Sveikaskaitis programavimas

A. Domarkas, VU, 2014

1 pavyzdys

Rasti maksimumą sveikųjų skaičių aibėje.

```
(%i1) f:30*x1 + 20*x2;
(%o1) 20 x2+30 x1
```

```
(%i2) apr:[3*x1+2*x2<=20, 2*x1<=14, x1+x2<=16, x1+2*x2<=18, x1>=0,x2>=0];
(%o2) [2 x2+3 x1<=20,2 x1<=14,x2+x1<=16,2 x2+x1<=18,x1>=0,x2>=0]
```

Detalus sprendimas:

```
(%i3) v:listofvars([f,apr]);
(%o3) [x1,x2]
```

```
(%i4) n:length(v);
(%o4) 2
```

```
(%i5) sritis:apply("and",apr);
(%o5) 2 x2+3 x1<=20∧2 x1<=14∧x2+x1<=16∧2 x2+x1<=18∧x1>=0∧x2>=0
```

```
(%i6) g(x):=ev(f,[x1=x[1],x2=x[2]]);
(%o6) g(x):=ev(f,[x1=x_1,x2=x_2])
```

```
(%i7) h(x):=ev(sritis,[x1=x[1],x2=x[2]]);
(%o7) h(x):=ev(sritis,[x1=x_1,x2=x_2])
```

```
(%i8) s:create_list([i,j],i,0,10,j,0,10)$
```

```
(%i9) s1:sublist(s,lambda([x],h(x)));
(%o9) [[0,0],[0,1],[0,2],[0,3],[0,4],[0,5],[0,6],[0,7],[0,8],[0,9],[1,0],[1
,1],[1,2],[1,3],[1,4],[1,5],[1,6],[1,7],[1,8],[2,0],[2,1],[2,2],[2,3],[2,4]
,[2,5],[2,6],[2,7],[3,0],[3,1],[3,2],[3,3],[3,4],[3,5],[4,0],[4,1],[4,2],[4
,3],[4,4],[5,0],[5,1],[5,2],[6,0],[6,1]]
```

```
(%i10) length(%);
(%o10) 43
```

```
(%i11) fv:map(g,s1);
(%o11) [0,20,40,60,80,100,120,140,160,180,30,50,70,90,110,130,150,170,190,60,80,
100,120,140,160,180,200,90,110,130,150,170,190,120,140,160,180,200,150,170,190,180
,200]
```

```
(%i12) m:lmax(fv);
(%o12) 200
```

```
(%i13) sublist(s1,lambda([x],g(x)=m));
(%o13) [[2,7],[4,4],[6,1]]
```

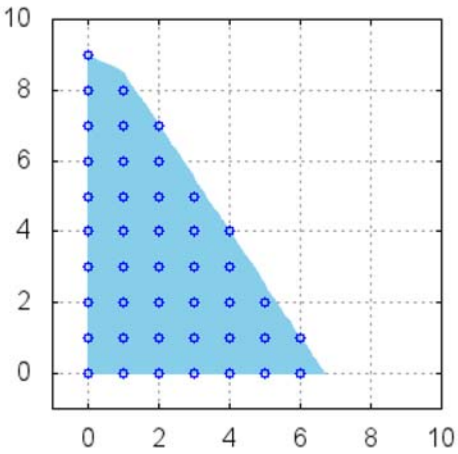
Atsakymas: f_max=200, kai x1=2, x2=7; x1=4, x2=4; x1=6, x2=1.

```
(%i76) load(draw)$
```

```
(%i15) set_draw_defaults(
           x_voxel = 30,
           y_voxel = 30,
           xrange = [-1,10],
           yrange = [-1,10],
           grid   = true,
           proportional_axes = xy,
           fill_color = skyblue)$
```

```
(%i16) wxdraw2d(region(sritis,x1,0,10,x2,0,10),
           point_type    = circle,
           point_size    = 1,
           points(s1));
```

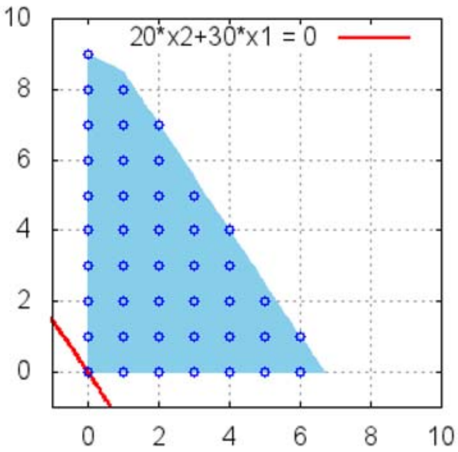
(%t16)



