Решения на задачи от писмен изпит по Логическо програмиране

23 януари 2019

Съдържание

1 Първа задача на пролог

Дядо Коледа има четири еленчета, които тръгват да тичат едновременно от една и съща стартова линия. Те тичат с постоянни еднопосочни скорости, перпендикулярни на стартовата линия и с големини съответно v_1 м/мин, v_2 м/мин, v_3 м/мин и v_4 м/мин, които са положителни цели числа. С цел да не се отдалечават прекомерено едно от друго следват следната обща стратегия:

II.1
На всяка кръгла минута, онези от тях, които са на една и съща челна позиция, на която до този момент няма други еленчете, спират и изчакват всички останали да ги задминат, и тогава отново на кръгла минута тръгват да тичат.
Да се дефинира на пролог двуместен предикат $D(V, X)$, който по даден списък от четири положителни числа V при n-тото преудовлетворяване генерира в X списъка от изминалите разстояния на всяко от еленчетата в $(n+1)$-та минута от началото.

II.2
На всяка кръгла минута, онези от тях, които са на една и съща челна позиция, на която до този момент няма други еленчете, спират и изчакват докато има еленчета зад тях, и тогава отново на кръгла минута тръгват да тичат.
Да се дефинира на пролог двуместен предикат $dist(V, X)$, който по даден списък от четири положителни числа V при n-тото преудовлетворяване генерира в X списъка от изминалите разстояния на всяко от еленчетата в $(n+1)$-та минута от началото.

1.1 Примерно решение

Предикатът $d(V, X, ProblemNumber)$ е на три аргумента като последния при 1-ца се изпълнява решението на група 1, а при 2-ка изпълнява на група 2.

```
count([], _, 0).
count([H|T], H, N) :-
    count(T, H, M),
    N is M+1.
count([H|T], X, N) :-
    H\=X,
    count(T, X, N).

merge([], [], []).
merge([A|AT], [B|BT], [[A, B]|T]) :-
    merge(AT, BT, T).
```

```
max2(A, A, B) :-
    A>B.
max2(B, A, B) :-
    not(A>B).

min2(A, A, B) :-
    A<B.
min2(B, A, B) :-
    not(A<B).

min(M, [M]).
min(M, [H|T]) :-
    min(N, T),
    min2(M, H, N).

max(M, [M]).
max(M, [H|T]) :-
    max(N, T),
    max2(M, H, N).
```

Listing 1: Utility predicates

```
d(V, X, ProblemNumber) :-
    d(V, X, _, ProblemNumber).

d(_, [0, 0, 0, 0], [0, 0, 0, 0], _).
d(V, X, W, ProblemNumber) :-
    d(V, X1, W1, ProblemNumber),
    move(V, W1, X1, X),
    setWaiting(X, W1, W, ProblemNumber).

move([], [], [], []).
move([V|VT], [W|WT], [P|PT], [N|NT]) :-
    incrementIfNotWaiting(V, W, P, N),
    move(VT, WT, PT, NT).

incrementIfNotWaiting(_, 1, X, X).
incrementIfNotWaiting(V, 0, Y, X) :-
    X is V+Y.

setWaiting(X, W, NW, ProblemNumber) :-
    min(Min, X),
    max(Max, X),
    merge(X, W, MergedDearNStatus),
    setWaiting(X,
```

```
               W,
               NW,
               MergedDearNStatus,
               Min,
               Max,
               ProblemNumber).

setWaiting([], [], [], _, _, _, _).
setWaiting([X|XT], [OldWait|RestOWait], [NewWait|RestNWait],
↪  MergedDearNStatus, Min, Max, 1) :-
    wait1(X, OldWait, NewWait, MergedDearNStatus, Min, Max),
    setWaiting(XT, RestOWait, RestNWait, MergedDearNStatus, Min,
     ↪  Max, 1).
setWaiting([X|XT], [OldWait|RestOWait], [NewWait|RestNWait],
↪  MergedDearNStatus, Min, Max, 2) :-
    wait2(X, OldWait, NewWait, MergedDearNStatus, Min, Max),
    setWaiting(XT, RestOWait, RestNWait, MergedDearNStatus, Min,
     ↪  Max, 2).

% Group 1
% If I'm not Max and can move, I continue to move.
wait1(X, 0, 0, _, _, Max) :-
    X<Max.
% I am not the first one to reach this position.
wait1(Max, 0, 0, L, _, Max) :-
    count(L, [Max, 1], N),
    N>0.
% I am the first one to reach this position.
wait1(Max, 0, 1, L, _, Max) :-
    count(L, [Max, 1], 0).
% All are strictly ahead me and I can move.
wait1(Min, 1, 0, L, Min, _) :-
    count(L, [Min, 0], 0).
% There is still someone who is last with me.
wait1(Min, 1, 1, L, Min, _) :-
    count(L, [Min, 0], N),
    N>0.
% All must be ahead of me so I can move.
wait1(X, 1, 1, _, Min, _) :-
    X>Min.

% Group 2
% If I'm not Max and can move, I continue to move.
wait2(X, 0, 0, _, _, Max) :-
    X<Max.
% I am not the first one to reach this position.
```

```
wait2(Max, 0, 0, L, _, Max) :-
    count(L, [Max, 1], N),
    N>0.
% I am the first one to reach this position.
wait2(Max, 0, 1, L, _, Max) :-
    count(L, [Max, 1], 0).
% All are ahead me and maybe there are some last with me and I
↪  can move.
wait2(Min, 1, 0, _, Min, _).
% All must be ahead of me or at least last with me so I can move.
wait2(X, 1, 1, _, Min, _) :-
    X>Min.
```

