Numbermind

Luis Marcos López Casines

University of Padova
Department of Mathematics

June 24, 2022

Problem to solve

Guess a secret code (sequence of digits) by generating guesses with the solutions of a SAT solver.

After each guess we are told how many coincidences there are with the secret code.

There is a coincidence iff both digit and its position are the same in the guess and the secret code.

Approach to solve the problem

- Encode each digit and position into a unique number $(digit \times length(code) + position + 1\ )$.

Approach to solve the problem

- Encode each digit and position into a unique number ($digit \times length(code) + position + 1$).
- Generate first guess from list of encoded numbers.

Approach to solve the problem

- Encode each digit and position into a unique number ($digit \times length(code) + position + 1$).
- Generate first guess from list of encoded numbers.
- Determine coincidences with secret code and:

Approach to solve the problem

- Encode each digit and position into a unique number ($digit \times length(code) + position + 1$).
- Generate first guess from list of encoded numbers.
- Determine coincidences with secret code and:
  - If coincidences = length(code) print *Found*.

Approach to solve the problem

- Encode each digit and position into a unique number ($digit \times length(code) + position + 1$).
- Generate first guess from list of encoded numbers.
- Determine coincidences with secret code and:
  - If coincidences = length(code) print *Found*.
  - If 0 coincidences: pass each of the numbers as negative clauses to the solver, remove all numbers from the encoded list and take new numbers.

Approach to solve the problem

- Encode each digit and position into a unique number ($digit \times length(code) + position + 1$).
- Generate first guess from list of encoded numbers.
- Determine coincidences with secret code and:
  - If coincidences = length(code) print *Found*.
  - If 0 coincidences: pass each of the numbers as negative clauses to the solver, remove all numbers from the encoded list and take new numbers.
  - Else add clauses according to the number of matches and build the new guess out of the satisfiable solutions of the solver.

Approach to solve the problem

- Encode each digit and position into a unique number ($digit \times length(code) + position + 1$).
- Generate first guess from list of encoded numbers.
- Determine coincidences with secret code and:
  - If coincidences = length(code) print *Found*.
  - If 0 coincidences: pass each of the numbers as negative clauses to the solver, remove all numbers from the encoded list and take new numbers.
  - Else add clauses according to the number of matches and build the new guess out of the satisfiable solutions of the solver.
- Keep iterating until the secret code is found.

Cardinality constraint

Exactly k

The constraint "exactly k propositional variables in X are true" can be rephrased as the conjunction of "at least k" and "at most k".

$$\bigwedge_{\substack{I\subseteq[n]\\|I|=n-k+1}} \bigvee x_i \wedge \bigwedge_{\substack{I\subseteq[n]\\|I|=k+1}} \bigvee_{i\in I} \neg x_i$$

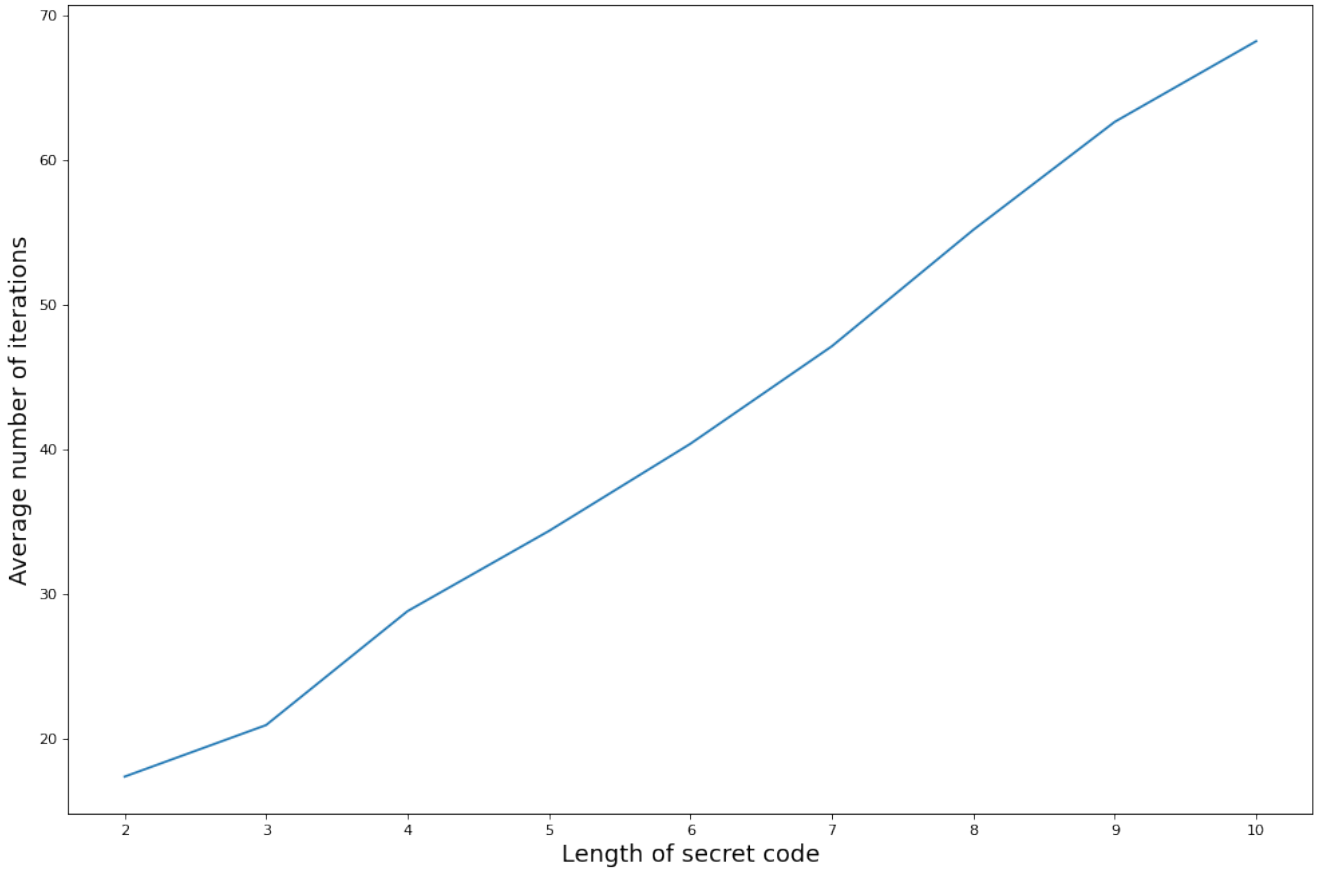
Example (Exactly 2 among X={a, b, c, d})

$$(a \vee b \vee c) \wedge (a \vee b \vee d) \wedge (b \vee c \vee d) \wedge (\neg a \vee \neg b \vee \neg c) \wedge$$

$$(\neg a \vee \neg b \vee \neg d) \wedge (\neg a \vee \neg c \vee \neg d) \wedge (\neg b \vee \neg c \vee \neg d)$$

Results and Conclusions

- Secret code is correctly predicted.
- Average number of iterations with respect to length of secret code:



Thank you for your attention