Proiect Inteligenta Artificiala 2

Marcu Mihai-Alexandru

08.12.2020

1 Introducere

In acest proiect sunt prezentate rezolvarile in interpretorul Prover9 a trei tipuri de puzzle-uri.

2 Problema 1: Gas Station Puzzle

2.1 Cerinta:

La o benzinarie se afla cinci masini diferite. Fiecare masina are ca si caracteristici un sofer, o culoare, un tip, un kilometraj, pretul care a fost platit pentru a alimenta si un producator. Sa se afle pentru fiecare masina ce caracteristici ii apartin.
Se dau urmatoarele indicii:
-Masina care va alimenta de 20 dolari se afla undeva intre masina lui Daniel si masina care va alimenta de 5 dolari, in aceasta ordine.
-Masina lui Alex se afla pe a 5-a pozitie.
-Masina galbena este exact in stanga masinii cu 30000 de mile.
-Masina produsa in Italia este intr-unul din capete.
-Masina cu 40000 de mile se afla undeva intre masina care va alimenta de 25 dolari si masina produsa in Franta, in aceasta ordine.
-Masina de tip sedan este exact in stanga masinii cu cele mai multe mile.
-Masina lui Jacob se afla pe a 4-a pozitie.
-Masina albastra este pozitionata undeva intre masina rosie si masina produsa in Italia, in aceasta ordine.
-Masina produsa in Korea se afla pe a 4-a pozitie.
-Crossover-ul se afla langa masina cu 30000 de mile.
-Masina cu 40000 de mile se alfa langa masina care va alimenta de 5 dolari.
-SUV-ul va alimenta de 5 dolari.
-Masina produsa in Germania se afla undeva intre masina produsa in America si cea produsa in Franta, in aceasta ordine.
-Masina lui Jacob este langa SUV.
-Intr-unul dintre capete se afla masina care a alimentat de 15 dolari.
-Masina albastra este undeva intre Pickup si masina produsa in Franta, in aceasta ordine.
-Masina cu 20000 de mile se afla pe a 4-a pozitie.
-Masina produsa in America este intr-unul dintre capete.
-Masina galbena este exact in dreapta masinii albe.
-Masina lui William este undeva intre masina produsa in America si masina lui Logan, in aceasta ordine.
-Pe a 5-a pozitie se afla masina cu 30000 de mile.