Listing 2: Main predicates

2 Втора задача на пролог

Правоъгълна торта с размери MxN трябва да се разреже на K правоъгълни парчета с целочислени страни с отношение за група II.1 - 3:1, а за II.2 - 1:2. Да се дефинира на пролог триемтен предикат $cake(M, N, K)$, който да разпознава точно онези тройки $\langle M, N, K\rangle$, за който това е възможно.

2.1 Примерно решение

Предикатът $cake(M, N, K, ProblemNumber)$ е на четири аргумента като последния при 1-ца се изпълнява решението на група 1, а при 2-ка изпълнява на група 2.

```
append([], L2, L2).
append([H|T], L2, [H|R]) :-
    append(T, L2, R).

member(X, L) :-
    append(_, [X|_], L).

length([], 0).
length([_|T], N) :-
    length(T, M),
    N is M+1.

between(A, B, A) :-
    A=<B.
between(A, B, R) :-
    A<B,
    A1 is A+1,
    between(A1, B, R).

n_th_element(X, 0, [X|_]).
n_th_element(X, N, [_|T]) :-
    n_th_element(X, M, T),
    N is M+1.

prettyWrite(L, V) :-
    length(L, N),
    prettyWriteList(L, N, V).
prettyWriteList([], 0, _) :-
    nl.
prettyWriteList([H|T], N, V) :-
    N>0,
    N1 is N-1,
    N1 mod V=\=0,
```

```
    write(H),
    prettyWriteList(T, N1, V).
prettyWriteList([H|T], N, V) :-
    N>0,
    N1 is N-1,
    N1 mod V=:=0,
    write(H),
    nl,
    prettyWriteList(T, N1, V).

min2(A, A, B) :-
    A<B.
min2(B, A, B) :-
    not(A<B).

min(M, [M]).
min(M, [H|T]) :-
    min(N, T),
    min2(M, H, N).
```

Listing 3: Main predicates

```
cake(M, N, K, ProblemNumber) :-
    All is M*N,
    generateListOfElement(All, 0, Matrix),
    min(Min, [M, N]),
    cover(M, N, Min, K, Matrix, ProblemNumber).

generateListOfElement(0, _, []).
generateListOfElement(N, Filling, [Filling|T]) :-
    N>0,
    N1 is N-1,
    generateListOfElement(N1, Filling, T).

cover(_, N, _, 0, Matrix, _) :-
    allFilled(Matrix),
    prettyWrite(Matrix, N).
cover(M, N, Min, K, Matrix, 1) :-
    K>0,
    K1 is K-1,
    between(1, Min, Value),
    ValueI is Value*2,
    ValueJ is Value*3,
    n_th_element(0, Idx, Matrix),
    I is Idx div N,
    J is Idx mod N,
```

```
    (   I+ValueI=<M,
        J+ValueJ=<N,
        replace(I,
                J,
                ValueI,
                ValueJ,
                Matrix,
                NewMatrix,
                K,
                N)
    ;   I+ValueJ=<M,
        J+ValueI=<N,
        replace(I,
                J,
                ValueJ,
                ValueI,
                Matrix,
                NewMatrix,
                K,
                N)
    ),
    cover(M, N, Min, K1, NewMatrix, 1).

cover(M, N, Min, K, Matrix, 2) :-
    K>0,
    K1 is K-1,
    between(1, Min, ValueI),
    ValueJ is ValueI*2,
    n_th_element(0, Idx, Matrix),
    I is Idx div N,
    J is Idx mod N,
    (   I+ValueI=<M,
        J+ValueJ=<N,
        replace(I,
                J,
                ValueI,
                ValueJ,
                Matrix,
                NewMatrix,
                K,
                N)
    ;   I+ValueJ=<M,
        J+ValueI=<N,
        replace(I,
                J,
                ValueJ,
```

```
                ValueI,
                Matrix,
                NewMatrix,
                K,
                N)
    ),
    cover(M, N, Min, K1, NewMatrix, 2).

allFilled(Matrix) :-
    not(( member(Y, Matrix),
          Y=:=0
        )).

replace(_, _, 0, _, Matrix, Matrix, _, _).
replace(I, J, SideI, SideJ, Matrix, ResultMatrix, K, N) :-
    SideI>0,
    SideI1 is SideI-1,
    I1 is I+1,
    Idx is I*N+J,
    replaceSublist(Matrix, Idx, SideJ, TempMatrix, K),
    replace(I1,
            J,
            SideI1,
            SideJ,
            TempMatrix,
            ResultMatrix,
            K,
            N).

replaceSublist(Row, J, SideJ, NewRow, K) :-
    append(A, B, Row),
    length(A, J),
    append(C, D, B),
    length(C, SideJ),
    not(( member1(X, C),
          X=\=0
        )),
    generateListOfElement(SideJ, K, E),
    append(E, D, F),
    append(A, F, NewRow).
```

