these notes are from

Chapter 7

Linear Programming and Reductions

in the class text

`http://algorithmics.lsi.upc.edu/docs/Dasgupta-Papadimitriou-Vazirani.pdf`

7.1.1 example: chocolate production

```
production * x1 boxes/day of type P  profit 1/box
           * x2       "          N         6

demand     * P       at most 200 boxes/day
           * N               300
           * P+N             400

      maximize x1 + 6x2

      subject to x1 <= 200
                 x2 <= 300
                 x1 + x2 <= 400
                 x1, x2 >= 0
                 x1, x2  rational
......................................

vector/matrix notation

c  | 1 |   x  | x1 |    A  | 1  0 |    b | 200 |
   | 6 |      | x2 |       | 0  1 |      | 300 |
                           | 1  1 |      | 400 |

       maximize    c^T x
       such that   A x <= b
                     x >= 0
                     x rational
```

7.1.1 example: chocolate production

```
production * x1 boxes/day of type P  profit 1/box
           * x2       "          N         6

demand     * P       at most 200 boxes/day
           * N               300
           * P+N             400

      maximize x1 + 6x2

      subject to x1 <= 200
                 x2 <= 300
                 x1 + x2 <= 400
                 x1, x2 >= 0
                 x1, x2  rational

solution: x1 = 100, x2 = 300, objective = 1900

proof of optimality?

x2      <= 300         5*x2  <= 5*300 = 1500
x1 + x2 <= 400     x1 +  x2  <=          400
                   -------------------------
              sum  x1 + 6x2  <=         1900

magic LP trick :)
```

expanded problem: augment c, x, A, b

```
c |  1 |    x  | x1 |     A  | 1 0 0 |     b | 200 |
  |  6 |       | x2 |        | 0 1 0 |       | 300 |
  | 13 |       | x3 |        | 1 1 1 |       | 400 |
                             | 0 1 3 |       | 600 |
```

```
      max         c^T x
      s.t.          A x <= b
                      x >= 0
```

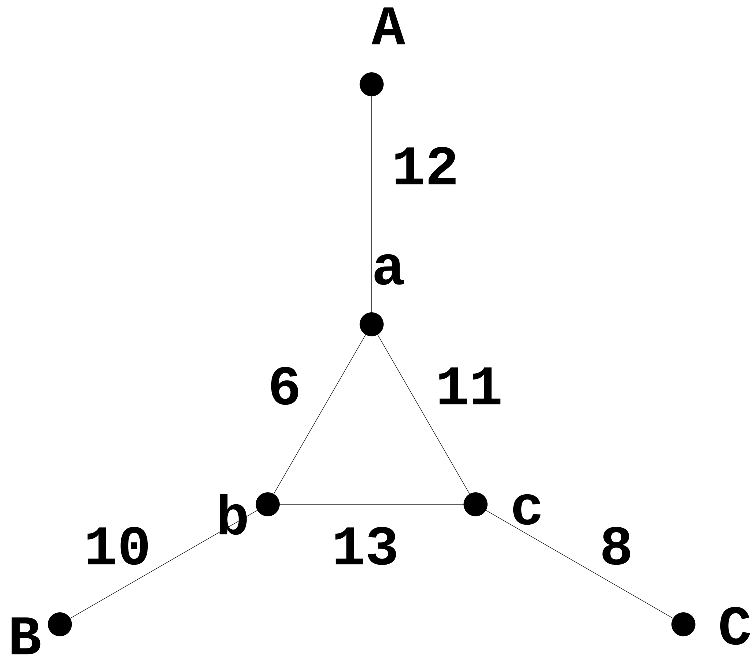
- **evaluate at** `https://sagecell.sagemath.org/`

```
p = MixedIntegerLinearProgram()
v = p.new_variable(real=True, nonnegative=True)
c, x, y, z = v["c"], v["x"], v["y"], v["z"]
p.set_objective(c)
p.add_constraint(c == x + 6*y + 13*z)
p.add_constraint(x <= 200)
p.add_constraint(y <= 300)
p.add_constraint(x + y + z   <= 400)
p.add_constraint(    y + 3*z <= 600)
round(p.solve(), 2)
p.get_values(c,x,y,z)
```

