12, 16 were all fairly easy. This continues up until about m = 19, where the time becomes > 1s, with 20s for m = 20 and 25 - 27s for m = 24. After that, it again drops to being very easy. This is mostly due to m being too high, and thus leading to unsolvable puzzles. Based on the % solvable and the average time, the hardest R would be around when m = 24, or R = 0.296.




You can see that for both heuristic combinations, the average time remains small, except for a few spikes. At R = 0.0988, 0.247, and 0.296, there are large spikes in average time. Now, for 0.0988, this can be explained by an unfortunate difficult puzzle that was generated, which skewed the average time upward. The other two can be explained by it being the hardest R for the solver.




Except for the massive spikes at R = 0.0988, 0.247, 0.296, the average number of nodes stays relatively the same.




The solvability drops as R increases, as expected. When R increases, m increases. As m increases, the likelihood of the generated puzzle being solvable drops as there are much more constraints that may not necessarily be consistent.

## LARGEST N COMPLETABLE AT THE HARDEST R

The largest n will show us how well the solver scales with size. This was tested using 10 random puzzles per m, n, p, q parameters. The hardest m was calculated as $m = n^2 * R$, using the hardest R calculated in the previous part

With only MRV and LCV.

| Hardest m | n | p | q | Avg. # Nodes | Avg. # Dead Ends | Avg. Time | % Completed | % Solved |
|---|---|---|---|---|---|---|---|---|
| 43 | 12 | 3 | 4 | 432 560 | 432 501 | 180.25s | 40% | 20% |
| 67 | 15 | 3 | 5 | 275 902 | 275 825 | 180.145s | 40% | 10% |
| 76 | 16 | 4 | 4 | 299 246 | 299 128 | 219.613s | 30% | 20% |
| 96 | 18 | 3 | 6 | 307 319 | 307 181 | 300s | 0% | 0% |

With only MRV, LCV, ACP, MAC, and FC.

| Hardest m | n | p | q | Avg. # Nodes | Avg. # Dead Ends | Avg. Time | % Completed | % Solved |
|---|---|---|---|---|---|---|---|---|
| 43 | 12 | 3 | 4 | 376 982 | 376 922 | 180.304s | 40% | 20% |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 67 | 15 | 3 | 5 | 244 316 | 244 239 | 180.19s | 40% | 10% |
| 76 | 16 | 4 | 4 | 263 653 | 263 534 | 220.281s | 30% | 20% |
| 96 | 18 | 3 | 6 | 298 727 | 300 162 | 300s | 0% | 0% |

The above tables show the solver trying to solve larger puzzles at the hardest R. As you can see, the solver has a lot of trouble solving larger puzzles at that R. After a certain size, n = 16, the solver simply could not solve the puzzles within 5 minutes. Even before at sizes 12 and 15, the percentage of completion if low and continues to drop until size 18. The largest n solvable at the hardest R would then be either 12 or 15. However, one can argue that because of the low completion of size 12 and 15 puzzles, the largest n solvable at the hardest R is in fact 9. For the purposes of the next part, we will say that the largest n is 12.

## HARDEST R FOR LARGEST N

This is to test whether the hardest R value for a 9x9 puzzle scales as n grows. This was tested using 15 random 12x12 puzzles per m value.

With only MRV and LCV.