2.2 Cod:

```
% Saved by Prover9-Mace4 Version 0.5, December 2007.

set(ignore_option_dependencies). % GUI handles dependencies
```

```
4
5  if(Prover9). % Options for Prover9
6    assign(max_seconds, 60).
7  end_if.
8
9  if(Mace4).   % Options for Mace4
10   assign(max_seconds, 60).
11 end_if.
12
13 formulas(assumptions).
14
15 differentFrom(a,b).
16 differentFrom(a,c).
17 differentFrom(a,d).
18 differentFrom(a,e).
19 differentFrom(b,c).
20 differentFrom(b,d).
21 differentFrom(b,e).
22 differentFrom(c,d).
23 differentFrom(c,e).
24 differentFrom(d,e).
25
26 differentFrom(x,y)->differentFrom(y,x).
27
28 rightNeighbour(a,b).
29 rightNeighbour(b,c).
30 rightNeighbour(c,d).
31 rightNeighbour(d,e).
32
33 -rightNeighbour(a,a).
34 -rightNeighbour(a,c).
35 -rightNeighbour(a,d).
36 -rightNeighbour(a,e).
37
38 -rightNeighbour(b,a).
39 -rightNeighbour(b,b).
40 -rightNeighbour(b,d).
41 -rightNeighbour(b,e).
42
43 -rightNeighbour(c,a).
44 -rightNeighbour(c,b).
45 -rightNeighbour(c,c).
46 -rightNeighbour(c,e).
47
48 -rightNeighbour(d,a).
49 -rightNeighbour(d,b).
50 -rightNeighbour(d,c).
51 -rightNeighbour(d,d).
52
53 -rightNeighbour(e,a).
54 -rightNeighbour(e,b).
55 -rightNeighbour(e,c).
56 -rightNeighbour(e,d).
57 -rightNeighbour(e,e).
```

```
rightNeighbour(x,y) | rightNeighbour(y,x)<->neighbour(x,y).

between(a,b,c).
between(a,b,d).
between(a,b,e).
between(a,c,d).
between(a,c,e).
between(a,d,e).
between(b,c,d).
between(b,c,e).
between(b,d,e).
between(c,d,e).

-between(a,a,a).
-between(a,a,b).
-between(a,a,c).
-between(a,a,d).
-between(a,a,e).
-between(a,b,a).
-between(a,b,b).
-between(a,c,a).
-between(a,c,b).
-between(a,c,c).
-between(a,d,a).
-between(a,d,b).
-between(a,d,c).
-between(a,d,d).
-between(a,e,a).
-between(a,e,b).
-between(a,e,c).
-between(a,e,d).
-between(a,e,e).
-between(b,a,a).
-between(b,a,b).
-between(b,a,c).
-between(b,a,d).
-between(b,a,e).
-between(b,b,a).
-between(b,b,b).
-between(b,b,c).
-between(b,b,d).
-between(b,b,e).
-between(b,c,a).
-between(b,c,b).
-between(b,c,c).
-between(b,d,a).
-between(b,d,b).
-between(b,d,c).
-between(b,d,d).
-between(b,e,a).
-between(b,e,b).
-between(b,e,c).
-between(b,e,d).
-between(b,e,e).
```

```
-between(c,a,a).
-between(c,a,b).
-between(c,a,c).
-between(c,a,d).
-between(c,a,e).
-between(c,b,a).
-between(c,b,b).
-between(c,b,c).
-between(c,b,d).
-between(c,b,e).
-between(c,c,a).
-between(c,c,b).
-between(c,c,c).
-between(c,c,d).
-between(c,c,e).
-between(c,d,a).
-between(c,d,b).
-between(c,d,c).
-between(c,d,d).
-between(c,e,a).
-between(c,e,b).
-between(c,e,c).
-between(c,e,d).
-between(c,e,e).
-between(d,a,a).
-between(d,a,b).
-between(d,a,c).
-between(d,a,d).
-between(d,a,e).
-between(d,b,a).
-between(d,b,b).
-between(d,b,c).
-between(d,b,d).
-between(d,b,e).
-between(d,c,a).
-between(d,c,b).
-between(d,c,c).
-between(d,c,d).
-between(d,c,e).
-between(d,d,a).
-between(d,d,b).
-between(d,d,c).
-between(d,d,d).
-between(d,d,e).
-between(d,e,a).
