Documentatie proiect 2

Proiect realizat de : Stecko Daiana,

Timis Iulia,

Viman andrei

1.

Prima problema implementata de noi consta in exemplificarea gradului de rudenie. Avem doua familii: Anca si Florin care il au drept copil pe Alex, si George care are 2 copii, pe Marius si Maria.

George si Florin sunt frati. Vrem sa aflam daca Maria si Alex sunt verisori.

In fisierul proiect1.in am declarant 6 nume corespunzatoare fiecarui membru al familiei. Am declarant formula prin care reiese ca Alex este copilul lui Florin si Anca, dupa am dat gradul de rudenie dintre Florin si George, acestia fiind frati. Am dat relatia dintre George si Marius, acesta fiind copilul lui George, iar la final am specificat relatia ditre Marius si Maria, acestia fiind frati. Trebuie sa demonstram ca Alex si Maria sunt Verisori.

**Aici vom pune codul pentru prima problema**

assign(max\_seconds, 30).

set(binary\_resolution).

set(print\_gen).

%Presupunem ca ANCA si FLORIN sunt soti si au un copil pe nume ALEX. Fratele

%lui FLORIN, GEORGE are un copil pe nume MARIUS. MARIUS are o sora pe nume %maria. MARIA si ALEX sunt verisori?

formulas(assumptions).

ANCA.

FLORIN.

ALEX.

GEORGE.

MARIUS.

MARIA.

ANCA & FLORIN -> ALEX.

FLORIN & GEORGE -> FRATI.

GEORGE -> MARIUS .

MARIUS & MARIA -> FRATI.

end\_of\_list.

formulas(goals).

ALEX -> MARIA.

end\_of\_list.

**Aici este rezultat dat dupa rularea cu prouver9:**

THEOREM PROVED

THEOREM PROVED

Exiting with 1 proof.

2.

Al 2-lea exercitiu consta in demonstrarea faptului ca porcul Porky nu poate sa alerge repede.

In fisierul proiect2.in, am declarat un nume corespunzator porcului: Porky. Incepem prin a-I atribui lui Porky statutul de porc (pig) . Spunem ca porcii (pig) sunt animale (animal) si de asemenea, porcii (pig) sunt animale grase (fat). Dupa spunem ca animalele grase (fat) nu pot fugi (run). Trebuie sa demonstram ca Porky nu poate sa fuga, rezultatul fiind adevarat.

**Aici vom pune codul pentru problema 2:**

[18:16, 02/12/2021] Viman Andrei: https://users.utcluj.ro/~agroza/puzzles/maloga/codes.html

[18:46, 02/12/2021] Viman Andrei: assign(max\_seconds, 30).

set(binary\_resolution).

set(print\_gen).

%1. Porky is a pig.

%2. Pigs are animals.

%3. Pigs are fat.

%4. No fat animals run fast.

%∴Porky does not run fast.

formulas(assumptions).

porky -> pig.

pig -> animal.

pig -> fat.

fat -> -run.

end\_of\_list.

formulas(goals).

porky -> -run.

end\_of\_list.

**Aici este rezultat dat dupa rularea cu prouver9:**

THEOREM PROVED

THEOREM PROVED

Exiting with 1 proof.

3.

Al 3-lea exercitiu doreste demonstrarea faptului ca randunelele zboara si mananca nuci. Acest exemplu are la baza pasarile

In fisierul proiect3.in incepem prin a specifica faptul ca pasarile (p) fac parte din categoria animalelor (a). Dupa spunem ca unele animale (a) au abilitatea de a zbura (fly). Pasarile (p) au aptitudinea de a manca nuci (n). Am specificat faptul ca randunelele ( r) sunt pasari (p). Dorim sa demonstarm ca randunelele au aptitudinea de a zbura si putinta de a manca nuci.

**Aici vom pune codul pentru problema 3:**

assign(max\_seconds, 30).

set(binary\_resolution).

set(print\_gen).

%1. Porky is a pig.

%2. Pigs are animals.

%3. Pigs are fat.

%4. No fat animals run fast.

%∴Porky does not run fast.

formulas(assumptions).

porky -> pig.

pig -> animal.

pig -> fat.

fat -> -run.

end\_of\_list.

formulas(goals).

porky -> -run.

end\_of\_list.

