```
unit rules200;
   interface

          uses
           globals,cmmnds1,pusherr,pushStack,ruleAtoF;



         procedure r151; procedure r152; procedure r153; procedure
r155; procedure r156;
            procedure r157; procedure r158; procedure r159;
procedure r160; procedure r161;
            procedure r162; procedure r163; procedure r164;
procedure r165; procedure r166;
            procedure r167; procedure r168; procedure r169;
procedure r170;  procedure r171;
            procedure r172; procedure r173; procedure r174;
procedure r175; procedure r176;
            procedure r178; procedure r179; procedure r180;
procedure r181; procedure r182;
            procedure r183; procedure r184; procedure r185;
procedure r186; procedure r187;
            procedure r188; procedure r189; procedure r190;
procedure r191; procedure r192;
            procedure r193; procedure r194; procedure r195;
procedure r196; procedure r197;
            procedure r198; procedure r199; procedure r200;



implementation

procedure r151;
(*********************************************************)
(*                                                       *)
(*  if P > 2 then                                        *)
(*       if reg and Nconn=1 then echr=maxdeg+1           *)
(*                                                       *)
(*********************************************************)
begin
  if (activerule[151]) and (min[nodes] > 2) and (max[reg] = 1)
      and (min[nconn] <= 1) and (max[nconn] >= 1)then
    begin
      rule:='151/ ';
      z:=0;
      if (min[reg]=1) and (min[nconn]=max[nconn]) and
         (min[nconn]=1) then
           rulea(maxdeg,echr,-1)
        else
          if max[echr]=min[maxdeg] then
               if min[reg]=1 then
```

```
                  begin
                    if min[nconn]=1 then
                         begin
                           z:=2;
                           pushmin(nconn);
                         end
                       else
                         if max[nconn]=1 then pushmax(nconn)
                    end
                  else
                    if (min[nconn]=max[nconn]) and (min[nconn]=1)
then
                            pushmax(reg);
    end;
end;

procedure r152;
(*********************************************)
(*                                           *)
(*    if clique < 3 and 2P/5 <= nind < P/2 then *)
(*       E <= 5*nind**2 - 4*nind*P+P**2         *)
(*                                           *)
(*********************************************)
begin
  if (activerule[152]) and (min[clique] = 2) and (max[nodes]
< infinity) then
    begin
      rule:='152/ ';
      if (max[clique] < 3) and (2*max[nodes]/5 <= max[nind]) and
         (max[nind] < min[nodes] div 2) then
           begin
             k:=max[nind];
             z:=max[nodes];
             z:=5*k*k-4*z*k+z*z;
             if z < max[edges] then pushmax(edges);
             z:=min[edges]-k*k;
             if z >= 0 then
                  begin
                    z:=2*k+round(sqrt(z)+hf);
                    if z > min[nodes] then pushmin(nodes);
                  end;
             k:=max[nodes];
             z:=5*min[edges]-k*k;
             if z >= 0 then
                 begin
                   z:=round((2*k+sqrt(z))/5+hf);
                   if z > min[nind] then pushmin(nind);
                 end;
           end
         else
           begin
             k:=max[nind];
```

```
             z:=max[nodes];
             rz:=k;
             rz:=5*rz*k-(4*rz-z)*z;
             if rz < infinity then
               begin
                 if min[edges] > 5*k*k-4*k*z+z*z then
                  if (2*z/5 <= min[nind]) and (k < min[nodes] div
2) then
                      begin
                        z:=3;
                        pushmin(clique);
                      end;
               end;
            end;
    end;
end;

procedure r153;
(************************************************)
(*                                              *)
(*   Bwidth <=maxdeg*(maxdeg-1)**(rad-1)        *)
(*                                              *)
(************************************************)
begin
  if activerule[153] then
    begin
      rule:='153/ ';
      z:=max[maxdeg];
      if (z=2) or (max[radius]=1) then
            begin
              if z < max[bwidth] then pushmax(bwidth);
              z:=min[bwidth];
              if z > min[maxdeg] then pushmin(maxdeg);
            end
         else
            if (max[radius] < infinity) and (z < infinity) then
                      begin
                        power(z-1,max[radius]-1,z);
                        if z < infinity then
                           begin
                             z:=z*max[maxdeg];
                             if z < max[bwidth] then
pushmax(bwidth);
                           end;
                      end;
      if (min[bwidth] > 1) and (min[maxdeg]=1) then
                begin
                  z:=2;
                  pushmin(maxdeg);
                end;
     end;
end;
```

```
(*    RETIRED by R265
procedure r154;           *)
(*********************************************************)
(*                                                       *)
(*       if maxdeg > 2 then                              *)
(*         P <= 1+maxdeg*((maxdeg-1)**diam -1)/(maxdeg-2)  *)
(*                                                       *)
(*********************************************************)


procedure r155;
(*************************************************)
(*                                               *)
(*     if mindeg=2 then                          *)
(*       P <= (2+max(4,maxdeg))*eind/2           *)
(*                                               *)
(*************************************************)
begin
  if (activerule[155]) and (min[mindeg] <= 2) and (max[mindeg] >=
2) then
     begin
       rule:='155/ ';
       k:=max[maxdeg];
       if k > 4 then k:=2+k
                else k:=6;
       if (min[mindeg]=max[mindeg]) and (min[mindeg]=2) then
          begin
            if k < infinity then
               begin
                 if max[eind] < infinity then
                    begin
                      z:=k*max[eind] div 2;
                      if z < max[nodes] then pushmax(nodes);
                    end;
                 z:=(2*min[nodes]-1) div k+1;
                 if z > min[eind] then pushmin(eind);
               end;
             if max[eind] < infinity then
                begin
                  z:=(2*min[nodes]-1) div max[eind]-1;
                  if (z > 4) and (z > min[maxdeg]) then
pushmin(maxdeg);
               end;
           end
         else
           if min[nodes] > k*max[eind] div 2 then
                 if min[mindeg]=2 then
                      begin
                        z:=3;
```

```
                      pushmin(mindeg);
                    end
                  else
                    if max[mindeg]=2 then
                      begin
                        z:=1;
