Grafo gaineko algoritmoak:
sakonerako eta zabalerako korritzeak

R. Arruabarrena
LSI - UPV/EHU

Sarrera

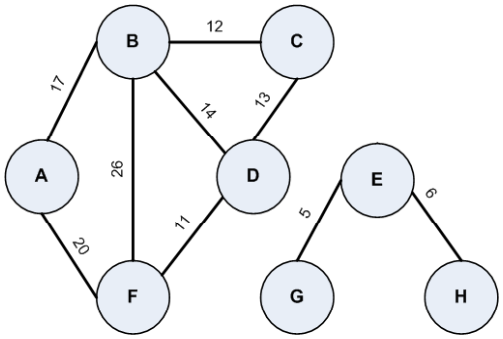
- Problema algoritmiko asko grafoen bidez modelatzen dira
  - Bide motzenak, hedapen zuhaitz minimoak, osagai konexuak,...
- Grafo datu-egiturak eta hauen gaineko korritzeak ematen dute soluzioa maiz
- Korritze esplizituak
  - Sakonerako korritzea
  - Zabalerako korritzea
- Korritze inplizituak
  - backtrack algoritmoak


```
procedure  GrafoaKorritu (G: in Grafoa) is
   Aztertua: constant Boolean := True;
   Bisitak: Taula(1..N):=(Others=> not (Aztertua));
    -- Erpin kopuruaren esparrua ≡ 1..N
begin

   for  Erpin_Bat  in  1..N loop
       if  not MarkatuaDago(Bisitak, ErpinBat)  then
              MarkaIpini(Bisitak, ErpinBat, Aztertua);
              SK(G, ErpinBat);          -- edo  ZK(G,
   ErpinBat);
       end if;
   end loop;

end GrafoaKorritu;
```



```
procedure  SK  (G: in Grafoa; A: in Erpina) is
   AAuzokideenKopia: ErpinenL;
   Lehena: Erpina;

Begin
   AAuzokideen_Kopia:= Auzokideak(G, A);

   while  not( HutsaDaL? (AAuzokideenKopia))  loop
       Lehena:= LehenengoaL(AAuzokideenKopia);
       Hondarra(A_AuzokideenKopia);

       if not Markatua_Dago(Bisitak, Lehena)  then
              Marka_Ipini(Bisitak, Lehena, Aztertua);
              S_K(G, Lehena);
       end if;

   end loop;
end S_K;
```

```
procedure  ZK  (G: in Grafoa; X: in Erpina) is
 Ilara: IlaraDMA:=HutsaI; YrenAuzokideak: ErpinenL;
 Y,Z: Erpina;
begin
   IlarariGehitu(Ilara, X);

   while not( HutsikDagoI?(Ilara))  loop
       Y:= Burua(Ilara);    Ilara:= IlaratikKendu(Ilara);
       YrenAuzokideak:= Auzokideak(Y).;
       while  not ( HutsikDagoL?(YRenAuzokideak))  loop
         Z:=LehenaL(YrenAuzokideak); Hondarra(YrenAuzokideak);

         if  not MarkatuaDago?(Bisitak, Z)  then
              MarkaIpini(Bisitak, Z, Aztertua);
              IlarariGehitu(Ilara, Z);
         end if;

       end loop;
   end loop;
end ZK;
```

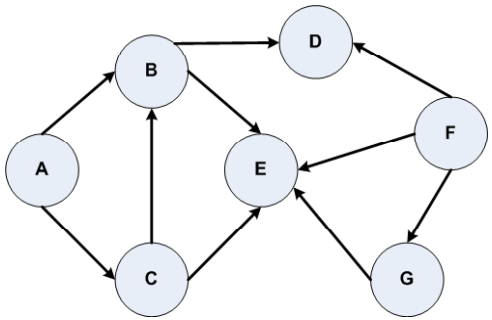
Grafo ez-zuzendu pisudunaren erpinen eta ertzen lehengo bisita ordenaren zerrendak:

- Sakonerako korritzean :

```
a (a,b) b (b,c) c (c,d) d (d,b) (d,f) f (f,a) (f,b) e (e,g) g (e,h) h
```

- Zabalerako korritzean:

```
a (a,b) (a,f) b (b,c) (b,d) (b,f) f (f,d) (c,d) e (e,g) (e,h) g h
```

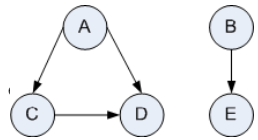
Grafo zuzenduaren erpinen eta arkuen lehengo bisita ordenaren zerrendak:

- Sakonerako korritzean : `a (a,b) b (b,d) d (b,e) e (a,c) c (c,b) (c,e) f (f,d) (f,e) (f,g) g (g,e)`
- Zabalerako korritzean: `a (a,b) (a,c) b (b,d) (b,e) c (c,b) (c,e) d e f (f,d) (f,e) (f,g) g (g,e)`

Korritzeak,

- Sakonerakoa

```
a (a,c) c (c,d) d (a,d)  b (b,e)
```



- Zabalerakoa
  - Ilaratzerakoan (hasieraketa, markatzen denean)

```
a (a,c) c (a,d) d (c,d) b (b,e) e
```

  - Ilaratik ateratzerakoan (benetako tratamendua)

```
a (a,c) (a,d) c (c,d) d b (b,e) e
```

Zuhaitz baten maila populatuena

Z zuhaitza emanik bertako zein K mailak adabegi gehien dituen kalkulatuko duen algoritmoa idaztea eta bere ordena kalkulatzea eskatzen da. Horietako maila bat baino gehiago existituko balira, maila handiena itzuli beharko luke algoritmoak

- Soluzioa uler dadin hainbat argipen
  - Zuhaitzak gehienez erpin adina maila izango ditu
  - MAILAK izeneko bektore global bat izango dugu maila bakoitzeko erpin kopurua metatzeko.
    - Mailetako kontagailuak hasieran zerora hasieratuko ditugu
  - Zabalerako korritzea erabiliz errotik urrunduz erpinak banaka aztertuko dira.
    - Erpinaren azterketak hau den mailako kontagailua inkrementatuko du
  - Korritzea bukatu ondoren, bektoreko balio maximoa maila populutsuenaren erpin kopurua da


```
procedure  GrafoaKorritu (G: in Grafoa, Erroa: in Erpina) is
   Aztertua: constant Boolean := True;
   Bisitak: Taula(ErpinKopEsparrua):=(Others=> not (Aztertua));
   Mailak: Taula(ErpinKopEsparrua):=(Others=> 0);

begin
   MarkaIpini(Bisitak, Erroa, Aztertua);
   ZK(G, Erroa);

   maxP:=maximoarenPosizioa(Mailak);
   writeln (Mailak[maxP], " erpin daude maila populatuenean,
           hau ", maxP,". Izanik.");

end GrafoaKorritu;
```



```
procedure  ZK  (G: in Grafoa; X: in Erpina) is
   Ilara: IlaraDMA:=HutsaI;  YrenAuzokideak: ErpinenL; Y,Z: Erpina;

begin
   IlarariGehitu(Ilara, (X, 0));                -- lehenengo maila 0-a
   while not( HutsikDagoI?(Ilara))  loop
        (Y, my):= Burua(Ilara);       Ilara:= IlaratikKendu(Ilara);
        Mailak[my]= Mailak[my]+1;

        YrenAuzokideak:= Auzokideak(Y).;
        while  not ( HutsikDagoL?(YRenAuzokideak))  loop
           Z:=LehenaL(YrenAuzokideak);Hondarra(YrenAuzokideak);
           if  not MarkatuaDago?(Bisitak, Z)  then
                MarkaIpini(Bisitak, Z, Aztertua);
                IlarariGehitu(Ilara, (Z, my+1));
           end if;
        end loop;
   end loop;
end ZK;
```


- Soluzioaren denbora analisia
  - `Mailak` bektorearen hasieratzea eta maximoaren kalkulua eragiketak linealak dira erpin kopuruan, $\Theta(n)$
  - Zabalerako korritzean erpin eta arku bakoitza behin tratatzen da
    - Erpin bakoitzeko kontagailu bat inkrementatuko da
    - Arkuak jomuga erpina harrapatzeko soilik aztertzen da
    - Eragiketa guztiak denboran konstantean burutzen direnez, korritzeak $\Theta(n+a)$ denbora du
- Baturaren erregela aplikatuz, $\Theta(n+(n+a)+n)=\Theta(n+a)$

Arbasoen zuhaitza

Demagun abizen jakin bateko zuhaitza genealogikoa dugula, erroa ezaguna izanik; hots, abizen hori eraman zuen lehenengo arbasoa.

Helburua sendiko edozein bi ahaideren arteko hurbilen duten lehengo arbaso komuna finkatuko duen prozedura egitea da.