Listing 4: Main predicates

3 Задача за определимост

Нека $\mathcal{L}$ е език за предикатно смятане с формално равенство и само един нелогически символ - двуместния функционален сумвол t. Нека $\mathcal{A}$ е структура за $\mathcal{L}$ с универсиум множеството на:

I.1
неотрицателните цели числа и функцията $t^{\mathcal{A}}$ е дефинирана с равенството $t^{\mathcal{A}} = 3^a(b+1)$.

a) Да се докаже, че следните множества са определими в $\mathcal{A}$ с формула от $\mathcal{L}$:

(1) $\{0\}$; (2) $\{1\}$; (3) $\{3^n \mid n \geq 0\}$; (4) $\{\langle a,b,c\rangle \mid a+b=c\}$.

b) Да се намерят всички автоморфизми на $\mathcal{A}$.

I.2
положителните цели числа и функцията $t^{\mathcal{A}}$ е дефинирана с равенството $t^{\mathcal{A}} = 5^{b-1}(a+1)$.

a) Да се докаже, че следните множества са определими в $\mathcal{A}$ с формула от $\mathcal{L}$:

(1) $\{1\}$; (2) $\{\langle a,a+1\rangle \mid a \geq 1\}$; (3) $\{5^n \mid n \geq 0\}$; (4) $\{\langle a,b,c\rangle \mid a = b+c\}$.

b) Да се намерят всички автоморфизми на $\mathcal{A}$.

3.1 Примерно решение за група 1

a)
$\varphi_0(x) \rightleftharpoons \neg\exists y\exists z(t(y,z) \doteq x))$.
$\varphi_1(x) \rightleftharpoons \exists y(\varphi_0(y) \,\&\, t(y,y) \doteq x)$.
$\varphi_{3^n}(x) \rightleftharpoons \exists y\exists z(\varphi_0(y) \,\&\, t(z,y) \doteq x)$.
$\varphi_+(x,y,z) \rightleftharpoons \exists w_1\exists w_2\exists w_3\exists v(\varphi_0(v) \,\&\, t(y,v) \doteq w_1 \,\&\, t(v,w_2) \doteq w_1$
$\&\, t(z,v) \doteq w_3 \,\&\, t(x,w_2) \doteq w_3)$.
b)
С индукция по $n \in \mathbb{N}$ ще покажем, че $\{n\}$ е определимо за n произволно и следователно тогава $Aut(\mathcal{A}) = \{Id_{\mathcal{A}}\}$.
За n = 0,1 ги имаме, нека видим за n = 2.
$\varphi_2(x) \rightleftharpoons \exists y\exists z(\varphi_0(y) \,\&\, \varphi_1(z) \,\&\, t(y,z) \doteq x)$.
Нека допуснем, че за някое $n \in \mathbb{N}, n \geq 3$ имаме $\varphi_n(x)$.
А за $(n+1)$?
$\varphi_{n+1}(x) \rightleftharpoons \exists y\exists z(\varphi_0(y) \,\&\, \varphi_n(z) \,\&\, t(y,z) \doteq x)$.
Т.е. за $\forall n \in \mathbb{N}$, то $\{n\}$ е определимо.

