Tiesiami keliai

Toks uždavinys vadinamas *Maksimalių srautų* uždaviniu.

Maksimalių srautų uždavinys yra vienas lengviausių iš *Srautų tinklų* uždavinių klasės, sprendžiamas *Srautų leidimo žemyn* (*Preflow-push*) metodu.

Maksimalių srautų uždavinys

Turime orientuotą grafą, kurį sudaro viršūnių ir briaunų aibė. Grafas yra svorinis, t. y. kiekviena briauna (u, v) turi neneigiamą svorį $sv(u, v) > 0$. Tą svorį vadinkime briaunos *pralaidumu*. Dvi grafo viršūnės išsiskiria iš kitų viršūnių: tai pradinė viršūnė p ir galinė viršūnė g. Susitarkime, kad grafas pasižymi tokia savybe: iš pradinės viršūnės p galima pasiekti bet kurią kitą viršūnę ir iš bet kurios viršūnės galime pasiekti galinę viršūnę g.

Toks grafas gali modeliuoti ne tik kelių žemėlapį. Tai gali būti ir elektros grandinė ar namo šildymo sistema. Pavyzdžiui, į namą ateina storas vamzdis su karštu vandeniu ir išsišakoja į storesnius ir plonesnius vamzdžius. Tie vamzdžiai išvedžiojami po namą, tai sujungiami, tai vėl atskiriami įvairiuose aukštuose, kol ataušęs vanduo surenkamas kitame vamzdyje. Žinomi visų vamzdžių pralaidumai. Koks didžiausias kiekis vandens gali ateiti į storąjį vamzdį.

Srautai s yra apibrėžti visoms grafo briaunoms. Jie pasižymi šiomis savybėmis:

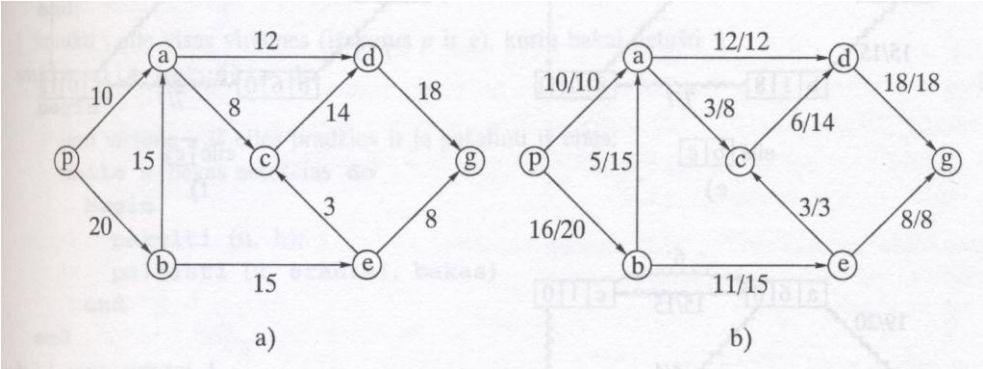
1) $s(u, v) \le sv(u, v)$. Tai reiškia, kad briauna praleidžiami srautai negali viršyti briaunos pralaidumo. Pavyzdžiui, jei siauroje gatvelėje tiesiamas kelias ir jame tilps tik dvi juostos, negalima leisti mašinoms važiuoti trimis juostomis.

2) $s(u, v) = -s(v, u)$. Į viršūnę ateinantys srautai žymimi teigiamu skaičiumi, iš jos išeinantys – neigiamu.

3) Kiekvienai viršūnei, išskyrus p ir g, galioja taisyklė: *Į ją ateinančių srautų suma lygi iš jos išeinančių srautų sumai.* Jei taisyklės nebūtų laikomasi galėtų susidaryti nekorektiška situacija. Pavyzdžiui, į sankryžą mašinos atvažiuoja penkiomis eismo juostomis, išvažiuoja tik viena. Arba iš pastato išeina daugiau žmonių nei įėjo. Matyt, kažkas lipo per langą....

Srautu grafe vadinsime visų iš pradinės viršūnės p išeinančių srautų sumą.

