```
unit rules450;
    interface
         uses
          globals,cmmnds1,pusherr,pushStack,ruleAtoF;


         procedure r401; procedure r402; procedure r403;
procedure r404; procedure r405;
     procedure r406; procedure r407; procedure r408; procedure
r409; procedure r410;
           procedure r411; procedure r412; procedure r413;
procedure r414; procedure r415;
           procedure r416; procedure r417; procedure r418;
procedure r419; procedure r420;
           procedure r421; procedure r422; procedure r423;
procedure r424; procedure r425;
           procedure r426; procedure r427; procedure r428;
procedure r429; procedure r430;
        procedure r431; procedure r432; procedure r433; procedure
r434; procedure r435;
           procedure r436; procedure r437; procedure r438;
procedure r439; procedure r440;
           procedure r441; procedure r442; procedure r443;
procedure r444; procedure r445;
           procedure r446; procedure r447; procedure r448;
procedure r449; procedure r450;


      implementation


procedure r401;
(*********************************************)
(*                                           *)
(*  if maximal plnar and maxdeg <= mindeg+1    *)
(*           then nconn = mindeg              *)
(*                                           *)
(*********************************************)
begin
  if (activerule[401]) and (max[plnar] = 1) then
    begin
      rule:='401/ ';
      if (min[plnar] = 1) and (min[edges] = 3*max[nodes]-6) then
         begin
           z:=max[maxdeg]-2;
           if z < max[nconn] then z:=max[nconn];
           if z < max[mindeg] then pushmax(mindeg);
         end;
      if (min[plnar] = 1) and (min[edges] = 3*max[nodes]-6)
         and (max[maxdeg] <= min[mindeg]+1) then
           begin
```

```
              z:=min[mindeg];
              if z > min[nconn] then pushmin(nconn);
            end
          else
            if (max[nconn] < min[mindeg]) and (min[plnar] = 1)
                and (min[edges] = 3*max[nodes]-6) then
                  begin
                    z:=min[mindeg]+2;
                    if z > min[maxdeg] then pushmin(maxdeg);
                  end
           else
             if (max[maxdeg] <= min[mindeg]+1) and (max[nconn]
< min[mindeg])
                 and (min[plnar] = 1) then
                  begin
                    if max[nodes] < infinity then
                     begin
                       z:=3*max[nodes]-7;
                       if z < max[edges] then pushmax(edges);
                     end;
                    z:=(min[edges]+5) div 3;
                    if z > min[nodes] then pushmin(nodes);
                  end
           else
             if (min[edges] = 3*max[nodes]-6) and (max[maxdeg] <=
min[mindeg]+1)
                  and (max[nconn] < min[mindeg]) then
                  begin
                    z:=0;
                    pushmax(plnar);
                  end;
    end;
end;

procedure r402;
(*************************************************)
(*                                               *)
(*  Bw <= p-(mindeg+1)(Nc-1)-1-FL((nind-Nc+1)/2) *)
(*                                               *)
(*************************************************)
begin
  if activerule[402] then
    begin
      rule:='402/ ';
      if max[nodes] < infinity then
        begin
          z:=max[nodes]-(min[mindeg]+1)*(min[ncomp]-1)-1;
          z:=z-(min[nind]-min[ncomp]+1) div 2;
          if z < max[bwidth] then pushmax(bwidth);
          z:=max[nodes]-min[bwidth]-1-(min[nind]-min[ncomp]+1)
div 2;
          if min[ncomp] > 1 then
```

```
            begin
              z:=z div (min[ncomp]-1)-1;
              if z < max[mindeg] then pushmax(mindeg);
            end;
          z:=2*(max[nodes]-min[bwidth])-(min[ncomp]-1)*(2
*min[mindeg]+1)-1;
          if z < max[nind] then pushmax(nind);
          z:=(2*(max[nodes]-min[bwidth])-min[nind]-1) div (2
*min[mindeg]+1)+1;
          if z < max[ncomp] then pushmax(ncomp);
         end;
       z:=min[bwidth]+(min[mindeg]+1)*(min[ncomp]-1);
       z:=z+(min[nind]-min[ncomp]+1) div 2;
       if z > min[nodes] then pushmin(nodes);
    end;
end;

procedure r403;
(*************************************************)
(*                                               *)
(*  regular, mindeg > nconn then eind >=(p-t)/2  *)
(*     where t is same parity as p and where     *)
(*      p <= (t+3)*(2*CL(mindeg/2)+1) + x        *)
(*        and x = 0, mindeg, 2*mindeg-2          *)
(*          if nconn >= 1, 2, 3, respectively    *)
(*                                               *)
(*************************************************)
begin
  if (activerule[403]) and (min[nconn] > 0) and (max[reg] = 1)
then
    begin
      rule:='403/ ';
      z:=0;
      if min[nconn] >= 3 then z:= 2*min[mindeg]-2
               else if min[nconn] = 2 then z:=min[mindeg];
      z1:=2*((min[mindeg]+1) div 2)+1;
      z1:=(max[nodes]-z-1) div z1-2;
      if ((odd (max[nodes])) and (not(odd(z1)))) or
         ((not(odd(max[nodes]))) and (odd(z1))) then z1:=z1+1;
      z:= (min[nodes]-z1+1) div 2;
      if (min[reg] = 1) and (min[mindeg] > max[nconn]) then
          begin
            if z > min[eind] then pushmin(eind);
            z:=2*max[eind]+z1;
            if z < max[nodes] then pushmax(nodes);
          end
        else
          if (max[eind] < z) and (min[reg] = 1) then
              begin
                z:=max[nconn];
                if z < max[mindeg] then pushmax(mindeg);
                z:=min[mindeg];
```

```
                if z > min[nconn] then pushmin(nconn);
              end
            else
              if (max[eind] < z) and (min[mindeg] > max[nconn])
then
                 begin
                   z:=0;
                   pushmax(reg);
                 end;
    end;
end;

procedure r404;
(*******************************************************)
(*                                                     *)
(*  if mindeg > econn = nconn then p >= mindeg+maxdeg  *)
(*                                                     *)
(*******************************************************)
begin
  if activerule[404] then
    begin
      rule:='404/ ';
      z1:=max[nodes];
      if (max[econn] = min[nconn]) and (z1 < infinity) then
           begin
             z:=z1-min[maxdeg];
             if z < max[nconn] then z:=max[nconn];
             if z < max[mindeg] then pushmax(mindeg);
           end;
      if (min[mindeg] > max[econn]) and (max[econn] = min[nconn])
then
         begin
           z:=min[mindeg]+min[maxdeg];
           if z > min[nodes] then pushmin(nodes);
           if z1 < infinity then
               begin
                 z:=z1-min[mindeg];
                 if z < max[maxdeg] then pushmax(maxdeg);
                 z:=z1-min[maxdeg];
                 if z < max[mindeg] then pushmax(mindeg);
               end;
         end
       else
         if (z1 < min[mindeg]+min[maxdeg]) and
            (min[mindeg] > max[econn]) then
              begin
                z:=min[nconn]+1;
                if z > min[econn] then pushmin(econn);
                z:=max[econn]-1;
                if z < max[nconn] then pushmax(nconn);
              end
            else
```

```
              if (max[econn] = min[nconn]) and
                 (z1 < min[mindeg]+min[maxdeg]) then
                   begin
                     z:=min[mindeg];
                     if z > min[econn] then pushmin(econn);
                     z:=max[econn];
                     if z < max[mindeg] then pushmax(mindeg);
                   end;
    end;
end;

procedure r405;
(*********************************************)
(*                                           *)
(* if mindeg > econn=nconn and diam=3 then   *)
(*           dom <= econn+1                  *)
(*                                           *)
(*********************************************)
begin
  if (activerule[405]) and (min[diam] <= 3) and (max[diam] >= 3)
then
    begin
      rule:='405/ ';
      if (max[diam] = 3) and (min[diam] = 3) and (max[econn] =
min[nconn]) then
          begin
            z:=min[dom]-1;
            if z > min[mindeg] then z:=min[mindeg];
            if z > min[econn] then pushmin(econn);
          end;
      if (min[mindeg] > max[econn]) and (max[econn] = min[nconn])
and
         (max[diam] = 3) and (min[diam] = 3) then
            begin
              z:=max[econn]+1;
              if z < max[dom] then pushmax(dom);
            end
         else
           if (min[dom] > max[econn]+1) and (min[mindeg] >
max[econn])
              and (max[econn] = min[nconn]) then
               begin
                 if min[diam] = 3 then
                      begin
                        z:= 4;
                        pushmin(diam);
                      end
                   else if max[diam] = 3 then
                           begin
                             z:=2;
                             pushmax(diam);
                           end;
```

```
              end
            else
             if (max[diam] = 3) and (min[diam] = 3) and
                (min[dom] > max[econn]+1) and (min[mindeg] >
max[econn]) then
                  begin
                    z:=min[nconn]+1;
                    if z > min[econn] then pushmin(econn);
                    z:=max[econn]-1;
                    if z < max[nconn] then pushmax(nconn);
                   end
               else
                 if (max[econn] = min[nconn]) and (max[diam] =
3)
                    and (min[diam] = 3) and (min[dom] >
max[econn]+1) then
                       begin
                         z:=max[econn];
                         if z < max[mindeg] then
pushmax(mindeg);
                       end;
    end;
end;

procedure r406;
(*********************************************)
(*                                           *)
