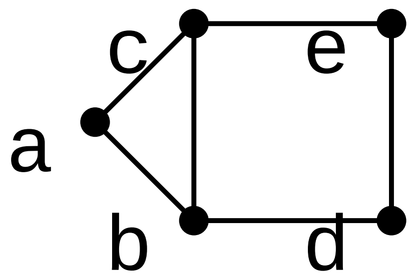
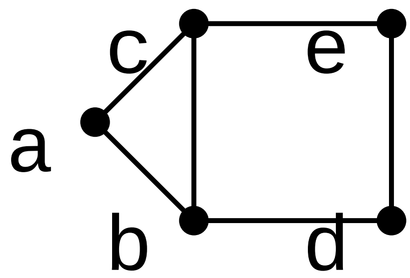
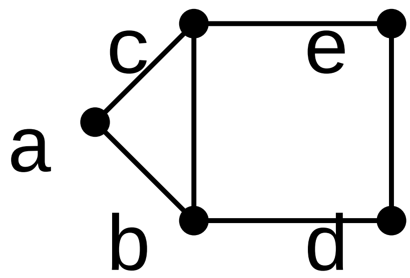


from IS to IP to LP



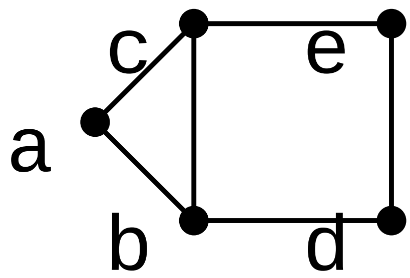
find max IS?

```
* NP-complete  :(
* restrict instances ?
* trees?    k-IS in P  :)
- method 1:     dynamic programming
- forest has isolated node or leaf (1 nbr)

- method 2: transform IS-to-LP
- not necessarily answer-preserving:
  can have non-integer solution :(
- trees ?
  * integer solution guaranteed :)
  * size of LP polynomial in size of graph,
- perfect graphs ?
  * integer solution guaranteed :)
```



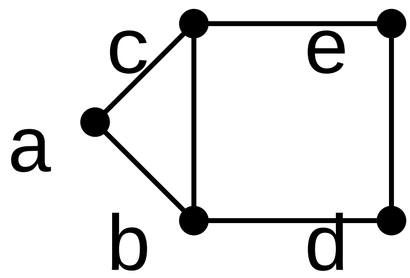
find max IS? formulate as IP

```
--- idea: edge constraints ---

max  a + b + c + d + e    why?

s.t.

     a + b  <= 1          why?
     a + c  <= 1
     b + c  <= 1
     b + d  <= 1
     c + e  <= 1
     d + e  <= 1

     a, b, c, d, e   each in {0, 1}
```

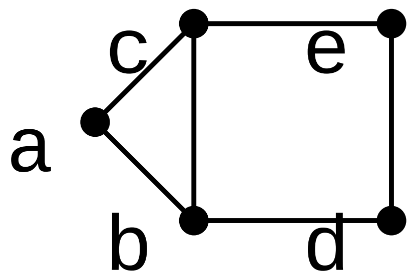


rewrite IP matrix form

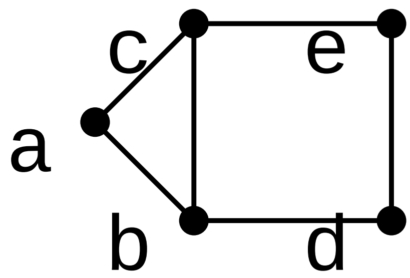
```
max [1 1 1 1 1] x  s.t.

  |1 1 . . .|        |1|
  |1 . 1 . .|        |1|
  |. 1 1 . .| X <=   |1|
  |. 1 . 1 .|        |1|
  |. . 1 . 1|        |1|
  |. . . 1 1|        |1|

                     |0|
                     |0|
              X >=   |0|
                     |0|
                     |0|
```




```
LP relaxation solution

|.5 .5 .5 .5 .5|  z = 2.5

opt IP solution   z = 2        :(
```




```
--- new idea: maximal clique constraints ---

  max [1 1 1 1 1] x  s.t.

    |1 1 1 . .|        |1| {a,b,c} max'l clique
    |. 1 . 1 .| X <=   |1|   {b,d}        "
    |. . 1 . 1|        |1|   {c,e}        "
    |. . . 1 1|        |1|   {d,e}        "

                       |0|
                       |0|
                  X >= |0|
                       |0|
                       |0|
```