Proposatutako soluzioaren denbora-analisia eta -ordena kalkula itzazu

- Soluzioa uler dadin hainbat argipen
  - Grafoa zuhaitza da, baina ez du zertan bitarra izan behar
    - odoleko zuzenen kopurua biren desberdina izan bailiteke
  - Bi ahaideak A1 eta A2 izango dira, zuhaitzean daude eta honen edozein mailatan.
  - Ahaide errepikaturik ez dago

- Sakonerako korritzea behar dugu problema ebazteko
- Erpin bakoitzeko aldagai pare bat egongo da; hots, topatu behar den ahaide bakoitzeko aldagai bat.
  - Hasieran pare guztiak False-ra hasieratuko dira
- Ahaide komun lehena topatzeko bi kasu daude
  - Ahaideak adar berean egotea
  - Ahaideak adar desberdinetan egotea

- Ahaideak adar berean
  - gurasoak pareko ahaide bat True izango du eta umeren batetik zintzilikatzen du beste ahaidea. Bigarrena, atzerako prozesaketaren ondorioz jasoko du, baina gurasoan detektatuko da adar batean daudela biak.
- Ahaideak adar desberdinetan
  - umeren bateko korritzeak ahaide bat topatzen duenean, post-prozesaketa bidez jasoko du gurasoak (adar horretatik ahaidea dagoela); beste umearekin berdin. Gurasoaren ume guztietatik abiatutako adarkatzeetatik biak topatu direnean, guraso hori da ahaide komun lehena.


```
procedure  AhaideKomunLehena (G: in Grafoa;
          Erroa, A1,A2: in Erpina;     ArbasoLehena: out Erpina) is

   Amaitua: constant Boolean := False;
   Ahaideak: PareenTaula (1..N) :=(Others=> (False,False));
   Lehena: Erpina;

begin
   if Erroa = A1 then Ahaideak(Erroa).A1:= True;
   else if Erroa = A2 then Ahaideak(Erroa).A2:= True; end if;
   end if;

   SK(G, Erroa);
   ArbasoLehena:=Lehena;

end AhaideKomunLehena;
```

```
procedure  SK  (G: in Grafoa; UnekoErp: in Erpina) is
   AAuzokideenKopia: ErpinenL;
   Erp: Erpina;
begin
   AAuzokideen_Kopia:= Auzokideak(G, UnekoErp);
   while  not( HutsaDaL? (AAuzokideenKopia))
            and not Amaitua                    loop
      Erp:= LehenengoaL(AAuzokideenKopia);
      Hondarra(A_AuzokideenKopia);
      if Erp = A1 then  Ahaideak(Erp).A1:= True;
      else if Erroa = A2 then  Ahaideak(Erp).A2:= True; end if;
      end if;

      S_K(G, Erp);
      OrBaturaGurasoEtaUmearenArtean(UnekoErp,Erp);
      If (Ahaideak(E).A1) and (Ahaideak(E).A2))
      then Lehena:= UnekoErp; Amaitua:=True;
      end if;
   end loop;

end S_K;
```



```
OrBaturaGurasoEtaUmearenArtean(E1,E2):
Ahaideak(E1).A1 := (Ahaideak(E1).A1) or (Ahaideak(E2).A1);
Ahaideak(E1).A2 := (Ahaideak(E1).A2) or (Ahaideak(E2).A2);
```

Algoritmoaren denbora analisia

- Funtsean algoritmoak zuhaitza bat sakoneran korritzen du. Erpin eta arku bakoitzeko egiten den tratamendua denbora konstantean egingarria da honakoa izanik.
  - Erpin bakoitzeko:
    - Erpinaren prozesaketa hasi aurretik: bi aldagaien hasieratzea
    - Erpinaren umeen azterketa ondoren: BiakTrue?
  - Arku bakoitzeko:
    - Arkuko jomuga erpina ahaidetako bat bada, erregistratu jomuga erpinari
    - Arku horretatik sortutako adartzetik ahaiderik zintzilik badago, gurasoak ere badu (OrBatura)
- Hortaz, soluzioren denbora-ordena O(n+a) =O(n) da, n erpin kopurua izanik eta a arku kopurua (a=n-1 baita).

Topaketako afaria. Grafo bipartigarria?

Topaketa bateko partaideen arteko *elkar ezinikusi erlazioa* ezaguna denez, topaketako antolatzaileek azken eguneko gala afarian ahalik eta girorik egon dadin, Timanfaya eta Garajonay izeneko bi jangela dituen jatetxe bat topatu dute eta haietan banatu nahi dituzte partaideak.