**Aici este rezultat dat dupa rularea cu prouver9:**

THEOREM PROVED

THEOREM PROVED

Exiting with 1 proof.

4.

Exemplul 4 doreste sa demonstreze faptul ca intr-o camera nu se pot afla comori si pirati.

In fisierul proiect4.in am numerotat cele 3 comori ( c1, c2 si c3), si cei 3 pirati ( p1, p2, p3). Am impus ca intr-o camera sa nu fie in acelasi timp si o comoara si un pirat. Daca in camera 1 se afla comoara nu se poate afla un pirat. Daca in camera 2 se afla comoara nu se poate afla un pirat. Daca in camera 3 se afla comoara nu se poate afla un pirat.

**Aici vom pune codul pentru problema 4:**

% sunt 3 camere in care se pot afla comori si pirati

% intr -o camera nu se pot afla si comori si pirati

formulas(assumptions).

c1 | c2 | c3.

p1 | p2 | p3.

c1 -> -p1.

c2 -> -p2.

c3 -> -p3.

end\_of\_list.

formulas(goals).

%l2.

end\_of\_list.

**Aici este rezultat dat dupa rularea cu mace4:**

=== Mace4 starting on domain size 2. ===

------ process 4741 exit (all\_models) ------

interpretation( 2, [number = 1,seconds = 0], [

relation(c1, [0]),

relation(c2, [0]),

relation(c3, [1]),

relation(p1, [0]),

relation(p2, [1]),

relation(p3, [0])]).

interpretation( 2, [number = 2,seconds = 0], [

relation(c1, [0]),

relation(c2, [0]),

relation(c3, [1]),

relation(p1, [1]),

relation(p2, [0]),

relation(p3, [0])]).

interpretation( 2, [number = 3,seconds = 0], [

relation(c1, [0]),

relation(c2, [0]),

relation(c3, [1]),

relation(p1, [1]),

relation(p2, [1]),

relation(p3, [0])]).

interpretation( 2, [number = 4,seconds = 0], [

relation(c1, [0]),

relation(c2, [1]),

relation(c3, [0]),

relation(p1, [0]),

relation(p2, [0]),

relation(p3, [1])]).

interpretation( 2, [number = 5,seconds = 0], [

relation(c1, [0]),

relation(c2, [1]),

relation(c3, [0]),

relation(p1, [1]),

relation(p2, [0]),

relation(p3, [0])]).

interpretation( 2, [number = 6,seconds = 0], [

relation(c1, [0]),

relation(c2, [1]),

relation(c3, [0]),

relation(p1, [1]),

relation(p2, [0]),

relation(p3, [1])]).

interpretation( 2, [number = 7,seconds = 0], [

relation(c1, [0]),

relation(c2, [1]),

relation(c3, [1]),

relation(p1, [1]),

relation(p2, [0]),

relation(p3, [0])]).

interpretation( 2, [number = 8,seconds = 0], [

relation(c1, [1]),

relation(c2, [0]),

relation(c3, [0]),

relation(p1, [0]),

relation(p2, [0]),

relation(p3, [1])]).

interpretation( 2, [number = 9,seconds = 0], [

relation(c1, [1]),

relation(c2, [0]),

relation(c3, [0]),

relation(p1, [0]),

relation(p2, [1]),

relation(p3, [0])]).

interpretation( 2, [number = 10,seconds = 0], [

relation(c1, [1]),

relation(c2, [0]),

relation(c3, [0]),

relation(p1, [0]),

relation(p2, [1]),

relation(p3, [1])]).

interpretation( 2, [number = 11,seconds = 0], [

relation(c1, [1]),

relation(c2, [0]),

relation(c3, [1]),

relation(p1, [0]),

relation(p2, [1]),

relation(p3, [0])]).

interpretation( 2, [number = 12,seconds = 0], [

relation(c1, [1]),

relation(c2, [1]),

relation(c3, [0]),

relation(p1, [0]),

relation(p2, [0]),

relation(p3, [1])]).

5.

In cazul problemei cu numarul 5 se doreste demonstrarea unicitatii comorii, si de asemenea a faptului ca nu se afla mai mult de una intr-o camera.