- **you should get this output**

```
[3100.0, 0.0, 300.0, 100.0]
```

7.1.3 example: optimum bandwidth allocation



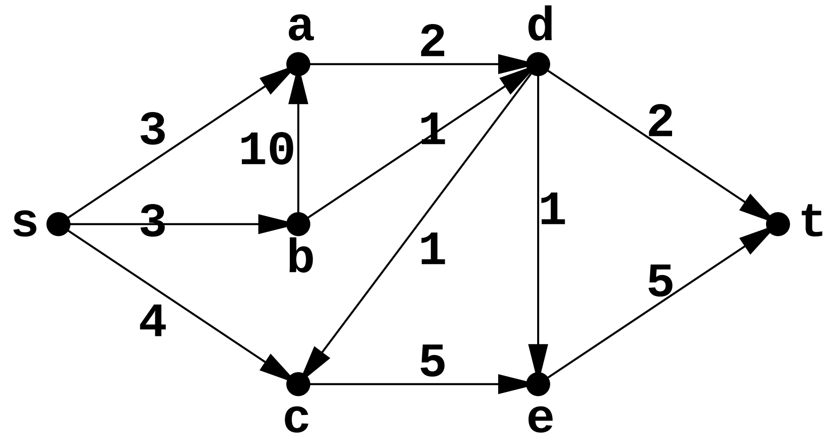
* users A B C
* between each pair users, at least 2 units bandwidth
* profit A-B $3/unit bandwidth, B-C $2/ub, A-C $4/ub
* inter-user communication routing:
  direct (3 hops) or indirect (4 hops)

```
max 3xAB + 3x'AB + 2xBC + 2x'BC + 4xAC + 4x'AC

s.t. xAB + x'AB + xBC + x'BC                 <= 10  edge (b,B)
     xAB + x'AB               + xAC + x'AC <= 12  edge (a,A)
                 xBC + x'BC + xAC + x'AC <=  8  edge (c,C)
     xAB +             x'BC +        x'AC <=  6  edge (a,b)
           x'AB + xBC +              x'AC <= 13  edge (b,c)
           x'AB +      x'BC + xAC         <= 11  edge (a,c)
     xAB + x'AB                            >= 2
                 xBC + x'BC                >= 2
                              xAC + x'AC   >= 2
     xAB,  x'AB,  xBC,  x'BC,  xAC,  x'AC >= 0
```