3.2 Примерно решение за група 2

a)
$\varphi_1(x) \rightleftharpoons \neg\exists y\exists z(t(y,z) \doteq x))$.
$\varphi_{<x,x+1>}(x,y) \rightleftharpoons \exists z(\varphi_1(z) \,\&\, t(z,x) \doteq y)$.

$\varphi_4(x) \rightleftharpoons \exists v \exists y \exists z(\varphi_1(v) \& \varphi_{<x,x+1>}(v,y) \,\&\, \varphi_{<x,x+1>}(y,z) \,\&\, \varphi_{<x,x+1>}(z,x))$.
$\varphi_{5^n}(x) \rightleftharpoons \exists y \exists z(\varphi_0(y) \,\&\, t(z,y) \doteq x)$.
Аналогично можем да дефинираме и φ_{24}.
$\varphi_+(x,y,z) \rightleftharpoons \exists w_1 \exists w_2 \exists w_3 \exists v \exists q \exists w(\varphi_{24}(v) \,\&\, \varphi_2(w) \,\&\, t(y,v) \doteq w_1 \,\&$
$t(w,w_2) \doteq w_1 \,\&\, t(x,v) \doteq w_3 \,\&\, \varphi_{<x,x+1>}(z,q) \,\&\, t(q,w_2) \doteq w_3)$.
b)
С индукция по $n \in \mathbb{N}_{>0}$ ще покажем, че $\{n\}$ е определимо за n произволно и следователно тогава $Aut(\mathcal{A}) = \{Id_{\mathcal{A}}\}$.
За n = 1 го имаме, нека видим за n = 2.
$\varphi_2(x) \rightleftharpoons \exists y(\varphi_1(y) \,\&\, t(y,y) \doteq x)$.
Нека видим и за n = 3.
$\varphi_3(x) \rightleftharpoons \exists y \exists z(\varphi_1(y) \,\&\, \varphi_2(z) \,\&\, t(y,z) \doteq x)$.
Нека допуснем, че за някое $n \in \mathbb{N}, n \geq 4$ имаме $\varphi_n(x)$.
А за $(n+1)$?
$\varphi_{n+1}(x) \rightleftharpoons \exists y \exists z(\varphi_1(y) \,\&\, \varphi_n(z) \,\&\, t(y,z) \doteq x)$.
Т.е. за $\forall n \in \mathbb{N}_{>0}$, то $\{n\}$ е определимо.

4 Задача за изпълнимост

Нека p е двуместен предикатен символ, а f е едноместен функционален символ. Да се докаже, че множеството от следните три формули е изпълнимо:

I.1
$\forall x(\neg p(f(x), x) \,\&\, \exists y p(f(x), y))$
$\forall x \forall y(p(x, y) \Longrightarrow \exists z(p(x, z) \,\&\, \neg p(z, z) \,\&\, p(z, f(y))))$
$\neg \forall x \forall y \forall z(p(x, y) \,\&\, p(y, z) \Longrightarrow p(f(x), z))$

I.2
$\forall x(\neg p(f(x), x) \,\&\, \exists y p(f(x), y))$
$\forall x \forall y(p(x, y) \Longrightarrow \exists z(p(x, z) \,\&\, \neg p(f(z), f(z)) \,\&\, p(z, y)))$
$\neg \forall x \forall y \forall z(p(x, y) \,\&\, p(y, z) \Longrightarrow \neg p(f(x), z))$

4.1 Примерни решения

$$S = (\mathbb{R}, p^S, f^S)$$
$$p^S \rightleftharpoons <$$
$$f^S(x) \rightleftharpoons x + 1, x \in \mathbb{R}$$

$$S = (\mathbb{R}, p^S, f^S)$$
$$p^S \rightleftharpoons \neq$$
$$f^S(x) \rightleftharpoons x, x \in \mathbb{R}$$

5 Задача за резолюция

Нека φ_1, φ_2 и φ_3 са следните три формули:

I.1
$\forall x \neg \forall y(q(y,x) \Longrightarrow \exists z(q(y,z) \& r(z,x)))$,
$\forall x(\forall y(q(x,y) \Longrightarrow \exists z(q(y,z) \& q(x,z))) \Longrightarrow \neg \exists z q(x,z))$,
$\forall z \forall y(\exists x(q(y,x) \& \neg q(z,x)) \Longrightarrow r(z,y))$.

I.2
$\forall x \neg \forall y(\forall z(p(z,y) \Longrightarrow r(x,z)) \Longrightarrow \neg p(x,y))$,
$\forall x(\exists y p(y,x) \Longrightarrow \exists y(p(y,x) \& \neg \exists z(p(z,y) \& p(z,x))))$,
$\forall x \neg \exists y(p(x,y) \& \neg \forall z(r(y,z) \Longrightarrow p(x,z)))$.

