Easy

Work yourself up, character by character.

For each character, try all letters of the alphabet several times to get significant timing. 10 does the trick!

Check out "Timing attack" on the internet! This is a technique which has actually been demonstrated on many systems!

It elegantly reduces the brute force time from $O(n^m)$ to $O(n*m)$.

Medium

Partitioning

Test each digit on a block of size N, setting all other values to invalid.

If that makes the lock loose test each position of that block one by one.

For a block of N digits you try, in the worst case, once for each of 10 possible digits (whole block), then N for the first digit, N-1 for the second digit etc...

That makes $10 + N(N+1)/2$ for a block of N, or $10/N + (N+1)/2$ per digit.

That makes 5 attempts per digit for a block size of 4 or 5

Hard

Many ways to solve this. Ours (~125 queries) was:

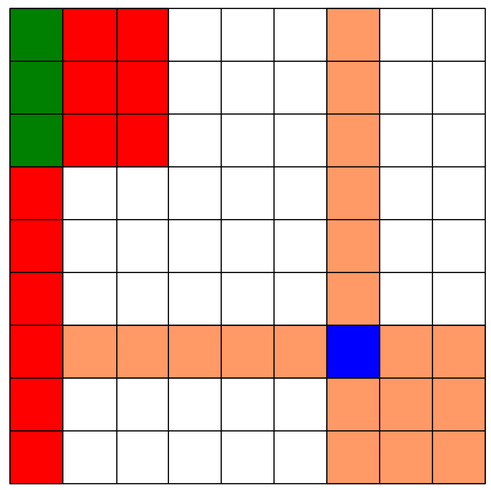
Select queries for maximal information

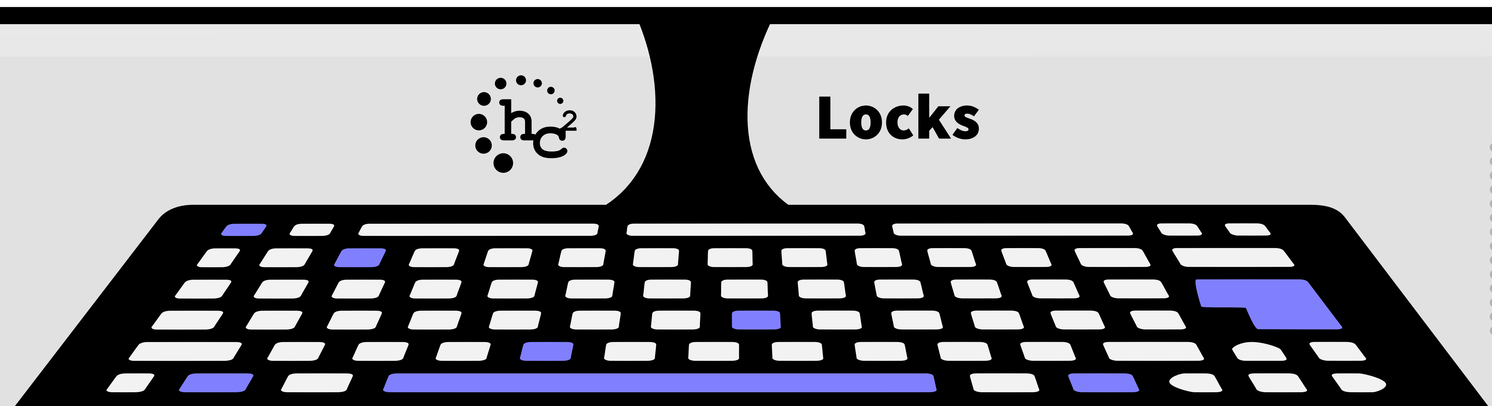
Treat the problem as a Sudoku problem, ask:
What query would give me most information?

Query triples (green) of biggest gain. Information gain = p(number present) * infromation gain in affected fields (red) + p(number absent) * information gain in triple
Information gain per field: A heuristic function which favors reducing possible numbers in field.

Mix with some simple sudoku solver, e.g. if a field is determined (blue), affected fields (orange) can't have its value.

Iterate, backtracking is not required.





AdNovum

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

open systems

PolyProg