-between(d,e,b).
-between(d,e,c).
-between(d,e,d).
-between(d,e,e).
-between(e,a,a).
-between(e,a,b).
-between(e,a,c).
-between(e,a,d).
-between(e,a,e).
```

```
-between(e,b,a).
-between(e,b,b).
-between(e,b,c).
-between(e,b,d).
-between(e,b,e).
-between(e,c,a).
-between(e,c,b).
-between(e,c,c).
-between(e,c,d).
-between(e,c,e).
-between(e,d,a).
-between(e,d,b).
-between(e,d,c).
-between(e,d,d).
-between(e,d,e).
-between(e,e,a).
-between(e,e,b).
-between(e,e,c).
-between(e,e,d).
-between(e,e,e).

Alex(x) | Daniel(x) | Jacob(x) | Logan(x) | William(x).
blue(x) | green(x) | red(x) | white(x) | yellow(x).
crossover(x) | hatchback(x) | pickup(x) | sedan(x) | suv(x).
10k(x) | 20k(x) | 30k(x) | 40k(x) | 50k(x).
$5(x) | $10(x) | $15(x) | $20(x) | $25(x).
American(x) | French(x) | German(x) | Italian(x) | Korean(x).

Alex(x) & Alex(y)-> -differentFrom(x,y).
Daniel(x) & Daniel(y)-> -differentFrom(x,y).
Jacob(x) & Jacob(y)-> -differentFrom(x,y).
Logan(x) & Logan(y)-> -differentFrom(x,y).
William(x) & William(y)-> -differentFrom(x,y).
blue(x) & blue(y)-> -differentFrom(x,y).
green(x) & green(y)-> -differentFrom(x,y).
red(x) & red(y)-> -differentFrom(x,y).
white(x) & white(y)-> -differentFrom(x,y).
yellow(x) & yellow(y)-> -differentFrom(x,y).
crossover(x) & crossover(y)-> -differentFrom(x,y).
hatchback(x) & hatchback(y)-> -differentFrom(x,y).
pickup(x) & pickup(y)-> -differentFrom(x,y).
sedan(x) & sedan(y)-> -differentFrom(x,y).
suv(x) & suv(y)-> -differentFrom(x,y).
10k(x) & 10k(y)-> -differentFrom(x,y).
20k(x) & 20k(y)-> -differentFrom(x,y).
30k(x) & 30k(y)-> -differentFrom(x,y).
40k(x) & 40k(y)-> -differentFrom(x,y).
50k(x) & 50k(y)-> -differentFrom(x,y).
$5(x) & $5(y)-> -differentFrom(x,y).
$10(x) & $10(y)-> -differentFrom(x,y).
$15(x) & $15(y)-> -differentFrom(x,y).
$20(x) & $20(y)-> -differentFrom(x,y).
$25(x) & $25(y)-> -differentFrom(x,y).
American(x) & American(y)-> -differentFrom(x,y).
```

```
220 French(x) & French(y)-> -differentFrom(x,y).
221 German(x) & German(y)-> -differentFrom(x,y).
222 Italian(x) & Italian(y)-> -differentFrom(x,y).
223 Korean(x) & Korean(y)-> -differentFrom(x,y).
224
225 Daniel(x) & $20(y) & $5(z)->between(x,y,z).
226 Alex(e).
227 yellow(x) & 30k(y)->rightNeighbour(x,y).
228 Italian(a) | Italian(e).
229 $25(x) & 40k(y) & French(z)->between(x,y,z).
230 sedan(x) & 50k(y)->rightNeighbour(x,y).
231 Jacob(d).
232 red(x) & blue(y) & Italian(z)->between(x,y,z).
233 Korean(d).
234 crossover(x) & 30k(y)->neighbour(x,y).
235 40k(x) & $5(y)->neighbour(x,y).
236 suv(x)<->$5(x).
237 American(x) & German(y) & French(z)->between(x,y,z).
238 Jacob(x) & suv(y)->neighbour(x,y).
239 $15(a) | $15(e).
240 pickup(x) & blue(y) & French(z)->between(x,y,z).
241 20k(d).
242 American(a) | American(e).
243 white(x) & yellow(y)->rightNeighbour(x,y).
244 American(x) & William(y) & Logan(z)->between(x,y,z).
245 30k(e).
246
247 end_of_list.
248
249 formulas(goals).
250
251 end_of_list.
```