(*  if (p-1)**2 div 4 < e <= (p-1)*(p-2)/2 then *)
(*    mindeg <= econn-1+((p-sqrt(4e+2p-p**2))/2 *)
(*                                           *)
(*********************************************)
var z2,z3:longint;
begin
  if (activerule[406]) and (max[nodes] < infinity) then
    begin
      rule:='406/ ';
      z:=max[nodes];
      z:=4*min[edges]+2*z-z*z;
      if z >= 0 then
         begin
           z:=max[econn]-1+trunk((max[nodes]-sqrt(z))/2);
           z1:=max[nodes]-1;
           z1:=(z1*z1) div 4;
           z2:=min[nodes]-1;
           z2:=z2*(z2-1) div 2;
           if (z1 < min[edges]) and (max[edges] <= z2) then
              begin
                if z < max[mindeg] then pushmax(mindeg);
                z:=min[mindeg]+max[econn]-z;
                if z > min[econn] then pushmin(econn);
                z:=min[mindeg]-max[econn]+1;
                k:=max[mindeg]-min[econn]+1;
```

```
                if z > 1 then
                   begin
                     z1:=max[nodes];
                     z:=z1*(z1-1) div 2-z1*z+z*z;
                     z3:=z1*(z1-1) div 2-z1*k+k*k;
                     if z3 > z then z:= z3;
                     if z < max[edges] then pushmax(edges);
                   end;
              end
            else
             if (min[mindeg] > z) and (z1 < min[edges]) then
                begin
                  z:=z2+1;
                  if min[edges] < z then pushmin(edges);
                  z:=trunk((3+sqrt(8*max[edges]-3))/2);
                  if z < max[nodes] then pushmax(nodes);
                end
              else
                if (min[mindeg] > z) and (max[edges] <= z2) then
                  begin
                    z:=z1;
                    if z < max[edges] then pushmax(edges);
                    z:=round(2*sqrt(min[edges])+hf)+1;
                    if z > min[nodes] then pushmin(nodes);
                  end;
         end;
    end;
end;

procedure r407;
(*********************************************)
(*                                           *)
(*  dom <= (p-maxdeg-1)*(p-mindeg-2)/(p-1)+2 *)
(*                                           *)
(*********************************************)
begin
  if (activerule[407]) and (min[dom] > 1) then
    begin
      rule:='407/ ';
      z:=min[mindeg]+min[maxdeg]+min[dom]-1;
      rz:=z*z-4*min[maxdeg]*(min[mindeg]+1);
      if rz >= 0.0 then
           begin
             z:=round((z+2+sqrt(rz))/2+hf);
             if z > min[nodes] then pushmin(nodes);
           end;
      z:=max[nodes];
      if z < infinity then
           begin
             if min[nodes] = z then
               begin
                 z:=((z-min[maxdeg]-1)*(z-min[mindeg]-2)) div
```

```
(z-1)+2;
                if z < max[dom] then pushmax(dom);
              end;
             z1:=(min[dom]-2)*(max[nodes]-1)-1;
             if z1 >= 0 then
              begin
                z:=max[nodes]-min[mindeg]-2;
                if z > 0 then
                   begin
                     z:=max[nodes]-2-z1 div z;
                     if z < max[maxdeg] then pushmax(maxdeg);
                   end;
                z:=max[nodes]-min[maxdeg]-1;
                if z > 0 then
                   begin
                     z:=max[nodes]-3-z1 div z;
                     if z < max[mindeg] then pushmax(mindeg);
                   end;
              end;
            end;
    end;
end;

procedure r408;
(*******************************************************)
(*                                                     *)
(* let m = maxdeg+2 and s = (m-sqrt(m**2-4p))/2        *)
(*  if s <= FL((p/2)**(1/3)) and diam =3                *)
(*               then e >= p+s*(s-1)/2-1               *)
(*                                                     *)
(*******************************************************)
begin
  if activerule[408]then
    begin
      rule:='408/ ';
      z:=min[maxdeg]+2;
      z1:=z*z-4*min[nodes];
      if z1 >= 0 then
        begin
          z:=trunk((z-sqrt(z1))/2);
          z1:=trunk(root(min[nodes]/2,3));
          if z1 >= z then
             begin
               z:=min[nodes]-1+(z*(z-1)) div 2;
               if (min[diam] = 3) and (max[diam] = 3) then
                 begin
                   if z > min[edges] then pushmin(edges);
                 end
               else
                 if max[edges] < z then
                   begin
                     z:=4;
```

```
                    if min[diam] = 3 then pushmin(diam)
                      else
                        if max[diam] = 3 then
                          begin
                            z:=2;
                            pushmax(diam);
                          end;
                  end;
              end;
          end;
    end;
end;

procedure r409;
(************************************************)
(*                                              *)
(* if p >= 4, hamil, and clique = 2 then        *)
(*          e <= FL((p-4)/2)*Fl(p/2)+p          *)
(*                                              *)
(************************************************)
begin
  if (activerule[409]) and (max[nodes] < infinity) and
(max[hamil] = 1)
      and (min[clique] = 2) and (max[nodes] >= 4) then
    begin
      rule:='409/ ';
      z:=max[nodes] div 2;
      z:=z*z-2*z+max[nodes];
      if (min[hamil] = 1) and (max[clique] = 2) then
          begin
            if z < max[edges] then pushmax(edges);
          end
        else
          if (z < min[edges]) and (min[hamil] = 1) then
            begin
              z:=3;
              pushmin(clique);
            end
          else
            if (z < min[edges]) and (max[clique] = 2) then
              begin
                z:=0;
                pushmax(hamil);
              end;
    end;
end;

procedure r410;
(*****************************)
(*                           *)
(*     spectral >= mindeg    *)
(*                           *)
```

```
(***************************)
begin
  if activerule[410] then
    begin
      rule:='410/ ';
      rz:=min[mindeg];
      if rz > lammin then pushlammin;
      z:=trunk(lammax);
      if z < max[mindeg] then pushmax(mindeg);
    end;
end;

procedure r411;
(*************************************************)
(*                                               *)
(*  if connected and p >= 2*mindeg+2 then        *)
(*     diam <= 3*(p div (mindeg+1))-3+k          *)
(*        where k = 0 if p = s*(mindeg+1)        *)
(*                = 1 if p = s*(mindeg+1)+1      *)
(*                = 2 otherwise                  *)
(*                    for s an longint           *)
(*                                               *)
(*************************************************)
var k:longint;
begin
  if (activerule[411]) and (max[diam] < infinity) then
    begin
      rule:='411/ ';
      if min[nodes] >= 2*max[mindeg]+2 then
        begin
          if max[nodes] < infinity then
            begin
              z:=max[nodes] div (min[mindeg]+1);
              z1:=max[nodes]-(min[mindeg]+1)*z;
              if (max[nodes] > min[nodes]) or (max[mindeg] >
min[mindeg])
                  or (z1 > 2) then z1:=2;
              z:=3*z-3+z1;
              if z < max[diam] then pushmax(diam);
              k:=(min[diam]+5-z1) div 3;
              if k > 0 then
                begin
                  z:=max[nodes] div k-1;
                  if z < max[mindeg] then pushmax(mindeg);
                end;
            end
           else k:=(min[diam]+3) div 3;
          z:=k*(min[mindeg]+1);
          if z > min[nodes] then pushmin(nodes);
        end
       else
         if (max[nodes] < infinity) and
```

```
            (min[diam] > 3*(max[nodes] div (min[mindeg]+1))-1)
then
            begin
              z:=2*max[mindeg]+1;
              if z < max[nodes] then pushmax(nodes);
              z:=min[nodes] div 2;
              if z > min[mindeg] then pushmin(mindeg);
            end;
    end;
end;

procedure r412;
(*********************************************************)
(*                                                       *)
(* if not a forest and 3 <= mindeg <= maxdeg-3 then      *)
(*   eind >= (p*(min-1)+(max-min+1)*T)/(max+min-1)       *)
(*                                                       *)
(*   where:  t = (girth-1) div 2                         *)
(*           S = ((min-1)**(t-1)-1)/(min*(min-2))        *)
(*           k = (1-(-1)**(t-1))/2                       *)
(*           T = max*S-(max-min)*k/min,    if g is odd   *)
(*             = (max+min-2)*S-(max-2)*k/min, if g is even *)
(*                                                       *)
(*********************************************************)
var t,k:longint;
    s,t1:real;
begin
  if (activerule[412]) and (min[mindeg] >= 3) and
     (min[mindeg] <= min[maxdeg]) and (min[girth] < infinity)
then
    begin
      rule:='412/ ';
      t:=(min[girth]-1) div 2;
      if odd(t) then k:=0
                else k:=1;
      if t = 1 then z:=1
         else if t = 2 then z:=min[mindeg]-1
           else power(min[mindeg]-1,t-1,z);
      if z < infinity then
       begin
         s:=(z-1)/(min[mindeg]*(min[mindeg]-2));
         if odd(min[girth]) then
           t1:=min[maxdeg]*s-(min[maxdeg]-
min[mindeg])*k/min[mindeg]
          else
            t1:=(min[maxdeg]+min[mindeg]-2)*s-
(min[maxdeg]-2)*k/min[mindeg];
         rz:=min[nodes]*(min[mindeg]-1)+(min[maxdeg]-min[mindeg]+
1)*t1;
         z:=round(rz/(max[maxdeg]+min[mindeg]-1));
         if max[mindeg] <= min[maxdeg]-3 then
             begin
```

```
              if z > min[eind] then pushmin(eind);
            end
          else
            if z > max[eind] then
             begin
               z:=min[maxdeg]-2;
               if z > min[mindeg] then pushmin(mindeg);
               z:=max[mindeg]+2;