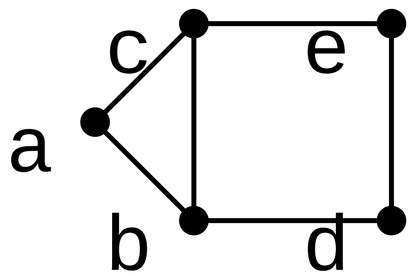


LP relaxation solution

|0 .5 .5 .5 .5| optimal, z = 2   :) :(

another solution

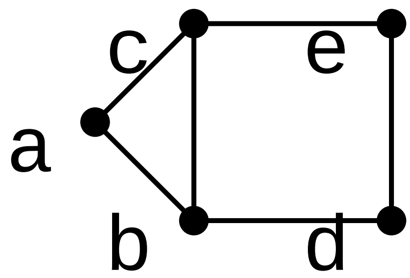
|1 0 0 0 1| optimal, z = 2   :) :)



graph-theoretic meaning of the dual?

minimize sum of clique weights, such that

each node is in at least one clique (clique cover)



why maximal clique formulation? recall perfect graph:

no induced odd C_k>=5 in graph or complement

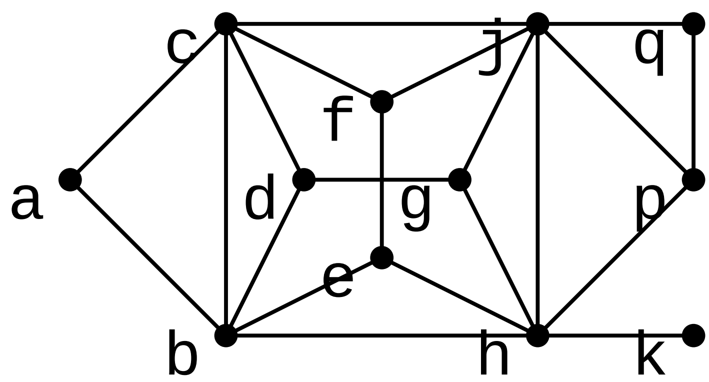
input graph perfect? then

* each feasible region extreme point all-integer

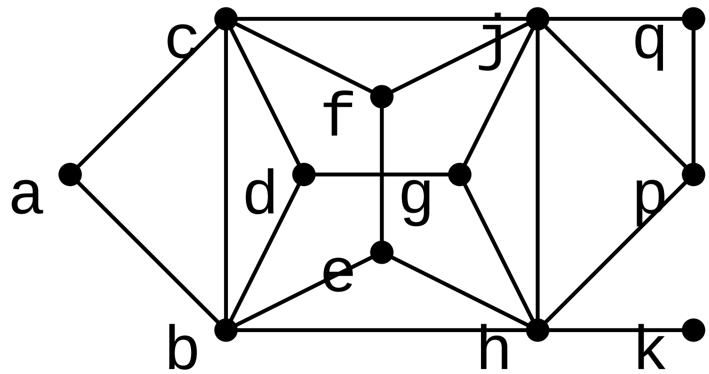
* each max'l-clique form'n LP relax'n solution all-in

* for perfect graphs, LP relaxation solves IP

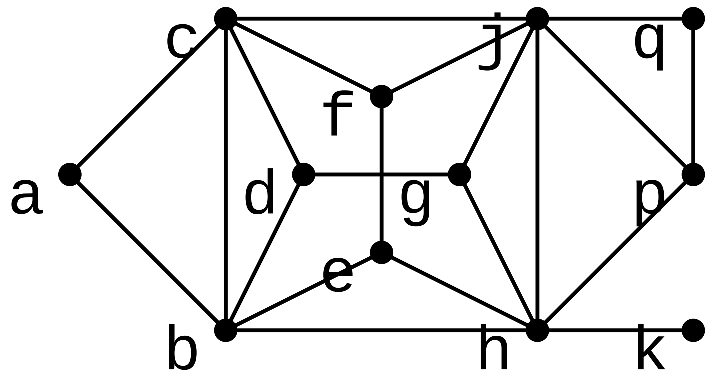
* warning: number maximal cliques can be exp'l in num




```
nodes x_1 .. x_12

max'l cliques  y_1 .. y_10

abc bcd beh cfj  dg  ef ghj  hk hjp jpq

max [1 1 .. 1] X s.t. AX <= [1 1 .. 1]^t
        a b c d e f g h j k p q
   A   |1 1 1                  |
       |  1 1 1                |
       |  1     1     1        |
       |    1     1     1      |
       |      1     1          |
       |        1 1            |
       |            1 1 1      |
       |              1   1    |
       |              1 1   1  |
       |                1   1 1|
```