(%o16)

```
(%i17) with_slider_draw(
          z, makelist(50*i, i, 0, 4),
          region(sritis, x1, -1, 10, x2, -1, 9),
          key   = string(ev(f,nouns)=z),
          color=red,
          line_width = 2,
          implicit(f=z,x1,-1,10,x2, -1, 9),
          key="",
          color=blue,
          point_type=circle,
          points(s1)
          );
```

(%t17)



(%o17)

Ištirkime atvejį, kai reikia rasti maksimumą realiųjų skaičių aibėje.

Su simplex visų sprendinių negauname:

```
(%i18) load(simplex)$
```

```
(%i19) maximize_lp(f,apr);
```

(%o19) $[200,[x2=\frac{17}{2},x1=1]]$

```
(%i20) subst(%[2],[x1,x2]);
```

(%o20) $[1,\frac{17}{2}]$

Su nopt: ((C), A.Domarkas)

```
(%i21) load(nopt)$
```

```
(%i22) spr:maximize_nopt(f,apr);
```

(%o22) $[200,[x1=1,x2=\frac{17}{2}],[x1=\frac{20}{3},x2=0]]$

```
(%i23) A:subst(spr[2],[x1,x2]);
```

(%o23) $[1,\frac{17}{2}]$

```
(%i24) B:subst(spr[3],[x1,x2]);
```

(%o24) $[\frac{20}{3},0]$

```
(%i25) ats:A*(1-t)+B*t;
```

(%o25) $[\frac{17\,t}{3}+1,\frac{17\,(1-t)}{2}]$

kai 0 <= t <=1.

2 ilp programa ((C) A.Domarkas)

```
(%i26) minimize_ilp(f,apr):=block([v,n,i,k,simplex,xv,fv,g,h,s,m],
       load(simplex),
       v:listofvars([f,apr]),
       n:length(v),
       for k thru n do
       (minimize_lp(v[k],apr)[1],if numberp(%%) then m[k]:round(%%) else m[k]:-10,
       maximize_lp(v[k],apr)[1],if numberp(%%) then M[k]:round(%%) else M[k]:10),
       xv:makelist(v[i]=x[i],i,1,n),
       g(x):=ev(f,xv),
       h(x):=apply("and",ev(apr,xv)),
       flatten(makelist([v[i],m[i],M[i]],i,1,n)),
       cons (v,%%),
       apply(create_list,%%),
       s:sublist(%%,lambda([x],h(x))),
       fv:map(g,s),
       m:lmin(fv),
       sublist(s,lambda([x],g(x)=m)),
       cons(m,%%)
       )$
```

```
(%i27) maximize_ilp(f,apr):=block([k],minimize_ilp(-f,apr),
       cons(-%%[1],makelist(%%[k],k,2,length(%%))))$
```

1 pavyzdžio sprendimas su ilp :

```
(%i28) maximize_ilp(f,apr);
(%o28) [200,[2,7],[4,4],[6,1]]
```

3 ilp taikymo pavyzdžiai

2 pavyzdys. Rasti maksimumą sveikųjų skaičių aibėje.

```
(%i29) f:9*x1+4*x2;
(%o29) 4 x2+9 x1
```

```
(%i30) apr:[4*x1+3*x2<=11,x1>=0,x2>=0];
(%o30) [3 x2+4 x1<=11, x1>=0, x2>=0]
```

```
(%i31) maximize_ilp(f,apr);
(%o31) [22, [2, 1]]
```

Atsakymas: f_max=22, kai x1=2, x2=1.

3 pavyzdys([3], 57 p.) Rasti maksimumą sveikųjų skaičių aibėje.

```
(%i35) f:30*x1+20*x2;
(%o35) 20 x2+30 x1
```

```
(%i36) apr:[10*x1+5*x2<=300,5*x1+4*x2<=200,15*x1+10*x2<=600,x1>=0,x2>=0];
(%o36) [5 x2+10 x1<=300, 4 x2+5 x1<=200, 10 x2+15 x1<=600, x1>=0, x2>=0]
```

```
(%i37) maximize_ilp(f,apr);
(%o37) [1060, [12, 35], [14, 32]]
```

Atsakymas: f_max=1060, kai x1=12, x2=35; x1=14, x2=32; x1=6.