7.1.4 variants, standard inequality form

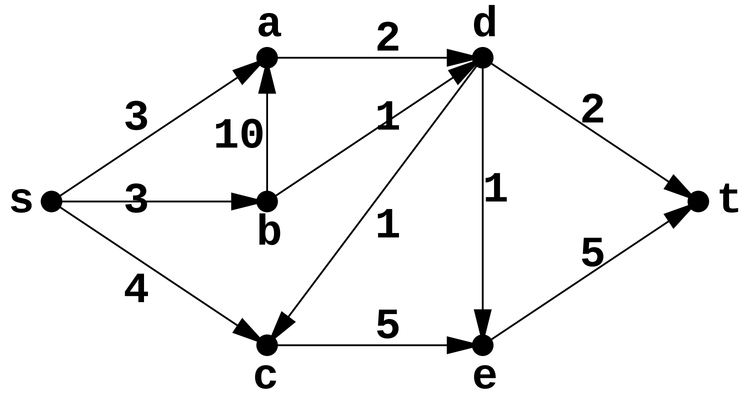
```
* max cx ...                    can be max or min
* s.t. 3x1 - 2x9 <= 3                  <=, >=, =
*      x1 >= 0                         >= 0, <= 0, unrestricted

 -------------------------------------------
|    less-equal form:    |   standard form: |
|      max cx            |   max cx         |
|      s.t. Ax <= b      |   s.t. Ax =  b   |
|            x >= 0      |         x >= 0   |
 -------------------------------------------
e.g. put this into                       min 3x1 - 4x2
  standard less-equal form               s.t. x1 + x2  = 7
                                              x1 - x3 >= 8
                                              x1,  x3  >= 0

max -3x1 + 4x2
s.t.  x1 +  x2      <=  7
     -x1 -  x2      <= -7
     -x1 +      x3  <= -8
      x1,       x3  >= 0

max -3x1 + 4(x4-x5)
s.t.  x1 +  (x4-x5) <=  7
     -x1 -  (x4-x5) <= -7
     -x1 +  x3      <= -8
      x1, x3, x4, x5 >= 0

max -3x1  +     4x4  - 4x5
s.t.  x1  +      x4  -  x5   <=  7
     -x1       - x4  +  x5   <= -7
     -x1  + x3               <= -8
      x1,   x3,   x4,   x5   >=  0
```

7.2.2 max flow

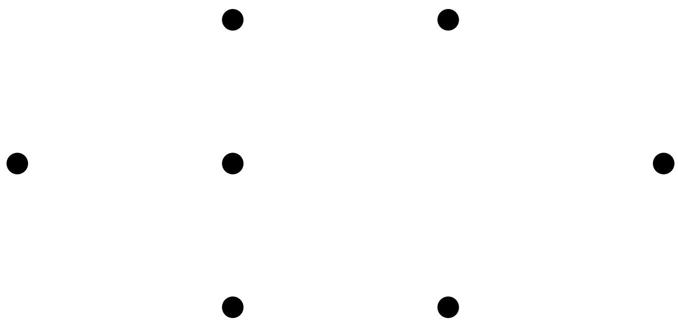



```
network input
  * source s, terminus t
  * on each arc, capacity

flow
  * on each arc, non-negative flow f(arc) (e.g. 73 litre/s)
      that does not exceed capacity(arc)
  * at each node except for source and terminus,
       *  in-flow(node)  [sum, over  in-arcs j, of f(j)]   =
         out-flow(node)  [sum, over out-arcs k, of f(k)]

flow volume
  * out-flow(source), or in-flow(terminus)

problem:
  * given network, find a max volume flow
```