Žemiau pateiktas pavyzdys iliustruoja grafą ir per jį praleidžiamus srautus:



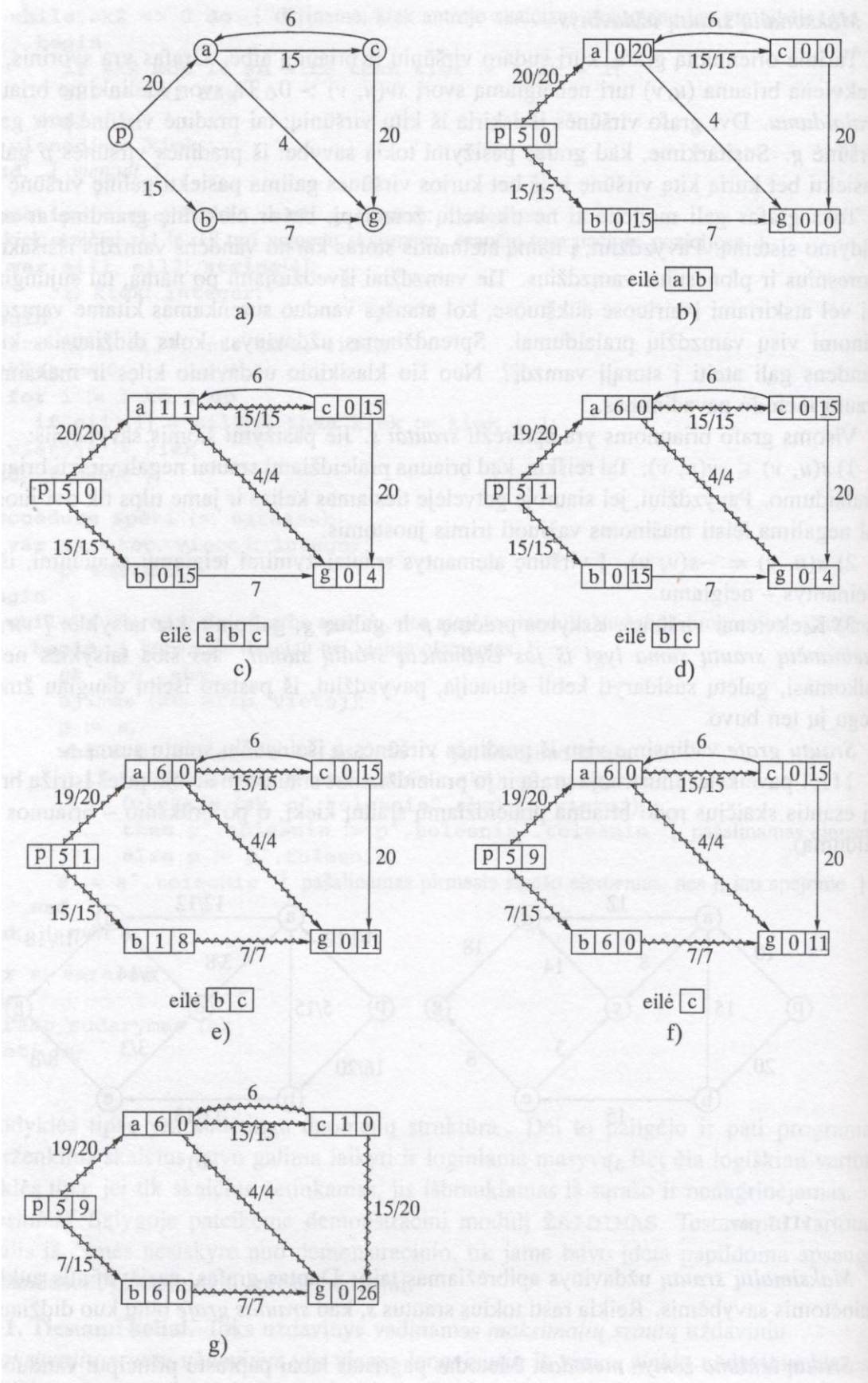
Maksimalių srautų uždavinys apibrėžiamas taip: Duotas grafas, pasižymintis, aukščiau minėtomis savybėmis. Rasti tokius srautus *s*, kad *srautas grafe* būtų kuo didžiausias.

Srautų leidimo žemyn metodas

Metodas remiasi labai paprastu principu: vanduo teka iš aukštesnės vietos į žemesnę. Metodą aiškinsime vandens tekėjimo vamzdžiais terminais. Sakykime, kad vamzdžius jungia *m* mazgų. Kiekvienas mazgas turi baką, kuriame laikomas į jį atėjęs ir iš jo dar neišėjęs vanduo. Pradžioje visi mazgai, išskyrus pradinį yra nuliniame aukštyje. Pradinis mazgas pakeltas į aukštį *m*. Iš pradinio mazgo į su juo sujungtus mazgus išbėga tiek vandens, koks yra jungiančių vamzdžių pralaidumas. Tas vanduo subėga į bakus.

Imamas mazgas, kurio bake yra vandens, jei galima, jis pakeliamas ir iš jo vanduo bėga į kitų vamzdžių bakus. Veiksmai kartojami, kol nei vieno mazgo (išskyrus pradinį ir galinį) bake nelieka vandens.