4 pavyzdys([3], 59 p.) Rasti maksimumą sveikųjų skaičių aibėje.

```
(%i38) f:0.8*x1+0.4*x2;
(%o38) 0.4 x2+0.8 x1
```

```
(%i39) apr:[31*x1+19.5*x2<=1500,2*x1+3*x2>=200,x1>=0,x2>=0];
(%o39) [19.5 x2+31 x1<=1500, 3 x2+2 x1>=200, x1>=0, x2>=0]
```

```
(%i40) maximize_ilp(f,apr);
(%o40) [32.40000000000001, [10, 61]]
```

Atsakymas: f_max=32.4, kai x1=10, x2=61.

5 pavyzdys([2], 151 p.) Rasti maksimumą sveikųjų skaičių aibėje.

```
(%i41) f:7*x1+4*x2;
(%o41) 4 x2+7 x1
```

```
(%i42) apr:[3*x1+2*x2<=21,6*x1+3*x2<=37,x1>=0,x2>=0];
(%o42) [2 x2+3 x1<=21, 3 x2+6 x1<=37, x1>=0, x2>=0]
```

```
(%i43) maximize_ilp(f,apr);
(%o43) [45, [3, 6]]
```

Atsakymas: f_max=45, kai x1=3, x2=6.

6 pavyzdys ([1], 191 p.) Rasti minimumą sveikųjų skaičių aibėje.

```
(%i44) f:5*x1-x2+x3;
(%o44) x3-x2+5 x1
```

```
(%i45) apr:[3*x1-x2-x3=4, x1-x2+x3-x4=1, 2*x1+x2+2*x3+x5=7, x1>=0,x2>=0,x3>=0, x4>=0,x5>=0];
(%o45) [-x3-x2+3 x1=4, -x4+x3-x2+x1=1, x5+2 x3+x2+2 x1=7, x1>=0, x2>=0, x3>=0, x4>=0, x5
>=0]
```

```
(%i46) minimize_ilp(f,apr);
(%o46) [10,[2,1,1,1,0]]
```

```
(%i47) maximize_ilp(f,apr);
(%o47) [10,[2,1,1,1,0]]
```

Gauname, kad yra tik vienas sveikaskaitis taškas [2,1,1,1,0], kuris tenkina apribojimus.
Todėl sveikų skaičių aibėje maksimumas ir minimumas sutampa

Atsakymas: f_min=f_max=10, kai x1=2, x2=1, x3=1, x4=1, x5=0.

7 pavyzdys. Rasti minimumą sveikųjų skaičių aibėje.

```
(%i48) f:2*x1+3*x2+4*x3-x4;
```
(%o48) $-x4+4\,x3+3\,x2+2\,x1$

```
(%i49) apr:[x1+2*x2>=9,3*x2+x3>=9,x2+x4<=10,x1>=0,x2>=0,x3>=0,x4>=0];
```
(%o49) $[2\,x2+x1>=9, x3+3\,x2>=9, x4+x2<=10, x1>=0, x2>=0, x3>=0, x4>=0]$

```
(%i50) minimize_ilp(f,apr);
(%o50) [8,[1,4,0,6],[3,3,0,7]]
```

Atsakymas: f_min=10, kai x1=1, x2=4, x3=0, x4=6 arba x1=3, x2=3, x3=0, x4=7.

4 Netiesinis sveikaskaitis programavimas

Rasti funkcijos minimumą ir maksimumą, kai x1 ir x2 yra sveikieji skaičiai.

```
(%i51) f:(x1-13)^2+(x2-14)^2;
```
(%o51) $(x2-14)^2+(x1-13)^2$

```
(%i52) apr:[(x1-8)^2+(x2-9)^2<=49,x1>=2,x2<=13,x1+x2<=24];
```
(%o52) $[(x2-9)^2+(x1-8)^2<=49, x1>=2, x2<=13, x2+x1<=24]$

```
(%i53) v:listofvars([f,apr]);
```
(%o53) $[x1, x2]$

```
(%i54) n:length(v);
(%o54) 2
```

```
(%i55) sritis:apply("and", apr);
```
(%o55) $(x2-9)^2+(x1-8)^2<=49 \wedge x1>=2 \wedge x2<=13 \wedge x2+x1<=24$