2.3 Analiza cod:

Intre linia de cod 15 si linia de cod 26 este implementata functie differentFrom. De la linia 28 pana la linia 57 este definita functia pentru vecinul din dreapta: rightNeighbour, dupa care este definita functia de vecini: neighbour. In continuare, in intervalul liniilor 60-185 este definita functia between. De la linia 194 pana la linia 223 este pusa conditia ca o caracteristica sa nu se repete pentru doua masini. Intre linia 225 si linia 245 sunt traduse indiciile date.

2.4 Solutie:

	Car #1	Car #2	Car #3	Car #4	Car #5
Color	red	blue	white	yellow	green
Driver	Daniel	William	Logan	Jacob	Alex
Type	pickup	sedan	SUV	crossover	hatchback
Mileage	10,000	40,000	50,000	20,000	30,000
Gasoline	$ 25	$ 20	$ 5	$ 10	$ 15
Manufacturer	American	German	French	Korean	Italian

```
interpretation( 5, [number = 1,seconds = 0], [
    function(a, [0]),
    function(b, [1]),
    function(c, [2]),
    function(d, [3]),
    function(e, [4]),
    relation($10(_), [0,0,0,1,0]),
    relation($15(_), [0,0,0,0,1]),
    relation($20(_), [0,1,0,0,0]),
    relation($25(_), [1,0,0,0,0]),
    relation($5(_), [0,0,1,0,0]),
    relation(10k(_), [1,0,0,0,0]),
    relation(20k(_), [0,0,0,1,0]),
    relation(30k(_), [0,0,0,0,1]),
    relation(40k(_), [0,1,0,0,0]),
    relation(50k(_), [0,0,1,0,0]),
    relation(Alex(_), [0,0,0,0,1]),
    relation(American(_), [1,0,0,0,0]),
    relation(Daniel(_), [1,0,0,0,0]),
    relation(French(_), [0,0,1,0,0]),
    relation(German(_), [0,1,0,0,0]),
    relation(Italian(_), [0,0,0,0,1]),
    relation(Jacob(_), [0,0,0,1,0]),
    relation(Korean(_), [0,0,0,1,0]),
    relation(Logan(_), [0,0,1,0,0]),
    relation(William(_), [0,1,0,0,0]),
    relation(blue(_), [0,1,0,0,0]),
    relation(crossover(_), [0,0,0,1,0]),
    relation(green(_), [0,0,0,0,1]),
    relation(hatchback(_), [0,0,0,0,1]),
    relation(pickup(_), [1,0,0,0,0]),
    relation(red(_), [1,0,0,0,0]),
    relation(sedan(_), [0,1,0,0,0]),
    relation(suv(_), [0,0,1,0,0]),
    relation(white(_), [0,0,1,0,0]),
    relation(yellow(_), [0,0,0,1,0]),
```

2.5 Explicatie:

In urmatorul paragraf voi descrie procesul de gandire care a condus la solutie:
Din al doilea si din ultimul indiciu reiese faptul ca masina de pe a 5-a pozitie este detinuta de Alex si are kilometrajul de 30000 de mile. Asemenea se deduce si faptul ca a 4-a masina ii apartine lui Jacob, are 20000 de mile si este produsa in Korea. Stim faptul ca masina de tip crossover se afla langa masina cu 30000 de mile, de aici rezultand faptul ca masina numarul 4 este crossover, deoarece masina cu 30000 de mile este in capat, pe pozitia 5. De asemenea din al 3-lea indiciu reiese faptul ca masina numarul 4 este galbena. Stim ca masina galbena este in dreapta celei albe, rezulta ca masina numarul 3 este alba. Deoarece masina din Germania este pozitionata intre cea din America si cea din Franta, iar cea din America si cea din Italia se afla in colturi, rezulta faptul ca masina numarul 1 este din America si cea numarul 5 este din Italia. Aditional stim ca masina numarul 4 este din Korea, iar deoarece cea din Germania este inaintea celeia din Franta rezulta faptul ca masina numarul 2 este produsa in Germania, iar masina numarul 3 in Franta. Deoarece masina albastra se afla intre masina rosie si cea din Italia, iar cea din Italia este pe ultima pozitie, reiese ca masina rosie se afla pe prima pozitie, cea albastra pe a doua, iar cea verde pe ultima. Deoarece masina albastra se afla intre pickup si cea din Franta, rezulta faptul ca masina numarul 1 este un pickup. Din penultimul indiciu reiese faptul ca masina lui William este cea de pe a doua pozitie si faptul ca cea a lui Logan este pe a treia pozitie, astfel prima masina apartinandu-i lui Daniel. Se stie ca masina cu 40000 de mile se afla intre masina care alimenteaza de 25 de dolari si cea produsa in Franta, rezultand ca masina de pe a doua pozitie are kilometrajul de 40000 de mile si ca prima masina alimenteaza de 25 de dolari. Fiindca se stie ca prima masina alimenteaza de 25 de dolari, iar cea de-a doua are 40000 de mile, din al 11-lea indiciu reiese faptul ca a treia masina alimenteaza de 5 dolari, aceasta fiind un SUV, fapt ce reiese din alt indiciu. Din primul indiciu reiese faptul ca cea de-a doua masina va alimenta de 20 de dolari. Masina de pe ultima pozitie va alimenta de 15 dolari, iar prin proces eliminatoriu rezulta ca cea de-a 4 masina va alimenta de 10 dolari. Deoarece al 6-lea indiciu spune ca sedanul se afla in stanga masinii cu 50000 de mile, rezulta faptul ca aceasta masina cu 50000 de mile nu poate fi pe prima pozitie, fiind pe cea de a treia, sedanul fiind pe a doua, rezultand ca prima masina are 10000 de mile prin proces eliminatoriu. De asmenea ramane faptul ca masina de pe ultima pozitie este un hatchback.