               if z < max[maxdeg] then pushmax(maxdeg);
             end;
        end;
    end;
end;

procedure r413;
(*********************************************)
(*                                           *)
(*  e >= m(p-chr)+chr(chr-1)/2-(nind-1)m(m+1)/2 *)
(*     where  m = (p-chr) div (nind-1)       *)
(*                                           *)
(*********************************************)
begin
  if (activerule[413]) and (max[nind] > 1) and (max[nind]
< infinity) then
    begin
      rule:='413/ ';
      z:=min[nodes]-min[chr];
      z1:=z div (max[nind]-1);
      z:=z1*z+min[chr]*(min[chr]-1) div 2-(max[nind]-1)*z1*(z1+1)
div 2;
      if z > min[edges] then pushmin(edges);
    end;
end;

procedure r414;
(*********************************************)
(*                                           *)
(*  if d <= 4 then edges <=                  *)
(*    (p-2)*(p-3)/2-(p-2)*(d-4)*k-2k*(k-1)+  *)
(*            k**2*((d-2)*(d-3))/2           *)
(*     where k = nconn and d = diam          *)
(*                                           *)
(*********************************************)
var z2,z3:longint;
begin
  if (activerule[414]) and (min[nconn] > 0) and (min[diam] <= 4)
     and (min[diam] >= 2) and (max[nodes] < infinity) then
    begin
      rule:='414/ ';
      z:=max[nodes];
      z1:=min[diam];
      k:=min[nconn];
```

```
      z2:=((z-2)*(z-3)-2*(z-2)*(z1-4)*k-4
*k*(k-1)+k*k*(z1-2)*(z1-3)) div 2;
      k:=max[nconn];
      z3:=((z-2)*(z-3)-2*(z-2)*(z1-4)*k-4
*k*(k-1)+k*k*(z1-2)*(z1-3)) div 2;
      if z2 < z3 then z:=z3
                 else z:=z2;
      if z < min[edges] then
               begin
                 z:=5;
                 pushmin(diam);
               end
             else if z < max[edges] then pushmax(edges);
    end;
end;

procedure r415;
(*****************************)
(*                           *)
(*   dom <= (p+2-mindeg)/2   *)
(*                           *)
(*****************************)
begin
  if activerule[415] then
    begin
      rule:='415/ ';
      if max[nodes] < infinity then
           begin
             z:=(max[nodes]+2-min[mindeg]) div 2;
             if z < max[dom] then pushmax(dom);
             z:=max[nodes]+2-2*min[dom];
             if z < max[mindeg] then pushmax(mindeg);
           end;
      z:=2*min[dom]+min[mindeg]-2;
      if z > min[nodes] then pushmin(nodes);
    end;
end;

procedure r416;
(*********************************************)
(*                                           *)
(*  p is even and maxdeg = p-2 then          *)
(*   echr = maxdeg+1 iff                     *)
(*                e >= 2((p-2)/2)**2+1+mindeg*)
(*                                           *)
(*********************************************)
begin
  if (activerule[416]) and (max[nodes] = min[nodes]) and
     (not(odd(max[nodes]))) and (min[maxdeg] = max[maxdeg]) and
     (min[maxdeg] = min[nodes]-2) then
    begin
      rule:='416/ ';
```

```
      z1:=max[nodes]-2;
      z:=(z1*z1) div 2;
      if min[echr] = min[maxdeg]+1 then
                begin
                  z:=z+1+min[mindeg];
                  if z > min[edges] then pushmin(edges);
                  if max[edges] < infinity then
                    begin
                      z:=max[edges]-z+min[mindeg];
                      if z < max[mindeg] then pushmax(mindeg);
                    end;
                 end
               else
                  if max[echr] = min[maxdeg] then
                     begin
                       z:=z+max[mindeg];
                       if z < max[edges] then pushmax(edges);
                       z:=min[edges]-z+max[mindeg];
                       if z > min[mindeg] then pushmin(mindeg);
                     end
                  else
                    if min[edges] >= z+1+max[mindeg] then
                        begin
                          z:=min[maxdeg]+1;
                          if z > min[echr] then pushmin(echr);
                        end
                     else
                       if max[edges] < z+1+min[mindeg] then
                          begin
                            z:=min[maxdeg];
                            if z < max[echr] then pushmax(echr);
                          end;
    end;
end;

procedure r417;
(*********************************************)
(*                                           *)
(*  if maxdeg <= 3 and triangle-free then    *)
(*       edges >= 13*p/2-14*nind             *)
(*                                           *)
(*********************************************)
begin
  if (activerule[417]) and (min[clique] <= 2) and (min[maxdeg] <=
3) then
    begin
      rule:='417/ ';
      if (max[maxdeg] <= 3) and (max[clique] <= 2) then
           begin
             if max[nind] < infinity then
                begin
                  z:=(13*min[nodes]+1) div 2-14*max[nind];
```

```
                  if z > min[edges] then pushmin(edges);
                  if max[edges] < infinity then
                     begin
                       z:=(2*max[edges]+28*max[nind]) div 13;
                       if z < max[nodes] then pushmax(nodes);
                     end;
                end;
             z:=(13*min[nodes]+1) div 2-max[edges];
             if z > 0 then
               begin
                 z:=(z+13) div 14;
                 if z > min[nind] then pushmin(nind);
               end;
          end
        else
          if (max[edges] < infinity) and (max[nind] < infinity)
then
            begin
              z:=(13*min[nodes]+1) div 2-14*max[nind];
              if (max[edges] < z) and (max[maxdeg] <= 3) then
                 begin
                   z:=3;
                   pushmin(clique);
                 end
               else
                 if (max[edges] < z) and (max[clique] <= 2) then
                    begin
                      z:=4;
                      pushmin(maxdeg);
                    end;
            end;
    end;
end;

procedure r418;
(*************************************************)
(*                                               *)
(* if clique = 2 and maxdeg <= 2 then            *)
(*               e >= 7p-15nind                  *)
(*                                               *)
(*************************************************)
begin
  if (activerule[418]) and (min[clique] <= 2) and (min[maxdeg] <=
2) then
    begin
      rule:='418/ ';
      if (max[clique] <= 2) and (max[maxdeg] <= 2) then
         begin
           if max[nind] < infinity then
             begin
               z:=7*min[nodes]-15*max[nind];
               if z > min[edges] then pushmin(edges);
```

```
              if max[edges] < infinity then
                 begin
                   z:=(max[edges]+15*max[nind]) div 7;
                   if z < max[nodes] then pushmax(nodes);
                 end;
            end;
           z:=7*min[nodes]-max[edges];
           if z > 0 then
                 begin
                   z:=(z+14) div 15;
                   if z > min[nind] then pushmin(nind);
                 end;
         end
        else
          if (max[edges] < infinity) and (max[nind] < infinity)
then
            begin
              z1:=7*min[nodes]-15*max[nind];
              z:=3;
              if (max[edges] < z1) and (max[clique] <= 2) then
pushmin(maxdeg)
               else
                 if (max[edges] < z1) and (max[maxdeg] <= 2)
then
                        pushmin(clique);
            end;

    end;
end;



procedure r419;
(*************************************************)
(*                                               *)
(* if clique = 2 and ncov <= 3p/5 then           *)
(*         ncov <= (3p-sqrt(5e-p**2))/5          *)
(*                                               *)
(*************************************************)
begin
  if (activerule[419]) and (min[clique] <= 2) then
    begin
      rule:='419/ ';
      z:=max[nodes];
      z:=5*min[edges]-z*z;
      if (max[clique] = 2) and (5*max[ncov] <= 3*min[nodes]) then
        begin
          if max[nodes] < infinity then
            begin
              if z >= 0 then
                 begin
                   z:=trunk((3*max[nodes]-sqrt(z))/5);
```

```
                    if z < max[ncov] then pushmax(ncov);
                  end;
              z:=3*max[nodes]-5*min[ncov];
              z:=(max[nodes]*max[nodes]+z*z) div 5;
              if z < max[edges] then pushmax(edges);
            end;
          z:=2*min[edges]-min[ncov]*min[ncov];
          if z >= 0 then
              begin
                z:=round((3*min[ncov]+sqrt(z))/2+hf);
                if z > min[nodes] then pushmin(nodes);
              end;
        end
      else
        if z >= 0 then
              begin
                z:=trunk(3*max[nodes]-sqrt(z)/5);
                if (min[ncov] > z) and (max[clique] = 2) then
                      begin
                        z:=(3*min[nodes]) div 5+1;
                        if z > min[ncov] then pushmin(ncov);
                        z:=(5*max[ncov]-1) div 3;
                        if z < max[nodes] then pushmax(nodes);
                      end
                    else
                      if (min[ncov] > z) and (5*max[ncov] <= 3
*min[nodes]) then
                        begin
                          z:=3;
                          pushmin(clique);
                        end;
             end;
    end;
end;

procedure r420;
(*********************************************)
(*                                           *)
(* if clique = 2 and nind >= 2p/5 then       *)
(*           nind >= (2p+sqrt(5e-p**2))/5    *)
(*                                           *)
(*********************************************)
begin
  if (activerule[420]) and (min[clique] <= 2) and (max[nodes]
< infinity) then
    begin
      rule:='420/ ';
