[Computational Logic Lab](https://drive.google.com/drive/folders/14HEbdjOlDulRVLwaqC0PLswX6fp2X__d?usp=sharing)

20/10/2021

# (1)

Use the SMT-solver z3 to solve the following problems.

(1) Check that the following formula is a tautology:

(p ∨ q → r) → (p → r) ∧ (q → r) .

; check tautology (p ∨ q → r) → (p → r) ∧ (q → r)

;declare 3 proposotional letters

(declare-const p Bool)

(declare-const q Bool)

(declare-const r Bool)

;negate everything

(assert(not(and (=> (=> (and p q) r) (=> p r)) (=> q r))))

(check-sat)

(get-model)

# (2)

(2) Show that 3 colors are not sufficient to color the part of the European

map that includes the following set of countries: France, Germany, Belgium,

Luxembourg, Netherlands.

[Hint: see the course notes.]

(3) In a pigeon-hole problem,1 we are given n pigeons and k holes. A pigeon

cannot stay inside two holes; each hole can contain only one pigeon and each

pigeon must stay inside some hole. In addition there are specific require-

ments, to be tested for consistency. In our case, we have n = 4, k = 4 and

the following requirements:

- pigeon 1 does not like holes 2 and 3;

- pigeon 2 does not like holes 3 and 4;

- pigeon 3 does not like holes 1 and 2:

- pigeon 4 does not like holes 3 and 4.

Check that the above problem has a solution and supply the solution.

[Hint: Use propositional letters P ij to express that pigeon i is in hole

j (for i, j = 1, . . . , 4). Then assert that each pigeon must stay in exactly

one hole.2 Then, with the help of another macro,3 assert that each hole can

contain at most one pigeon. Finally, formalize the above constraints.]

;pigeon hole problem

; 4 pigeons, 4 holes

;pigeon hole problem

; 4 pigeons, 4 holes

(declare-const P1H1 Bool)

(declare-const P1H2 Bool)

(declare-const P1H3 Bool)

(declare-const P1H4 Bool)

(declare-const P2H1 Bool)

(declare-const P2H2 Bool)

(declare-const P2H3 Bool)

(declare-const P2H4 Bool)

(declare-const P3H1 Bool)

(declare-const P3H2 Bool)

(declare-const P3H3 Bool)

(declare-const P3H4 Bool)

(declare-const P4H1 Bool)

(declare-const P4H2 Bool)

(declare-const P4H3 Bool)

(declare-const P4H4 Bool)

;exactly one out of three

(define-fun xor4

((x Bool) (y Bool) (z Bool) (w Bool))

Bool

(and

(or x y z w)

(not (and x y))

(not (and x z))

(not (and x w))

(not (and y z))

(not (and y w))

(not (and z w))

)

)

(assert (xor4 P1H1 P1H2 P1H3 P1H4))

(assert (xor4 P2H1 P2H2 P2H3 P2H4))

(assert (xor4 P3H1 P3H2 P3H3 P3H4))

(assert (xor4 P4H1 P4H2 P4H3 P4H4))

(assert (xor4 P1H1 P2H1 P3H1 P4H1))

(assert (xor4 P1H2 P2H2 P3H2 P4H2))

(assert (xor4 P1H3 P2H3 P3H3 P4H4))

(assert (xor4 P1H4 P2H4 P3H4 P4H4))

(assert (and (not P1H2) (not P1H3)))

(assert (and (not P2H3) (not P2H4)))

(assert (and (not P3H1) (not P3H2)))

(assert (and (not P4H3) (not P4H4)))

(check-sat)

(get-model)

# (4)

(4) A group of friends formed by A, B, C, D, E, wants to find an accommo-

dation in a hotel. In the hotel, only three rooms are available, all such rooms

can accommodate two people. These are the friends’ desiderata:

1This is a classical challenging problem per SAT solvers (it is often practically

impossible to solve it for large n, k).

2Use a 4-variables exclusive-or macro xor4 similar to the 3-variables exclusive-or

macro xor3 introduced for last week exercises.

3Such macro says that, among the propositional variables x, y, z, u, at most one of

them hold.

- A wants to share the room with either B or E;

- B wants to share the room with either A or C;

- C wants to share the room with either B or D;

- D wants to share the room with either C or E;

- E wants to share the room with either D or A.

Check whether these requirements are consistent or not.

[Hint: use propositional letters A1, A2, A3 to express that A is accom-

modated in room 1, 2, 3, respectively. Do the same for B, C, D, E. Then

write down a formula saying that A is accommodated in exactly one room,

B is accommodated in exactly one room, etc.4 With the help of a suitable

macro,5 write down formulae saying that room 1 is occupied by at most 2

people, room B is occupied by at most 2 people, room 3 is occupied by at most

two people. Finally formalize the above requirements.]

; declare propositional letter

(declare-const A1 Bool)

(declare-const A2 Bool)

(declare-const A3 Bool)

(declare-const B1 Bool)

(declare-const B2 Bool)

(declare-const B3 Bool)

(declare-const C1 Bool)

(declare-const C2 Bool)

(declare-const C3 Bool)

(declare-const D1 Bool)

(declare-const D2 Bool)

(declare-const D3 Bool)

(declare-const E1 Bool)

(declare-const E2 Bool)

(declare-const E3 Bool)

;exactly one out of three

(define-fun xor3

((x Bool) (y Bool) (z Bool))

Bool

(and

(or x y z)

(not (and x y))

(not (and x z))

(not (and y z))

)

)

(assert (xor3 A1 A2 A3))

(assert (xor3 B1 B2 B3))

(assert (xor3 C1 C2 C3))

(assert (xor3 D1 D2 D3))

(assert (xor3 E1 E2 E3))

; A wants to share the room with either B or E;

(assert (xor (xor3 (and A1 B1) (and A2 B2) (and A3 B3)) (xor3 (and A1 E1) (and A2 E2) (and A3 E3))))

; B wants to share the room with either A or C;

(assert (xor (xor3 (and B1 A1) (and B2 A2) (and B3 A3)) (xor3 (and B1 C1) (and B2 C2) (and B3 C3))))

; C wants to share the room with either B or D

(assert (xor (xor3 (and C1 B1) (and C2 B2) (and C3 B3)) (xor3 (and C1 D1) (and C2 D2) (and C3 D3))))

; - D wants to share the room with either C or E;

(assert (xor (xor3 (and D1 C1) (and D2 C2) (and D3 C3)) (xor3 (and D1 E1) (and D2 E2) (and D3 E3))))

;- E wants to share the room with either D or A.

(assert (xor (xor3 (and E1 D1) (and E2 D2) (and E3 D3)) (xor3 (and E1 A1) (and E2 A2) (and E3 A3))))

(check-sat)

(get-model)