rank_3_ex_minors_for_gf_5

Ben Clark

7/31/2017

```
import sage.matroids
from sage.matroids.advanced import *

def rank_3_matroids(n):
    '''
    Generates the simple isomorph-free sets of rank 3, simple \
   matroids up to size n <= 12
    '''
    M = matroids.Uniform(3,3)

    E = {3:[M],4:[],5:[],6:[],7:[],8:[],9:[],10:[],11:[],12:[]}

    for i in range(3,n):
        for M in E[i]:
            for N in M.extensions():
                if N.is_simple():
                    nIso = True
                    for P in E[i+1]:
                        if P.is_isomorphic(N):
                            nIso = False
                    if nIso and not N.has_line_minor(7):
                        E[i+1].append(N)
    return E

cat = rank_3_matroids(12)
print("size: number")
for i in range(8,13):
    print('%d: %d' %(i, len(cat[i])))
```

```
size: number
8: 44
9: 149
10: 492
11: 1302
12: 2279
```

```
#Generate the GF(5)-representable matroids and eliminate as excluded\
    minors

def rank_3_gf5(n):
    '''
    generates simple isomorph-free sets of GF(5)-representable \
   matroids up to size n <= 12
    '''
    M = Matroid(field=GF(5), matrix=[[1,0,0],[0,1,0],[0,0,1]])

    E = {3:[M], 4:[], 5:[], 6:[], 7:[], 8:[], 9:[], 10:[], 11:[], \
   12:[]}

    for i in range(3,n):
        for M in E[i]:
            for N in M.linear_extensions(simple=True):
                nIso = True
                for P in E[i+1]:
                    if P.is_isomorphic(N):
                        nIso = False
                if nIso:
                    E[i+1].append(N)
    return E

cat2 = rank_3_gf5(12)
print('size: number')
for i in range(8,13):
    print('%d : %d' %(i,len(cat2[i])))
```

size: number
8 : 34
9 : 82
10 : 168
11 : 296
12 : 476

```
#excluded minor candidates
ex = {3:[],4:[],5:[],6:[],7:[],8:[],9:[],10:[],11:[],12:[]}
for i in range(3,13):
    for M in cat[i]:
        rep = False
        for N in cat2[i]:
            if M.is_isomorphic(N):
                rep = True
        if not rep:
            ex[i].append(M)
```

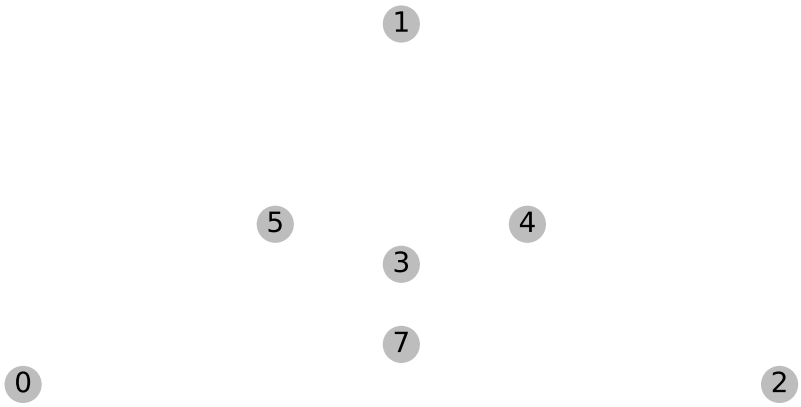
```
print('size: number')
for i in range(8,13):
    print('%d : %d' %(i,len(ex[i])))
```

size: number
8 : 10
9 : 67
10 : 324
11 : 1006
12 : 1803

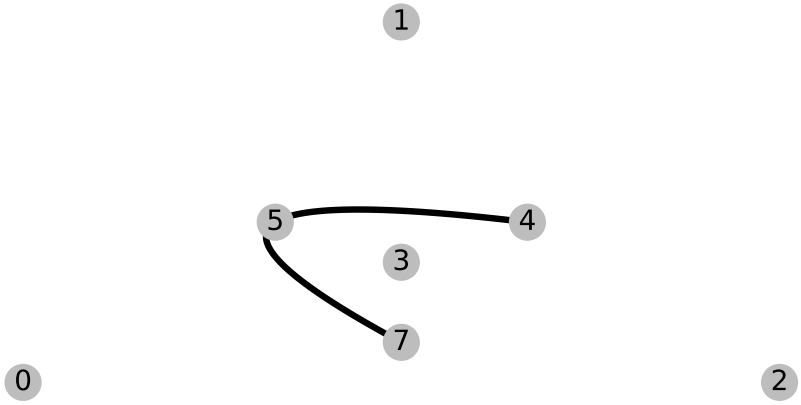
```
#Eliminate those candidate excluded-minors that have a single-\
   element deletion that is GF(5)-representable.
ex2 = {3:[],4:[],5:[],6:[],7:[],8:[],9:[],10:[],11:[],12:[]}
for i in range(7,13):
    for M in ex[i]:
        exm = True
        for e in M.groundset():
            N = M \ e
            for P in ex[i-1]:
                if N.is_isomorphic(P):
                    exm = False
        if exm:
            ex2[i].append(M)


print('size: number')
for i in range(7,13):
    print('%d : %d' %(i,len(ex2[i])))
```