                        pushmax(mindeg);
                      end;
    end;
end;

procedure r156;
(*******************************************)
(*                                         *)
(*       if mindeg <= P/2 then             *)
(*              eind >= mindeg            *)
(*                                         *)
(*******************************************)
begin
  if activerule[156] then
    begin
      rule:='156/ ';
      if 2*max[mindeg] <= min[nodes] then
           rulea(mindeg,eind,0)
         else
           if max[eind] < min[mindeg] then
               begin
                 z:=2*max[mindeg]-1;
                 if z < max[nodes] then pushmax(nodes);
                 z:=(min[nodes]+2) div 2;
                 if z > min[mindeg] then pushmin(mindeg);
               end;
     end;
end;

procedure r157;
(************************************************)
(*                                              *)
(*      if clique <= (maxdeg-1)/2 then          *)
(*              chr <= maxdeg-1                 *)
(*                                              *)
(************************************************)
begin
  if activerule[157] then
    begin
      rule:='157/ ';
      if max[clique] <= (min[maxdeg]-1) div 2 then
             rulea(chr,maxdeg,-1)
          else
            if min[chr] >= max[maxdeg] then
               begin
                 z:=(min[maxdeg]+1) div 2;
```

```
                if z > min[clique] then pushmin(clique);
                if max[clique] < infinity then
                        begin
                          z:=2*max[clique];
                          if z < max[maxdeg] then
pushmax(maxdeg);
                        end;
              end;
    end;
end;

procedure r158;
(*******************************************)
(*                                         *)
(*     if connected then                   *)
(*        if B <=  3 then genus  = 0       *)
(*             <=  7 then       <= 1       *)
(*             <= 10 then       <= 2       *)
(*          where B is the Betti number    *)
(*               =E - P + 1                *)
(*                                         *)
(*******************************************)
begin
  rule:='158/ ';
  k:=max[edges]-min[nodes]+1;
  if (activerule[158]) and (max[connct] = 1) then
    if min[connct]=1 then
         begin
           z:=max[genus];
           if k < 4 then z:=0
               else
                 if k < 8 then z:=1
                   else
                     if k < 11 then z:=2;
           if z < max[genus] then pushmax(genus);
           if min[genus] > 2 then rulea(nodes,edges,-10)
               else
                 if min[genus] > 1 then rulea(nodes,edges,-7)
                    else
                      if min[genus] > 0 then
rulea(nodes,edges,-3);
         end
       else
         if (((k < 4) and (min[genus] > 0)) or
             ((k < 8) and (min[genus] > 1)) or
             ((k < 11) and (min[genus] > 2))) then
                  begin
                    z:=0;
                    pushmax(connct);
                  end;
end;
```

```
procedure r159;
(*************************************************)
(*                                               *)
(*       nind >= (P-1)/(maxdeg+1) + 1/(mindeg+1) *)
(*                                               *)
(*************************************************)
var rk:real;
begin
  if activerule[159] then
    begin
      rule:='159/ ';
      if (max[nind] < infinity) and (max[mindeg] < infinity) then
         begin
           rz:=max[nind]-1/(max[mindeg]+1);
           z:=round((min[nodes]-1)/rz+hf)-1;
           if z > min[maxdeg] then pushmin(maxdeg);
         end;
      if max[maxdeg] < infinity then
        begin
          rz:=(min[nodes]-1)/(max[maxdeg]+1);
          z:=round(rz+1/(max[mindeg]+1)+hf);
          if z > min[nind] then pushmin(nind);
          if max[nind] < infinity then
            begin
              rk:=max[mindeg]+1;
              rz:=max[maxdeg]+1;
              rz:=rz*(max[nind]*rk-1)/rk+1;
              if rz < infinity then z:=trunk(rz)
                               else z:=infinity;
              if z < max[nodes] then pushmax(nodes);
              k:=max[maxdeg]+1;
              rz:=max[nind]-(min[nodes]-1)/k;
              if rz <> 0 then
                 begin
                   z:=round(1/rz+hf)-1;
                   if z > min[mindeg] then pushmin(mindeg);
                 end;
            end;
        end;
    end;
end;

procedure r160;
(*************************************************)
(*                                               *)
(*  if clique = 2 then                           *)
(*       if maxdeg >= 3 then                     *)
(*          nind >= P/(maxdeg-1/5)               *)
(*                                               *)
(*       if not (even path or odd cycle)         *)
(*            and connected and P > 2 then       *)
(*             nind >= P/maxdeg +1/(mindeg+1)    *)
```

```
(*                        -1/(maxdeg+1)         *)
(*                                              *)
(************************************************)
begin
  if (activerule[160]) and (min[clique] = 2) then
    begin
      rule:='160/ ';
      if max[clique] = 2 then
        begin
          if min[maxdeg] > 2 then
            begin
              if max[maxdeg] < infinity then
                   begin
                     z:=round(min[nodes]/(max[maxdeg]-0.2)+hf);
                     if z > min[nind] then pushmin(nind);
                     if max[nind] < infinity then
                        begin
                          z:=trunk(max[nind]*(max[maxdeg]-0.2));
                          if z < max[nodes] then pushmax(nodes);
                         end;
                    end;
              if max[nind] < infinity then
                   begin
                     z:=round(min[nodes]/max[nind]+0.2+hf);
                     if z > min[maxdeg] then pushmin(maxdeg);
                   end;
            end;
          if (min[connct]=1) and (min[nodes] > 2) and
             ((max[cycle]=0) or ((min[cycle]=1) and
(max[nodes]=min[nodes])
              and (not(odd(min[nodes])))))) and ((min[edges] >=
max[nodes]) or
              (min[maxdeg] > 2) or
              ((max[nodes]=min[nodes]) and (odd(max[nodes]))))
then
                begin
                  if max[maxdeg] < infinity then
                     begin
                       k:=max[maxdeg];
                       z:=max[mindeg];
                       z:=round(min[nodes]/k+1/(z+1)-1/(k+1)+hf);
                       if z > min[nind] then pushmin(nind);
                       if max[nind] < infinity then
                          begin
                            z:=trunk(k*(max[nind]-1/(max[mindeg]+
1)+1/(k+1)));
                            if z < max[nodes] then
pushmax(nodes);
                            rz:=max[nind]-min[nodes]/k+1/(k+1);
                            if rz > 0 then
                                begin
                                  z:=round(1/rz+hf) - 1;
```