Algoritmą iliustruoja šie paveikslėliai. Kursyvu rašomas viršūnės aukštis, pusjuodžiu šriftu – viršūnės bake esančių srautų kiekis. Pradinės ir galinės viršūnės į eilę neįtraukiame, nes baigus algoritmą pradinės bakas gali, o galinės privalo būti netuščias.


a)
b)
c)
d)
e)
f)
g)
eilė a b
eilė a b c
eilė b c
eilė c

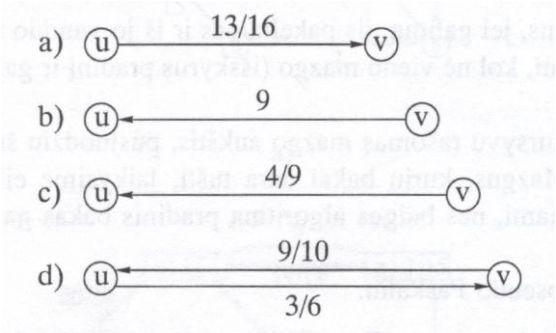
Pateiksime formalų algoritmo užrašą Pseudo-Paskaliu.

```
procedure max_srautai (p, g, m;      { pradinė, galinė viršūnės, viršūnių
skaičius }
                        sv;     { briaunų svoriai }
                        var s   { srautai }) ;
  var h,        { viršūnių aukščiai }
      bakas;    { bakas }
begin
  inicializuoti (h, bakas, s); { priskiriami nuliai }
  h[1] := m; { pradinės viršūnės aukštis }
  { paleidžiami srautai iš pradinės viršūnės }
  for u gretima viršūnei p do
    begin
      s[1, u] :=  sv[1, u];
      s[u, 1] := -sv[1, u];
      bakas[u] := sv[1, u];
    end;
  įtraukti į eilę visas viršūnes (išskyrus p ir g, , kurių bakai netušti }
  while eilė netuščia do
    begin
      imti viršūnę u iš eilės pradžios ir ją pašalinti iš eilės;
      while u bakas netuščias do
        begin
          pakelti (u, h);
          paleisti (u, s, bakas)
        end;
    end;
end; { max_srautai }
```

Apie darbą su eile galite pasiskaityti 83 uždavinio sprendimo aprašyme. Atskirai pateiksime procedūras *pakelti* ir *paleisti*. Šios procedūros atliekamos tik su tomis viršūnėmis, kurių bakai nėra tušti.

Viršūnės u aukštis didinamas tik tada, jei visų viršūnių, į kurias iš jos *galime paleisti srautus*, aukščiai yra didesni arba lygūs u. Jei nors vienos viršūnės aukštis yra žemesnis, tuomet nėra reikalo didinti viršūnės aukštį: galima iš karto paleisti srautus.

Reiktų detalizuoti sąvoką *galime paleisti srautus*. Paleisti srautus iš u į v galima tada, jei $sv[u, v] - s[u, v] > 0$. Panagrinėkime keletą pavyzdžių:



a) atveju iš *u* į *v* galime paleisti tris srautus, kurio kiekis yra 3. b) atveju iš u į v srautų paleisti negalime. c) atveju galime paleisti keturis srautus. Iš *u* į *v* atėjo keturi srautai, jeigu neturime kur juos nukreipti, galime grąžinti atgal. d) atveju galime paleisti keturiolika srautų.

```
procedure pakelti (u; { keliama viršūnė }
                   var h { viršūnių aukščiai });
  var min;
begin
  inicializuoti min;
  for v yra grafo viršūnė do
    if (sv[u, v] - s[u, v] > 0) ⇒ (h[u] <= h[v])
       then if h[v] < h[mini]
                 then mini := v
       else pakelti negalima;
  if galima pakelti
     then h[u] := h[mini] + 1
end; { pakelti }
```

Jei viršūnės *u* bakas netuščias, baką reikia ištuštinti. Randamos tokios viršūnės, kurios yra žemesnės nei *u*, ir į kurias galima nukreipti srautus ir tai padaroma. Iš principo, iš *u* srautus galima paleisti į bet kurią viršūnę, kuri yra žemiau nei *u*, tačiau panagrinėję aukščiau pateiktą pavyzdį, pamatysite, kad užtenka imti tik tas viršūnes, kurios yra lygiai vienetu žemesnės nei *u*.

```
procedure paleisti (u; var s; var bakas);
{ paleidžia viršūnės u bake esančius srautus }
  var kiekis;
begin
  for v yra grafo viršūnė do
    if (h[v] + 1 = h[u]) and (sv[u, v] - s[u, v] > 0)
        { jei v žemesnė ir į ją dar galima paleisti srautus }
        then begin
                kiekis := min (bakas[u], sv[u, v] - s[u, v]);
                bakas[u] := bakas[u] - kiekis;
                s[u, v] := s[u, v] + kiekis;
                s[v, u] := -s[u, v];
                bakas[v] := bakas[v] + kiekis;
                if v nėra pradinė ar galinė viršūnė ir jos bakas netuščias
                    then įtraukti į eilę v;
             end;
end; { paleisti }
```

