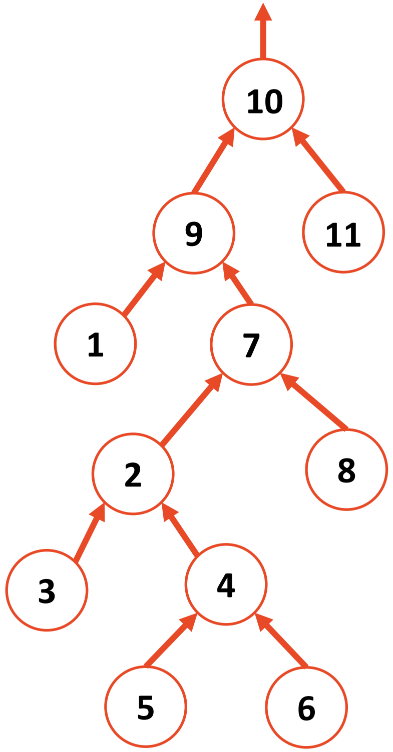
CS 225

Data Structures

Nov. 13 – Introduction to Graphs

Wade Fagen-Ulmschneider

UpTree



Disjoint Sets Find

```
int DisjointSets::find(int i)  {
  if ( arr_[i] < 0 ) { return i; }
  else { return              find( arr_[i] ); }
}
```

```
void DisjointSets::unionBySize(int root1, int root2)  {
  int newSize = arr_[root1] + arr_[root2];

  // If arr_[root1] is less than (more negative), it is the larger set;
  // we union the smaller set, root2, with root1.
  if ( arr_[root1] < arr_[root2] ) {
    arr_[root2] = root1;
    arr_[root1] = newSize;
  }

  // Otherwise, do the opposite:
  else {
    arr_[root1] = root2;
    arr_[root2] = newSize;
  }
}
```

Exam Information w/ Mattox

Now: Exam 10 – Programming Exam

You should have seen the **.h** files on Piazza.

Next Week: Exam 11 – Theory Exam

Hash Tables
Heaps
Disjoint Sets
Hash Functions (SUHA)

POTDs

POTDs

We have exhausted the initial set. We'll be making more, but we may have some "gap days". The POTDs will be more puzzle-like in nature (you won't be told what data structure or algorithm you need to solve it).

No POTDs over Fall Break.

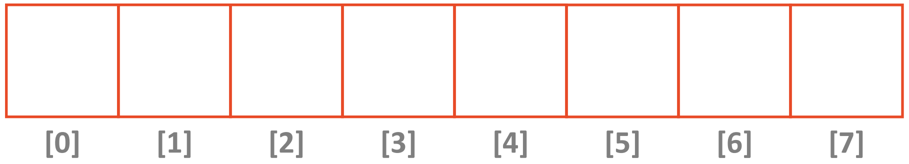
In Review: Data Structures

Array

- Sorted Array
- Unsorted Array
  - Stacks
  - Queues
  - Hashing
  - Heaps
    - Priority Queues
  - UpTrees
    - Disjoint Sets

List

- Doubly Linked List
- Skip List
- Trees
  - BTree
  - Binary Tree
    - Huffman Encoding
    - kd-Tree
    - AVL Tree



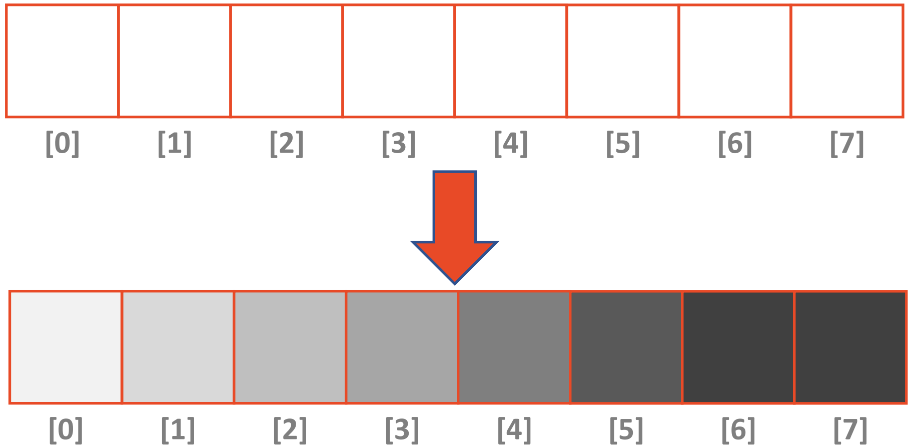
Array

- Constant time access to any element, given an index

  a[k] is accessed in O(1) time, no matter how large the array grows

- Cache-optimized

  Many modern systems cache or pre-fetch nearby memory values due the “Principle of Locality”. Therefore, arrays often perform faster than lists in identical operations.