size: number
7 : 5
8 : 2
9 : 9
10 : 1
11 : 1
12 : 0

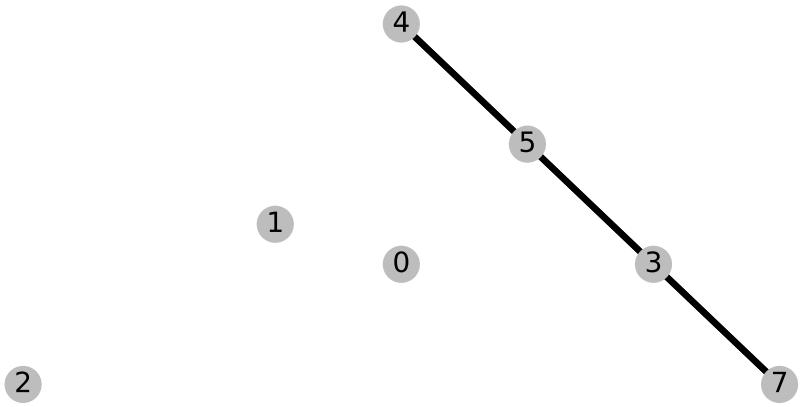
```
ex2[7][0].show()
```



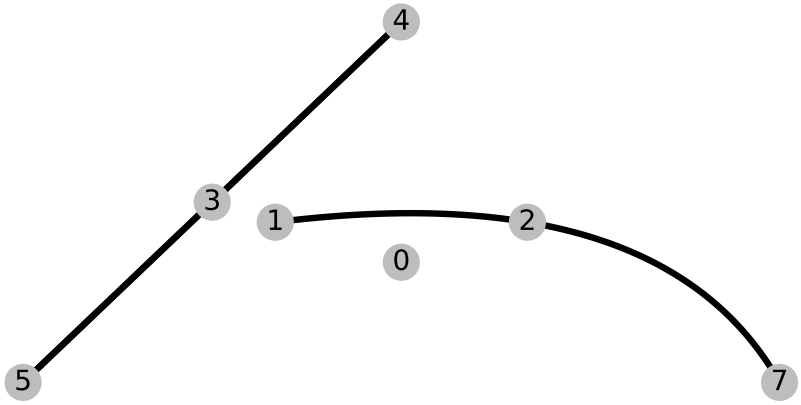

```
ex2[7][1].show()
```



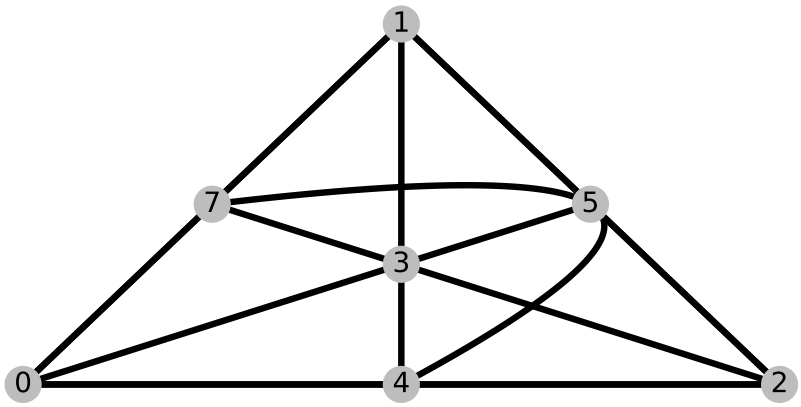

```
ex2[7][2].show()
```




```
ex2[7][3].show()
```




```
ex2[7][4].show()
```




```
#Circuit closures of the 8-element rank-3 excluded minors
ex2[8][0].circuit_closures()
```

{2: set([frozenset([2, '4', '6']), frozenset(['3', '5', '4']), frozenset([1, '5', '6']), frozenset([2, '5', '7']), frozenset([0, '7', '6']), frozenset([1, '4', '7'])]), 3: set([frozenset([0, 1, 2, '3', '5', '4', '7', '6'])])}

```
ex2[8][1].circuit_closures()
```

{2: set([frozenset(['4', '7', '6']), frozenset([0, '3', '7']), frozenset([0, '5', '4']), frozenset([1, 2, '7']), frozenset([1, '3', '4']), frozenset([1, '5', '6']), frozenset([2, '5', '3']), frozenset([0, 2, '6'])]), 3: set([frozenset([0, 1, 2, '3', '5', '4', '7', '6'])])}

```
#Circuit closures of the 9-element rank-3 excluded minors
ex2[9][0].circuit_closures()
```

{2: set([frozenset(['3', '5', '7']), frozenset([0, '9', 2]), frozenset([2, '4', '7']), frozenset(['9', '3', '4']), frozenset([1, '4', '6']), frozenset([2, '5', '6']), frozenset(['9', 1, '7']), frozenset([0, '7', '6'])]), 3: set([frozenset([0, 1, 2, '3', '5', '4', '7', '6', '9'])])}

```
ex2[9][1].circuit_closures()
```