      if (max[clique] = 2) and (5*max[nind] >= 2*max[nodes]) then
        begin
          z:=max[nodes];
          z1:=5*min[edges]-z*z;
          if z1 >= 0 then
```

```
                  begin
                    z:=round((2*z+sqrt(z1))/5+hf);
                    if z > min[nind] then pushmin(nind);
                  end;
            z:=max[nind];
            if z < infinity then
              begin
                z1:=min[edges]-z*z;
                if z1 >= 0 then
                   begin
                     z:=trunk(2*z+sqrt(z1));
                     if z > min[nodes] then pushmin(nodes);
                   end;
                z:=max[nind];
                z1:=max[nodes]-2*z;
                z:=z1*z1+z*z;
                if z < max[edges] then pushmax(edges);
              end;
         end
      else
        if max[nind] < infinity then
          begin
            z:=max[nodes];
            z:=5*min[edges]-z*z;
            if z >= 0 then
              begin
                z:=round((2*min[nodes]+sqrt(z))/5+hf);
                if (max[nind] < z) and (max[clique] = 2) then
                   begin
                     z:=(2*max[nodes]-1) div 5;
                     if z < max[nind] then pushmax(nind);
                     z:=(5*min[nind]+2) div 2;
                     if z > min[nodes] then pushmin(nodes);
                   end
                  else
                    if (max[nind] < z) and (5*min[nind] >= 2
*max[nodes]) then
                       begin
                         z:=3;
                         pushmin(clique);
                       end;
              end;
          end;
    end;
end;

procedure r421;
(*************************************************)
(*                                               *)
(*   if nind = 2 and clique >= 2p/5 then         *)
(*        clique >= (2p+sqrt(p(3p-5)/2-5e))/5    *)
(*                                               *)
```

```
(*********************************************)
begin
  if (activerule[421]) and (min[nind] <= 2) and (max[nodes]
< infinity) then
    begin
      rule:='421/ ';
      z:=min[nodes];
      z1:=(z*(3*z-5)) div 2-5*max[edges];
      if (max[nind] = 2) and (5*min[clique] >= 2*max[nodes]) then
         begin
           if z1 >= 0 then
              begin
                z:=round((2*z+sqrt(z1))/5+hf);
                if z > min[clique] then pushmin(clique);
              end;
           if max[clique] < infinity then
               begin
                 z:=min[nodes];
                 z:=max[clique]*(4*z-5*max[clique])-(z*(z+1)) div
2;
                 if z > min[edges] then pushmin(edges);
               end;
         end
        else
         if (max[clique] < infinity) and (max[edges] < infinity)
            and (z1 >= 0) then
              begin
                z:=round((2*min[nodes]+sqrt(z1))/5+hf);
                if (max[clique] < z) and (max[nind] <= 2) then
                    begin
                      z:=2*max[nodes]-1;
                      if z >= 10 then
                         begin
                           z:=z div 5;
                           if z < max[clique] then
pushmax(clique);
                         end;
                      z:=(5*min[clique]+5) div 2;
                      if z > min[nodes] then pushmin(nodes);
                    end
                  else
                   if (max[clique] < z) and (5*min[clique] >= 2
*max[nodes]) then
                      begin
                        z:=3;
                        pushmin(nind);
                      end;
              end;
      end;
end;

procedure r422;
```

```
(*********************************************)
(*                                           *)
(* if not a forest then B >=                 *)
(*       (2ncov(girth-2)-p(girth-3))/(2p-2ncov) *)
(*                                           *)
(*********************************************)
begin
  if (activerule[422]) and (min[forest] = 0) then
    begin
      rule:='422/ ';
      if max[forest] = 0 then
         begin
           z1:=min[ncov];
           z:=max[bwidth];
           if z < infinity then
              begin
                z:=2*z+min[girth]-3;
                z:=(z1*(min[girth]-1)-1) div z+z1+1;
                if z > min[nodes] then pushmin(nodes);
              end;
           z:=max[nodes];
           if z < infinity then
             begin
               z:=2*(z-z1);
               z1:=2*z1*(min[girth]-2)-
max[nodes]*(min[girth]-3)-1;
               k:=2*z1*(max[girth]-2)-
max[nodes]*(max[girth]-3)-1;
               if k < z1 then z1:=k;
               z:=z1 div z+1;
               if z > min[bwidth] then pushmin(bwidth);
               z:=max[bwidth];
               if z < infinity then
                 begin
                   z1:=min[girth]-2;
                   z:=(max[nodes]*(2*z+z1-1)) div (2*z+2*z1);
                   if z < max[ncov] then pushmax(ncov);
                   z1:=2*min[ncov]-max[nodes];
                   if z1 > 0 then
                      begin
                        z:=2*max[bwidth]-1;
                        z:=2+(max[nodes]*z-2
*min[ncov]*max[bwidth]) div z1;
                        if z < max[girth] then pushmax(girth);
                      end;
                 end;
             end;
          end
         else
           if (max[nodes] < infinity) and (max[bwidth]
< infinity) then
              begin
```

```
                z1:=min[girth]-2;
                z:=2*min[ncov]*z1-max[nodes]*(z1-1);
                if 2*(max[nodes]-min[ncov])*max[bwidth] < z then
                  begin
                    z:=1;
                    pushmin(forest);
                  end;
               end;
    end;
end;

procedure r423;
(*********************************)
(*                               *)
(*  ncov <= p-maxdeg/(chr-1)     *)
(*                               *)
(*********************************)
begin
  if activerule[423] then
    begin
      rule:='423/ ';
      if (max[nodes] < infinity) and (max[chr] < infinity) then
         begin
           z1:=max[chr]-1;
           z:=(max[nodes]*z1-min[maxdeg]) div z1;
           if z < max[ncov] then pushmax(ncov);
           z:=(max[nodes]-min[ncov])*z1;
           if z < max[maxdeg] then pushmax(maxdeg);
         end;
       if max[nodes] < infinity then
         begin
           z:=(min[maxdeg]-1) div (max[nodes]-min[ncov])+2;
           if z > min[chr] then pushmin(chr);
         end;
       if max[chr] < infinity then
           begin
             z:=min[ncov]+(min[maxdeg]-1) div (max[chr]-1)+1;
             if z > min[nodes] then pushmin(nodes);
           end;
    end;
end;

procedure r424;
(*************************************************************
***)
(*
*)
(* if connected, clique = 2, and not odd cycle or even path, then
*)
(*        ncov <= p(x-1)/x-(p**2)/(p+2e)
*)
(*          where x = maxdeg**2+maxdeg
```

```
*)
(*
*)
(*************************************************************
***)
begin
  if (activerule[424]) and (max[connct] = 1) and (min[clique] =
2)
     and (max[cycle] = 0) then
    begin
      rule:='424/ ';
      if (min[connct] = 1) and (max[clique] = 2) and
         (max[maxdeg] > 2) then
           begin
             if (max[nodes] < infinity) and (max[maxdeg]
< infinity) then
                 begin
                   z1:=max[maxdeg];
                   z1:=z1*z1+z1;
                   z:=max[nodes];
                   rz:=z1*(z-min[ncov])/z-1;
                   if rz > 0 then
                     begin
                       z:=round((z+min[ncov]*z1)/(2*rz)+hf);
                       if z > min[edges] then pushmin(edges);
                     end;
                   if max[edges] < infinity then
                     begin
                       z:=max[nodes];
                              rz:=z1-1;
                              rz:=rz/z1-z/(z+2*max[edges]);
                       z:=trunk(z*rz);
                       if z < max[ncov] then pushmax(ncov);
                     end;
                  end;
              if (max[nodes] < infinity) and (max[edges]
< infinity) then
                 begin
                   z:=max[nodes];
                   rz:=(4*z-min[ncov])*(z+2*max[edges])+z*2
*max[edges];
                   rz:=rz/(2*z*max[edges]-min[ncov]*(z+2
*max[edges]));
                   z:=min[nodes];
                   rhb:=(4*z-min[ncov])*(z+2*max[edges])+z*2
*max[edges];
                   rhb:=rhb/(2*z*max[edges]-min[ncov]*(z+2
*max[edges]));
                   if rhb < rz then rz:=rhb;
                   if rz < infinity then
                     begin
                       z:=round((-1+sqrt(rz))/2+hf);
```

```
                    if z > min[maxdeg] then pushmin(maxdeg);
                  end;
                end;
            end;
    end;
end;

procedure r425;
(*************************************************)
(*                                               *)
(*  if plnar then mindeg <= p-ncov+2           *)
(*                                               *)
(*************************************************)
begin
  if (activerule[425]) and (max[plnar] = 1) then
    begin
      rule:='425/ ';
      z:=min[mindeg]+min[ncov]-2;
      if min[plnar] = 1 then
         begin
           if z > min[nodes] then pushmin(nodes);
           z1:=max[nodes];
           if z1 < infinity then
             begin
               z:=z1-min[ncov]+2;
               if z < max[mindeg] then pushmax(mindeg);
               z:=z1-min[mindeg]+2;
               if z < max[ncov] then pushmax(ncov);
             end;
         end
       else
         if z > max[nodes] then
           begin
             z:=0;
             pushmax(plnar);
           end;
    end;
end;

procedure r426;
(*************************************************)
(*                                               *)
(*  edges <= MAX{(p-ecov)(2p-2ecov+1),           *)