Array

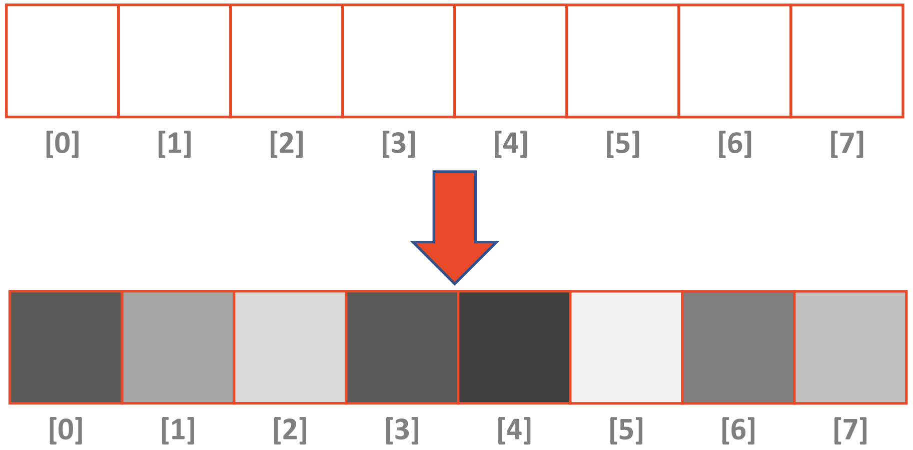
Sorted Array

- Efficient general search structure

  Searches on the sort property run in O(lg(n)) with Binary Search

- Inefficient insert/remove

  Elements must be inserted and removed at the location dictated by the sort property, resulting shifting the array in memory – an O(n) operation



Array

Unsorted Array

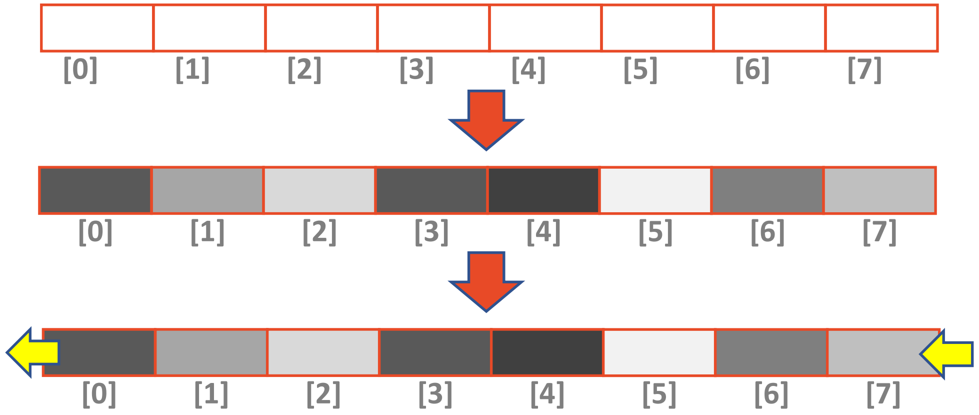
- Constant time add/remove at the beginning/end

  Amortized O(1) insert and remove from the front and of the array

  <u>Idea:</u> Double on resize

- Inefficient search structure

  With no sort property, all searches must iterate the entire array; O(1) time



Array

Unsorted Array

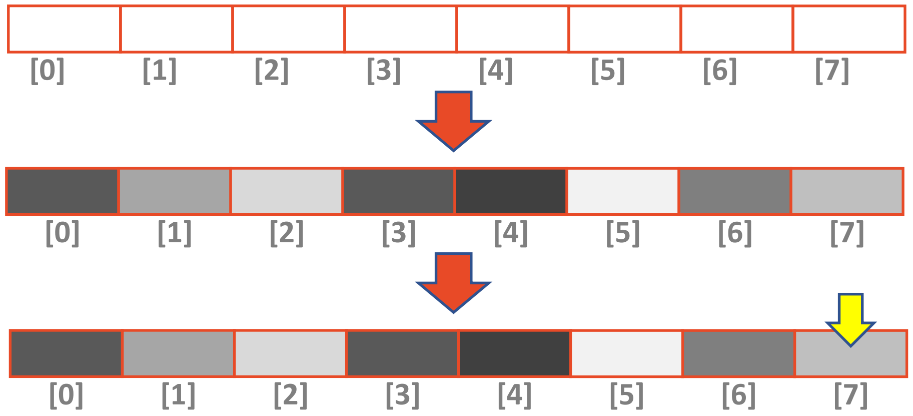
Queue (FIFO)

- First In First Out (FIFO) ordering of data

  Maintains an arrival ordering of tasks, jobs, or data

- All ADT operations are constant time operations

  enqueue() and dequeue() both run in O(1) time



Array

Unsorted Array

Stack (LIFO)