Randame min ir max su programa nopt:

```
(%i56) load(nopt)$
```

```
(%i57) minimize_nopt(f,apr);
```
(%o57) $[\frac{9}{2}, [x1=\frac{23}{2}, x2=\frac{25}{2}]]$

```
(%i58) maximize_nopt(f,apr);
```
(%o58) $[\left(-\frac{7\sqrt{2}-16}{2}-13\right)^2+\left(-\frac{7\sqrt{2}-18}{2}-14\right)^2, [x1=-\frac{7\sqrt{2}-16}{2}, x2=-\frac{7\sqrt{2}-18}{2}]]$

```
(%i59) ratsimp(%);
```
(%o59) $[35\,2^{3/2}+99, [x1=-\frac{7\sqrt{2}-16}{2}, x2=-\frac{7\sqrt{2}-18}{2}]]$

```
(%i60) float(%), numer;
```
(%o60) $[197.9949493661167, [x1=3.050252531694167, x2=4.050252531694167]]$

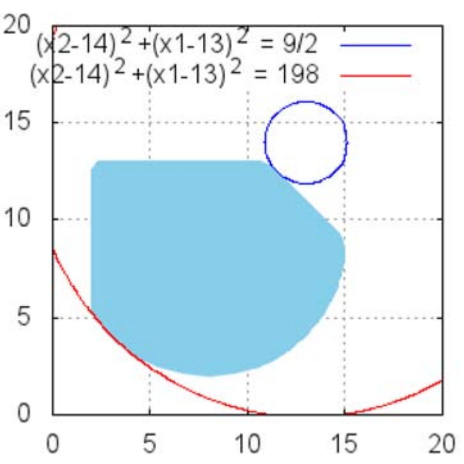
Sprendiniai nėra sveikaskaitiniai.

```
(%i61) load(draw)$
```

```
(%i62) set_draw_defaults(
                  x_voxel = 30,
                  y_voxel = 30,
                  xrange = [0,20],
                  yrange = [0,20],
                  grid   = true,
                  proportional_axes = xy,
                  fill_color = skyblue)$
```

```
(%i63) wxdraw2d( region(sritis, x1, 0, 20, x2, 0, 20),
                key    = string(ev(f,nouns)=9/2),
                implicit(f=9/2,x1,0,20,x2, 0, 20),
                color=red,
                key    = string(ev(f,nouns)=198),
                implicit(f=198,x1,0,20,x2, 0, 20));
```

(%t63)



(%o63)

Detalus spendimas:

```
(%i64) g(x):=ev(f,[x1=x[1],x2=x[2]]);
```

(%o64) $g(x):=ev(f,[x1=x_1,x2=x_2])$

```
(%i65) h(x):=ev(sritis,[x1=x[1],x2=x[2]]);
```

(%o65) $h(x):=ev(sritis,[x1=x_1,x2=x_2])$

```
(%i66) s:create_list([i,j],i,0,20,j,0,20)$
```

```
(%i67) s1:sublist(s,lambda([x],h(x)));
```

```
(%o67) [[2,6],[2,7],[2,8],[2,9],[2,10],[2,11],[2,12],[3,5],[3,6],[3,7],[3,8],
[3,9],[3,10],[3,11],[3,12],[3,13],[4,4],[4,5],[4,6],[4,7],[4,8],[4,9],[4,10]
,[4,11],[4,12],[4,13],[5,3],[5,4],[5,5],[5,6],[5,7],[5,8],[5,9],[5,10],[5,11
],[5,12],[5,13],[6,3],[6,4],[6,5],[6,6],[6,7],[6,8],[6,9],[6,10],[6,11],[6,
12],[6,13],[7,3],[7,4],[7,5],[7,6],[7,7],[7,8],[7,9],[7,10],[7,11],[7,12],[7
,13],[8,2],[8,3],[8,4],[8,5],[8,6],[8,7],[8,8],[8,9],[8,10],[8,11],[8,12],[8
,13],[9,3],[9,4],[9,5],[9,6],[9,7],[9,8],[9,9],[9,10],[9,11],[9,12],[9,13],[
10,3],[10,4],[10,5],[10,6],[10,7],[10,8],[10,9],[10,10],[10,11],[10,12],[10,13
],[11,3],[11,4],[11,5],[11,6],[11,7],[11,8],[11,9],[11,10],[11,11],[11,12],[
11,13],[12,4],[12,5],[12,6],[12,7],[12,8],[12,9],[12,10],[12,11],[12,12],[13,5
],[13,6],[13,7],[13,8],[13,9],[13,10],[13,11],[14,6],[14,7],[14,8],[14,9],[14
,10],[15,9]]
```