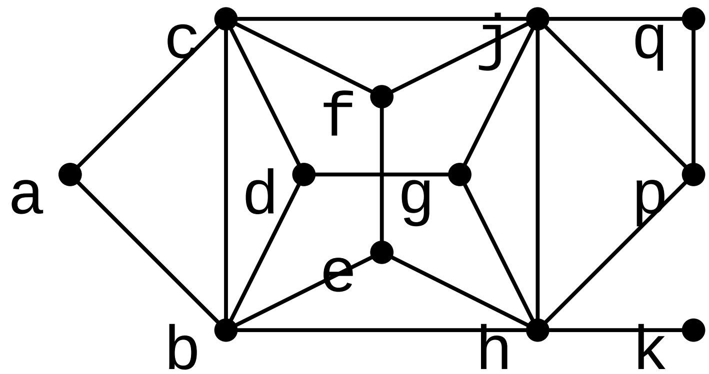



```
P = MixedIntegerLinearProgram()
v = P.new_variable(real=True, nonnegative=False)
a,b,c,d,e,f,g,h,j,k,p,q,z=v['a'],v['b'],v['c'],v['d'],v
P.set_objective(z)
P.add_constraint(z <= a+b+c+d+e+f+g+h+j+k+p+q)
P.add_constraint(a >= 0)
P.add_constraint(b >= 0)
...
P.add_constraint(q >= 0)
P.add_constraint(a+b+c <= 1)
P.add_constraint(b+c+d <= 1)
P.add_constraint(b+e+h <= 1)
P.add_constraint(c+f+j <= 1)
P.add_constraint(d+g   <= 1)
P.add_constraint(e+f   <= 1)
P.add_constraint(g+h+j <= 1)
P.add_constraint(h+k   <= 1)
P.add_constraint(h+j+p <= 1)
P.add_constraint(j+p+q <= 1)
P.solve()
P.get_values(z,a,b,c,d,e,f,g,h,j,k,p,q)
```

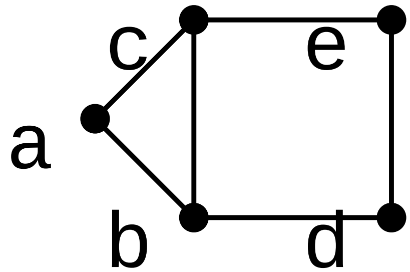


evaluate `https://sagecell.sagemath.org/`

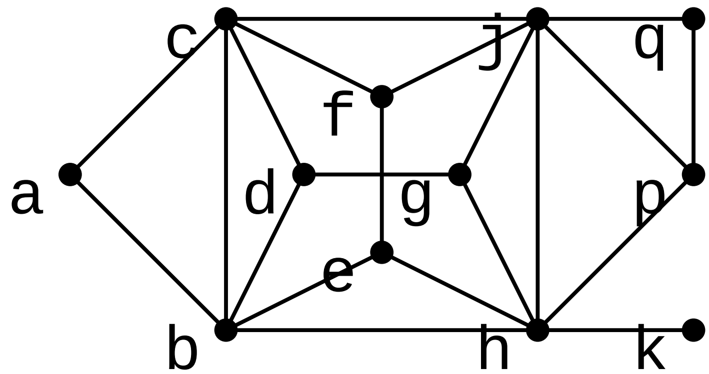
```
solution

[5.0, 1.0,-0.0,-0.0, 1.0, 1.0,-0.0,

     -0.0,-0.0, 1.0, 1.0,-0.0,-0.0]

  z    a    b    c    d    e    f

       g    h    j    k    p    q

max IS = {a, d, e, j, k}

can we easily verify this solution?  dual !
```