(*                   (p-ecov)(p+ecov-1)/2}       *)
(*                                               *)
(*************************************************)
begin
  if activerule[426] then
    begin
      rule:='426/ ';
      z:=max[nodes];
      z1:=min[ecov];
```

```
      if z < infinity then
        begin
          if 3*z+3 >= 5*z1 then
              begin
                z1:=z-z1;
                z:=z1*(2*z1+1);
               end
            else z:=((z-z1)*(z+z1-1)) div 2;
          if z < max[edges] then pushmax(edges);
          z:=max[nodes];
          z1:=min[edges];
          z:=trunk(0.5+sqrt(z*(z-1)-2*z1+0.25));
          z1:=trunk(max[nodes]+(1-sqrt(8*z1+1))/4);
          if z1 > z then z:=z1;
          if z < max[ecov] then pushmax(ecov);
         end;
        z:=min[edges];
        z1:=min[ecov];
        z:=round(z1-0.25+sqrt(8*z+1)/4+hf);
        rz:=2*min[edges]+z1*(z1-1)+1.0/4.0;
        if rz >= 0 then
           begin
             z1:=round(0.5+sqrt(rz)+hf);
             if z1 < z then z:=z1;
             if z > min[nodes] then pushmin(nodes);
           end;
    end;
end;

procedure r427;
(*********************************************)
(*                                           *)
(* if reg, and econn >= mindeg-2 >= 1 then   *)
(*                                           *)
(*             (p+2*t1)/2       if p is even *)
(*     ecov <=                               *)
(*             (p+max{2*t2-1,1} if p is odd  *)
(*                                           *)
(*         where x = mindeg*FL((mindeg+3)/2)-1 *)
(*              t1 = FL((p+1)/2x)            *)
(*              t2 = FL((p+1+x)/2x)          *)
(*                                           *)
(*********************************************)
var x:longint;
begin
  if (activerule[427]) and (max[reg] = 1) and (max[mindeg] >= 3)
     and (max[nodes] < infinity)  then
    begin
      rule:='427/ ';
      z:=min[mindeg];
      z1:=max[nodes];
      x:=z*((z+3) div 2)-1;
```

```
      if odd(z1) then
          begin
            z:=2*((z1+1+x) div (2*x))-1;
            if z < 1 then z:=1;
          end
         else z:=2*((z1+1) div (2*x));
       z:=(z1+z) div 2;
       if (min[reg] = 1) and (min[mindeg] >= 3) and
          (min[econn] >= max[mindeg]-2) then
             begin
               if z < max[ecov] then pushmax(ecov);
             end
           else
             if (min[ecov] > z) and (min[reg] = 1) and
(min[mindeg] >= 3) then
                begin
                  z:=max[mindeg]-3;
                  if z < max[econn] then pushmax(econn);
                  z:=min[econn]+3;
                  if z > min[mindeg] then pushmin(mindeg);
                end
              else
                if (min[econn] >= max[mindeg]-2) and (min[ecov] >
z)
                   and (min[reg] = 1) then
                   begin
                     z:=2;
                     pushmax(mindeg);
                   end
                 else
                   if (min[mindeg] >= 3) and (min[econn] >=
max[mindeg]-2)
                      and (min[ecov] > z) then
                       begin
                         z:=0;
                         pushmax(reg);
                       end;
    end;
end;

procedure r428;
(*********************************************************)
(*                                                       *)
(*  if cubic then ecov <= p/2+Fl((p+3)/18+Fl((Ncomp+4)/6) *)
(*                                                       *)
(*********************************************************)
begin
  if (activerule[428]) and (max[mindeg] >= 3) and (min[maxdeg] <=
3)
     and (max[nodes] < infinity) and ( max[ncomp] < infinity)
then
    begin
```

```
      rule:='428/ ';
      z:=max[nodes];
      z:=z div 2+(z+3) div 18+(max[ncomp]+4) div 6;
      if (min[mindeg] = max[maxdeg]) then
          begin
            if z < max[ecov] then pushmax(ecov);
            z:=(18*(min[ecov]-((max[ncomp]+4) div 6))+6) div 10;
            if z > min[nodes] then pushmin(nodes);
            z:=6*(min[ecov]-(max[nodes] div 2)-((max[nodes]+3)
div 18))-4;
            if z > min[ncomp] then pushmin(ncomp);
          end
        else
          if min[ecov] > z then
              if max[maxdeg] = 3 then
                    begin
                      z:=2;
                      pushmax(mindeg);
                    end
                  else
                    if min[mindeg] = 3 then
                        begin
                          z:=4;
                          pushmin(maxdeg);
                        end;
    end;
end;

procedure r429;
(*********************************************)
(*                                           *)
(*  if clique = 2 and maxdeg <= 4 then       *)
(*         edges >= 13*ncov-7*nodes          *)
(*                                           *)
(*********************************************)
begin
  if (activerule[429]) and (min[clique] <= 2) and (min[maxdeg] <=
4) then
    begin
      rule:='429/ ';
      if (max[clique] = 2) and (max[maxdeg] <= 4) then
        begin
          if max[nodes] < infinity then
             begin
               z:=13*min[ncov]-7*max[nodes];
               if z > min[edges] then pushmin(edges);
               if max[edges] < infinity then
                  begin
                    z:=(max[edges]+7*max[nodes]) div 13;
                    if z < max[ncov] then pushmax(ncov);
                  end;
             end;
```

```
           if max[edges] < infinity then
                begin
                  z:=13*min[ncov]-max[edges]+6;
                  if z > 14 then
                    begin
                      z:=z div 7;
                      if z > min[nodes] then pushmin(nodes);
                    end;
                end;
        end
      else
        if (max[nodes] < infinity) and (max[edges] < infinity)
then
             begin
               z:=13*min[ncov]-7*max[nodes];
               if (max[edges] < z) and (max[clique] = 2) then
                  begin
                    z:=5;
                    pushmin(maxdeg);
                  end
               else
                 if (max[maxdeg] <= 4) and (max[edges] < z) then
                    begin
                      z:=3;
                      pushmin(clique);
                    end;
             end;
    end;
end;

procedure r430;
(*********************************************)
(*                                           *)
(* if nconn >= 2 and ncov <= p-2 then        *)
(*        circ >= (4p-2ncov-4)/(p-ncov)      *)
(*                                           *)
(*********************************************)
begin
  if (activerule[430]) and (max[nconn] >= 2) then
    begin
      rule:='430/ ';
      if (min[nconn] >= 2) and (max[ncov] <= min[nodes]-2) then
        begin
          z:=max[nodes];
          if z < infinity then
             begin
               z:=(4*z-2*min[ncov]-5) div (z-min[ncov])+1;
               if z > min[circ] then pushmin(circ);
               z:=max[circ];
               if z < infinity then
                  begin
                    z:=(max[nodes]*(z-4)+4) div (z-2);
```

```
                    if z < max[ncov] then pushmax(ncov);
                  end;
              end;
             z:=max[circ];
             if (z < infinity) and (z > 4) then
               begin
                 z:=((z-2)*min[ncov]-5) div (z-4)+1;
                 if z > min[nodes] then pushmin(nodes);
               end;
           end
          else
            if (max[nodes] < infinity) and (max[circ] < infinity)
then
              begin
                z:=max[nodes];
                z:=(4*z-2*min[ncov]-5) div (z-min[ncov])+1;
                if (max[circ] < z) and (min[nconn] >= 2) then
                  begin
                    z:=min[nodes]-1;
                    if z > min[ncov] then pushmin(ncov);
                    z:= max[ncov]+1;
                    if z < max[nodes] then pushmax(nodes);
                  end
                 else
                   if (max[ncov] <= min[nodes]-2) and ( max[circ]
< z) then
                      begin
                        z:=1;
                        pushmax(nconn);
                      end;
               end;
    end;
end;

procedure r431;
(*************************************************)
(*                                               *)
(* if girth >= 6 then ncov <= (p*maxdeg**2)/x    *)
(*           where x = maxdeg**2+2*maxdeg-1      *)
(*                                               *)
(*************************************************)
begin
  if (activerule[431]) and (max[girth] >= 6) then
    begin
      rule:='431/ ';
      if min[girth] >= 6 then
        begin
          z:=max[maxdeg];
          z1:=max[nodes];
          if z < infinity then
               begin
                 z:=(min[ncov]*(2*z-1)-1) div (z*z)+min[ncov]+1;
```

```
                if z > min[nodes] then pushmin(nodes);
                if z1 < infinity then
                  begin
                    rz:=max[maxdeg];
                    z:=trunk(max[nodes]/((rz+2)/rz-1/(rz*rz)));
                    if z < max[ncov] then pushmax(ncov);
                  end;
               end;
            z:=min[ncov];
            z1:=max[nodes];
            if z1 <= 2*z then
               begin
                 z:=round((z+sqrt(z*(2*z-z1)))/(z1-z)+hf);
                 if z > min[maxdeg] then pushmin(maxdeg);
               end;
          end
        else
          if max[maxdeg] < infinity then
             begin
               z:=max[maxdeg];
               z:=(min[ncov]*(2*z-1)-1) div (z*z)+min[ncov]+1;
               if max[nodes] < z then
                  begin
                    z:=5;
                    pushmax(girth);
                  end;
               end;
    end;
end;

procedure r432;
