UNIVERSITÀ
DI TRENTO

Formal Method Mod. 1 (Automated Reasoning)

Laboratory 4

Giuseppe Spallitta
giuseppe.spallitta@unitn.it

Università degli studi di Trento

March 27, 2024

UNIVERSITÀ DEGLI STUDI
DI TRENTO

Outline

1. Advanced SMT solving
   - Cybersecurity applications
   - Private investigations
   - Pseudo-Boolean for SMT solving
2. Homeworks

Black hat hacker

UNIVERSITÀ DEGLI STUDI DI TRENTO

Exercise 4.1: hacking key

You want to access the UniTN database. Sadly the server is protected by a key. From reverse engineering you obtain the following part of code executed by the machine:

```
% Key is the concat of 3 32-bit numbers a,b and c
assert(isMultiple(a,5))
assert(or(a,b) == 2022))
assert(a - b > 1000)
assert(isAverage(c, [a,b]))
assert(c<0x76543210)
login()
```

Given you have one opportunity to log in and that if you fail you will be expelled, can you guess the key?

UNIVERSITÀ DEGLI STUDI DI TRENTO

Black hat hacker: properties

- Properties are trivial for most of the part, since they simply require to encode the content of the Python instructions *assert*.
- Be careful: we work with bit vectors, so do not forget to use the correct operators.
- Moreover, be sure that integers used as constants are also treated as bit vector (MathSAT does not provide implicit type conversion :( )

UNIVERSITÀ DEGLI STUDI DI TRENTO

Black hat hacker: constant conversion

- The simplest alternative is directly setting the number using the instruction:
  (_ bv<number> <size>)
- But when we manage negative numbers, `bv-1` does not work, so we require a different instruction, which maps integers to their equivalent BV representation assuming the size chosen is high enough:
  (_ to_bv 3) (- 2)
- You can also write numbers using the hexadecimal or binary representation (convenient when dealing with low numbers of bits) using prefixing respectively *#x* or *#b*.

Black hat hacker: variables

UNIVERSITÀ DEGLI STUDI DI TRENTO

As always, we first define the variables that efficiently describe the problem:

- 3 variables are necessary to store the three sub-parts of the entire key.
- The comment highlights that they are Bit vectors, so the type is also clearly defined.

UNIVERSITÀ DEGLI STUDI DI TRENTO

Black hat hacker: functions

- No function is mandatory for this problem; the two high-level operations can be encoded as functions if desired.
- *isMultiple* can be defined as a 2-arity function **(BitVector, Int) $\Rightarrow$ Bool**.
- *isAverage* can be defined as a 3-arity function **(BitVector, BitVector, BitVector) $\Rightarrow$ Bool**.
- Other than that,the encoding is simply to write using SMT-LIB functions.

Private investigations

UNIVERSITÀ DEGLI STUDI DI TRENTO

Exercise 4.2: who killed Agatha?

Someone who lives in Dreadbury Mansion killed Aunt Agatha. Agatha, the butler, and Charles live in Dreadbury Mansion, and are the only people who live therein. A killer always hates his victim, and is never richer than his victim. Charles hates no one that Aunt Agatha hates. Agatha hates everyone except the butler. The butler hates everyone not richer than Aunt Agatha. The butler hates everyone Aunt Agatha hates. No one hates everyone. Who killed Agatha?

First approach: without quantifiers

- MathSAT does not natively support quantifiers!
- If we iterate through all variables, we can simulate the behaviour of $\exists$ and $\forall$ thanks to the **Shannon expansion**.
- Can we use functions that maps variables to a truth value, in order to set if a relation holds?

Private investigations: variables

UNIVERSITÀ DEGLI STUDI DI TRENTO

First we must define constants, predicates and functions that efficiently describe the problem:

- **Constants**: agatha, butler, charles (for each person, I use a different `Int`.
- **Predicates/Functions**: hates(X,Y), richer(X,Y)

The killer will be an `Int` variable, constrained by all the conditions stated in the problem and the hidden ones.

UNIVERSITÀ DEGLI STUDI DI TRENTO

Private investigations: SMT formulae (1)

- A killer always hates his victim...
  $hates(killer, agatha))$
- and is never richer than his victim.
  $\neg\ richer(killer, agatha))$
- Charles hates no one that Aunt Agatha hate.
  $\forall x.(hates(agatha, x) \rightarrow \neg\ hates(charles, x)))$

Private investigations: SMT formulae (2)

UNIVERSITÀ DEGLI STUDI DI TRENTO

- Agatha hates everyone except the butler.
  $\forall x.(x \mathrel{!}= butler \rightarrow (hates(agatha, x))$
- The butler hates everyone not richer than Aunt Agatha
  $\forall x.(\neg\ richer(x, agatha) \rightarrow (hates(butler, x))$
- The butler hates everyone Aunt Agatha hates.
  $\forall x.(hates(agatha, x) \rightarrow hates(butler, x)))$
- No one hates everyone.
  $\forall x \exists y.(\neg\ hates(x,y))$

Private investigations: hidden conditions

UNIVERSITÀ DEGLI STUDI DI TRENTO

Richer is a function such as:

- It is non-reflexive:
  $\forall x.(\neg richer(x, x))$
- It is non-symmetric:
  $\forall xy.(richer(x, y) \leftrightarrow \neg richer(y, x))$

UNIVERSITÀ DEGLI STUDI DI TRENTO

Defining functions

- Every time you define a function mapping one or more values to a Boolean or a sorted type, be sure to encode also its hidden properties; this could drastically change the behaviour of the solver!
- A brief list includes:
  - (non-)Simmetry
  - (non-)Reflexivity
  - (non-)Transitivity

UNIVERSITÀ DEGLI STUDI
DI TRENTO

Second approach: with quantifiers

- Z3 supports quantifiers!
- pysmt offers the two shortcuts: **ForAll** e **Exists**, accepting the list of quantified variables and the formula.
- Can we use Z3 to easily encode the same problem?