Pateikiame Paskaliu užrašytą algoritmą.

```
const MAX_M = 100; { maksimalus miestų skaičius }
type lentelė1 = array [1..MAX_M] of integer;
     lentelė2 = array [1..MAX_M, 1..MAX_M] of integer;
     aibė = set of 1..MAX_M

procedure max_srautai (m: integer; sv: lentelė2;
                       var s: lentelė2);
{ randa maksimalius srautus grafe }
  var e, { eilė }
      h, bakas: lentelė1;   { viršūnių aukščiai ir bakas }
      u, galva, uodega: integer;
      virš: aibė;

  procedure inicializuoti (var h, bakas: lentelė1;
                           var s: lentelė2 { srautai });
  { inicializuoja lenteles h, s, bakas }
    var u, v: integer;
  begin
    for u := 1 to MAX_M do
      begin
        h[u] := 0;
        bakas[u] := 0;
      end;
```

```
  for u := 1 to MAX_M do
    for v := 1 to MAX_M do
      begin
        s[u, v] := 0;
        s[v, u] := 0;
      end;
end; { inicializuoti }

procedure įtraukti_į_eilę (u: integer; var virš: aibė;
                           var e: lentelė1; var uodega:
integer);
  { įtraukia į eilės pabaigą viršūnę u }
  begin
    virš := virš + [u];
    e[uodega] := u;
    if uodega = MAX_M
       then uodega := 1
       else uodega := uodega + 1;
  end; { įtraukti_į_eilę }

  procedure pašalinti_iš_eilės (var galva: integer;
                                var virš: aibė);
  begin
    virš := virš - [u];
    if galva = MAX_M
       then galva := 1
       else galva := galva + 1
  end; { pašalinti_iš_eilės }

  procedure pakelti (u: integer; { keliama viršūnė }
                     var h: lentelė1 { viršūnių aukščiai });
  { padidinamas u aukštis }
  { vartojami globalūs kintamieji m, s, sv }
  var mini, v: integer;
      galima: boolean;
  begin
    galima := true; { ar galima pakelti viršūnę }
    mini := 0;
    for v := 1 to m do
      if (sv[u, v] - s[u, v] > 0)
         then if (h[u] <= h[v])
                 then begin
                        if mini = 0
                           then mini := v
                           else if h[v] < h[mini]
```

```
                                        then mini := v;
                          end
                     else galima := false;
      if galima
         then h[u] := h[mini] + 1
   end; { pakelti }

   function min (x, y: integer): integer;
   begin
      if x < y
         then min := x
         else min := y
   end; { min }

   procedure paleisti (u: integer;
                       var s: lentelė2; { srautai }
                       var bakas: lentelė1);
   { paleidžia viršūnės u bake esančius srautus }
   { vartojami globalūs kintamieji m, h, sv, e, uodega }
      var v, kiekis: integer;
   begin
      for v := 1 to m do
        if (h[v] + 1 = h[u]) and (sv[u, v] - s[u, v] > 0)
           { jei v žemesnė ir į ją dar galima paleisti srautus }
           then begin
                  kiekis := min (bakas[u], sv[u, v] - s[u, v]);
                  bakas[u] := bakas[u] - kiekis;
                  s[u, v] := s[u, v] + kiekis;
                  s[v, u] := -s[u, v];
                  bakas[v] := bakas[v] + kiekis;
                  if (bakas[v] > 0) and (v <> 1)
                  then if (v <> m) and (not (v in virš))
                       then įtraukti_į_eilę (v, virš, e,
uodega);
               end;
      end; { paleisti }

begin
   inicializuoti (h, bakas, s);
   galva := 1; uodega := 1; { eilė tuščia }
 virš := [];
   h[1] := m; { pradinės viršūnės aukštis }
   { paleidžiame srautus iš pradinės viršūnės }
   for u := 1 to m do
      if sv[1, u] > 0
         then begin
```

```
            s[1, u] :=  sv[1, u];
            s[u, 1] := -sv[1, u];
            bakas[u] := sv[1, u];
            if u <> m
               then įtraukti_į_eilę (u, virš, e, uodega);
         end;
  while virš <> [] do
    begin
      u := e[galva];
      pašalinti_iš_eilės (galva, virš);
      while bakas[u] > 0 do
        begin
          pakelti (u, h);
          paleisti (u, s, bakas)
        end;
    end;
end;  { max_srautai }
```