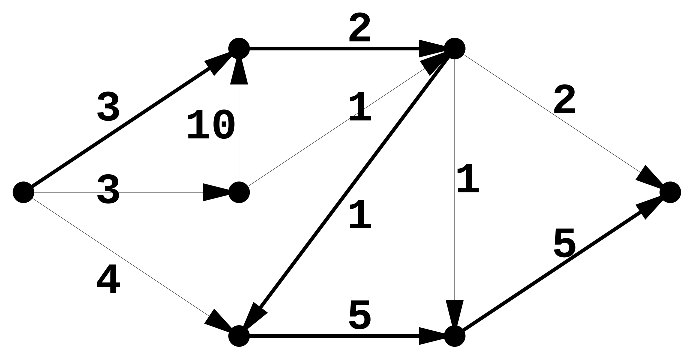
network



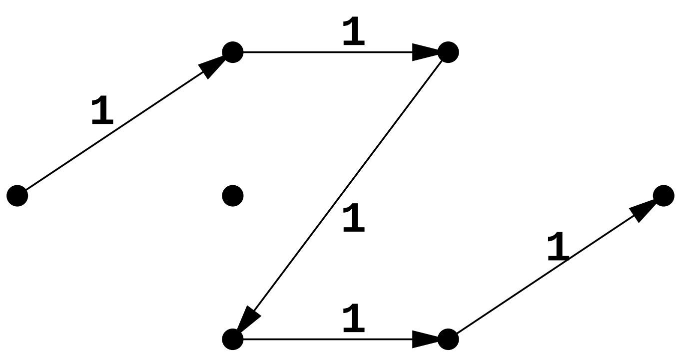
flow 0



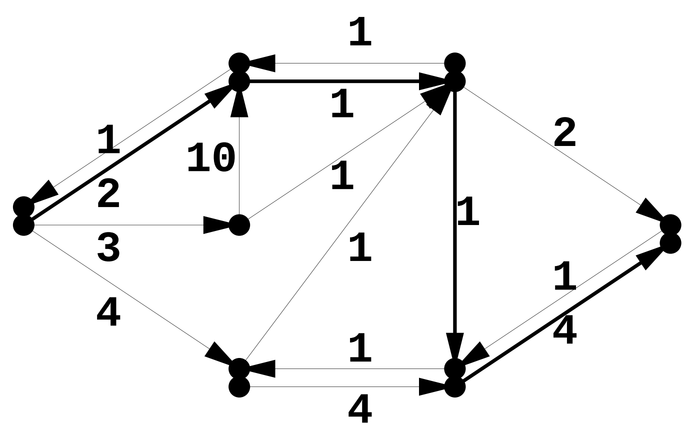
residual 0



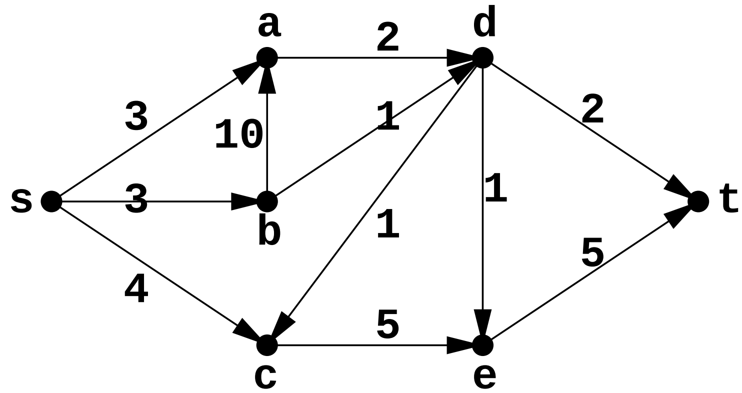
flow 1



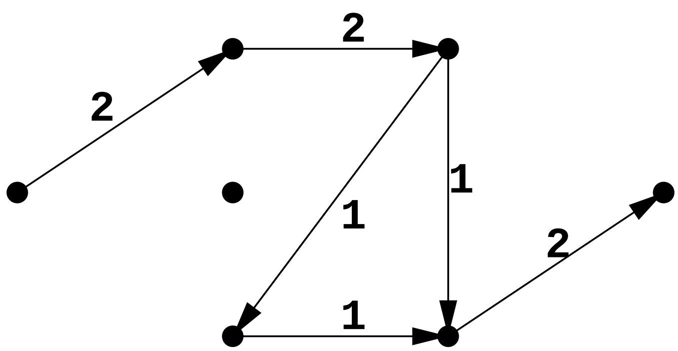
residual 1



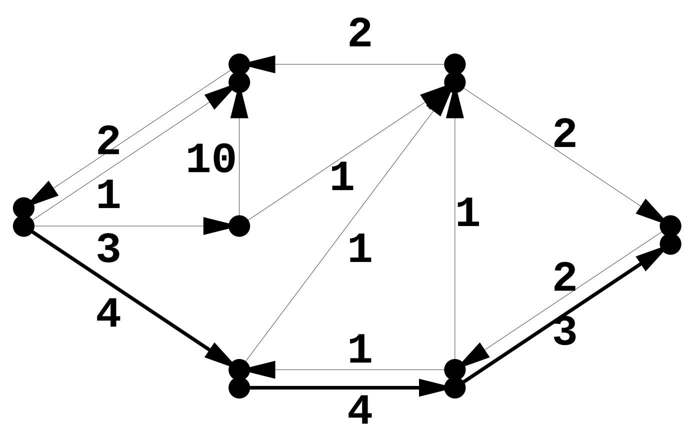
network



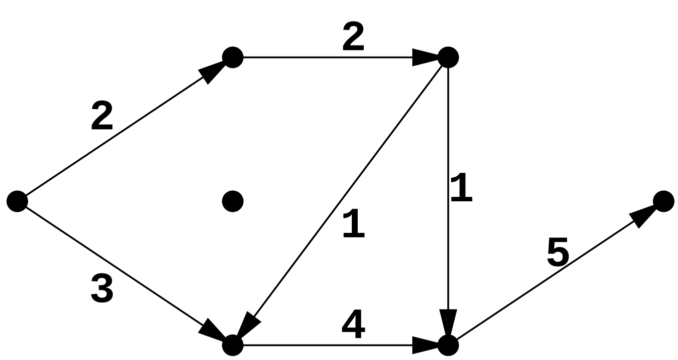
flow 2



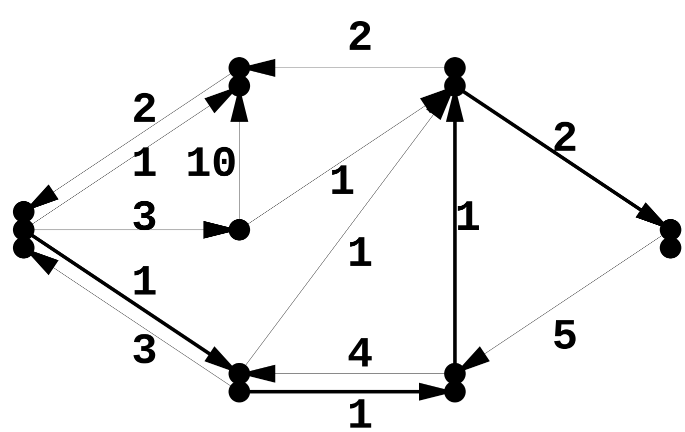
residual 2



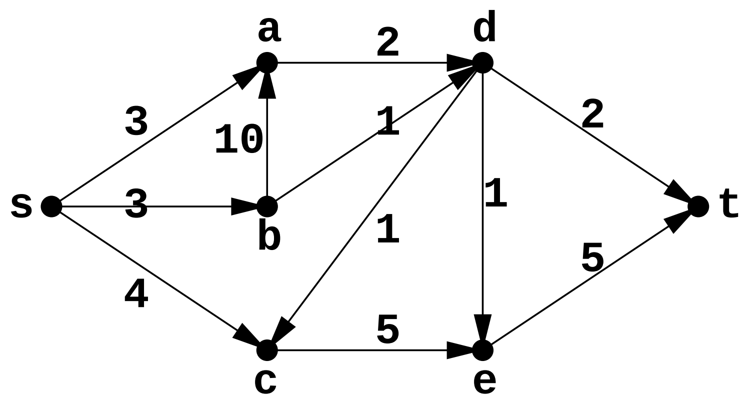
flow 3



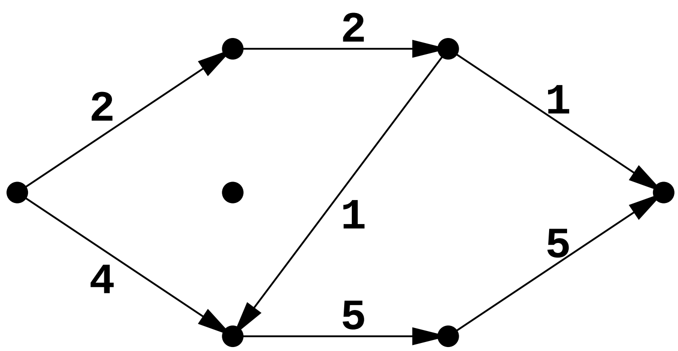
residual 3