(*************************************************)
(*                                               *)
(*  if cubic then ncov <= 33p/52, if girth >= 6 *)
(*                    <= 33p/53, if girth >= 8 *)
(*                                               *)
(*************************************************)
begin
  if (activerule[432]) and (max[mindeg] >= 3) and (min[maxdeg] <=
3)
     and (max[girth] >= 6) then
    begin
      rule:='432/ ';
      if min[mindeg] = max[maxdeg] then
         begin
           if min[girth] >= 8 then
             begin
               if max[nodes] < infinity then
                  begin
                    z:=(33*max[nodes]) div 53;
                    if z < max[ncov] then pushmax(ncov);
                  end;
```

```
              z:=(53*min[ncov]+32) div 33;
              if z > min[nodes] then pushmin(nodes);
            end
          else
            if min[girth] >= 6 then
                 begin
                   if max[nodes] < infinity then
                      begin
                        z:=(33*max[nodes]) div 52;
                        if z < max[ncov] then pushmax(ncov);
                      end;
                   z:=(52*min[ncov]+32) div 33;
                   if z > min[nodes] then pushmin(nodes);
                 end
              else
                if max[nodes] < infinity then
                    if min[ncov] > (33*max[nodes]) div 52 then
                         begin
                           z:=5;
                           pushmax(girth);
                         end
                       else
                         if min[ncov] > (33*max[nodes]) div 53
then
                              begin
                                z:=7;
                                if z < max[girth] then
pushmax(girth);
                              end;
          end
        else
          if max[nodes] < infinity then
            if ((min[girth] >= 6) and (min[ncov] > (33
*max[nodes]) div 52)) or
               ((min[girth] >= 8) and (min[ncov] > (33
*max[nodes]) div 53)) then
                   if max[maxdeg] = 3 then
                         begin
                           z:=2;
                           pushmax(mindeg);
                         end
                       else
                         if min[mindeg] = 3 then
                            begin
                              z:=4;
                              pushmin(maxdeg);
                            end;
    end;
end;

procedure r433;
(*********************************************)
```

```
(*                                                  *)
(* if reg. and ecov > p/2 then echr = maxdeg+1  *)
(*                                                  *)
(**************************************************)
begin
  if (activerule[433]) and (max[reg] = 1) then
    begin
      rule:='433/ ';
      if (min[reg] = 1) and (2*min[ecov] > max[nodes]) then
           begin
             z:=min[maxdeg]+1;
             if z > min[echr] then pushmin(echr);
             z:=max[echr]-1;
             if z > max[maxdeg] then pushmax(maxdeg);
           end
         else
           if (max[echr] = max[maxdeg]) and (min[reg] = 1) then
               begin
                 if max[nodes] < infinity then
                      begin
                        z:=max[nodes] div 2;
                        if z < max[ecov] then pushmax(ecov);
                      end;
                 z:=2*min[ecov];
                 if z > min[nodes] then pushmin(nodes);
               end
              else
                if (2*min[ecov] > max[nodes]) and
                   (max[echr] = max[maxdeg]) then
                     begin
                       z:=0;
                       pushmax(reg);
                     end;
    end;
end;

procedure r434;
(**************************************************)
(*                                                *)
(*  if reg. then ecov <= p(maxdeg+2)/(2maxdeg+2) *)
(*                                                *)
(**************************************************)
begin
  if (activerule[434]) and (max[reg] = 1) then
    begin
      rule:='434/ ';
      if min[reg] = 1 then
        begin
          z:=min[maxdeg];
          z:=(2*min[ecov]*(z+1)-1) div (z+2)+1;
          if z > min[nodes] then pushmin(nodes);
          if max[nodes] < infinity then
```

```
            begin
              z:=min[maxdeg];
              z:=(max[nodes]*(z+2)) div (2*z+2);
              if z < max[ecov] then pushmax(ecov);
              z:=2*min[ecov]-max[nodes];
              if z > 0 then
                 begin
                   z:=(max[nodes]-z) div z;
                   if z < max[maxdeg] then pushmax(maxdeg);
                 end;
            end;
          end
        else
          if max[nodes] < infinity then
             if min[ecov]*(2*min[maxdeg]+2) >
max[nodes]*(min[maxdeg]+2) then
                begin
                  z:=0;
                  pushmax(reg);
                end;
    end;
end;

procedure r435;
(*************************************************)
(*                                               *)
(* if reg. and p = 2*maxdeg+1 then ncov >= p-nconn *)
(*                                               *)
(*************************************************)
begin
  if (activerule[435]) and (max[reg] = 1) and (min[nodes] =
max[nodes])
     and (min[maxdeg] = max[maxdeg]) and (min[nodes] = 2
*max[maxdeg]+1) then
    begin
      rule:='435/ ';
      if min[reg] = 1 then
         begin
           if max[nconn] < infinity then
              begin
                z:=min[nodes]-max[nconn];
                if z > min[ncov] then pushmin(ncov);
              end;
           if max[ncov] < infinity then
               begin
                 z:=min[nodes]-max[ncov];
                 if z > min[nconn] then pushmin(nconn);
               end;
         end
        else
          if max[ncov] < min[nodes]-max[nconn] then
                  begin
```

```
                    z:=0;
                    pushmax(reg);
                  end;
    end;
end;


procedure r436;
(*****************************************************)
(*                                                   *)
(* if t=(gi-2) div 2 >= 1 and mindeg=2 then          *)
(*   ncov <= p-maxdeg*((t+1) div 2))  {+1 if t even} *)
(*                                                   *)
(*****************************************************)
var t,z2:longint;
begin
  if (activerule[436]) and (max[mindeg] >= 2)
      and (min[mindeg] <= 2) and (min[girth] >= 4)then
    begin
      rule:='436/ ';
      t:=(min[girth]-2) div 2;
      if odd(t) then z1:=0
                else z1:=1;
      z2:=min[maxdeg]*((t+1-z1) div 2)+z1;
      z:=max[nodes]-z2;
      if min[mindeg]=max[mindeg] then
           begin
             if max[nodes] < infinity then
               begin
                 if z < max[ncov] then pushmax(ncov);
                 z:=(2*(max[nodes]-min[ncov]-z1)) div (t+1-z1);
                 if z < max[maxdeg] then pushmax(maxdeg);
                 z:=4*((max[nodes]-min[ncov]-z1) div
min[maxdeg])+2*z1+1;
                 if z < max[girth] then pushmax(girth);
               end;
              z:=min[ncov]+z2;
              if z > min[nodes] then pushmin(nodes);
            end
          else
            if min[ncov] > z then
               begin
                 z:=1;
                 pushmax(mindeg);
               end;
    end;
end;

procedure r437;
(*********************************************************)
(*                                                       *)
(* if girth >= 4 and mindeg >= 3 then                    *)
```

```
(*     ncov <= p-(maxdeg*(mindeg-1)*s+k)                    *)
(*                                                          *)
(*      where t = (girth-2) div 2 and                       *)
(*         s = ((mindeg-1)**t-1) div (mindeg*(mindeg-2))    *)
(*         and k = 1 if t even, else = maxdeg/mindeg        *)
(*                                                          *)
(************************************************************)
var t,z2:longint;
begin
  if (activerule[437]) and (min[girth] >= 4) and (min[girth]
< infinity) then
    begin
      rule:='437/ ';
      t:=(min[girth]-2) div 2;
      if (max[nodes] < infinity) and (t >= 2) then
          begin
            if t = 2 then z:=1+(max[nodes]-min[ncov]) div
min[maxdeg]
              else z:=1+trunk(root((max[nodes]-
min[ncov])/min[maxdeg],t-1));
           if z < 2 then z:=2;
           if z < max[mindeg] then pushmax(mindeg);
          end;
      z:=min[mindeg];
      if z >= 3 then
        begin
          if t > 1 then
           begin
             power(z-1,t,z1);
             z1:=(z-1)*((z1-1) div (z*(z-2)));
           end;
          if odd(t) then
            begin
              z2:=min[maxdeg];
              if t > 1 then z2:=z2*z1+1+(min[maxdeg]-1) div z;
              z:=z2+min[ncov];
              if z > min[nodes] then pushmin(nodes);
              if max[nodes] < infinity then
                 begin
                   z:=max[nodes]-z2;
                   if z < max[ncov] then pushmax(ncov);
                   z:=max[nodes]-min[ncov];
                   if t > 1 then  z:=(min[mindeg]*z) div (z1
*min[mindeg]+1);
                   if z < max[maxdeg] then pushmax(maxdeg);
                 end;
            end
           else
            begin
              z2:=min[maxdeg]*z1+1;
              z:=min[ncov]+z2;
              if z > min[nodes] then pushmin(nodes);
```

```
               if max[nodes] < infinity then
                  begin
                    z:=max[nodes]-z2;
                    if z < max[ncov] then pushmax(ncov);
                    z:=(max[nodes]-min[ncov]-1) div z1;
                    if z < max[maxdeg] then pushmax(maxdeg);
                  end;