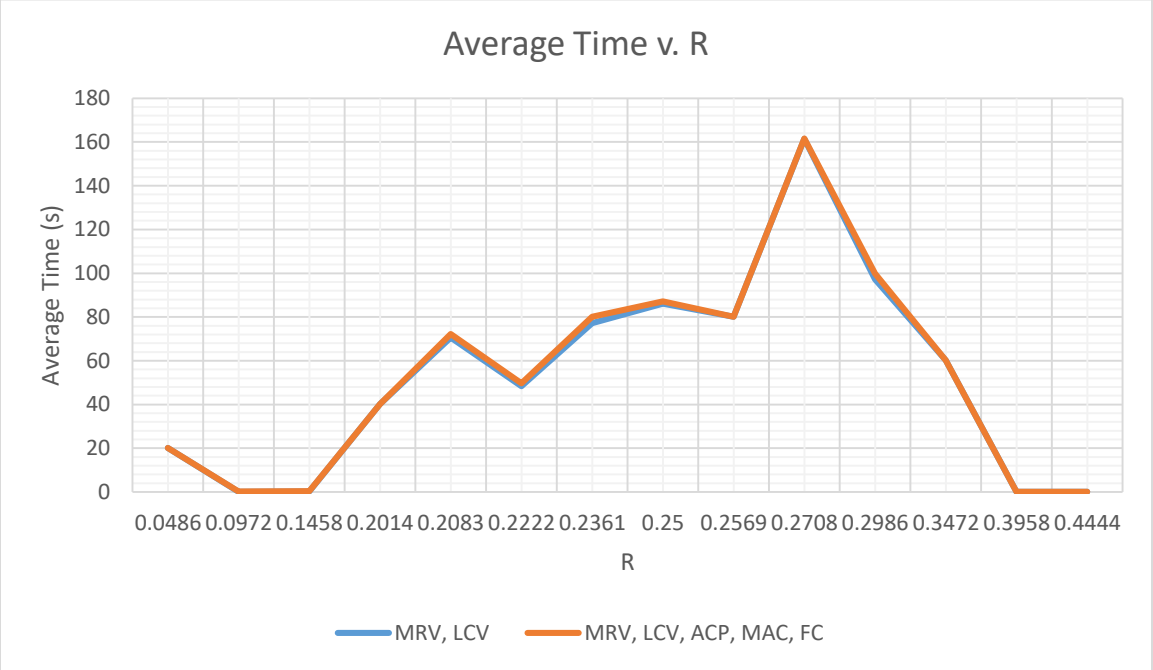
| m | R | Avg. # Nodes | Avg. # Dead Ends | Avg. Time | % Completed | % Solved |
|---|---|---|---|---|---|---|
| 7 | 0.0486 | 59 984 | 59 849 | 20.0736s | 93.3% | 93.3% |
| 14 | 0.0972 | 167 | 37 | 0.0775s | 100% | 100% |
| 21 | 0.1458 | 528 | 405 | 0.2056s | 100% | 100% |
| 29 | 0.2014 | 95 415 | 94 314 | 40.1124s | 86.7% | 80% |
| 30 | 0.2083 | 176 899 | 176 799 | 70.6323s | 80% | 73.3% |
| 32 | 0.2222 | 110 966 | 110 859 | 48.2862s | 86.7% | 86.7% |
| 34 | 0.2361 | 182 517 | 182 428 | 77.13s | 80% | 66.7% |
| 36 | 0.25 | 207 477 | 207 387 | 86.1668s | 83.3% | 66.7% |
| 37 | 0.2569 | 185 223 | 185 139 | 80.0714s | 73.3% | 60% |
| 39 | 0.2708 | 383 863 | 383 797 | 161.491s | 46.7% | 53.3% |
| 43 | 0.2986 | 230 518 | 230 455 | 97.2483s | 73.3% | 46.7% |
| 50 | 0.3472 | 138 098 | 137 082 | 60.219s | 80% | 66.7% |
| 57 | 0.3958 | 5 | 5 | 0.0049s | 100% | 0% |
| 64 | 0.4444 | 0 | 0 | 0.0029s | 100% | 0% |

With only MRV, LCV, ACP, MAC, and FC.