In fisierul proiect5.in incepem prin a specifica faptul ca avem 3 camere, 3 comori (c1, c2, c3) si 3 pirati (p1,p2,p3). Daca in camera 1 se afla comoara nu se poate afla un pirat. Daca in camera 2 se afla comoara nu se poate afla un pirat. Daca in camera 3 se afla comoara nu se poate afla un pirat. Daca comoara 1 se afla in camera 1, aceasta nu poate fii in camera2 sau camera3. Daca comoara 2 se afla in camera 2, aceasta nu poate fii in camera1 sau camera3. Daca comoara 3 se afla in camera 3, aceasta nu poate fii in camera2 sau camera1.

**Aici vom pune codul pentru problema 5:**

% sunt 3 camere in care se pot afla comori si pirati

% intr -o camera nu se pot afla si comori si pirati

% comoara este unica si nu se afla mai mult de una intr o camera

formulas(assumptions).

c1 | c2 | c3.

p1 | p2 | p3.

c1 -> -p1.

c2 -> -p2.

c3 -> -p3.

c1 -> -c2 & -c3.

c2 -> -c1 & -c3.

c3 -> -c1 & -c2.

end\_of\_list.

formulas(goals).

end\_of\_list.

**Aici este rezultat dat dupa rularea cu mace4:**

=== Mace4 starting on domain size 2. ===

------ process 4835 exit (all\_models) ------

interpretation( 2, [number = 1,seconds = 0], [

relation(c1, [0]),

relation(c2, [0]),

relation(c3, [1]),

relation(p1, [0]),

relation(p2, [1]),

relation(p3, [0])]).

interpretation( 2, [number = 2,seconds = 0], [

relation(c1, [0]),

relation(c2, [0]),

relation(c3, [1]),

relation(p1, [1]),

relation(p2, [0]),

relation(p3, [0])]).

interpretation( 2, [number = 3,seconds = 0], [

relation(c1, [0]),

relation(c2, [0]),

relation(c3, [1]),

relation(p1, [1]),

relation(p2, [1]),

relation(p3, [0])]).

interpretation( 2, [number = 4,seconds = 0], [

relation(c1, [0]),

relation(c2, [1]),

relation(c3, [0]),

relation(p1, [0]),

relation(p2, [0]),

relation(p3, [1])]).

interpretation( 2, [number = 5,seconds = 0], [

relation(c1, [0]),

relation(c2, [1]),

relation(c3, [0]),

relation(p1, [1]),

relation(p2, [0]),

relation(p3, [0])]).

interpretation( 2, [number = 6,seconds = 0], [

relation(c1, [0]),

relation(c2, [1]),

relation(c3, [0]),

relation(p1, [1]),

relation(p2, [0]),

relation(p3, [1])]).

interpretation( 2, [number = 7,seconds = 0], [

relation(c1, [1]),

relation(c2, [0]),

relation(c3, [0]),

relation(p1, [0]),

relation(p2, [0]),

relation(p3, [1])]).

interpretation( 2, [number = 8,seconds = 0], [

relation(c1, [1]),

relation(c2, [0]),

relation(c3, [0]),

relation(p1, [0]),

relation(p2, [1]),

relation(p3, [0])]).

interpretation( 2, [number = 9,seconds = 0], [

relation(c1, [1]),

relation(c2, [0]),

relation(c3, [0]),

relation(p1, [0]),

relation(p2, [1]),

relation(p3, [1])]).

6.

In cazul problemei cu numarul 6 se doreste demonstrarea unicitatii comorii, de asemenea a faptului ca nu se afla mai mult de o comoara intr-o camera, iar in celelalte camere se afla piratii.

In fisierul proiect6.in incepem prin a specifica faptul ca avem 3 camere, 3 comori (c1, c2, c3) si 3 pirati (p1,p2,p3).

Daca in camera 3 se afla o comoara, in camera 2 va fi un pirat si in camera1 va fi un pirat. Daca in camera 1 se afla o comoara, in camera 2 va fi un pirat si in camera3 va fi un pirat. Daca in camera 2 se afla o comoara, in camera 1 va fi un pirat si in camera 3 va fi un pirat.

