Homework 1 Report

R04922034 吳軒衡

Compilation:

Execute “make” will compile the codes that generate the database as well as that solves a given puzzle.

Project Layout:

Hw1-----*5x5---------- |-------bin/ ----- Where binaries are located
| |--------db/ ----- Where precomputed db is located
| |--------testcase/ ----- Where testcases are located
| |--------log/ ---- Whter logs are located
| |--------Makefile
| |--------new_db.cpp ----- size 6 pattern db generator
| |---------run.cpp ----- Problem Solver
4x4---------- |--------bin/ ----- Where binaries are located
|--------db/ ----- Where precomputed db is located
|--------testcase/test.in ----- Where testcases are located
|--------log/ ---- Where logs are located
|--------Makefile
|--------new_db_sub.cpp ----- size 4 pattern db generator
|--------new_run_sub.cpp ----problem solver using size 4 pattern
|--------new_run_db_sub_fh.cpp ---using only first pattern as heuristic
|--------new_run_db_sub_sh.cpp ---using only the second patterns as heursitic
|--------new_db.cpp ----- size 8 pattern db generator
|--------new_run.cpp ----problem solver using size 8 pattern

Execution Notice:

- The precomputed database will be cleared whenever the generator is run
- Make clean will also clear the database
- Testcase is loaded from the firstline in testcase/test.in

Testcase Layout:

In test.in, the first line should be a integer N indicating the number of test cases, limited by 100.

The following N lines will have n^2 numbers that should be filled into an empty board following row order. (The board is 0 base-indexed therefore the number n^2-1 indicates the empty tile.)

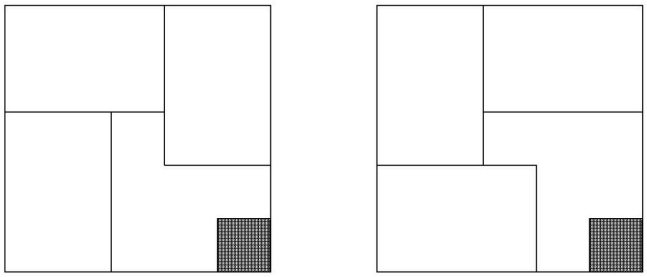
Implementation Detail:

A. Precomputing the Database Patterns

A* with Manhattan distance as heuristic is used as the way to generate the precomputed database.

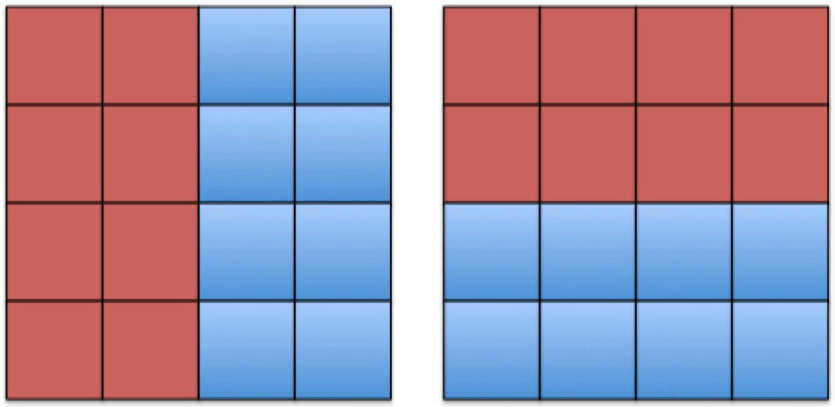
1. 5x5

This implementation basically follows “Disjoint pattern database heuristics “ (Korf 2002) , with the patterns flipped horizontally and vertically according to the normal definition of the problem.



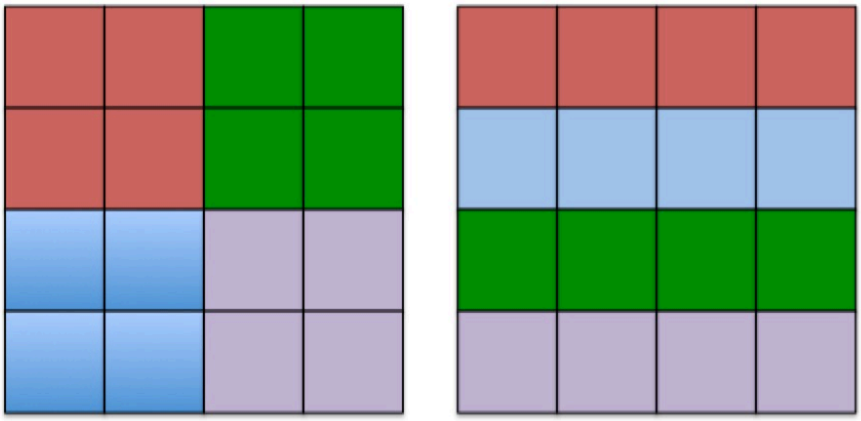
2. 4x4 using 2x4 patterns

In Korf 2002. , disjoint patterns of size 8 and 7 are used, but for the simplicity of implementation, disjoing patterns of size 8 and 8 are used.



3. 4x4 using 2x2 and 1x4 patterns

The use of 2x2 patterns further reduce the number of patterns needed to be generated.



B. Problem Solving:

Perform Iterative Deepening A* using the larger of the sum of two sets of precomputed pattern as heuristic. A Priority is used , and the child of each node is visited in a fix order (by iterating through all possible directions). A close list is also maintained in order to be able to track the previous board . (A print_history function is implemented with recursion in order to replay the history of board movements.

Experiment:

BFS has been tried to generate precomputed database, however, the runtime of the BFS is intolerable. A single run of BFS also fails to generate all additive patterns.

An experiment on 5 random test-cases shows that taking the maximum of two heuristic did improve the performance of IDA*. (h1 is the result of new_run_sub_fh.cpp, using the pattern shown on the left as heuristic while h2 is the result of new_run_sub_sh.cpp using the pattern shown on the right as heuristic).

	max(h1,h2)	h1	h2
1	0.0006	0.00083	0.000669
2	2.311	2.07	12.416741
3	0.000137	0.000142	0.000119
4	0.000079	0.000098	0.00013
5	0.052651	0.100884	0.586012