Checking algorithms

UNIVERSITÀ DEGLI STUDI DI TRENTO

Exercise 4.3: check security... again

You are given the following piece of code that analyzes a 5-digit number, between 10000 and 99999 where all digits are different:

```
void check_security(num) {
        security = 4
        if num is multiple of 3 or 5, but not both  {
            security = security - 1
        }
        if the sum of digits is a multiple of 10 {
            security = security - 1
        }
        if the number is palindrome {
            security = security - 1
        }
        if the digits are in ascending order (including equality) {
            security = security - 1
        }
        return security
    }
```

You wonder if there exists an input that provides the security value 2. Can you get it using MathSAT?

Pseudo-Boolean variables

- On one hand, we could set one *ExactlyTwo* condition having a Boolean variable for each condition: but what would happen if we had an *ExactlyFour*, do you really want to encode it?
- The idea is having a counter so that every time a condition is falsified it is incremented by 1;
- To this extent, it is useful to map Booleans to 0-1 values so that getting their sum answer our *ExactlyTwo* conditions!

UNIVERSITÀ DEGLI STUDI DI TRENTO

Pseudo-Boolean variables: how to implement it

- **Solution 1**: define each Bool variable as Int, bounding them to the range 0-1.
- **Solution 2**: define each variable as Bool B_i and then use a sum of (ite B_i 1 0) to easily get their combined sum.

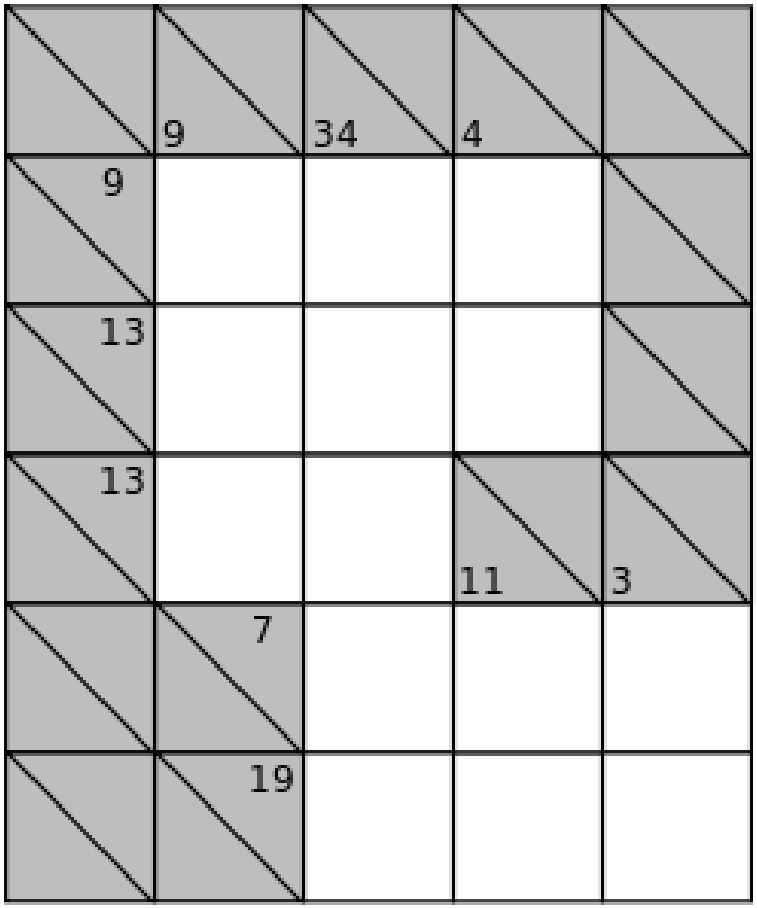
UNIVERSITÀ DEGLI STUDI
DI TRENTO

Outline

1. Advanced SMT solving
2. Homeworks

Solving Kakuro

Homework 4.1: kakuro



Kakuro is a puzzle in which one must put the numbers 1 to 9 in the different cells such that they satisfy certain constraints. If a clue is present in a row or column, the sum of the cell for that row should be equal to the value. Within each sum all the numbers have to be different, so to add up to 4 we can have 1+3 or 3+1. Can we find a solution using SMT solvers?

Homeworks

UNIVERSITÀ DEGLI STUDI DI TRENTO

Exercise 4.2: task manager

Your PC needs to complete 5 different tasks (A,B,C,D and E) to correctly save a file. There are some constraints about the order execution of the tasks:

- We can execute A after D is completed.
- We can execute B after C and E are completed.
- We can execute E after B or D are completed.
- We can execute C after A is completed.

Which is the task that will execute for last?