**Binary Matroids Without K5-minors**

Table 1: Gordon Royle’s computations

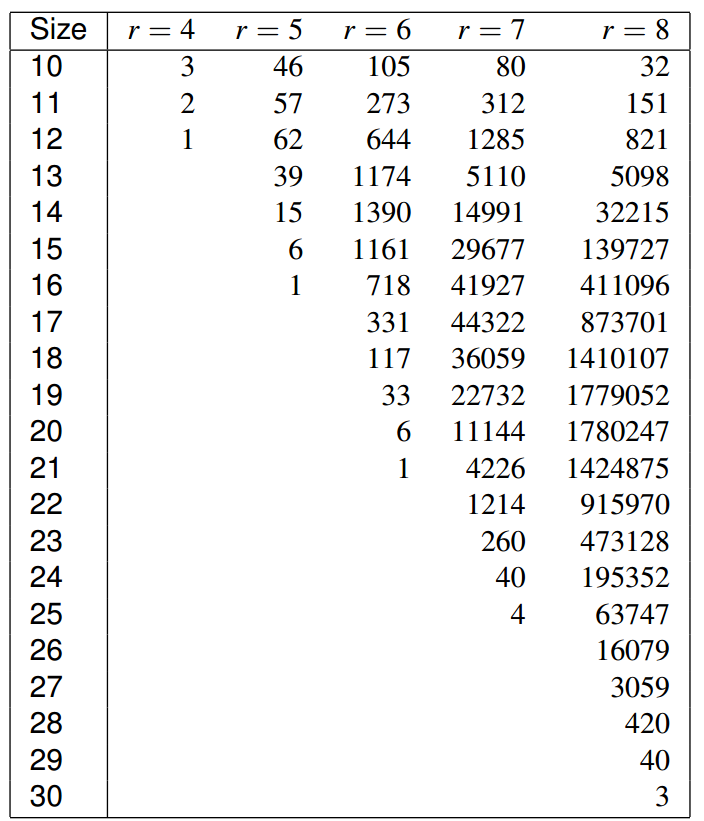


Table 2: Sage computation

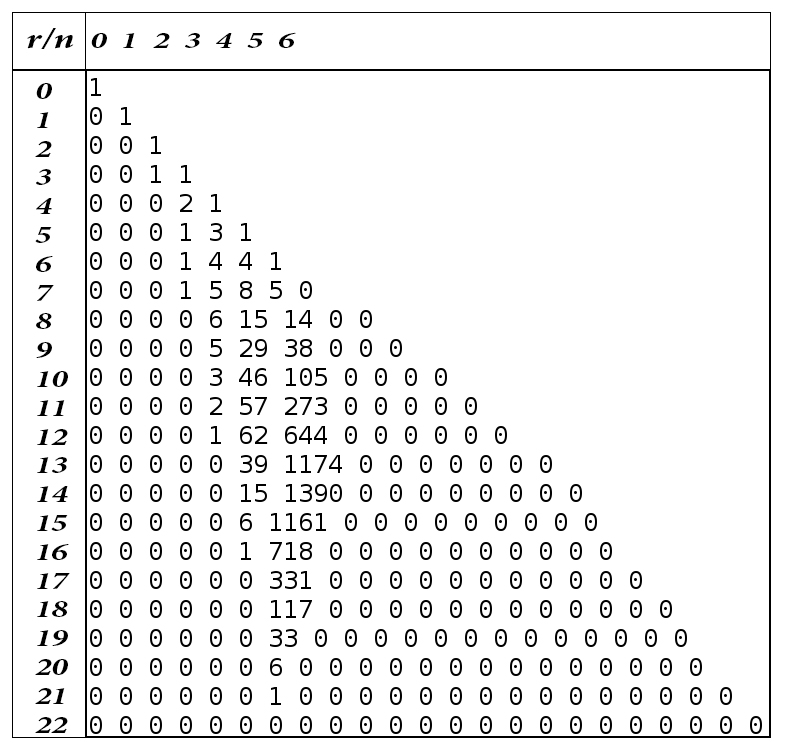
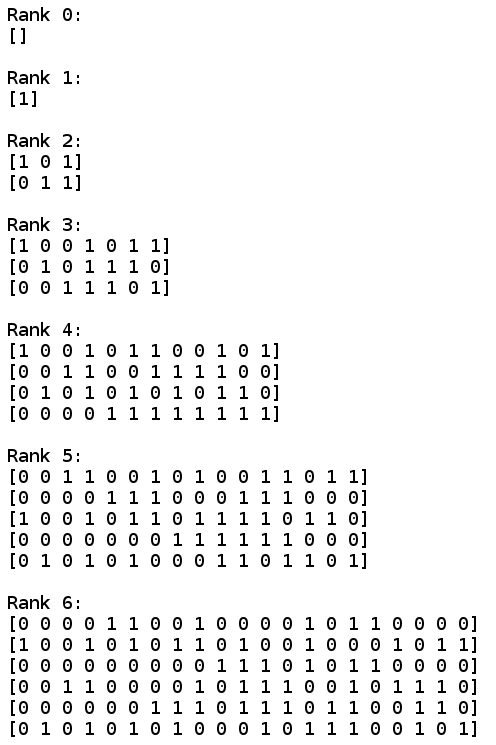