С метода на резолюцията да се докаже, че
$\varphi_1, \varphi_2, \varphi_3 \models \forall x \exists y(p(x,x) \Longrightarrow \exists z(r(z,y) \& \neg r(z,y)))$.
* Тъй като $(r(z,y) \& \neg r(z,y))$ е винаги лъжа, то ψ дефинираме като:
$\neg \forall x \exists y(\neg p(x,x)) \equiv \exists x p(x,x)$.
Респективно q за вариант I.1.

5.1 Примерно решение

I.1
Получаваме следните формули като приведем в ПНФ, СНФ и КНФ:
$\varphi_1^S \rightleftharpoons \forall x \forall z(q(f(x),x) \& (\neg q(f(x),z) \vee \neg r(z,x)))$.
$\varphi_2^S \rightleftharpoons \forall x \forall z \forall t((q(x,g(x)) \vee \neg q(x,t)) \& (\neg q(g(x),z) \vee \neg q(x,z) \vee \neg q(x,t))$.
$\varphi_3^S \rightleftharpoons \forall z \forall y \forall x(r(z,y) \vee \neg q(y,x) \vee q(z,x))$.
$\psi^S \rightleftharpoons q(a,a)$.
Дизюнктите са (нека ги номерираме променливите по принадлежност към дизюнкт):
$D_1 = \{q(f(x_1),x_1)\}$;
$D_2 = \{\neg q(f(x_2),z_2), \neg r(z_2,x_2)\}$;
$D_3 = \{q(x_3,g(x_3)), \neg q(x_3,t_3)\}$;
$D_4 = \{\neg q(g(x_4),z_4), \neg q(x_4,z_4), \neg q(x_4,t_4)\}$;
$D_5 = \{r(z_5,y_5), \neg q(y_5,x_5), q(z_5,x_5)\}$;
$D_6 = \{q(a,a)\}$.

I.2
Получаваме следните формули като приведем в ПНФ, СНФ и КНФ:
$\varphi_1^S \rightleftharpoons \forall x \forall z(p(x,f(x)) \& (\neg p(z,f(x)) \vee r(x,z)))$.
$\varphi_2^S \rightleftharpoons \forall x \forall t \forall z((p(g(x),x) \vee \neg p(t,x)) \& (\neg p(z,g(x)) \vee \neg p(z,x) \vee \neg p(t,x))$.
$\varphi_3^S \rightleftharpoons \forall x \forall y \forall z(\neg r(y,z) \vee \neg p(x,y) \vee p(x,z))$.
$\psi^S \rightleftharpoons p(a,a)$.
Дизюнктите са (нека ги номерираме променливите по принадлежност към дизюнкт):

$D_1 = \{p(x_1, f(x_1))\}$;
$D_2 = \{\neg p(z_2, f(x_2)), r(x_2, z_2)\}$;
$D_3 = \{p(g(x_3), x_3), \neg p(t_3, x_3)\}$;
$D_4 = \{\neg p(z_4, g(x_4)), \neg p(z_4, x_4), \neg p(t_4, x_4)\}$;
$D_5 = \{\neg r(y_5, z_5), \neg p(x_5, y_5), p(x_5, z_5)\}$;
$D_6 = \{p(a, a)\}$.

И за двата варианта един примерен резолютивен извод на $\blacksquare$ е:
$D_7 = Res(D_2\{x_2/y_5, z_2/z_5\}, D_5) =$
$\{\neg q(f(y_5), z_5), \neg q(y_5, x_5), q(z_5, x_5)\}$;
$D_8 = Res(D_3\{x_3/f(y_5)\}, D_7\{z_5/g(f(y_5))\}) =$
$\{\neg q(f(y_5), t_3), \neg q(y_5, x_5), q(g(f(y_5)), x_5)\}$;
$D_9 = Res(D_4\{x_4/f(y_5), z_4/x_5\}, D_8) =$
$\{\neg q(f(y_5), t_4), \neg q(f(y_5), x_5), \neg q(f(y_5), t_3), \neg q(y_5, x_5)\}$;
$D_{10} = Collapse(D_9\{t_3/x_5, t_4/x_5\}) =$
$\{\neg q(f(y_5), x_5), \neg q(y_5, x_5)\}$;
$D_{11} = Res(D_1, D_{10}\{y_5/x_1, x_5/x_1\}) =$
$\{\neg q(x_1, x_1)\}$;
$Res(D_{11}\{x_1/a\}, D_6) = \blacksquare$.

За вариант I.2 заменете q с p.