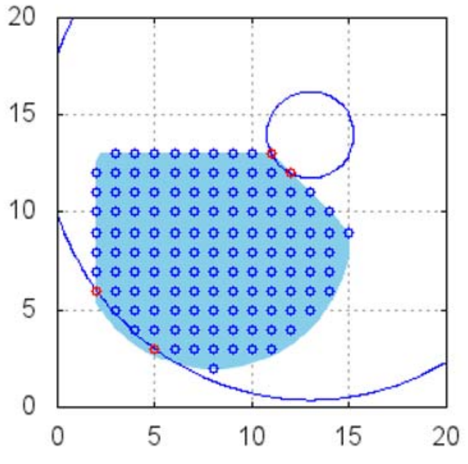
```
(%i68) length(%);
(%o68) 126
```

```
(%i69) fv:map(g,s1);
(%o69) [185,170,157,146,137,130,125,181,164,149,136,125,116,109,104,101,181,162,145
,130,117,106,97,90,85,82,185,164,145,128,113,100,89,80,73,68,65,170,149,130,113,
98,85,74,65,58,53,50,157,136,117,100,85,72,61,52,45,40,37,169,146,125,106,89,74,
61,50,41,34,29,26,137,116,97,80,65,52,41,32,25,20,17,130,109,90,73,58,45,34,25,
18,13,10,125,104,85,68,53,40,29,20,13,8,5,101,82,65,50,37,26,17,10,5,81,64,49,36
,25,16,9,65,50,37,26,17,29]

(%i70) lmax(fv);
(%o70) 185

(%i71) s2:sublist(s1,lambda([x],g(x)=185));
(%o71) [[2,6],[5,3]]

(%i72) lmin(fv);
(%o72) 5

(%i73) s3:sublist(s1,lambda([x],g(x)=5));
(%o73) [[11,13],[12,12]]
```

Atsakymas: f_max=185, kai x1=2, x2=6 arba x1=5, x2=3;
f_min=5, kai x1=11, x2=13 arba x1=12, x2=12.

```
(%i74) wxdraw2d(region(sritis,x1,0,15,x2,0,15),
               point_type    = circle,
               point_size    = 1,
               implicit(f=185,x1,0,20,x2, 0, 20),
               implicit(f=5,x1,0,20,x2, 0, 20),
               points(s1),
               color=red,
               points(s2),
               points(s3)
        );
```

(%t74)

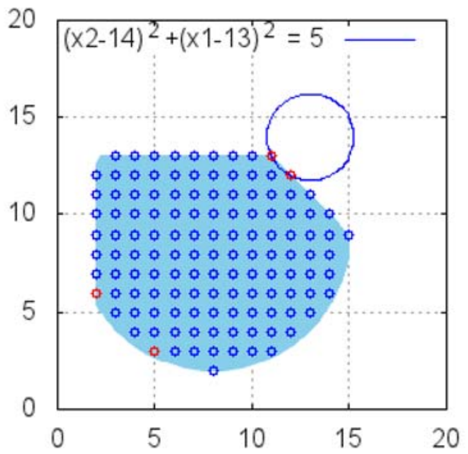


(%o74)

Brėžinys su animacija:

```
(%i75) with_slider_draw(
            z, makelist(5+20*i, i, 0, 9),
            region(sritis, x1, 0, 15, x2, 0, 15),
            key   = string(ev(f,nouns)=z),
            implicit(f=z,x1,0,20,x2,0,20),
            key="",
            color=blue,
            point_type=circle,
            points(s1),
            color=red,
            points(s2),
            points(s3));
```

(%t75)



(%o75)

5 Literatūra

[1] A.Apynis. Optimizavimo metodai, VU, Vilnius, 2005
[2] V.Čiočys, R.Jasilionis, Matematinis programavimas, Vilnius, Mokslas, 1990
[3] S.Puškoius, Sprendimų priėmimo teorija, Vilnius, MRU, 2009
[4] V.Bubelis, T.Medaiskis, A.Morkeliūnas, Operacijų tyrimo įvadas, Vilnius, VU, 2008