3 Problema 2: Cavaleri, talhari si spioni.

3.1 Cerinta:

Pe o insula se afla trei tipuri de persoane: cavaleri, talhari si spioni. Cavalerii spun doar adevarul, talharii mint intotdeauna, iar spioni raspund ori cu adevar ori cu minciuna, depinzand de vointa lor.Explorand insula, intalnim trei oameni: unul imbracat in rosu, unul in albastru si unul in verde. Se stie ca unul este un cavaler, unul este un talhar si unul este un spion. Din pura curiozitate caracterul nostru isi pune intrebarea urmatoare: "Cine este spionul?".
Interactiunile cu cei trei locatari sunt urmatoarele:
-Omul imbracat in albastru spune: "Omul imbracat in rosu este spionul!".
-Omul imbracat in rosu spune: "Nu! Omul imbracat in verde este spionul!".
-Omul imbracat in verde spune: "Intr-adevar omul imbracat in rosu este spionul!".

3.2 Cod:

```
% Saved by Prover9-Mace4 Version 0.5, December 2007.

set(ignore_option_dependencies). % GUI handles dependencies

if(Prover9). % Options for Prover9
  assign(max_seconds, 60).
end_if.
```

```
8
9  if(Mace4).   % Options for Mace4
10   assign(max_seconds, 60).
11 end_if.
12
13 formulas(assumptions).
14
15 knight(blue) | knight(red) | knight(green).
16 knave(blue) | knave(red) | knave(green).
17 spy(blue) | spy(red) | spy(green).
18
19 spy(x)->-knight(x) & -knave(x).
20 knight(x)->-spy(x) & -knave(x).
21 knave(x)->-knight(x) & -spy(x).
22
23 knight(blue)->spy(red).
24 knave(blue)->-spy(red).
25 knight(red)->spy(green).
26 knave(red)->-spy(green).
27 knight(green)->spy(red).
28 knave(green)->-spy(red).
29
30 end_of_list.
31
32 formulas(goals).
33
34 end_of_list.
```

3.3 Analiza cod:

De la linia 19 pana la 21 este descris faptul ca daca o persoana este de un anume tip atunci aceasta nu poate fi de alt tip. Intre liniile 23 si 28 sunt traduse indiciile date.

3.4 Solutie:

```
interpretation( 3, [number = 1,seconds = 0], [
    function(blue, [0]),
    function(green, [1]),
    function(red, [2]),
    relation(knave(_), [1,0,0]),
    relation(knight(_), [0,0,1]),
    relation(spy(_), [0,1,0])]).
```

Din solutie reiese raspunsul corect: Rosu este cavaler, albastru este talhar, iar verde este spion.

3.5 Explicatie:

Pentru simplitate ma voi referi despre fiecare persoana dupa culoarea pe care o poarta. Albastru si verde spun ca rosu este spionul, presupunere care este falsa, deoarece daca acesta ar fi intr-adevar spionul atunci ambii spun adevarul, fapt ce l-ar face pe rosu talhar nu spion. Atunci se stie ca cei doi mint, rezultand

faptul ca rosu este cavaler, iar acesta ne informeaza ca verde este in fapt spionul, albastru dovedindu-se a fi talharul.