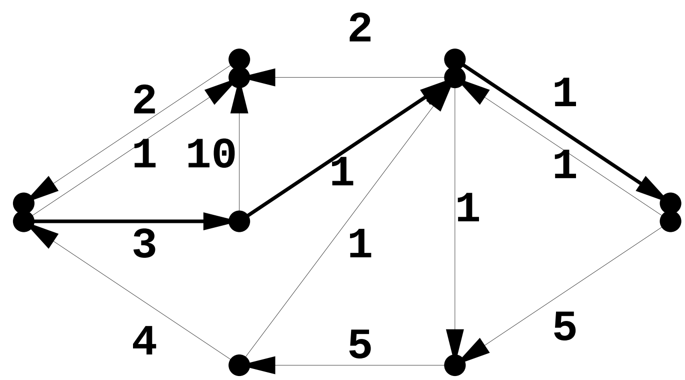
network



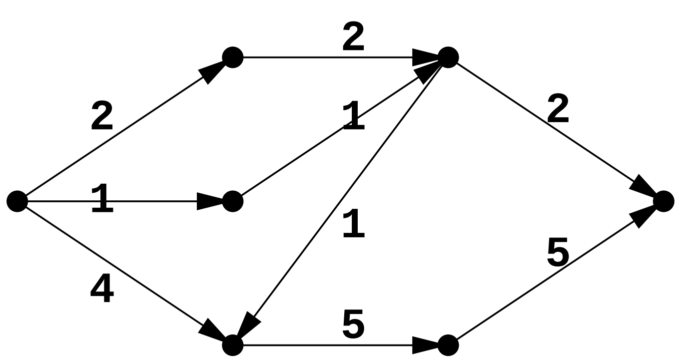
flow 4



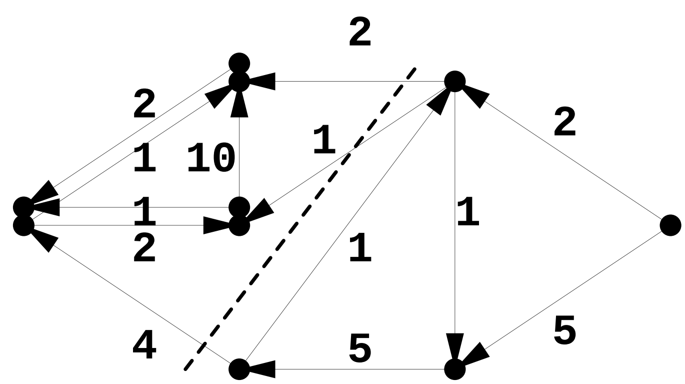
residual 4



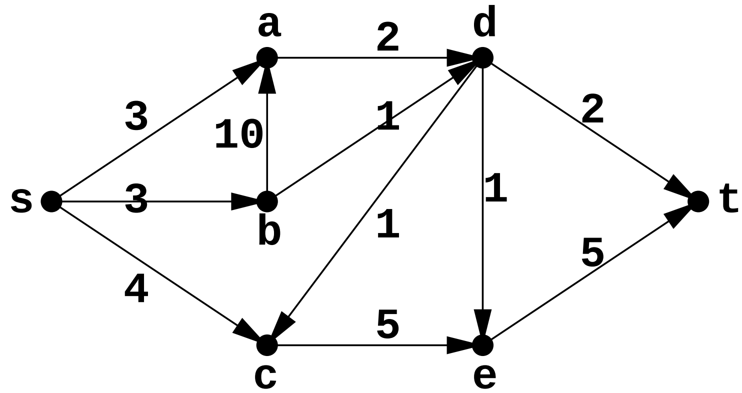
flow 5



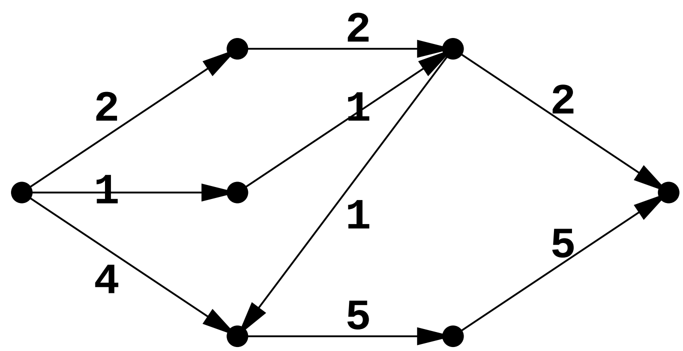
residual 5



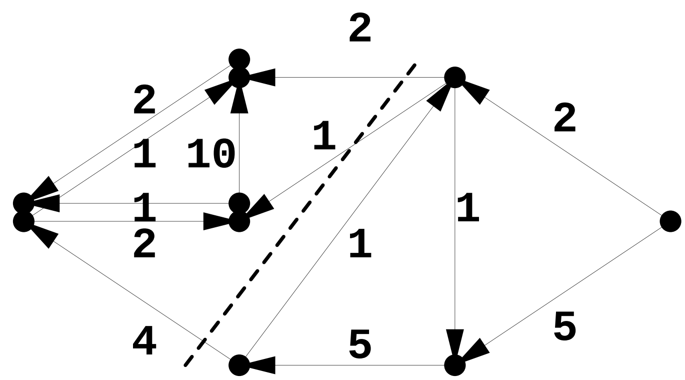
network



flow 5



residual 5




```
min cut?

in final residual network,

   { nodes reachable from s}     { other nodes }

here
         { s, a, b }   { c, d, e, t }

why?
```

7.2.2 max flow transforms to LP

```
