Conjecture_2-1_Size10

Rupert Li

3/10/2019

```
'''
Restrictions placed on G,S,j:
    G must be finite, connected, undirected
    G must not be a complete graph, an odd cycle graph, or a tree
    G must have vertex connectivity greater than 1 ("irreducible")
Results:
    Passed all sizes at most 7.
    Failed at size 8: 3 different
    Passed size 9
    Size 10: Count reached 100k, no flagged graphs
            As skipping for count takes too long, switched to num. \
   Set progress to 100k for num, despite massive loss.
            Looks like size 10 will take a long time (size 9 had 261\
    thousand, so size 10 is a lot larger...)
Note:
    There are roughly 9.7 million biconnected graphs with 10 \
   vertices.
    Eliminated -d2 condition, as well as vertex_connectivity check \
   as that's redundant from -C. Greatly speeds up the process.
    Changed output from every 1000 to every 10000, takes \
   approximately 30 seconds per output.
    Takes approximately 3-4 minutes to run through 5 million pre-\
   progress graphs.
'''
import copy
import sys

sage_server.MAX_OUTPUT_MESSAGES = 10000
sage_server.MAX_OUTPUT = 100000000

def known(G,i):
    if G==graphs.CompleteGraph(i):          #complete
        return True
    elif G.is_cycle() and i%2==1:           #odd cycle
        return True
```

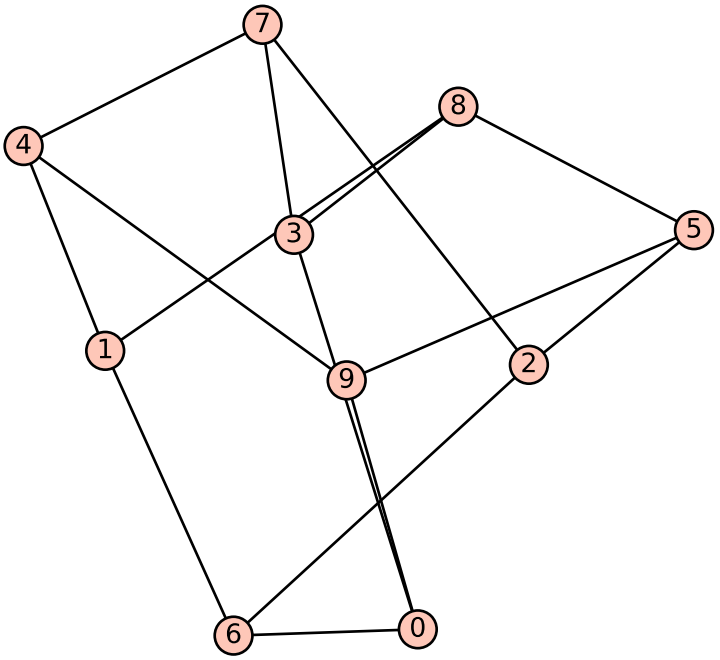
```
    elif len(G.edges()) == i-1:                #tree
        return True
    return False

for i in range(10,11): #online range(2,6) takes a while, but range\
   (2,5) is pretty quick
    num = 0
    progress = 649 * 10000
    for G in graphs.nauty_geng(str(i) + " -C"):
        num += 1
        if (num <= progress):
            if (num == progress):
                print("*")
            continue
        if num % 10000 == 0:
            print("%d" % (num/10000))
        q = True
        for j in range (0,i):
            S = Sandpile(G,j)
            if S.identity() != S.max_stable():
                q = False
                break
        if q:
            if known(G,i):
                continue
            else:
                print "fail: %d" % i
                G.show()
    print "Done: %d" % i
```

'\nRestrictions placed on G,S,j:\n G must be finite, connected, undirected\n G must not be a complete graph, an odd cycle graph, or a tree\n G must have vertex connectivity greater than 1 (“irreducible”)\nResults:\n Passed all sizes at most 7.\n Failed at size 8: 3 different\n Passed size 9\n Size 10: Count reached 100k, no flagged graphs\n As skipping for count takes too long, switched to num. Set progress to 100k for num, despite massive loss.\n Looks like size 10 will take a long time (size 9 had 261 thousand, so size 10 is a lot larger...)\nNote:\n There are roughly 9.7 million biconnected graphs with 10 vertices.\n Eliminated -d2 condition, as well as vertex_connectivity check as that\'s redundant from -C. Greatly speeds up the process.\n Changed output from every 1000 to every 10000, takes approximately 30 seconds per output.\n'

*
456
457
458
459
460

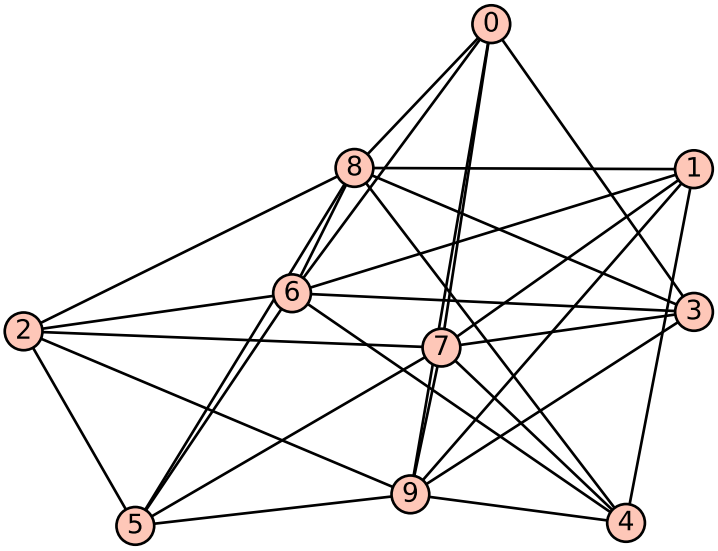
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601

602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
fail: 10



618
619
620
621
622
623
624
625
626
627

628
629
630
631
632
633
fail: 10



634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649