4 Problema 3: Princes and tigers

4.1 Cerinta:

Intr-o zi o domnisoara s-a prezentat in fata reginei unui regat prosper cu propunerea de a primi aprobarea acesteia asupra casatoriei sale cu unul dintre fii acesteia. Pentru a-i testa tenacitatea si inteligenta, regina i-a dat domnisoarei trei teste: in capatul unui coridor al castelului se afla 2 usi, in care se pot afla ori un tigru flamand ori un print. Regina i-a spus domnisoarei ca daca reuseste sa ghiceasca usa in care se afla printul de trei ori la rand, aceasta o sa le dea binecuvantarea. Conditia era faptul ca, deoarece regina avea mai multi fii, ambele usi puteau contine cate un print, insa ambele nu puteau contine tigrii, trebuind sa existe un print macar dupa una dintre usi.
1. Primul test: pe fiecare usa se afla cate un semn. Pe prima usa scria: "In aceasta camera se afla un print, iar in cealalta un tigru.", iar pe cea de a doua scria: "Intr-una dintre aceste camere se afla un print, iar in cealalta un tigru.". Regina ii spune domnsioarei ca exact unul din semne spune adevarul, celalalt mintind.
2. Al doilea test: Regina ii spune domnisoarei faptul ca daca dupa prima usa se afla un print, atunci semnul de pe aceasta este adevarat, iar daca se afla un tigru, acesta este fals. Pentru a doua usa se aplica opusul, adica daca se afla un print dupa cea de-a doua usa, semnul este fals, iar daca se afla un tigru atunci este adevarat. Pe ambele semne scrie: "Ambele camere contin printi.".
3. Pentru ultimul test, regina ii spune domnisoarei faptul ca aceleasi reguli ca si la testul precedent se aplica, insa semnele de pe usi au cazut, rezultand faptul ca nu se stie care semn ii corespunde carei usi. Pe unul dintre semne scrie:"Aceasta camera contine un tigru.", iar pe celalalt: "Ambele camere contin tigri.".

4.2 Cod:

1. Primul set de usi:

```
% Saved by Prover9-Mace4 Version 0.5, December 2007.

set(ignore_option_dependencies). % GUI handles dependencies

if(Prover9). % Options for Prover9
  assign(max_seconds, 60).
end_if.

if(Mace4).   % Options for Mace4
  assign(max_seconds, 60).
end_if.

formulas(assumptions).

t1->-t2.
t2->-t1.
p1->-t1.
p2->-t2.

(s1 & (-s2)) | ((-s1) & s2).

s1->p1 & t2.
s2->(p1&t2) | (p2&t1).
```

```
24
25  end_of_list.
26
27  formulas(goals).
28
29  end_of_list.
```

2. Al doilea set de usi:

```
1   % Saved by Prover9-Mace4 Version 0.5, December 2007.
2
3   set(ignore_option_dependencies). % GUI handles dependencies
4
5   if(Prover9). % Options for Prover9
6     assign(max_seconds, 60).
7   end_if.
8
9   if(Mace4).   % Options for Mace4
10    assign(max_seconds, 60).
11  end_if.
12
13  formulas(assumptions).
14
15  t1->-t2.
16  t2->-t1.
17  p1->-t1.
18  p2->-t2.
19
20  s1->p1.
21  -s1->t1.
22  -s2->p2.
23  s2->t2.
24
25  s1->p1 & p2.
26  s2->p1 & p2.
27
28  end_of_list.
29
30  formulas(goals).
31
32  end_of_list.
```