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

procedure r161;
(*******************************************************)
(*                                                     *)
(*     if genus > 1, girth >= 4, and                   *)
(*          mindeg >= trunc((5+sqrt(16*genus+1))/2)    *)
(*         then reg, hamiltonian,                      *)
(*              if mindeg=(3+sqrt(16*genus+9))/2       *)
(*                       then P=2*mindeg+2             *)
(*                       else P=2*mindeg               *)
(*                                                     *)
(*******************************************************)
begin
  if (activerule[161]) and (max[genus] < infinity) and
     (min[genus] > 1) and (min[girth] > 3) then
         begin
           rule:='161/ ';
           z:=trunk((5+sqrt(16*max[genus]+1))/2);
           if min[mindeg] >= z then
              begin
                z:=1;
                if min[reg] = 0 then pushmin(reg);
                if min[hamil] = 0 then pushmin(hamil);
                z:=2*max[mindeg]+2;
                if z < max[nodes] then pushmax(nodes);
                z:=(min[nodes]-1) div 2;
                if z < min[mindeg] then pushmin(mindeg);
                z:=max[nodes] div 2;
                if z < max[mindeg] then pushmax(mindeg);
                z:=2*min[mindeg];
                if z > min[nodes] then pushmin(nodes);
                rz:=(3+sqrt(16*min[genus]+9))/2;
                rhb:=(3+sqrt(16*max[genus]+9))/2;
                if (max[mindeg] < rz) or (min[mindeg] > rhb) then
                     begin
                       z:=(min[nodes]+1) div 2;
                       if z < min[mindeg] then pushmin(mindeg);
                       z:=2*max[mindeg];
                       if z < max[nodes] then pushmax(nodes);
                      end;
                if (min[mindeg]=max[mindeg]) and
(min[genus]=max[genus]) then
                      begin
```

```
                        z:=2*min[mindeg]-3;
                        if z*z=16*min[genus]+9 then
                              begin
                                z:=2*min[mindeg]+2;
                                if z > min[nodes] then
pushmin(nodes);
                              end;
                      end;
              end;
       end;
end;

procedure r162;
(*******************************************************)
(*                                                     *)
(*       if Nconn >= 2 and regular and                 *)
(*           ((if P is even and                        *)
(*                    mindeg>= (P-sqrt(2*P))/2)        *)
(*         or (if P is odd and                         *)
(*                    mindeg >= (P-sqrt(P))/2 ))       *)
(*          then Hamiltonian                           *)
(*                                                     *)
(*******************************************************)
begin
  if (activerule[162]) and (min[hamil] <> max[hamil]) and
(min[reg] = 1)
     and (min[nconn] >= 2) then
    begin
      rule:='162/ ';
      if ((min[mindeg] >=(max[nodes]-sqrt(max[nodes]))/2) or
         ((max[nodes]=min[nodes]) and (not(odd(max[nodes]))) and
         (min[mindeg] >= (max[nodes]-sqrt(2*max[nodes]))/2)))
then
                  begin
                    z:=1;
                    pushmin(hamil);
                  end;
    end;
end;

procedure r163;
(*****************************************************)
(*                                                   *)
(*       if reg and Nconn >= 2 and P <= 3*mindeg     *)
(*            then Hamilitonian                      *)
(*                                                   *)
(*****************************************************)
begin
  if (activerule[163]) and (min[hamil] = 0) and (max[reg] = 1)
     and (max[nconn] >= 2) then
    begin
      rule:='163/ ';
```

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

procedure r164;
(*****************************************************)
(*                                                   *)
(*     if spectr > sqrt(E) then girth=3              *)
(*                                                   *)
(*****************************************************)
begin
  if activerule[164] then
    begin
      rule:='164/ ';
      if lammin*lammin > max[edges]+0.001 then
          begin
            z:=3;
            if max[girth] > z then pushmax(girth);
```

```
          end
         else
           if min[girth] > 3 then
              begin
                z:=round(lammin*lammin+hf);
                if z > min[edges] then pushmin(edges);
                if max[edges] < infinity then
                  begin
                    rz:=sqrt(max[edges]);
                    if rz < lammax then pushlammax;
                  end;
              end;
    end;
end;

procedure r165;
(*********************************************************)
(*                                                       *)
(*       spectr >= sqrt(maxdeg)                          *)
(*                                                       *)
(*********************************************************)
begin
  if activerule[165] then
    begin
      rule:='165/ ';
      rz:=sqrt(min[maxdeg]);
      if rz > lammin then pushlammin;
      z:=trunk(lammax*lammax);
      if z < max[maxdeg] then pushmax(maxdeg);
    end;
end;

procedure r166;
(*************************************************)
(*                                               *)
(*     if diam ,Nconn > 1 then                   *)
(*        E >= (P*d-2*d-1)/(d-1)                 *)
(*                                               *)
(*************************************************)
begin
  if (activerule[166]) and (max[nconn] > 1) then
    begin
      rule:='166/ ';
      if (min[diam] > 1) and (min[nconn] > 1) then
         begin
           z:=max[diam];
           k:=min[nodes];
           if z < infinity then
              begin
                z:=(k*z-z-3) div (z-1);
                if z > min[edges] then pushmin(edges);
              end;
```

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

procedure r167;
(*********************************************)
(*                                           *)
(*      if girth > 4 then                    *)
(*          chr <= 2*(maxdeg+3)/3            *)
(*                                           *)
(*********************************************)
begin
  if (activerule[167]) and (max[girth] > 4) then
    begin
      rule:='167/ ';
      if min[girth] > 4 then
         begin
           if max[maxdeg] < infinity then
               begin
```

```
                z:=2*(max[maxdeg]+3) div 3;
                if z < max[chr] then pushmax(chr);
              end;
            z:=(3*min[chr]-5) div 2;
            if z > min[maxdeg] then pushmin(maxdeg);
          end
        else
          if 3*min[chr] > 2*max[maxdeg]+6 then
              begin
                z:=4;
                pushmax(girth);
              end;
     end;
end;

procedure r168;
(*********************************************)
(*                                           *)
(*   if girth >= 2*maxdeg**2 then            *)
(*       chr <= (maxdeg+4)/2                 *)
(*                                           *)
(*********************************************)
begin
  if activerule[168] then
    begin
      rule:='168/ ';
      z:=round((sqrt(min[girth]+1)/2)+hf);
      if z > 2*min[chr]-4 then z:=2*min[chr]-4;
      if z > min[maxdeg] then pushmin(maxdeg);
      z:=max[maxdeg];
      rz:=z;
      rz:=2*rz*rz;
      if min[girth] >= rz then
         begin
           z:=(z+4) div 2;
           if z < max[chr] then pushmax(chr);
         end
        else
          if 2*min[chr] > max[maxdeg]+4 then
             begin
               z:=2*max[maxdeg]*max[maxdeg]-1;
               if z < max[girth] then pushmax(girth);
             end;
     end;
end;

procedure r169;
(*****************************************************)
(*                                                   *)
(*     nind >= P**2/(2*E+P)                         *)
(*                                                   *)
(*****************************************************)
```