variables sa sb sc ad ba bd ce dc de dt et

max    sa + sb + sc          flow out of source
                             flow conservation
s.t.   sa + ba = ad              node a
       sb = ba + bd                   b
       sc + dc = ce                   c
       ad + bd = dc + de + dt         d
       ce + de = et                   e
                                 capacities
       sa <= 3
       sb <= 3
       sc <= c
       ad <= 2
       ba <= 10
       bd <= 1
       ce <= 5
       dc <= 1
       de <= 1
       dt <= 2
       et <= 5

       sa,sb,sc,ad,ba,bd,ce,dc,de,dt,et >= 0
```

conclusion: max flow polytime transforms into LP

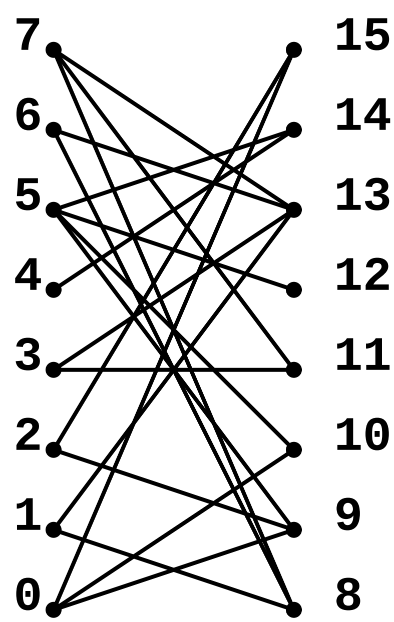
max flow integrality observation:

if all capacities are integer, then there exists

an all-integer max flow   (why?)

7.3 max bipartite matching

```
instance:  bipartite graph G = (V0, V1, E)
query:     what is the size of a maximum matching in G?
```



7.3 max bipartite matching

```
instance:  bipartite graph G = (V0, V1, E)
query:     what is the size of a maximum matching in G?
```