3. Al treilea set de usi:

```
1   % Saved by Prover9-Mace4 Version 0.5, December 2007.
2
3   set(ignore_option_dependencies). % GUI handles dependencies
4
5   if(Prover9). % Options for Prover9
6     assign(max_seconds, 60).
7   end_if.
8
9   if(Mace4).   % Options for Mace4
10    assign(max_seconds, 60).
```

```
11  end_if.
12
13  formulas(assumptions).
14
15  t1->-t2.
16  t2->-t1.
17  p1->-t1.
18  p2->-t2.
19
20  (door1sign1 & door2sign2) | (door1sign2 & door2sign1).
21  door1sign1->-door1sign2.
22  door2sign1->-door2sign2.
23  door1sign2->-door1sign1.
24  door2sign2->-door2sign1.
25
26  door1sign1 & s1->p1.
27  door1sign1 & -s1->t1.
28  door1sign2 & s2->p1.
29  door1sign2 & -s2->t1.
30  door2sign1 & s1->t2.
31  door2sign1 & -s1->p2.
32  door2sign2 & s2->t2.
33  door2sign2 & -s2->p2.
34
35  door1sign1 & s1->t1.
36  door2sign1 & s1->t2.
37  s2->t1 & t2.
38
39  end_of_list.
40
41  formulas(goals).
42
43  end_of_list.
```

4.3 Analiza cod:

De la linia 15 pana la linia 18 sunt puse conditiile ca sa existe cel putin un print intr-una dintre camere si ca nu pot sa existe un tigru si un print in aceeasi camera.
Dupa aceea am scris conditiile pentru fiecare set de usi si am tradus indiciile. Pentru ultimul set de usi am introdus patru variabile care spun care semn ii corespunde carei usi.

4.4 Solutie:

1. Primul set de usi:

```
interpretation( 2, [number = 1,seconds = 0], [
    relation(p1, [0]),
    relation(p2, [1]),
    relation(s1, [0]),
    relation(s2, [1]),
    relation(t1, [1]),
    relation(t2, [0])]).
```

Din solutie reiese faptul ca cel de-al doilea semn spune adevarul, rezultand ca dupa prima usa se afla un tigru, iar dupa cea de-a doua usa un print.

2. Al doilea set de usi:

```
interpretation( 2, [number = 1,seconds = 0], [
    relation(p1, [0]),
    relation(p2, [1]),
    relation(s1, [0]),
    relation(s2, [0]),
    relation(t1, [1]),
    relation(t2, [0])]).
```

Din solutie reiese faptul ca ambele semne mint, rezultand ca dupa prima usa se afla un tigru, iar dupa cea de-a doua usa un print.

3. Al treilea set de usi:

```
interpretation( 2, [number = 1,seconds = 0], [
    relation(door1sign1, [0]),
    relation(door1sign2, [1]),
    relation(door2sign1, [1]),
    relation(door2sign2, [0]),
    relation(p1, [0]),
    relation(p2, [1]),
    relation(s1, [0]),
    relation(s2, [0]),
    relation(t1, [1]),
    relation(t2, [0])]).
```

Din solutie reiese faptul ca ambele semne mint, iar in prima camera se alfa un tigru si in cea de-a doua camera un print.

4.5 Explicatie:

1. Primul set de usi:
Deoarece al doilea semn spune sigur adevarul rezulta faptul ca primul semn minte, atunci in prima camera este un tigru, iar in cea de-a doua un print.
2. Al doilea set de usi:
Daca ambele camere ar contine printi atunci un semn ar minti si celalt ar spune adevarul, dar ambele semne spun acelasi mesaj, rezulta faptul ca ambele semne mint. Deoarece ambele mint, atunci dupa prima usa se afla un tigru, iar dupa ce-a de-a doua usa un print.
3. Al treilea set de usi:
Al doilea semn minte fiindca ar rezulta ca nu exista printi in niciuna dintre camere. De asemenea si primul semn minte deoarece daca ar spune adevarul si ar apartine de oricare dintre camere atunci se ajunge la o contradictie. Deoarece ambele semne mint rezulta faptul ca primul semn apartine celei de-a doua camere, in aceasta aflandu-se un print, iar ce-l de-al doilea semn apartine primei camere care contine un tigru.

5 Referinte:

Problema 1: https://www.brainzilla.com/logic/zebra/gas-station/.
Problema 2: https://www.popularmechanics.com/science/math/a15388659/riddle-of-the-week-48-knights-and-knaves-part-6/.