```
begin
  if activerule[169] then
    begin
      rule:='169/ ';
      z:=min[nodes];
      k:=max[edges];
      if k < infinity then
         begin
           rz:=z;
           rz:=rz*z/(2*k+z);
           if rz < infinity then
             begin
               z:=(z*z-1) div (2*k+z)+1;
               if z > min[nind] then pushmin(nind);
             end;
           z:=max[nind];
           if z < infinity then
             begin
               rz:=z;
               rz:=rz*z+8*k*rz;
               if rz < infinity then
                  begin
                    z:=trunk((z+sqrt(z*z+8*k*z))/2);
                    if z < max[nodes] then pushmax(nodes);
                  end;
             end;
         end;
       k:=max[nind];
       if k < infinity then
           begin
             z:=min[nodes];
             rz:=z;
             rz:=rz*z/k-z;
             if rz < infinity then
               begin
                 z:=round((z*z/k-z)/2+hf);
                 if z > min[edges] then pushmin(edges);
               end;
           end;
     end;
end;

procedure r170;
(*******************************************************)
(*                                                     *)
(*    if connected and not complete then               *)
(*            nind >= (P**3+3*P+1)/(2*E*P+P*P)        *)
(*                                                     *)
(*******************************************************)
begin
  if (activerule[170]) and (max[connct] = 1) then
     begin
```

```
       rule:='170/ ';
       k:=min[nodes];
       z:=max[edges];
       if (min[connct] = 1) and (max[compl] = 0) then
          begin
            if z < infinity then
              begin
                z:=round((k*k+3+1/k)/(2*z+k)+hf);
                if z > min[nind] then pushmin(nind);
              end;
            z:=max[nind];
            if z < infinity then
                begin
                  z:=round((k*k+3+1/k-z*k)/(2*z)+hf);
                  if z > min[edges] then pushmin(edges);
                end;
           end
         else
             if max[nind] < round((k*k+3+1/k)/(2*z+k)+hf) then
                 begin
                   z:=0;
                   if max[compl]=0 then pushmax(connct)
                       else
                         if min[connct]=1 then
                            begin
                              z:=1;
                              pushmin(compl);
                            end;
                 end;
     end;
end;

procedure r171;
(*******************************************************)
(*                                                     *)
(*   if genus > 1 and girth > 3 then                   *)
(*            mindeg<= 2+2*sqrt(genus)                 *)
(*                                                     *)
(*******************************************************)
begin
  if (activerule[171]) and (max[genus] > 1) and (max[girth] > 3)
then
    begin
      rule:='171/ ';
      if (min[genus] > 1) and (min[girth] > 3) then
         begin
           if max[genus] < infinity then
             begin
               z:=2+trunk(2*sqrt(max[genus]));
               if z < max[mindeg] then pushmax(mindeg);
             end;
           z:=min[mindeg]-2;
```

```
          z:=(z*z+3) div 4;
          if z > min[genus] then pushmin(genus);
         end
        else
         if min[mindeg] > 2+2*sqrt(max[genus]) then
             if min[genus] > 1 then
                    begin
                      z:=3;
                      pushmax(girth);
                    end
                  else
                    if min[girth] > 3 then
                          begin
                            z:=1;
                            pushmax(genus);
                          end;
   end;
end;

procedure r172;
(*******************************************************)
(*                                                     *)
(*      if connected then     diam <= 2*ncov           *)
(*                                                     *)
(*******************************************************)
begin
  if (activerule[172]) and (max[connct]=1) then
   begin
     rule:='172/ ';
     z:=0;
     if min[connct] = 1 then
         begin
           z:=2*max[ncov];
           if z < max[diam] then pushmax(diam);
           z:=(min[diam]+1) div 2;
           if z > min[ncov] then pushmin(ncov);
         end
       else
         if min[diam] > 2*max[ncov] then pushmax(connct);
    end;
end;

procedure r173;
(*****************************************)
(*                                       *)
(*     nind>= 2*P/(maxdeg+clique+1)      *)
(*       ( replaced by R410 )            *)
(*                                       *)
(*****************************************)
begin
  if activerule[173] then
    begin
```

```
      rule:='173/ ';
      z:=max[maxdeg];
      z1:=max[nind];
      if (z1 < infinity) and (z < infinity)
         and (max[clique] < infinity) then
           begin
             z:=(z1*(z+max[clique]+1)) div 2;
             if z < max[nodes] then pushmax(nodes);
           end;
        if (z1 < infinity) and (max[maxdeg] < infinity) then
            begin
              z:=(2*min[nodes]-1) div z1-max[maxdeg];
              if z > min[clique] then pushmin(clique);
            end;
         if (z1 < infinity) and (max[clique] < infinity) then
            begin
              z:=(2*min[nodes]-1) div z1-max[clique];
              if z > min[maxdeg] then pushmin(maxdeg);
            end;
          if (max[clique] < infinity) and (max[maxdeg]
< infinity) then
             begin
               z:=(2*min[nodes]-1) div (max[maxdeg]+max[clique]+
1)+1;
               if z > min[nind] then pushmin(nind);
             end;
    end;
end;

procedure r174;
(*************************************************)
(*                                               *)
(*  nind>=(P+2*maxdeg-clique-mindeg+1)/(maxdeg+1)   *)
(*                                               *)
(*************************************************)
begin
  if activerule[174] then
    begin
      rule:='174/ ';
      k1:=max[nind];
      k2:=max[maxdeg];
      k3:=max[mindeg];
      k4:=max[clique];
      if (k1 < infinity) and (k2 < infinity) and
         (k3 < infinity) and (k4 < infinity) then
           begin
             z:=k1+k3+k4+k2*(k1-2)-1;
             if z < max[nodes] then pushmax(nodes);
           end;
      if (k1 < infinity) and (k2 < infinity) then
          begin
            z1:=min[nodes]-k1-k2*(k1-2)+1;
```