              end;
         end;
    end;
end;


procedure r438;
(*****************************)
(*                           *)
(*  nind >= (2p-e+eind)/4    *)
(*                           *)
(*****************************)
begin
  if activerule[438] then
    begin
      rule:='438/ ';
      if max[edges] < infinity then
         begin
           z:=(2*min[nodes]-max[edges]+min[eind]+3) div 4;
           if z > min[nind] then pushmin(nind);
           if max[nind] < infinity then
              begin
                z1:=4*max[nind]+max[edges];
                z:=(z1-min[eind]) div 2;
                if z < max[nodes] then pushmax(nodes);
                z:=z1-2*min[nodes];
                if z < max[eind] then pushmax(eind);
              end;
          end;
        if max[nind] < infinity then
            begin
              z:=2*min[nodes]+min[eind]-4*max[nind];
              if z > min[edges] then pushmin(edges);
            end;
    end;
end;

procedure r439;
(*************************)
(*                       *)
(*  ncov <= (p+e+ecov)/4 *)
(*                       *)
(*************************)
begin
  if activerule[439] then
```

```
    begin
      rule:='439/ ';
      if (max[nodes] < infinity) and (max[ecov] < infinity) then
        begin
          z:=4*min[ncov]-max[nodes]-max[ecov];
          if z > min[edges] then pushmin(edges);
          if max[ecov] < infinity then
            begin
              z:=(max[nodes]+max[edges]+max[ecov]) div 4;
              if z < max[ncov] then pushmax(ncov);
            end;
         end;
      if max[edges] < infinity then
         begin
           z1:=4*min[ncov]-max[edges];
           if max[nodes] < infinity then
              begin
                z:=z1-max[nodes];
                if z > min[ecov] then pushmin(ecov);
              end;
           if max[ecov] < infinity then
              begin
                z:=z1-max[ecov];
                if z > min[nodes] then pushmin(nodes);
              end;
          end;
    end;
end;

procedure r440;
(*******************************)
(*                             *)
(*    nind >= (3p-e-ecov)/4    *)
(*                             *)
(*******************************)
begin
  if activerule[440] then
    begin
      rule:='440/ ';
      if (max[edges] < infinity) and (max[ecov] < infinity) then
          begin
            z:=(3*min[nodes]-max[edges]-max[ecov]+3) div 4;
            if z > min[nind] then pushmin(nind);
            if max[nind] < infinity then
                begin
                  z:=(4*max[nind]+max[edges]+max[ecov]) div 3;
                  if z < max[nodes] then pushmax(nodes);
                end;
           end;
        if max[nind] < infinity then
          begin
            z1:=3*min[nodes]-4*max[nind];
```

```
            if max[edges] < infinity then
               begin
                 z:=3*min[nodes]-4*max[nind]-max[edges];
                 if z > min[ecov] then pushmin(ecov);
               end;
            if max[ecov] < infinity then
               begin
                 z:=3*min[nodes]-4*max[nind]-max[ecov];
                 if z > min[edges] then pushmin(edges);
               end;
          end;
    end;
end;

procedure r441;
(*********************************************)
(*                                           *)
(*  if maxdeg >= 6 and clique < maxdeg then  *)
(*    ncov <= (p(maxdeg-1)-1)/maxdeg         *)
(*                                           *)
(*********************************************)
begin
  if (activerule[441]) and (max[maxdeg] >= 6) then
    begin
      rule:='441/ ';
      if (min[maxdeg] >= 6) and (max[clique] < min[maxdeg]) then
        begin
          z:=max[nodes];
          if z < infinity then
            begin
              z:=z div (z-min[ncov])+1;
              if z > min[maxdeg] then pushmin(maxdeg);
              z:=max[maxdeg];
              if z < infinity then
                begin
                  z:=(max[nodes]*(z-1)-1) div z;
                  if z < max[ncov] then pushmax(ncov);
                end;
            end;
          z:=max[maxdeg];
          if z < infinity then
              begin
                z:=(z*min[ncov]) div (z-1)+1;
                if z > min[nodes] then pushmin(nodes);
              end;
        end
      else
        if (max[nodes] < infinity) and (max[maxdeg] < infinity)
then
           begin
             z:=(max[nodes]*(max[maxdeg]-1)-1) div max[maxdeg];
             if min[ncov] > z then
```

```
                if min[maxdeg] >= 6 then
                    begin
                      z:=min[maxdeg];
                      if z > min[clique] then pushmin(clique);
                      z:=max[clique];
                      if z < max[maxdeg] then pushmax(maxdeg);
                    end
                  else
                    if max[clique] < min[maxdeg] then
                       begin
                         z:=5;
                         pushmax(maxdeg);
                       end;
            end;
    end;
end;


procedure r442;
(*********************************************)
(*                                           *)
(* if clique = 2 then ncov <=                *)
(*    p-p(2e/p*ln(2e/p)-2e/p+1)/(2e/p-1)**2  *)
(*                                           *)
(*       where ln is the natural log.        *)
(*       ln(x) > .693148*log2(x)             *)
(*                                           *)
(*********************************************)
var dbar:real;
begin
  if (activerule[442]) and (max[edges] < infinity)
     and (min[clique] = 2) and (max[nodes] < infinity) then
    begin
      rule:='442/ ';
      if 2*max[edges] = min[nodes] then z:=min[nodes] div 2
        else
          begin
            dbar:=2*max[edges]/min[nodes];
            rz:=dbar*0.693148*log2(dbar)-dbar+1;
                rz:=rz/((dbar-1)*(dbar-1));
                z:=trunk(max[nodes]*(1-rz));
          end;
      if max[clique] = 2 then
        begin
          if z < max[ncov] then pushmax(ncov);
             z:=min[edges];
          if 2*z > min[nodes] then
                    begin
                        dbar:=(2*z)/min[nodes];
                  rz:=(dbar*0.693148*log2(dbar)-dbar+
1)/((dbar-1)*(dbar-1));
                        rz:=trunk(max[nodes]*(1-rz));
```

```
                        while (rz < min[ncov]) and (z <=
max[edges]) do
                                begin
                                    z:=z+1;
                                    dbar:=2*z/min[nodes];
                                    rz:=(dbar*0.693148
*log2(dbar)-dbar+1)/((dbar-1)*(dbar-1));
                                  rz:=trunk(max[nodes]*(1-rz));
                                  end;
                           if z > min[edges] then pushmin(edges);
                     end;
                  z:=max[nodes];
                  if 2*max[edges] > z then
                     begin
                         dbar:=(2*max[edges])/z;
                         rz:=(dbar*0.693148*log2(dbar)-dbar+
1)/((dbar-1)*(dbar-1));
                         rz:=trunk(z*(1-rz));
                         while (rz < min[ncov]) and (z >=
min[nodes]) do
                                begin
                                    z:=z-1;
                                    dbar:=2*max[edges]/z;
                                    rz:=(dbar*0.693148
*log2(dbar)-dbar+1)/((dbar-1)*(dbar-1));
                                  rz:=trunk(z*(1-rz));
                                  end;
                           if z < max[nodes] then pushmax(nodes);
                    end;
        end
       else
         if min[ncov] > z then
            begin
              z:=3;
              pushmin(clique);
            end;
    end;
end;


procedure r443;
(*************************************************)
(*                                               *)
(*  Bw <= p-(mindeg+1)(Nc-1)-1-FL((p-ncov-Nc+1)/2) *)
(*                                               *)
(*************************************************)
begin
  if activerule[443] then
    begin
      rule:='443/ ';
      if max[nodes] < infinity then
        begin
```

```
          if max[ecov] < infinity then
            begin
              z:=max[nodes]-(min[mindeg]+1)*(min[ncomp]-1)-1;
              z:=z-((max[nodes]-max[ncov]-min[ncomp]+1) div 2);
              if z < max[bwidth] then pushmax(bwidth);
              z:=max[nodes]-max[ncov]-min[ncomp]+1;
              z:=max[nodes]-min[bwidth]-1-(z div 2);
              if min[ncomp] > 1 then
                begin
                  z:=z div (min[ncomp]-1)-1;
                  if z < max[mindeg] then pushmax(mindeg);
                end;
              z:=2*min[mindeg]+1;
              z:=(max[nodes]-2*min[bwidth]+max[ncov]-1) div z+1;
              if z < max[ncomp] then pushmax(ncomp);
             end;
          z:=2*min[bwidth]+(min[ncomp]-1)*(2*min[mindeg]+1)+1-
max[nodes];
          if z > min[ncov] then pushmin(ncov);
         end;
        if max[ncov] < infinity then
            begin
              z:=2*min[bwidth]+(2*min[mindeg]+1)*(min[ncomp]-1)-
max[ncov]+1;
              if z > min[nodes] then pushmin(nodes);
            end;
    end;
end;

procedure r444;
(*********************************************)
(*                                           *)
(*  regular, mindeg > nconn then ecov <=(p+t)/2 *)
(*     where t is same parity as p and where    *)
(*      p <= (t+3)*(2*CL(mindeg/2)+1) + x       *)
(*        and x = 0, mindeg, 2*mindeg-2         *)
(*         if nconn = 1, 2, and >=3, respect.   *)
(*                                           *)
(*********************************************)
begin
  if (activerule[444]) and (min[nconn] > 0) and (max[reg] = 1)
then
    begin
      rule:='444/ ';
      z:=0;
      if min[nconn] >= 3 then z:= 2*min[mindeg]-2