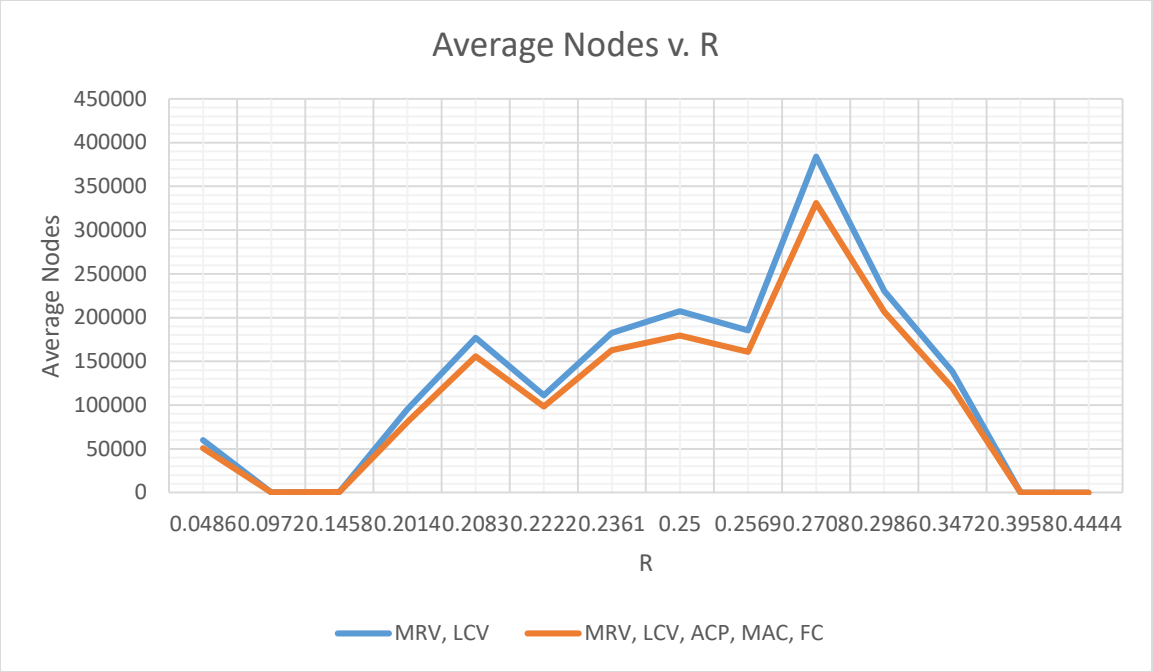
| m | R | Avg. # Nodes | Avg. # Dead Ends | Avg. Time | % Completed | % Solved |
|---|---|---|---|---|---|---|
| 7 | 0.0486 | 50 772 | 50 636 | 20.1611s | 93.3% | 93.3% |
| 14 | 0.0972 | 167 | 37 | 0.1672s | 100% | 100% |
| 21 | 0.1458 | 528 | 405 | 0.3023s | 100% | 100% |
| 29 | 0.2014 | 80 884 | 80 782 | 40.1734s | 86.7% | 80% |

| | | | | | | |
|---|---|---|---|---|---|---|
| 30 | 0.2083 | 155 943 | 155 843 | 72.2758s | 80% | 73.3% |
| 32 | 0.2222 | 98 373 | 98 265 | 49.5991s | 80% | 86.7% |
| 34 | 0.2361 | 162 716 | 162 627 | 80.0514s | 80% | 66.7% |
| 36 | 0.25 | 179 620 | 179 529 | 87.1337s | 83.3% | 66.7% |
| 37 | 0.2569 | 160 918 | 160 836 | 80.1119s | 73.3% | 60% |
| 39 | 0.2708 | 331 069 | 331 003 | 161.637s | 46.7% | 53.3% |
| 43 | 0.2986 | 206 546 | 206 483 | 99.8994s | 73.3% | 46.7% |
| 50 | 0.3472 | 119 644 | 119 628 | 60.2703s | 80% | 66.7% |
| 57 | 0.3958 | 3 | 3 | 0.0131s | 100% | 0% |
| 64 | 0.4444 | 0 | 0 | 0.0046s | 100% | 0% |