Daca in camera 1 se afla comoara nu se poate afla un pirat. Daca in camera 2 se afla comoara nu se poate afla un pirat. Daca in camera 3 se afla comoara nu se poate afla un pirat. Daca comoara 1 se afla in camera 1, aceasta nu poate fii in camera2 sau camera3. Daca comoara 2 se afla in camera 2, aceasta nu poate fii in camera1 sau camera3. Daca comoara 3 se afla in camera 3, aceasta nu poate fii in camera2 sau camera1.

**Aici vom pune codul pentru problema 6:**

% sunt 3 camere in care se pot afla comori si pirati

% intr -o camera nu se pot afla si comori si pirati

% comoara este unica si nu se afla mai mult de una intr o camera

% se afla numai 2 pirati

formulas(assumptions).

c1 | c2 | c3.

p1 & p2 | p2 & p3 | p1 & p3.

c1 -> -p1.

c2 -> -p2.

c3 -> -p3.

c1 -> -c2 & -c3.

c2 -> -c1 & -c3.

c3 -> -c1 & -c2.

end\_of\_list.

formulas(goals).

end\_of\_list.

**Aici este rezultat dat dupa rularea cu mace4:**

=== Mace4 starting on domain size 2. ===

------ process 4888 exit (all\_models) ------

interpretation( 2, [number = 1,seconds = 0], [

relation(c1, [0]),

relation(c2, [0]),

relation(c3, [1]),

relation(p1, [1]),

relation(p2, [1]),

relation(p3, [0])]).

interpretation( 2, [number = 2,seconds = 0], [

relation(c1, [0]),

relation(c2, [1]),

relation(c3, [0]),

relation(p1, [1]),

relation(p2, [0]),

relation(p3, [1])]).

interpretation( 2, [number = 3,seconds = 0], [

relation(c1, [1]),

relation(c2, [0]),

relation(c3, [0]),

relation(p1, [0]),

relation(p2, [1]),

relation(p3, [1])]).

7.

In cazul problemei cu numarul 7 dorim sa demonstram daca bebelusul Alex nu poate face fata la o lupta cu un alligator.

Pentru inceput am tras concluzia ca un bebelus (bebelus) este irational (irational). Dupa ajungem la deductia ca un om inconstient (onInconstient) nu face fata unui alligator (nufacefatalaaligator). Ajungem la concluzia ca un om irational( irrational) este si inconstient (onInconstient). La final ii atribuim lui Alex statutul de bebelus.

**Aici vom pune codul pentru problema 7:**

% toti bebelusii sunt irationali

% oamenii incontienti nu fac fata confruntarii cu aligatori

% oamenii irationali sunt si inconstienti

% alex e un bebe care nu poate sa invinga in confruntaea cu aligatorul

assign(max\_seconds, 30).

set(binary\_resolution).

set(print\_gen).

formulas(assumptions).

bebelus -> irational.

omInconstient -> nufacefatalaaligator.

irational -> omInconstient.

alex -> bebelus.

end\_of\_list.

formulas(goals).

alex -> nufacefatalaaligator.

end\_of\_list.

**Aici este rezultat dat dupa rularea cu prouver9:**

THEOREM PROVED

THEOREM PROVED

Exiting with 1 proof.

8.

La un bal sunt printi si printese. Ei sunt locuitorii unui castel mare. La bal printesele danseaza si printii nu danseaza. Atat printii cat si printesele afirma ca danseaza. Primesti o invitatie la bal si esti intampinat de un print si de o printesa si ambii iti spun ca danseaza. Trebuie sa afli care dintre ei danseaza cu adevarat.

**Aici vom pune codul pentru problema 8:**

formulas(assumptions).

all x (locuitori(x) -> printi(x) | printese(x)).

all x ((printese(x) -> -printi(x)) & (printi(x) -> -printese(x))).

printese(x) -> danseaza(x).

printi(x) -> -danseaza(x).

locuitori(a) & locuitori(b).

danseaza(a) <-> printi(a) & printi(b).

end\_of\_list.

formulas(goals).

printi(a) & printese(b).

end\_of\_list.

**Aici este rezultat dat dupa rularea cu prouver9:**

THEOREM PROVED

THEOREM PROVED

Exiting with 1 proof.