                else if min[nconn] = 2 then z:=min[mindeg];
      z1:=2*((min[mindeg]+1) div 2)+1;
      z1:=(max[nodes]-z-1) div z1-2;
      if ((odd (max[nodes])) and (not(odd(z1)))) or
         ((not(odd(max[nodes]))) and (odd(z1))) then z1:=z1+1;
      z:= (max[nodes]+z1) div 2;
```

```
      if (min[reg] = 1) and (min[mindeg] > max[nconn]) then
          begin
            if z < max[ecov] then pushmax(ecov);
            z:=2*min[ecov]-z1;
            if z > min[nodes] then pushmin(nodes);
          end
        else
          if (min[ecov] > z) and (min[reg] = 1) then
              begin
                z:=max[nconn];
                if z < max[mindeg] then pushmax(mindeg);
                z:=min[mindeg];
                if z > min[nconn] then pushmin(nconn);
              end
           else
             if (min[ecov] > z) and (min[mindeg] > max[nconn])
then
                 begin
                   z:=0;
                   pushmax(reg);
                 end;
    end;
end;

procedure r445;
(************************************************)
(*                                              *)
(*  e >= m(p-chr)+chr(chr-1)/2-(p-ncov-1)m(m+1)/2 *)
(*     where  m = (p-chr) div (p-ncov-1)        *)
(*                                              *)
(************************************************)
begin
  if (activerule[445]) and (max[nodes] < infinity) then
    begin
      rule:='445/ ';
      z:=max[nodes]-min[chr];
      z1:=max[nodes]-min[ncov]-1;
      if z1 > 0 then
        begin
          z1:=z div (max[nodes]-min[ncov]-1);
          z:=z1*z+(min[chr]*(min[chr]-1)) div 2;
          z:=z-((max[nodes]-min[ncov]-1)*z1*(z1+1) div 2);
          if z > min[edges] then pushmin(edges);
        end;
    end;
end;

procedure r446;
(************************************************)
(*                                              *)
(* if clique <= 2 and maxdeg <= 3 then          *)
(*       edges >= 14ncov-15p div 2              *)
```

```
(*                                              *)
(************************************************)
begin
  if (activerule[446]) and (min[clique] <= 2) and (min[maxdeg] <=
3) then
    begin
      rule:='446/ ';
      z:=max[edges];
      if (max[clique] = 2) and (max[maxdeg] <= 3) then
          begin
            z:=max[edges];
            if z < infinity then
                 begin
                   z:=28*min[ncov]-2*z+14;
                   if z > 30 then
                        begin
                          z:= z div 15;
                          if z > min[nodes] then pushmin(nodes);
                        end;
                    z:=max[nodes];
                    if z < infinity then
                        begin
                          z:=((15*z) div 2+max[edges]) div 14;
                          if z < max[ncov] then pushmax(ncov);
                        end;
                 end;
            z:=max[nodes];
            if z < infinity then
                  begin
                    z:=14*min[ncov]-(15*z) div 2;
                    if z > min[edges] then pushmin(edges);
                  end;
          end
         else
           if (z < infinity) and (max[nodes] < infinity) then
             begin
               z:=14*min[ncov]-(15*max[nodes]) div 2;
               if max[edges] < z then
                  if max[clique] = 2 then
                     begin
                       z:=4;
                       pushmin(maxdeg);
                     end
                   else
                     begin
                       z:=3;
                       pushmin(clique);
                     end;
             end;
    end;
end;
```

```
procedure r447;
(*************************************************)
(*                                               *)
(*  if clique = 2 and maxdeg <= 2 then           *)
(*        e >= 15*ncov-8*p                       *)
(*                                               *)
(*************************************************)
begin
  if (activerule[447]) and (min[clique] = 2) and (min[maxdeg] <=
2) then
    begin
      rule:='447/ ';
      if (max[clique] = 2) and (max[maxdeg] <= 2) then
        begin
          if max[nodes] < infinity then
              begin
                z:=15*min[ncov]-8*max[nodes];
                if z > min[edges] then pushmin(edges);
                if max[edges] < infinity then
                   begin
                     z:=(max[edges]+8*max[nodes]) div 15;
                     if z < max[ncov] then pushmax(ncov);
                   end;
              end;
           if max[edges] < infinity then
              begin
                z:=15*min[ncov]-max[edges]+7;
                if z > 16 then
                  begin
                    z:=z div 8;
                    if z > min[nodes] then pushmin(nodes);
                  end;
              end;
        end
      else
        if (max[nodes] < infinity) and (max[edges] < infinity)
then
          begin
            z:=15*min[ncov]-8*max[nodes];
            if max[edges] < z then
              if max[clique] = 2 then
                 begin
                   z:=3;
                   pushmin(maxdeg);
                 end
                else
                  begin
                    z:=3;
                    pushmin(clique);
                  end;
          end;
    end;
```

```
end;

procedure r448;
(*************************************************)
(*                                               *)
(* e <= ((p-nccov)*(nccov+maxdeg-1)+mindeg)/2    *)
(*                                               *)
(*************************************************)
var z2:longint;
begin
  if activerule[448] then
    begin
      rule:='448/ ';
      if (max[nodes] < infinity) and (max[maxdeg] < infinity)
then
        begin
          z1:=max[nccov];
          z2:=(max[nodes]-z1)*(z1+max[maxdeg]-1);
          if z2 < (max[nodes]-
min[nccov])*(min[nccov]+max[maxdeg]-1) then
                begin
                  z1:=min[nccov];
                  z2:=(max[nodes]-z1)*(z1+max[maxdeg]-1);
                end;
          if max[mindeg] < infinity then
             begin
               z:=(z2+max[mindeg]) div 2;
               if z < max[edges] then pushmax(edges);
               z:=2*min[edges]-z2;
               if z > min[mindeg] then pushmin(mindeg);
               z2:=2*min[edges]-max[mindeg]-1;
               if (z2 > 0) and (max[nccov] = min[nccov]) then
                 begin
                   z:=z2 div (max[nodes]-z1)-z1+2;
                   if z > min[maxdeg] then pushmin(maxdeg);
                   z:=z2 div (z1+max[maxdeg]-1)+z1+1;
                   if z > min[nodes] then pushmin(nodes);
                 end;
             end;
        end;
    end;
end;

procedure r449;
(*************************************************)
(*                                               *)
(* if nind = 2 and p <= mindeg+5 then            *)
(*     e <= p(p-13)/2+13*clique                  *)
(*                                               *)
(*************************************************)
begin
  if (activerule[449]) and (min[nind] <= 2) and
```

```
     (max[nodes] >= 13) then
    begin
      rule:='449/ ';
      if max[clique] < infinity then
              begin
                z:=8*min[edges]-104*max[clique]+169;
                if z >= 0 then
                    begin
                      z:=trunk((13+sqrt(z))/2);
                      if z > min[mindeg]+6 then z:=min[mindeg]+6;
                    end;
               end
             else z:=0;
      if min[nind] = max[nind] then
        begin
          if (max[clique] < infinity) and (z > min[nodes]) then
pushmin(nodes);
          if max[nodes] <= min[mindeg]+5 then
               begin
                 z:=max[nodes];
                 z:=(min[edges]-(z*(z-13)) div 2-1) div 13+1;
                 if z > min[clique] then pushmin(clique);
                 if max[clique] < infinity then
                   begin
                     z:=max[nodes];
                     z:=(z*(z-13)) div 2+13*max[clique];
                     if z < max[edges] then pushmax(edges);
                   end;
                end
              else
                if max[nodes] < z then
                    begin
                      z:=max[nodes]-6;
                      if z < max[mindeg] then pushmax(mindeg);
                    end;
        end
       else
        if max[nodes] < z then
              if min[nind] = 2 then
                 begin
                    z:=3;
                    pushmin(nind);
                 end
               else
                 if max[nind] = 2 then
                     begin
                       z:=1;
                       pushmax(nind);
                     end;
    end;
end;
```

```
procedure r450;
(*************************************************)
(*                                               *)
(* if nind < nccov = p-mindeg-1 then             *)
(*    p <= 2*mindeg+3  ( -1, if p >= mindeg+10) *)
(*                                               *)
(*************************************************)
var boole:boolean;
begin
  if (min[nodes] > 2*max[mindeg]+3) or
      ((min[nodes] > 2*max[mindeg]+2) and (min[nodes] >=
max[mindeg]+10)) then boole:=true
            else boole:=false;
  if (activerule[450]) and (max[nodes] = min[nodes]) and
     (max[mindeg] = min[mindeg]) and
     (boole)then
    begin
      rule:='450/ ';
      if max[nind] < min[nccov] then
          begin
            z:=max[nodes]-min[mindeg]-2;
            if z < max[nccov] then pushmax(nccov);
          end;
       if min[nccov] = max[nodes]-min[mindeg]-1 then
          begin
            z:=min[nccov];
            if z > min[nind] then pushmin(nind);
            z:= max[nind];
            if z < max[nccov] then pushmax(nccov);
          end;
    end;
end;


end.
```