```
            if k3 < infinity then
                  begin
                    z:=z1-k3;
                    if z > min[clique] then pushmin(clique);
                  end;
            if k4 < infinity then
                   begin
                     z:=z1-k4;
                     if z > min[mindeg] then pushmin(mindeg);
                   end;
          end;
       if (k3 < infinity) and (k4 < infinity) then
          begin
            if (k2 < infinity) and (max[compl]=0) then
              begin
                z:=(min[nodes]+2*k2-k3-k4) div (k2+1)+1;
                if z > min[nind] then pushmin(nind);
              end;
            if (k1 < infinity) and (k1 > 2) then
              begin
                z:=(min[nodes]-k1-k3-k4) div (k1-2)+1;
                if z > min[maxdeg] then pushmin(maxdeg);
              end;
          end;
     end;
end;

procedure r175;
(*********************************************)
(*                                           *)
(*      Bwidth >= (2P-1-sqrt((2P-1)**2-8E))/2   *)
(*                                           *)
(*********************************************)
begin
  if activerule[175] then
    begin
      rule:='175/ ';
      if max[nodes] < infinity then
           begin
             z:=2*max[nodes]-1;
             rz:=z;
             rz:=rz*rz-8*min[edges];
             if rz < infinity then
                begin
                  z:=round((z-sqrt(z*z-8*min[edges]))/2+hf);
                  if z > min[bwidth] then pushmin(bwidth);
                end;
             if max[bwidth] < max[nodes] then
                  begin
                    z:=max[bwidth];
                    rz:=z;
                    rz:=2*rz*max[nodes]-rz*rz-rz;
```

```
                    if rz < infinity then
                       begin
                         z:=(2*z*max[nodes]-z*z-z) div 2;
                         if z < max[edges] then pushmax(edges);
                       end;
                  end;
           end;
       rz:=sqrt(2*min[edges]);
       if (min[bwidth] <= rz) and (rz <= max[bwidth]) then
                 z:=round(rz+0.5+hf)
               else
                 begin
                   if rz > max[bwidth] then z:=max[bwidth]
                                       else z:=min[bwidth];
                   z:=round(min[edges]/z+z/2+0.5+hf);
                 end;
       if z > min[nodes] then pushmin(nodes);
    end;
end;

procedure r176;
(*****************************************)
(*                                       *)
(*      if clique=2 then                 *)
(*          Bwidth >= (3*mindeg-2)/2     *)
(*                                       *)
(*****************************************)
begin
  if (activerule[176]) and (min[clique]=2) then
    begin
      rule:='176/ ';
      if max[clique]=2 then
          begin
            z:=(3*min[mindeg]-1) div 2;
            if z > min[bwidth] then pushmin(bwidth);
            if max[bwidth] < infinity then
                 begin
                   z:=(2*max[bwidth]+2) div 3;
                   if z > max[mindeg] then pushmax(mindeg);
                 end;
          end
        else
          if 2*max[bwidth]+2 < 3*min[mindeg] then
              begin
                z:=3;
                pushmin(clique);
              end;
    end;
end;

(* procedure r177;  Replaced by r280 *)
(*****************************************)
```

```
(*                                                *)
(*     if a tree then Bwidth <= P/2               *)
(*                                                *)
(**************************************************)

procedure r178;
(*****************************************)
(*                                       *)
(*     t= max(4,nind+1)                 *)
(*    nccov <= P-mindeg-trunc((P-mindeg)/t) *)
(*                                       *)
(*****************************************)
begin
  if activerule[178] then
     begin
       rule:='178/ ';
       k:=max[nind]+1;
       if k < 4 then k:=4;
       if k < infinity then
           begin
             z:=min[mindeg]+(k*min[nccov]-1) div (k-1);
             if z > min[nodes] then pushmin(nodes);
             if max[nodes] < infinity then
                begin
                  z:=max[nodes]-min[mindeg];
                  z:=z-z div k;
                  if z < max[nccov] then pushmax(nccov);
                  z:=trunk(max[nodes]+1-k*min[nccov]/(k-1));
                  if z < max[mindeg] then pushmax(mindeg);
                end;
           end;
     end;
end;

procedure r179;
(*****************************************************)
(*                                                   *)
(*     if dom >= 2 then                              *)
(*          E <= (P-nind)*(P+nind-2*dom+2)/2         *)
(*                                                   *)
(*****************************************************)
begin
  if (activerule[179]) and (min[dom] > 1) then
    begin
      rule:='179/ ';
      k:=min[dom]-1;
      z:=min[nind];
      z:=k+round(sqrt(k*k+z*(z-2*k)+2*min[edges])+hf);
      if z > min[nodes] then pushmin(nodes);
      z:=max[nodes];
      if z < infinity then
            begin
```

```
              z:=k+round(sqrt(k*k+z*(z-2*k)-2*min[edges])+hf);
              if z < max[nind] then pushmax(nind);
              z:=max[nodes];
              k:=min[nind];
              z:=(z-k)*(z+k-2*min[dom]+2) div 2;
              if z < max[edges] then pushmax(edges);
              z:=max[nodes];
              rz:=max[nodes]-sqrt(2*min[edges]);
              if min[nind] > rz then k:=min[nind]
                    else if max[nind] <= rz then k:=max[nind]
                                 else k:=round(rz);
              z:=((z+k+2)*(z-k)-2*min[edges]) div (2*(z-k));
              if z < max[dom] then pushmax(dom);
            end;
    end;
end;

procedure r180;
(*********************************************)
(*                                           *)
(*  if P > 5 then                            *)
(*  if reg. and (P-1)/2 <= deg <= P-2 then   *)
(*      chr <= min( deg, and                 *)
(*              (2(P-deg)-3)*P /(3(P-deg)-4) *)
(*                                           *)
(*          if P-deg is odd replace the last *)
(*                term with 3*P/5            *)
(*                                           *)
(*********************************************)
begin
  if (activerule[180]) and (min[nodes] > 5) and (min[reg] = 1)
then
     if (max[nodes] div 2 <= min[maxdeg]) and
        (max[maxdeg] <= min[nodes]-2) then
           begin
             rule:='180/ ';
             if (max[nodes]=min[nodes]) and
(min[maxdeg]=max[maxdeg]) and
                (odd(max[nodes]-max[maxdeg])) then
                     z:=3*max[nodes] div 5
                else
                   begin
                     z:=max[nodes]-min[maxdeg];
                     z:=(2*z-3)*max[nodes] div (3*z-4);
                   end;
              if z > max[maxdeg] then z:=max[maxdeg];
              if z < max[chr] then pushmax(chr);
           end;
end;

