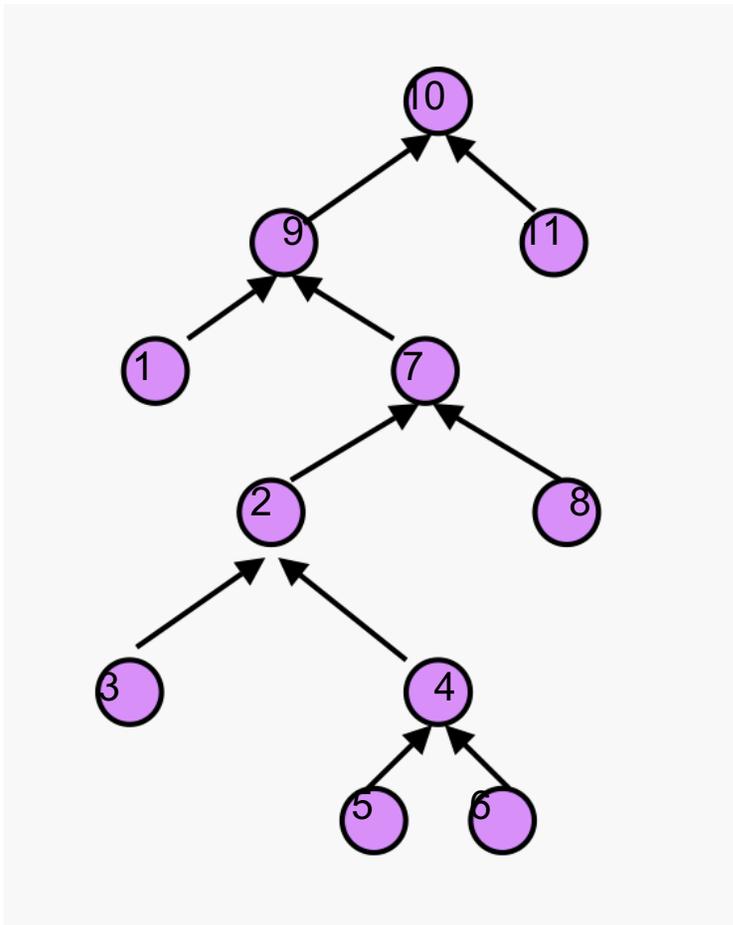


Today's announcements:

MP7 available, due 5/2, 11:59p. EC due 4/21, 11:59p.

Path Compression:



Path Compression:

```
int DS::Find(int i) {  
    if (s[i] < 0) return i;  
    else return      Find(s[i]);  
}
```

```
void DS::UnionBySize(int root1, int root2) {  
    int newSize = s[root1]+s[root2];  
    if (isBigger(root1,root2)) {  
        s[root2]= root1;  
        s[root1]= newSize;  
    }  
    else {  
        s[root1] = root2;  
        s[root2]= newSize;  
    }  
}
```

Analysis:

$$\log^* n := \begin{cases} 0 & \text{if } n \leq 1; \\ 1 + \log^*(\log n) & \text{if } n > 1 \end{cases}$$

Example:

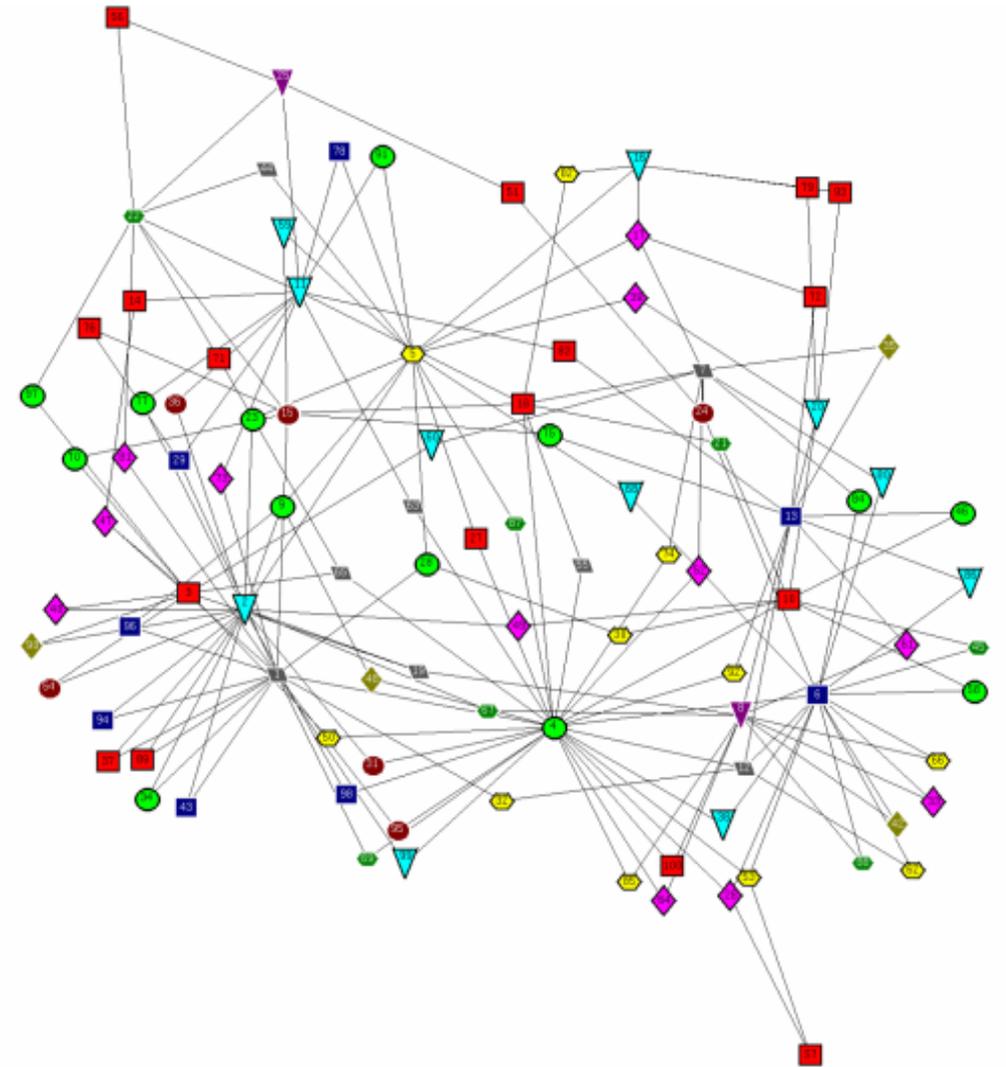