Idatz eta aztertu ezazu algoritmo bat, elkar ezinikusi erlazioa errespetatuz, erabakiko duena posiblea den ala ez partaideak bi jangeletan banatzea, eta hala balitz, banaketa itzuliko lukeena.

Beraz, edozein partaide jangela batean esleiturik dagoenean, honek ezinikusi dituenak beste jangelan egon beharko lukete

- Problema partiketa problema da:
  - bi multzo disjuntutan banagarriak dira jankideak?
  - Problema ebazteko zabalerako (nahiz sakonerako) korritzea erabil dezakegu
- Prozesua (zabalerako korritzea
  - Jankide bat abiapuntutzat aukeratzen da, eta adibidez A jangela esleitzen zaio
  - Abiapuntua (0 mailan dagoena) eta maila bikoitietan (ertz-distantzia bikoitietara) dauden guztiak ‘A’(edo 0) jantokira esleitugarriak izan beharko lukete.
  - Aldiz, maila bakoitietan eroriko liratekeena ‘B’ (edo 1) jangelara esleitugarriak izan beharko lukete.
  - Honela, jangela esleitua duen jankide bat berriz aztertu beharko bagenu (ziklorik dagoelako edo gurasoa delako edo jatorritik bide batek baino gehiago existitzen duelako)
    - problemarik ez legoke jangela berdina esleitu beharko bagenio.
    - aldiz, jangela desberdina tokatuko balitzaio, partiketarik ez luke existituko.

- Suposaketak eta erabakiak:
  - etsai erlazioa erreziprokoa (grafo ez-zuzendua) dela
  - gerta liteke grafoa ez-konexua izatea
  - abiapuntuak beti 0 jangelara esleituko ditugu


```
procedure  GrafoaKorritu (G: in Grafoa) is
   Aztertua: constant Boolean := True;
   Bisitak: Taula(1..N):= (Others=>  not (Aztertua));
   Jangela: array (1..N) of 0..1;
   EzDagoPartiketarik: Boolean:= False;
begin
   for  ErpinBat  in  1..N loop
        -- osagai konexu edo azpigrafo bakoitzak elkarren
        -- arteko ezinikusiak jasotzen ditu.
        -- Beharrezkoa da begizta
        if  not MarkatuaDago(Bisitak, ErpinBat)  then
              MarkaIpini(Bisitak, ErpinBat, Aztertua);
              Jangela(ErpinBat):= 0;
              ZK(G, ErpinBat,0);
        end if;
   end loop;
   if EzDagoPartiketarik then write ("ez dago partiketarik");
   else IdatziTaula(Jangela);
   end if;
end GrafoaKorritu;
```



```
procedure  ZK  (G: in Grafoa; X: in Erpina) is
Ilara: IlaraDatuMota:=HutsaI; YRenAuzokideak: ErpinenL; Y,Z:Erpina;
begin
   IlarariGehitu(Ilara, X);
   while not( HutsikDagoI?(Ilara)) and (not EzDagoPartiketarik) loop
      Y:= Burua(Ilara); Ilara:= IlaratikKendu(Ilara);
      YEnAuzokideak:= Auzokideak(Y);
      while  not ( HutsikDagoL?(YRenAuzokideak))  loop
        Z:= LehenaL(YRenAuzokideak);
        Hondarra(YRenAuzokideak);
        if  not MarkatuaDago?(Bisitak, Z)
        then    MarkaIpini(Bisitak, Z, Aztertua);
                Jangela(Z):= ( (1+Jangela(Y)) mod 2);
                IlarariGehitu(Ilara, Z);
        else    if Jangela(Z)=Jangela(Y) then
                 EzDagoPartiketarik:=True; exit;
         -- else markatuta dago eta esleitutako jangela egokia da!
         end if;
      end loop;
   end loop;
end ZK;
```


Algoritmoaren denbora analisia

- Zabalerako korritzeari denbora konstantea behar duten hainbat eragiketa esleitu zaio. Zehazki,
  - erpin edo jankide bakoitza behin markatu eta jangela batera esleituko da.
  - Ertz bakoitzeko bertako erpin bat dagoeneko marka duen ala ez begiratuko da.
  - Eragiketa hauek denak dira denbora konstantean egikarigarri eta gehitu diren eragiketa berriak.