procedure r181;
(*********************************************)
```

```
(*                                                *)
(*       if nccov > nind  then                    *)
(*           maxdeg >= 3*P/(3*nind-1) - 1         *)
(*                                                *)
(************************************************)
begin
  if activerule[181] then
 begin
   rule:='181/ ';
   z:=(3*min[nodes]+max[maxdeg]) div (3*max[maxdeg]+3)+1;
   if z > min[nccov] then z:=min[nccov];
   if z > min[nind] then pushmin(nind);
   if min[nccov] > max[nind] then
           begin
             z:=(3*min[nodes]-1) div (3*max[nind]-1);
             if z > min[maxdeg] then pushmin(maxdeg);
             if max[maxdeg] < infinity then
                   begin
                     z:=(max[maxdeg]+1)*(3*max[nind]-1) div 3;
                     if z < max[nodes] then pushmax(nodes);
                   end;
           end
         else
           if max[maxdeg] < 3*min[nodes]/(3*max[nind]-1)-1 then
                    rulea(nccov,nind,0);
  end;
end;

procedure r182;
(*****************************************)
(*                                       *)
(*   clique >= 2*P/(P-mindeg+nind)       *)
(*                                       *)
(*****************************************)
begin
  if activerule[182] then
    begin
      rule:='182/ ';
      k:=min[mindeg]-max[nind];
      if k > 0 then
         begin
           if max[nodes] < infinity then
               begin
                 z:=round(2/(1-k/max[nodes])+hf);
                 if z > min[clique] then pushmin(clique);
               end;
           if max[clique] < infinity then
               begin
                 z:=round(k/(1-2/max[clique])+hf);
                 if z > min[nodes] then pushmin(nodes);
               end;
         end;
```

```
      z:=max[nodes];
      k:=max[clique];
      if (z < infinity) and (k < infinity) then
           begin
             rz:=z*(k-2)/k;
             if max[nind] < infinity then
                 begin
                   z:=max[nind]+trunk(rz);
                   if z < max[mindeg] then pushmax(mindeg);
                 end;
             z:=round(min[mindeg]-rz+hf);
             if z > min[nind] then pushmin(nind);
           end;
    end;
end;


procedure r183;
(*********************************)
(*                               *)
(*    if nind <= 2 then          *)
(*      clique >=(-3+sqrt(9+8P))/2 *)
(*                               *)
(*********************************)
begin
  if (activerule[183]) and (min[nind] <= 2) then
    begin
      rule:='183/ ';
      if max[nind] <= 2 then
                    begin
                      z:=round((sqrt(8*min[nodes]+9)-3)/2+hf);
                      if z > min[clique] then pushmin(clique);
                      if max[clique] < infinity then
                         begin
                           k:=max[clique];
                           rz:=k;
                           rz:=rz*rz+3*rz;
                           if rz < infinity then
                              begin
                                z:=(k*k+3*k) div 2;
                                if z < max[nodes] then
pushmax(nodes);
                              end;
                         end;
                    end
                else
                  begin
                    rz:=max[clique];
                    rz:=rz*(rz+3);
                    if rz < infinity then
                        if max[clique]*(max[clique]+3) < 2
*min[nodes] then
```

```
                              begin
                                z:=3;
                                pushmin(nind);
                              end;
                  end;
    end;
end;

procedure r184;
(*****************************************)
(*                                       *)
(* complement of E<=ncov*maxdeg           *)
(*                                       *)
(*  P(P-1)/2 - E<=(P-clique)(P-mindeg-1) *)
(*                                       *)
(*****************************************)
begin
  if activerule[184] then
    begin
      rule:='184/ ';
      k1:=max[nodes];
      k2:=min[mindeg];
      k3:=min[clique];
      if k1 < infinity then
        begin
          z:=(k1*(k1-1)+1) div 2-(k1-k2-1)*(k1-k3);
          k1:=min[nodes];
          z1:=(k1*(k1-1)+1) div 2-(k1-k2-1)*(k1-k3);
          if z1 < z then z:=z1;
          if z > min[edges] then pushmin(edges);
          if max[edges] < infinity then
              begin
                z:=k1-k2-1;
                if z > 0 then
                    begin
                      z:=trunk(k1-(k1*(k1-1)-2*max[edges])/(2
*z));
                      k1:=max[nodes];
                      z1:=k1-k2-1;
                      if z1 > 0 then
                         begin
                           z1:=trunk(k1-(k1*(k1-1)-2
*max[edges])/(2*z1));
                           if z < z1 then z:=z1;
                           if z < max[clique] then
pushmax(clique);
                         end;
                    end;
                z:=k1-k3;
                if z > 0 then
                  begin
                    z:=trunk(k1-1-(k1*(k1-1)-2*max[edges])/(2
```

```
*z));
                    k1:=min[nodes];
                    z1:=k1-k3;
                    if z1 > 0 then
                      begin
                        z1:=trunk(k1-1-(k1*(k1-1)-2
*max[edges])/(2*z1));
                        if z < z1 then z:=z1;
                        if z < max[mindeg] then pushmax(mindeg);
                      end;
                  end;
              end;
          end;
    end;
end;


procedure r185;
(*************************************)
(*                                   *)
(*  if nccov=2 then clique=chr       *)
(*                                   *)
(*************************************)
begin
  if (activerule[185]) and (min[nccov] <= 2) then
     begin
       rule:='185/ ';
       z:=3;
       if max[nccov] <= 2 then
          begin
            rulea(clique,chr,0);
            rulea(chr,clique,0);
          end
        else
          if max[clique] < min[chr] then pushmin(nccov);
     end;
end;

procedure r186;
(************************************)
(*                                  *)
(*   if nind=2 and nccov >= 4 then  *)
(*            P >= 11               *)
(*                                  *)
(************************************)
begin
  if (activerule[186]) and (min[nind] <= 2) and (max[nccov] >= 4)
      and (min[nodes] < 11) then
    begin
      z:=11;
       rule:='186/ ';
      if (max[nind]=2) and (min[nccov] > 3) then pushmin(nodes)
```

```
       else
         if max[nodes] < 11 then
            begin
              z:=3;
              if max[nind] = 2 then pushmax(nccov)
                               else if min[nccov] >= 4 then
pushmin(nind);
            end;
     end;
end;

procedure r187;
(*************************************************************)
(*                                                           *)
(*    regular and maxdeg < P-1 then                          *)
(*          clique <= P/2-(nind-1)(nind-2)/(2(P-maxdeg-1))  *)
(*                                                           *)
(*************************************************************)
begin
  if (activerule[187]) and (max[reg] = 1) then
    begin
      rule:='187/ ';
      k1:=max[nodes];
      k2:=min[nind];
      k3:=min[maxdeg];
      k4:=min[clique];
      if (min[reg]=1) and (max[maxdeg]<=min[nodes]-2) then
          begin
            if k1 < infinity then
              begin
                z1:=k1-k3-1;
                if z1 > 0 then
                   begin
                     z:=(k1*z1-(k2-1)*(k2-2)) div (2*z1);
                     if z < max[clique] then pushmax(clique);
                    end;
                z:=z1;
                z1:=k1-2*k4;
                z:=trunk((3+sqrt(1+4*z1*z))/2);
                if z < max[nind] then pushmax(nind);
                if z1 > 0 then
                   begin
                     z:=((k1-1)*z1-(k2-1)*(k2-2)) div z1;
                     if z < max[maxdeg] then pushmax(maxdeg);
                   end;
              end;
            z:=2*k4*(k3+1)-(k2-1)*(k2-2);
            z1:=2*k4+k3+1;
            z:=z1*z1-4*z;
            if z >= 0 then
                begin
                  z:=round((z1+sqrt(z))/2+hf);
```