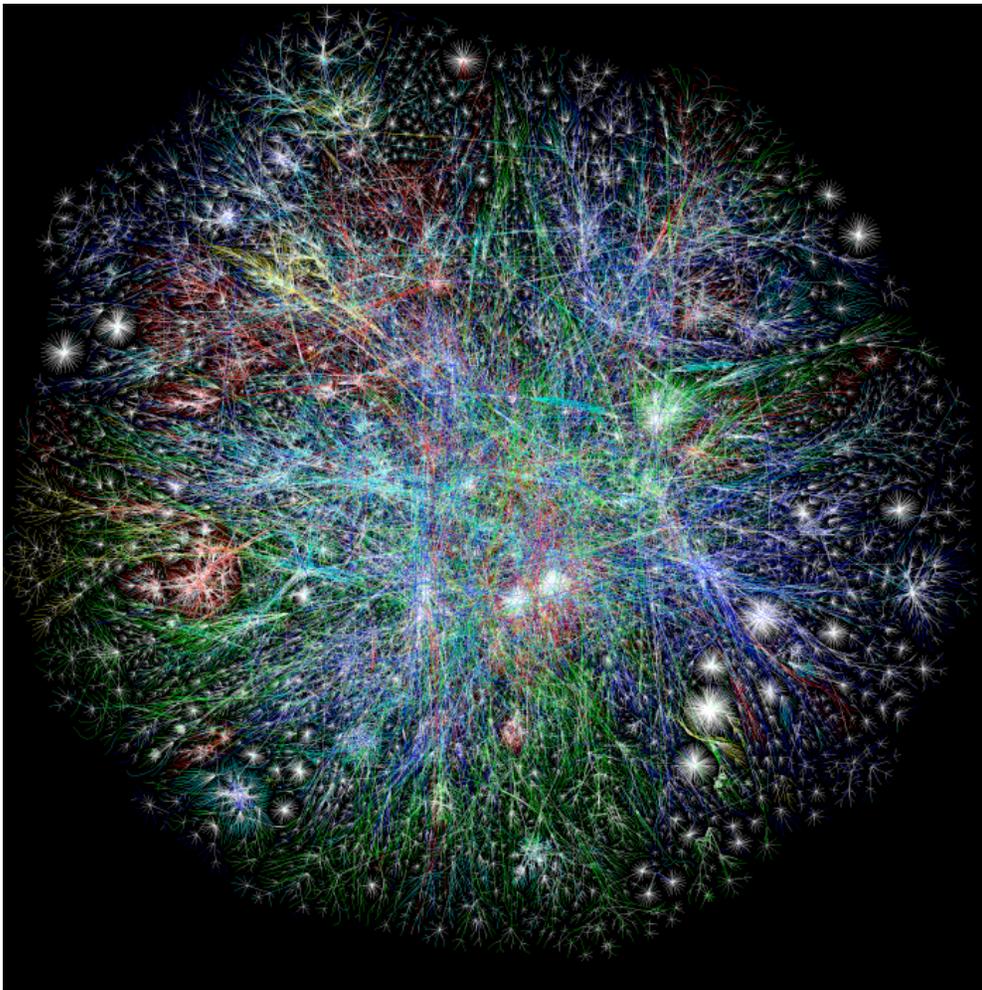
265536