  Ondorioz,
  - $\Theta(n+a)$ ordena du soluzioak baldin eta partiketak existitzen badu (ertz eta erpin guztiak behin tratatuko direlako);
  - aldiz, partiketarik ez balego, orduan $O(n+a)$ litzateke.

Bide kopuruak Erromaraino

Jakina da bide oro Erromara doala. Gida on bat erabiliz eta errepide mapa bat erabiliz, grafo zuzendu azikliko bat (DAG, ingeleratik, directed acyclic graph) marraztu dugu hamaika ibilbide on jasotzen dituena gure hiritik Erromara iristeko.

Emandako DAGa erabiliz, gure hiritik Erromara doazen ibilbide kopurua kalkulatuko duen algoritmoa diseinatzea eskatzen zaizu bai eta haren denbora-analisia kalkulatzea ere.

- Soluziorako argipenak
  - `BideKop(1..N)` bektorea erabilko dugu
    - `BideKop(H)=z` `H` grafoko erpinetik Erromaraino `z` bide dago
    - Bektorearen betetzeak atze prozesaketa behar du
      - **Erpin baten ume guztietatik dauden bide kopuruak ezagutzen direnean, gurasoaren kopurua finka liteke**

- Abiapuntu eta jomuga erpinak (`J` erpina):
  - Mapan abiapuntu bat baino gehiago egon liteke; hots, `J` abiabuntu erpina da baldin `InDegree(J)=0`.
    - Gure hiria mapako erpin izanik, ez du zertan abiapuntu erpina izan behar: gerta liteke `InDegree(GureHiria)≠0)`. Halere ,gure abiapuntua izango da.
  - Jomuga bat baino gehiago egon litezke; hots, `Outdegree(J)=0`. Erroma bat izan liteke edo ez, Erroma “barruko” erpin bat izan bailiteke.
  - Beraz, (GureHiria,Erroma) erpin bikote arteko bideen kopurua kalkulatu behar dugu baldintza horietan:
    - `BideKop(GureHiria)`
- Oharra: gure soluzioan `BideKop(K)` ez da kalkulatuko, gure hiria eta Erroma arteko bideen artean ez badago `K` erpina.

- Grafoa sakoneran korrituko da,
  - Markatzea BideKop taula bidez egingo da.
    - -1: oraindik azter gabea adieraziko du
    - ≥0: aztertua edo aztertzen hasia
  - gure hiritik egingo lehengo deia eta Erromatik aurrera ez da sakoneran korritzen jarraitu behar. Grafoa Korritu prozeduran honako hasieratzeak beharko dira:
    - `BideKop(HasiHiria)=0;` bide kontagailua hasieratzea
    - `BideKop(Erroma)=1;` Erromatik Erromara bide 1, k.n.
- `DaudenKop`, bi hiri horien arteko bideen kopurua jasoko du.
- Egin beharko den deia

  `GrafoaKorritu(Donostia, Erroma, DaudenKop).`


```
procedure  GrafoaKorritu (G: in Grafoa;
                          HasiHiria: in Erpina    -- Gure hiria
                          Erroma: in Erpina       -- Erroma, jo muga
                          DaudenKop: out Integer) is

   BideKop: Taula(ErpinKopEsparrua):= (Others=>  -1);
                          -- = Bisitatugabea

begin

  BideKop(Erroma):=1;    -- Erromatik Erromara bide bat.
                         -- Kasu nabari bat oraingoan hemen
                         -- Erromatik irteten direnak ez kalkulatzeko
  BideKop(HasiHiria):=0;
  SK(G, HasiHiria);
  DaudenKop:= BideKop(HasiHiria);

end GrafoaKorritu
```

```
procedure  SK  (G: in Grafoa; A: in Erpina) is
   AAuzokideenKopia: ErpinenL;        Lehena: Erpina;
Begin

   AAuzokideen_Kopia:= Auzokideak(G, A);
   while  not( HutsaDaL? (AAuzokideenKopia))  loop
        Lehena:= LehenengoaL(AAuzokideenKopia);
        HondarraL (A_AuzokideenKopia);
        if BideKop(Lehena)=-1            -- Ez da aurretik kalkulatu
        then BideKop(Lehena):=0;         -- Markatu,hasieratu+balioa lortu
             S_K(G, Lehena);
        end if;
        BideKop(A):= BideKop(A)+BideKop(Lehena);
        -- BideKop(Lehena)=0 denean Lehena hosto da edo Erromara ez
        -- doazen bideetan dago.
        -- BideKop(Lehena)=1 denean Lehena=Erroma
   end loop;
end S_K;
```