```
                  if z > min[nodes] then pushmin(nodes);
                end;
            end
          else
            begin
              z1:=k1-k3-1;
              if z1 > 0 then
                begin
                  z:=(k1*z1-(k2-1)*(k2-2)) div (2*z1);
                  if k4 > z then
                       if max[maxdeg]<=min[nodes]-2 then
                          begin
                            z:=0;
                            pushmax(reg);
                          end
                         else
                           if min[reg]=1 then
rulea(nodes,maxdeg,1);
                end;
            end;
      end;
end;

procedure r188;
(*********************************************)
(*                                           *)
(*   if girth is undefined then Thick = 1    *)
(*        else Thick >= E(1-2/g)/(P-2)       *)
(*                                           *)
(*********************************************)
begin
  if activerule[188] then
    begin
      rule:='188/ ';
      if min[girth]=infinity then
            begin
              z:=1;
              if max[thick] > 1 then pushmax(thick);
            end
        else
         if max[girth] < infinity then
          begin
            rz:=1-2/min[girth];
            if max[nodes] < infinity then
                begin
                  z:=round(min[edges]*rz/(max[nodes]-2)+hf);
                  if z > min[thick] then pushmin(thick);
                  if max[thick] < infinity then
                     begin
                       z:=trunk((max[nodes]-2)*max[thick]/rz);
                       if z < max[edges] then pushmax(edges);
                       z:=min[edges];
```

```
                        z1:=z-(max[nodes]-2)*max[thick];
                        if z1 > 0 then
                          begin
                            z:=(2*z) div z1;
                            if (z > 2) and (z < max[girth])
                               and (z <= min[nodes]) then
pushmax(girth);
                          end;
                      end;
                end;
            if max[thick] < infinity then
                      begin
                        z:=round(rz*min[edges]/max[thick]+hf)+2;
                        if z > min[nodes] then pushmin(nodes);
                      end;
           end;
   end;
end;

procedure r189;
(*************************************)
(*                                   *)
(*        thick <= (P+7)/6           *)
(*              <= 3  if P=9,10      *)
(*                                   *)
(*************************************)
begin
  if activerule[189] then
    begin
      rule:='189/ ';
      if max[nodes] < infinity then
         begin
           z:=(max[nodes]+7) div 6;
           if (max[nodes]=9) or (max[nodes]=10) then z:=3;
           if z < max[thick] then pushmax(thick);
         end;
      z:=6*min[thick]-7;
      if min[thick]=3 then z:=9;
      if z > min[nodes] then pushmin(nodes);
     end;
end;

procedure r190;
(*********************************************)
(*                                           *)
(*     thick <= (echr+1)/2                   *)
(*                                           *)
(*********************************************)
begin
  if activerule[190] then
    begin
      rule:='190/ ';
```

```
      if max[echr] < infinity then
        begin
          z:=(max[echr]+1) div 2;
          if z < max[thick] then pushmax(thick);
        end;
      z:=2*min[thick]-1;
      if z > min[echr] then pushmin(echr);
    end;
end;

procedure r191;
(*****************************************)
(*                                       *)
(*    thick <= max(Bwidth div 2,1)       *)
(*                                       *)
(*****************************************)
begin
  if activerule[191] then
    begin
      rule:='191/ ';
      if max[bwidth] < infinity then
          begin
            z:=max[bwidth] div 2;
            if z < 1 then z:=1;
            if z < max[thick] then pushmax(thick);
          end;
      z:=2*min[thick];
      if (z > 2) and (z > min[bwidth]) then pushmin(bwidth);
    end;
end;

procedure r192;
(*****************************************)
(*                                       *)
(*      if clique=2 then                 *)
(*            nind >= mindeg(diam+1)/4   *)
(*                                       *)
(*****************************************)
begin
  if (activerule[192]) and (min[clique] < 3)
      and (min[diam] < infinity) then
    begin
      rule:='192/ ';
      z:=min[mindeg]*((min[diam]+4) div 4);
      if max[clique]=2 then
           begin
             if z > min[nind] then pushmin(nind);
             if max[nind] < infinity then
                  begin
                    z:=(4*max[nind]) div (min[diam]+1);
                    if z < max[mindeg] then pushmax(mindeg);
                    if max[diam] < infinity then
```

```
                    begin
                      z:=(4*max[nind]) div min[mindeg] - 1;
                      if z < max[diam] then pushmax(diam);
                    end;
                  end;
             end
          else
            if max[nind] < z then
                   begin
                     z:=3;
                     pushmin(clique);
                   end;
     end;
end;

procedure r193;
(*****************************************)
(*                                       *)
(*    thick <= (ncov+1) div 2            *)
(*                                       *)
(*****************************************)
begin
  if activerule[193] then
     begin
       rule:='193/ ';
       if max[ncov] < infinity then
             begin
               z:= (max[ncov]+1) div 2;
               if z < max[thick] then pushmax(thick);
             end;
       z:=2*min[thick]-1;
       if z > min[ncov] then pushmin(ncov);
     end;
end;

procedure r194;
(*************************************)
(*                                   *)
(*    thick >= (clique+7) div 6      *)
(*          >= 3  if clique = 9 or 10 *)
(*                                   *)
(*************************************)
begin
  if activerule[194] then
     begin
       rule:='194/ ';
       z:=min[clique];
       if (z=9) or (z=10) then z:=3
              else z:=(z+7) div 6;
       if z > min[thick] then pushmin(thick);
       z:=max[thick];
       if z=2 then z:=8
```