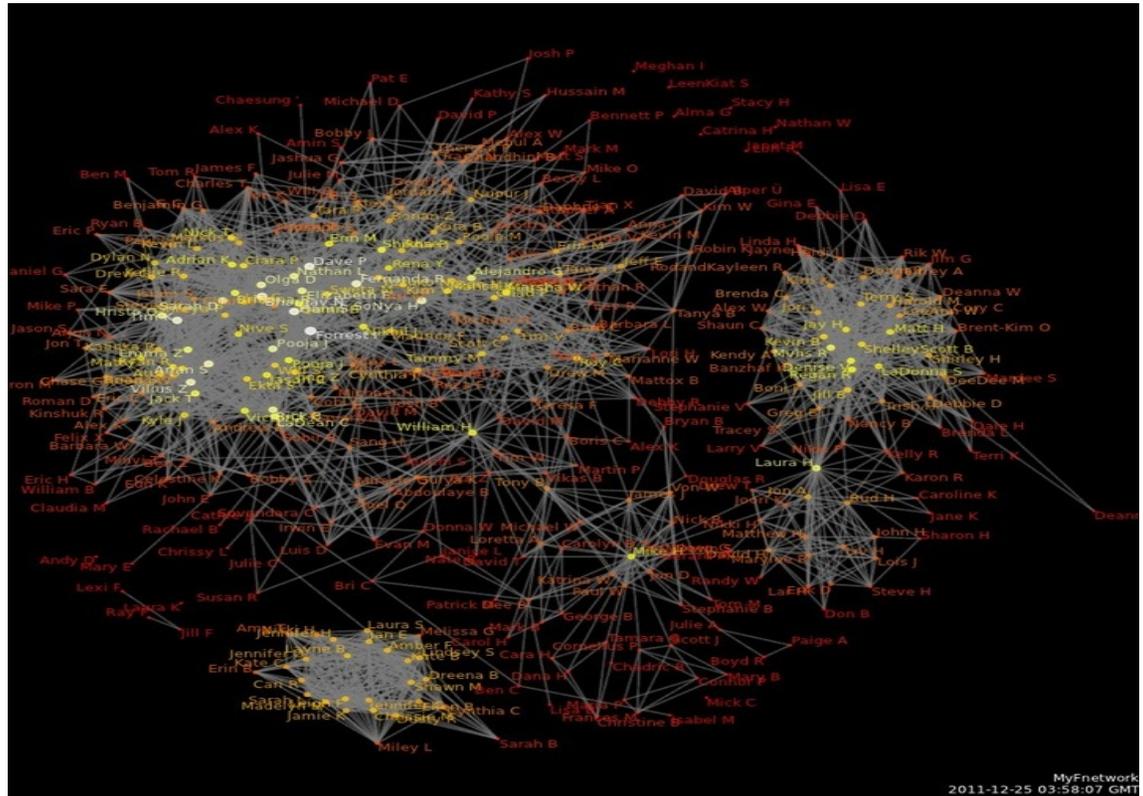
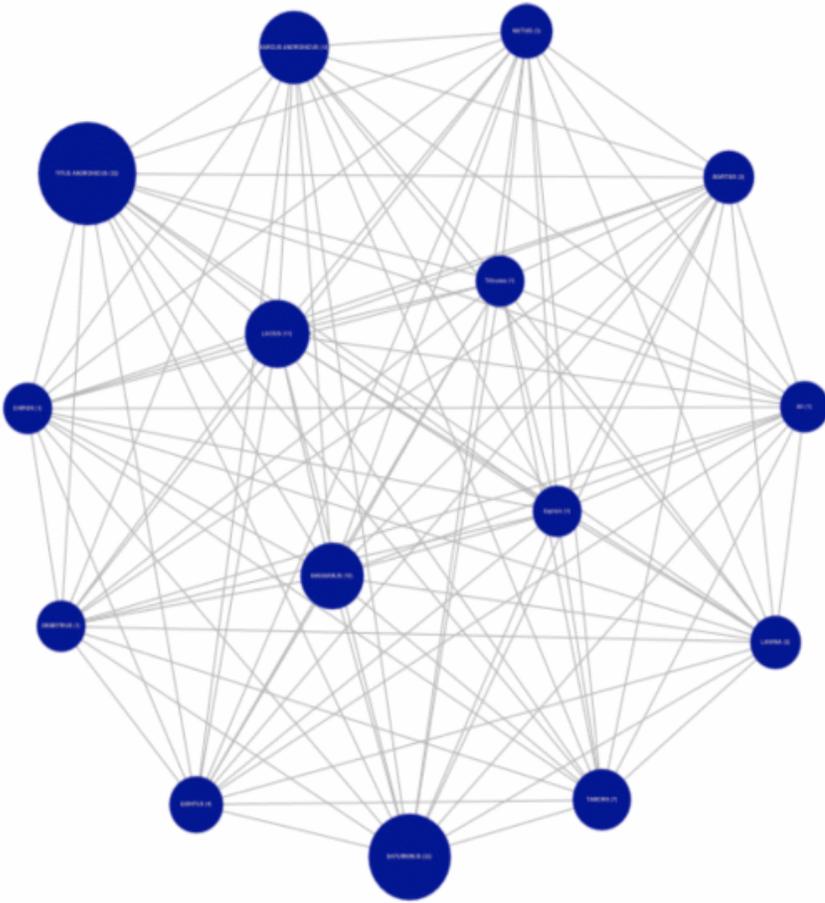
Relevant result:

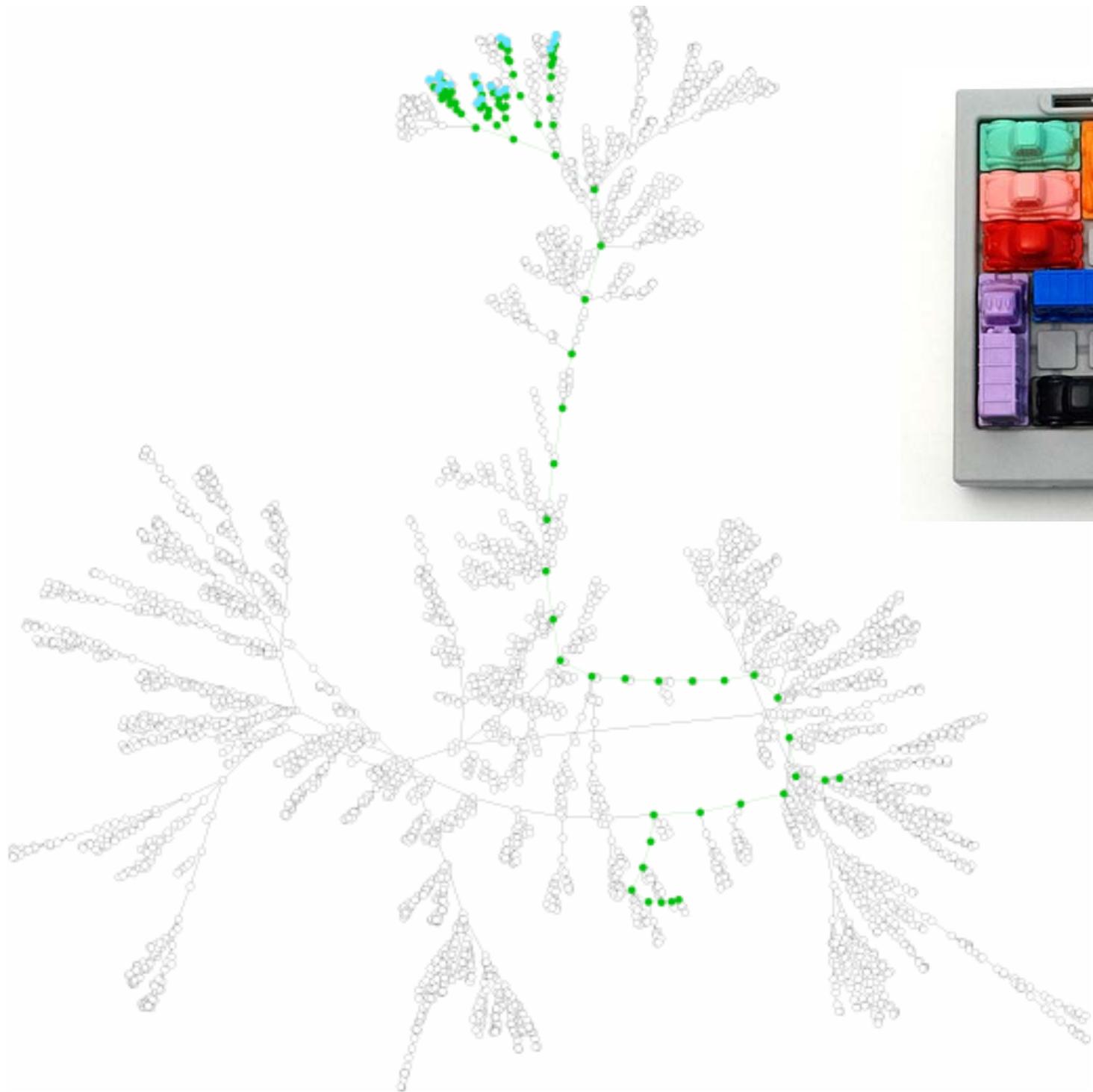
In an upTree implementation of Disjoint Sets using smart `union` and path compression upon `find`...