Fig 1: Largest matroids with respect to rank



Source Code:

#Creates all Matroids excluding K5 minor up to and including rank 6

from sage.matroids.advanced import \*

K5 = matroids.CompleteGraphic(5)

K5 = Matroid(ring=GF(2), reduced\_matrix=K5.representation(reduced=True, labels=False))

def all\_matroids(field, maxsize, maxrank):

D = {}

D[(0,0)] = [Matroid(field=field, groundset=[], matrix=[[]])]

for n in range(1, maxsize + 1):

e = n

D[(0,n)] = []

for r in range(n):

# extend:

for M in D[(r, n - 1 - r)]:

if M.rank() < maxrank + 1:

S = [N for N in M.linear\_extensions(element=e, simple=True) if e

in list(N.\_weak\_partition())[-1]]

for N in S:

if not any(N.is\_field\_isomorphic(NN) for NN in D[(r, n - r)])

and not N.has\_minor(K5):

D[(r, n - r)].append(N)

# coextend by coloop:

D[(r + 1, n - 1 - r)] = [M.linear\_coextension(e, cochain={f:0 for f in

M.groundset()}) for M in D[(r, n - 1 - r)] if M.rank() < maxrank]

return D

K5min = all\_matroids(GF(2), 22, 6)

for n in range(23):

s = ''

for r in range(n+1):

s = s + ' ' + str(len(K5min[(r,n-r)]))

print s

#Prints out the largest matroid of rank n without K5 as a minor

for x in K5min[(0, 0)]:

x = x.simplify() #removes the row of zeros that is left behind from algorithm

print "Rank 0:"

print Matrix(x)

print

for x in K5min[(1, 0)]:

x = x.simplify()

print "Rank 1:"

print Matrix(x)

print

for x in K5min[(2, 1)]:

x = x.simplify()

print "Rank 2:"

print Matrix(x)

print

for x in K5min[(3, 4)]:

x = x.simplify()

print "Rank 3:"

print Matrix(x)

print

for x in K5min[(4, 8)]:

x = x.simplify()

print "Rank 4:"

print Matrix(x)

print

for x in K5min[(5, 11)]:

x = x.simplify()

print "Rank 5:"

print Matrix(x)

print

for x in K5min[(6, 15)]:

x = x.simplify()

print "Rank 6:"

print Matrix(x)

print

Fig 2: Storing the matroids

(0, 0)

(0, 1) (1, 0)

(0, 2) (1, 1) (2, 0)

(0, 3) (1, 2) (2, 1) (3, 0)

(0, 4) (1, 3) (2, 2) (3, 1) (4, 0)

(0, 5) (1, 4) (2, 2) (3, 2) (4, 1) (5, 0)

(0, 6) (1, 5) (2, 3) (3, 3) (4, 2) (5, 1) (6, 0)

(0, 7) (1, 6) (2, 4) (3, 4) (4, 3) (5, 2) (6, 1) (7, 0)

(0, 8) (1, 7) (2, 5) (3, 5) (4, 4) (5, 3) (6, 2) (7, 1) (8, 0)

(0, 9) (1, 8) (2, 6) (3, 6) (4, 5) (5, 4) (6, 3) (7, 2) (8, 1) (9, 0)

The matroids are stored in a large dictionary of lists using keys of the form (r, k) where r = rank and k = n – r, where n = number of elements (columns) of the matroid. If you wanted access to all matroids of rank 4 and 9 elements, you would use “dictionary\_name[(4, 5)]”.