```
          else z:=6*z-2;
       if z < max[clique] then pushmax(clique);
     end;
end;


procedure r195;
(*****************************************************)
(*                                                   *)
(*     P(P-1)/2 - E >= P-clique+(nind-1)*(nind-2)/2  *)
(*                                                   *)
(*****************************************************)
begin
  if activerule[195] then
    begin
      rule:='195/ ';
      z1:=(min[nind]-1)*(min[nind]-2);
      if max[nodes] < infinity then
         begin
           rz:=max[nodes];
           rz:=rz*(rz-3)+min[edges];
           if rz < infinity then
             begin
               k:=max[nodes]*(max[nodes]-3);
               z:=min[edges]+z1 div 2 - k div 2;
               if z > min[clique] then pushmin(clique);
               z:=min[edges]-z+max[clique];
               if z < max[edges] then pushmax(edges);
               k:=1+8*max[clique]-8*min[edges]+4*k;
               z:=trunk((3+sqrt(k))/2);
               if z < max[nind] then pushmax(nind);
             end;
         end;
      if max[clique] < infinity then
           begin
             k:=9+8*min[edges]-8*max[clique]+4*z1;
             if k >= 0 then
                 begin
                   z:=round((3+sqrt(k))/2+hf);
                   if z > min[nodes] then pushmin(nodes);
                 end;
           end;
     end;
end;


procedure r196;
(*************************************************************)
(*                                                           *)
(*   P*(P-1)-2*E>=2*clique(P-maxdeg-1)+(nind-1)(nind-2)       *)
(*                                                           *)
(*************************************************************)
```

```
begin
  if activerule[196] then
    begin
      rule:='196/ ';
      z1:=(min[nind]-1)*(min[nind]-2);
      if max[nodes] < infinity then
           begin
             k:=max[maxdeg]+1;
             if k > infinity then k:=max[nodes];
             k1:=max[nodes]*(max[nodes]-2*min[clique]-1);
             z:=(k1+2*min[clique]*k-z1) div 2;
             if z < max[edges] then pushmax(edges);
             z:=max[nodes]-k;
             if (z > 0) and (min[nodes]=max[nodes]) then
                begin
                  z:=(max[nodes]*(max[nodes]-1)-z1-2*min[edges])
div (2*z);
                  if z < max[clique] then pushmax(clique);
                end;
             z:=(2*min[edges]+z1-k1-1) div (2*min[clique]);
             if z > min[maxdeg] then pushmin(maxdeg);
             z:=1-8*min[edges]+8*min[clique]*k+4*k1;
             if z >= 0 then
                  begin
                    z:=trunk((3+sqrt(z))/2);
                    if z < max[nind] then pushmax(nind);
                  end;
           end;
      if (max[maxdeg] < infinity) and (max[clique]=min[clique])
and
         (max[compl] = 0) then
           begin
             k:=2*min[clique]+1;
             z:=k*k+8*min[edges]-8*min[clique]*(max[maxdeg]+1)+4
*z1;
             if z >= 0 then
                    begin
                      z:=round((k+sqrt(z))/2+hf);
                      if z > min[nodes] then pushmin(nodes);
                    end;
           end;
    end;
end;

procedure r197;
(*********************************************)
(*                                           *)
(*  if P >= 3 then                           *)
(*     eccov <= thick*(2*P-Ncomp-3)          *)
(*                                           *)
(*********************************************)
begin
```

```
  if activerule[197] then
    begin
      rule:='197/ ';
      k:=max[nodes];
      k1:=max[thick];
      if (min[nodes] > 2) and (k < infinity) then
          begin
            if k1 < infinity then
              begin
                rz:=k;
                rz:=k1*(2*rz-min[ncomp]-3);
                if rz < infinity then
                   begin
                     z:=k1*(2*k-min[ncomp]-3);
                     if z < max[eccov] then pushmax(eccov);
                   end;
                z:=trunk(2*k-min[eccov]/k1-3);
                if z < max[ncomp] then pushmax(ncomp);
              end;
            z:=(min[eccov]-1) div (2*k-min[ncomp]-3)+1;
            if z > min[thick] then pushmin(thick);
          end;
      if (k1 < infinity) and (min[nodes] > 2) then
         begin
           z:=round((min[eccov]/k1+min[ncomp]+3)/2+hf);
           if z > min[nodes] then pushmin(nodes);
         end;
    end;
end;

procedure r198;
(*****************************)
(*                           *)
(*   ncov <= P-P/chr         *)
(*                           *)
(*****************************)
begin
  if activerule[198] then
    begin
      rule:='198/ ';
      if max[nodes] < infinity then
          begin
            if max[chr] < infinity then
               begin
                 z:=max[nodes]-max[nodes] div max[chr];
                 if z < max[ncov] then pushmax(ncov);
               end;
            z:=(max[nodes]-1) div (max[nodes]-min[ncov])+1;
            if z > min[chr] then pushmin(chr);
          end;
      if max[chr] < infinity then
         begin
```

```
           z:=round(min[ncov]/(1-1/max[chr])+hf);
           if z > min[nodes] then pushmin(nodes);
         end;
    end;
end;

procedure r199;
(*************************************************)
(*                                               *)
(*     Bwidth <= P-1-(P-ncov) div 2              *)
(*                                               *)
(*************************************************)
begin
  if activerule[199] then
     begin
       rule:='199/ ';
       if max[nodes] < infinity then
          begin
            if max[ncov] < infinity then
                begin
                  z:=max[nodes]-1-(max[nodes]-max[ncov]) div 2;
                  if z < max[bwidth] then pushmax(bwidth);
                end;
            z:=2*min[bwidth]-max[nodes]+1;
            if z > min[ncov] then pushmin(ncov);
          end;
        if max[ncov] < infinity then
             begin
               z:=2*min[bwidth]-max[ncov]+1;
               if z > min[nodes] then pushmin(nodes);
             end;
     end;
end;

procedure r200;
(*************************************)
(*                                   *)
(*    if P > 2*eind+1                *)
(*       then ncov <= 2*eind-Nconn   *)
(*                                   *)
(*************************************)
begin
  if activerule[200] then
     begin
       rule:='200/ ';
       z:=(min[ncov]+min[nconn]+1) div 2;
       if z > min[nodes] div 2 then z:=min[nodes] div 2;
       if z > min[eind] then pushmin(eind);
       if min[nodes] > 2*max[eind]+1 then
          begin
            z:=2*max[eind]-min[nconn];
            if z < max[ncov] then pushmax(ncov);
```

```
            z:=2*max[eind]-min[ncov];
            if z < max[nconn] then pushmax(nconn);
          end
        else
          if min[ncov] > 2*max[eind]-min[nconn] then
             begin
               z:=2*max[eind]+1;
               if z < max[nodes] then pushmax(nodes);
             end;
    end;
end;

end.
```