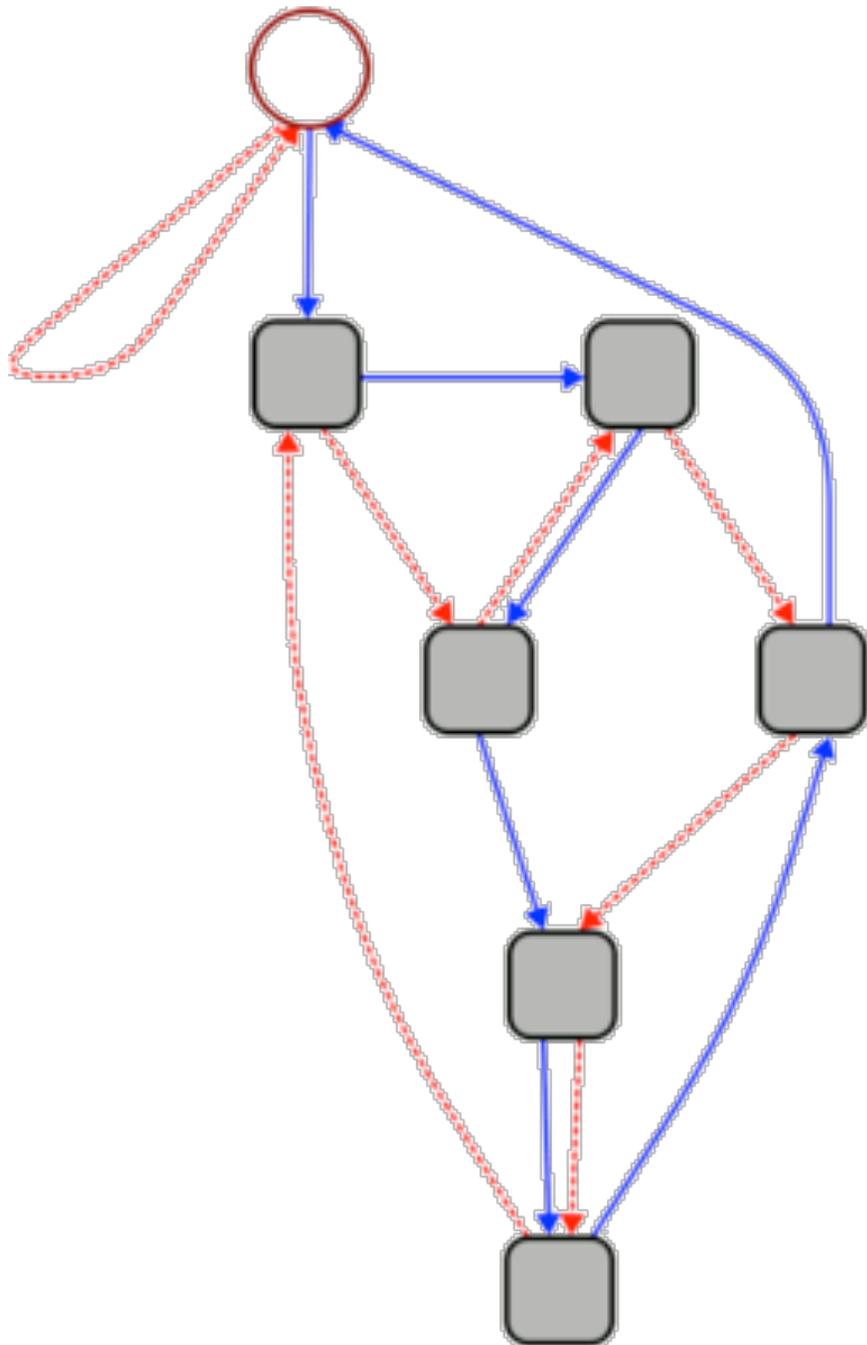
any sequence of m `union` and `find` operations results in worst case running time of $O(\text{_____})$, where n is the number of items.