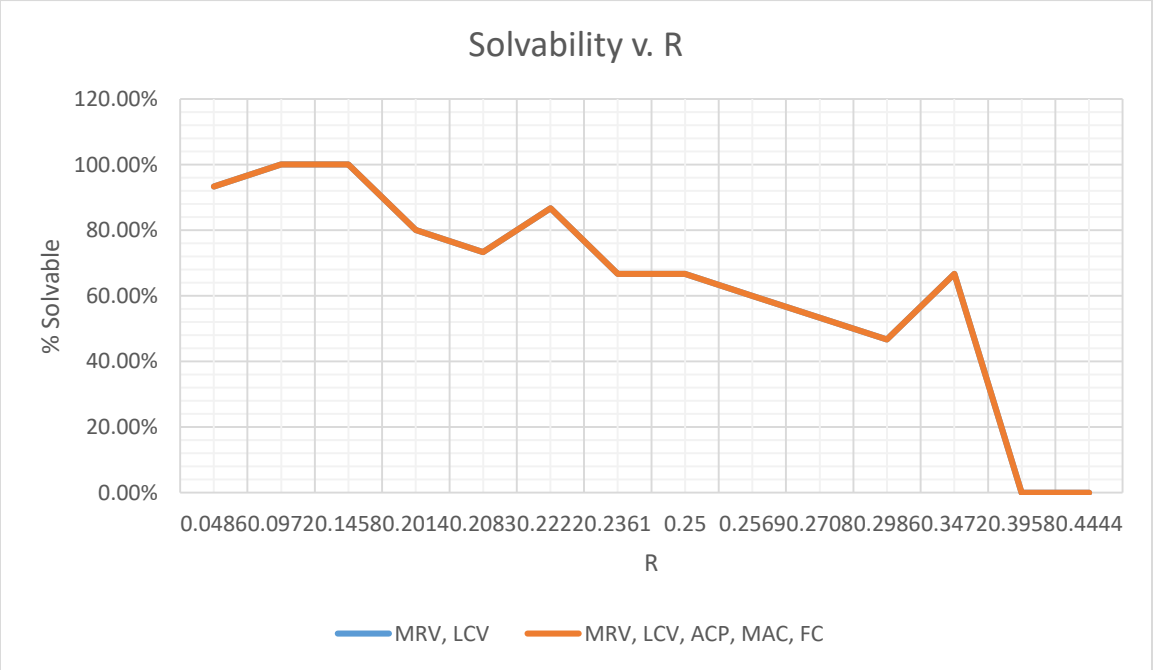
The distribution is similar to finding the hardest R for an 9x9 puzzle: At the low and high ends, the solver completes the puzzle very quickly. As it approaches around m = 39, the solver takes longer and longer to solve. As m increases, the solvability drops to 0% as expected. Comparing the two tables, you can see that the solver with only MRV and LCV heuristics still has a slight time edge over the other. This means that even though the ac3 and forward checking algorithms reduce the search space by eliminating nodes, it does not do this enough to make up for the time overhead. Based on the two tables, the hardest m for a 12x12 puzzle would be around m = 39, or R = 0.2708. This is close to the hardest R = 0.296 for an 9x9 puzzle; there is only a 0.2 difference. This difference can be from the difference in overall difficulties of the puzzles. The hardest R does seem to scale with n; however, it is very likely that the hardest R does not scale 1:1, and may eventually plateau.




As expected, as R approaches the hardest R, the average time increases. Then afterwards, it drops. The low and high R's have low average times.




This follows similarly to the average time, as time taken correlates with number of nodes. As expected, the solver with only the MRV and LCV heuristics has higher average nodes than the other.




The solvability drops to 0% as R increases.

## CONCLUSION

After a series of tests, we can conclude that while the solver does not scale very effectively with the size of the puzzle, it performs reasonably well for regular 9x9 puzzles. Some harder puzzles may take some time, but for most puzzles, the solver should solve it in a short time frame. The solver performs best with simply MRV and LCV heuristics, as the other heuristics do not shorten the search space enough to make up for their time overhead.