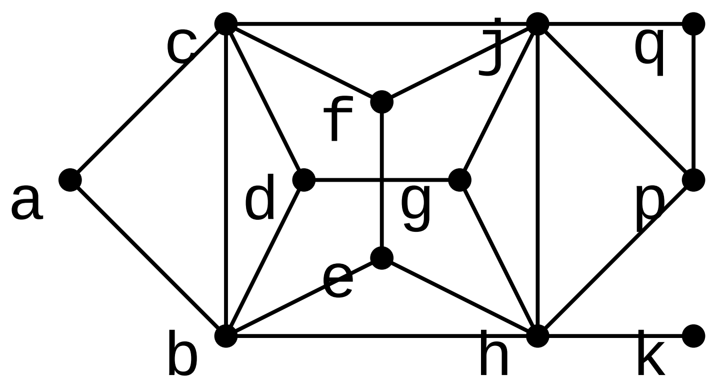

```
primal:   max C^t X s.t.    AX <= B

dual:     min B^t Y s.t. Y^t A >= C^t

dual (max form):

         max -B^t Y s.t. Y^t A >= C^t
```




```
primal      max   [1 1 1 1 1] X
            st    |1 1 1 . .| X  <= |1|
                  |. 1 . 1 .|       |1|
                  |. . 1 . 1|       |1|
                  |. . . 1 1|       |1|
all-integer?
* {x_j} (weighted node set) hits each clique <=1 time
* union{x_j} forms IS

dual        min     [1 1 1 1] Y
           st Y^t |1 1 1 . .|    >= [1 1 1 1 1]
                  |. 1 . 1 .|
                  |. . 1 . 1|
                  |. . . 1 1|
all-integer?
* {y_k} (weighted clq set) hits each node >=1 time
* union{y_k} forms clique cover (CC)

all-integer solution? dual. thm: same size IS, CC :)
```




```
D = MixedIntegerLinearProgram()
v = D.new_variable(real=True, nonnegative=False)
a,b,c,d,e,f,g,h,j,k,z=v['a'],v['b'],v['c'],v['d'],v['e'
D.set_objective(z)
D.add_constraint(z <=-a-b-c-d-e-f-g-h-j-k)
D.add_constraint(a >= 0)
    ..
D.add_constraint(k >= 0)
D.add_constraint(a     >= 1)
D.add_constraint(a+b+c >= 1)
D.add_constraint(a+b+d >= 1)
D.add_constraint(b+e   >= 1)
D.add_constraint(c+f   >= 1)
D.add_constraint(d+f   >= 1)
D.add_constraint(e+g   >= 1)
D.add_constraint(c+g+h+j >= 1)
D.add_constraint(d+g+j+k >= 1)
D.add_constraint(h     >= 1)
D.add_constraint(j+k   >= 1)
D.add_constraint(k     >= 1)
..
```



evaluate dual `https://sagecell.sagemath.org/`

dual solution

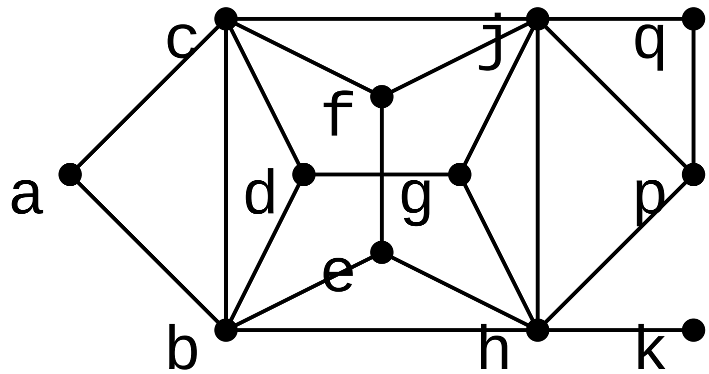
```
[-5.0,1.0,-0.0,-0.0,-0.0,1.0,1.0,-0.0,1.0,-0.0,1.0]
       *                  *   *         *        *
   z   a    b   c    d    e   f    g    h    j   k
```

min clique cover = {abc, dg, ef, hk, jpq}

we have found

* IS hitting every maximal clique,   size 5
* set of cliques hitting every node, size 5
  - so every IS set has size <= 5

clique cover easily verifies IS-is-max :)



for this graph,

how did we know that IS-to-LP solution

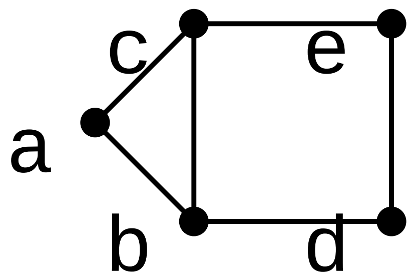
would be all-integer?

answer:

graph is in a class called *perfect*

perfect graphs have this property:

solution to IS-to-LP always all-int :)



review: to solve IS on perfect graphs, what happens if we formulate as

each node in S hits at most one node of every edge instead of

each node in S hits at most one node of every maximal clique ?

answer:

with edge constraints, feasible polytope can have extreme points with fractional values :(

example:

we solved edge constraint relaxation for above graph, solution |.5 .5 .5 .5 .5| :(