<http://research.cs.vt.edu/AVresearch/UF/>





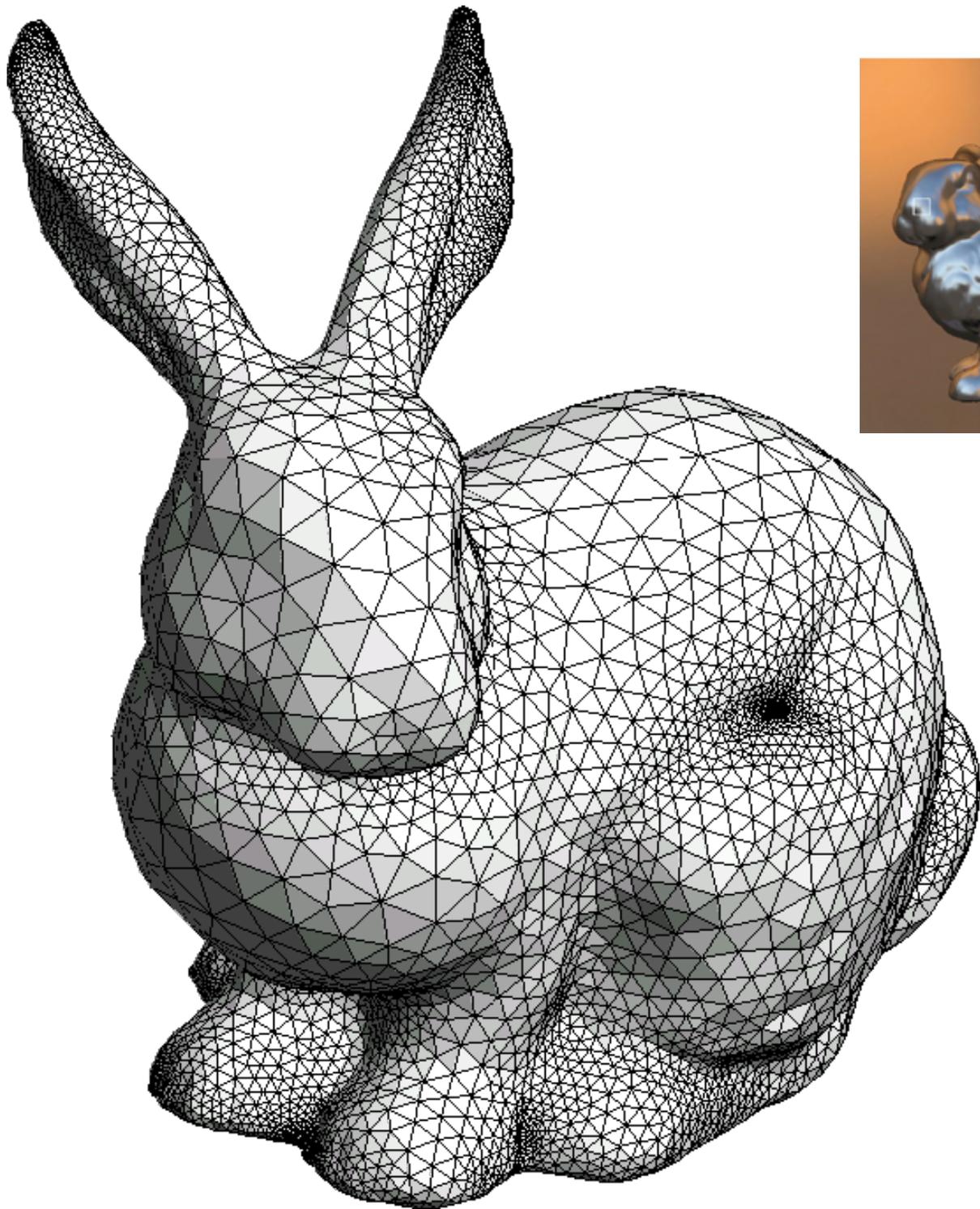




This graph can be used to quickly calculate whether a given number is divisible by 7.

1. Start at the circle node at the top.
2. For each digit d in the given number, follow d blue (solid) edges in succession. As you move from one digit to the next, follow 1 red (dashed) edge.
3. If you end up back at the circle node, your number is divisible by 7.

3703



Graphs

How do we get from here to there?
Need:

1. Common Vocabulary
2. Graph implementation
3. Traversal
4. Algorithms.