{2: set([frozenset(['3', '5', '7']), frozenset([0, '9', 2]), frozenset(['9', 1, '3']), frozenset([2, '4', '7']), frozenset([1, '4', '6']), frozenset([2, '5', '6']), frozenset([0, '7', '6']), frozenset(['9', '5', '4'])]), 3: set([frozenset([0, 1, 2, '3', '5', '4', '7', '6', '9'])])}

```
ex2[9][2].circuit_closures()
```

{2: set([frozenset(['3', '5', '7']), frozenset([2, '4', '7']), frozenset([1, '5', '6']), frozenset([0, '9', '3']), frozenset(['9', 1, 2]), frozenset([0, '4', '6'])]), 3:

set([frozenset([0, 1, 2, '3', '5', '4', '7', '6', '9'])])}

```
ex2[9][3].circuit_closures()
```
{2: set([frozenset(['3', '5', '7']), frozenset([2, '4', '7']), frozenset([1, '5', '6']), frozenset(['9', '7', '6']), frozenset([0, '9', '3']), frozenset(['9', 1, 2]), frozenset([0, '4', '6'])]), 3: set([frozenset([0, 1, 2, '3', '5', '4', '7', '6', '9'])])}

```
ex2[9][4].circuit_closures()
```
{2: set([frozenset(['3', '5', '7']), frozenset([0, '9', 2]), frozenset(['9', 1, '3']), frozenset([2, '4', '7']), frozenset([1, '5', '6']), frozenset(['9', '7', '6']), frozenset([0, '4', '6']), frozenset(['9', '5', '4'])]), 3: set([frozenset([0, 1, 2, '3', '5', '4', '7', '6', '9'])])}

```
ex2[9][5].circuit_closures()
```
{2: set([frozenset(['3', '5', '7']), frozenset([2, '4', '7']), frozenset([1, '5', '6']), frozenset([0, '9', '3']), frozenset(['9', 1, 2]), frozenset([0, '4', '6']), frozenset([2, '3', '6']), frozenset(['9', '5', '4'])]), 3: set([frozenset([0, 1, 2, '3', '5', '4', '7', '6', '9'])])}

```
ex2[9][6].circuit_closures()
```
{2: set([frozenset(['3', '5', '7']), frozenset([2, '4', '7']), frozenset([1, '5', '6']), frozenset(['9', '7', '6']), frozenset([0, '9', '3']), frozenset(['9', 1, 2]), frozenset([0, '4', '6']), frozenset([2, '3', '6']), frozenset(['9', '5', '4'])]), 3: set([frozenset([0, 1, 2, '3', '5', '4', '7', '6', '9'])])}

```
ex2[9][7].circuit_closures()
```
{2: set([frozenset([1, '5', '7']), frozenset([2, '4', '6']), frozenset(['9', 2, '5']), frozenset([2, '3', '7']), frozenset([0, '5', '6']), frozenset([1, '3', '6']), frozenset([0, '9', '3']), frozenset([0, '4', '7']), frozenset(['9', 1, '4'])]), 3: set([frozenset([0, 1, 2, '3', '5', '4', '7', '6', '9'])])}

```
ex2[9][8].circuit_closures()
```
{2: set([frozenset([1, '5', '7']), frozenset([2, '4', '6']), frozenset([2, '3', '7']), frozenset([0, '5', '6']), frozenset([1, '3', '6']), frozenset(['9', '7', '6']), frozenset([0, '9', '3']), frozenset(['9', 1, 2]), frozenset([0, '4', '7']), frozenset(['9', '5', '4'])]), 3: set([frozenset([0, 1, 2, '3', '5', '4', '7', '6', '9'])])}

```
#The 10-element rank-3 excluded minor
ex2[10][0].circuit_closures()
```
{2: set([frozenset([0, '4', '7', '6']), frozenset([0, '8', 2]), frozenset(['8', '3', '4']), frozenset([0, '9', '5']), frozenset([1, '3', '6']), frozenset([1, '5', '7', '8']), frozenset(['9', 2, '3', '7']), frozenset([2, '5', '6']), frozenset(['9', 1, '4'])]), 3: set([frozenset([0, 1, 2, '3', '5', '4', '7', '6', '9', '8'])])}

```
#The 11-element rank-3 excluded minor
```

```
ex2[11][0].circuit_closures()
```

{2: set([frozenset([0, 1, 'A']), frozenset([0, '8', '5']), frozenset(['A', '8', 2, '3', '7']), frozenset([0, '9', '4', '7', '6']), frozenset(['3', '5', '4']), frozenset([1, '3', '6']), frozenset([1, '4', '8']), frozenset(['9', 1, 2]), frozenset([2, '5', '6']), frozenset([1, '5', '7']), frozenset(['9', 'A', '5'])]), 3: set([frozenset([0, 1, 2, '3', 'A', '4', '7', '6', '9', '5', '8'])])}

```
ex2[8][0].show(B=['4', '5', '6'])
```