- Last In First Out (LIFO) ordering of data

  Maintains a “most recently added” list of data

- All ADT operations are constant time operations

  push() and pop() both run in O(1) time

In Review: Data Structures

Array
- **Sorted Array**
- **Unsorted Array**
  - **Stacks**
  - **Queues**
  - **Hashing**
  - **Heaps**
    - **Priority Queues**
  - **UpTrees**
    - **Disjoint Sets**

List
- **Doubly Linked List**
- **Skip List**
- **Trees**
  - **BTree**
  - **Binary Tree**
    - **Huffman Encoding**
    - **kd-Tree**
    - **AVL Tree**

In Review: Data Structures

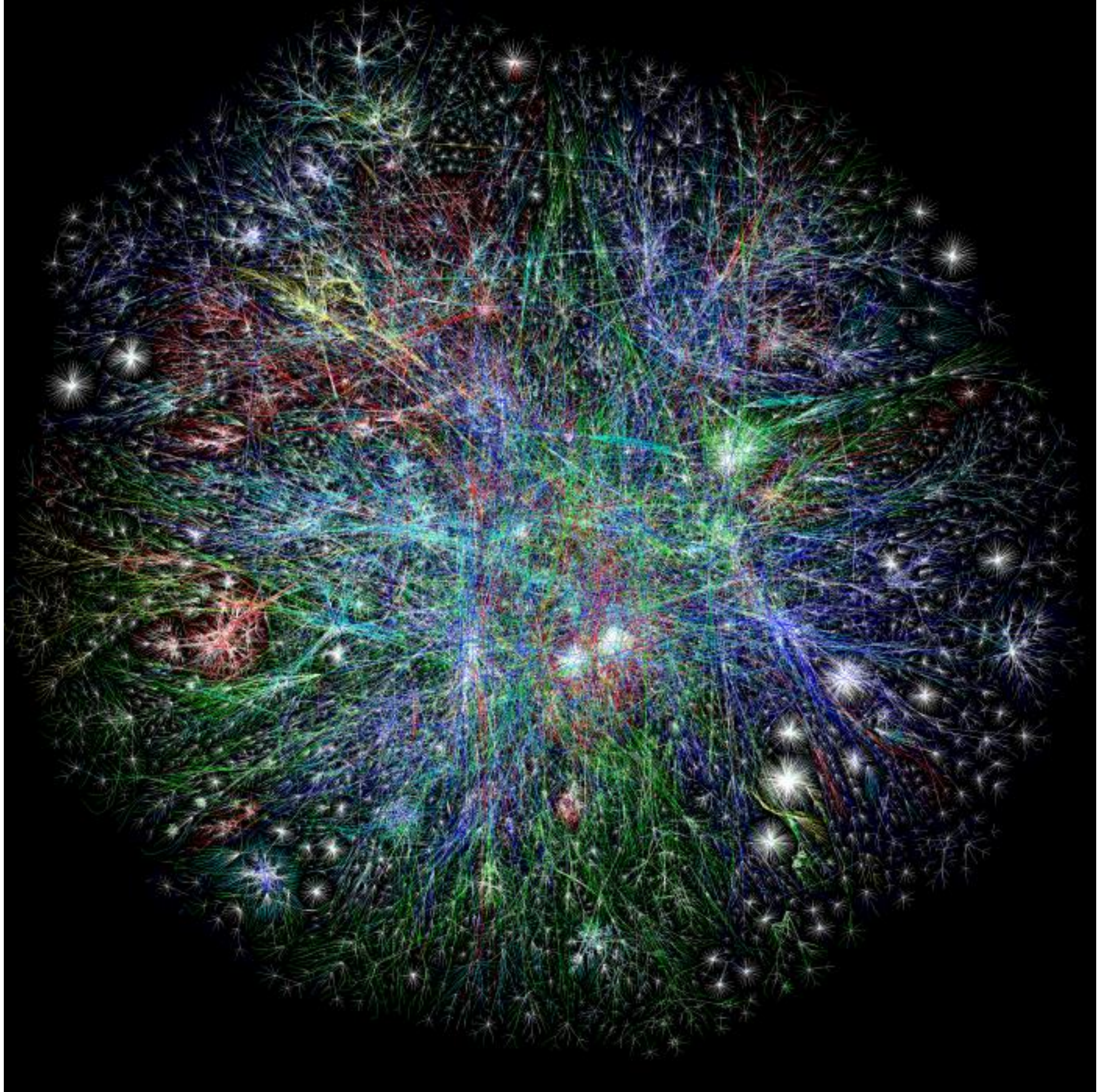
Array

- Sorted Array
- Unsorted Array
  - Stacks
  - Queues
  - Hashing
  - Heaps
    - Priority Queues
  - UpTrees
    - Disjoint Sets

Graphs

List

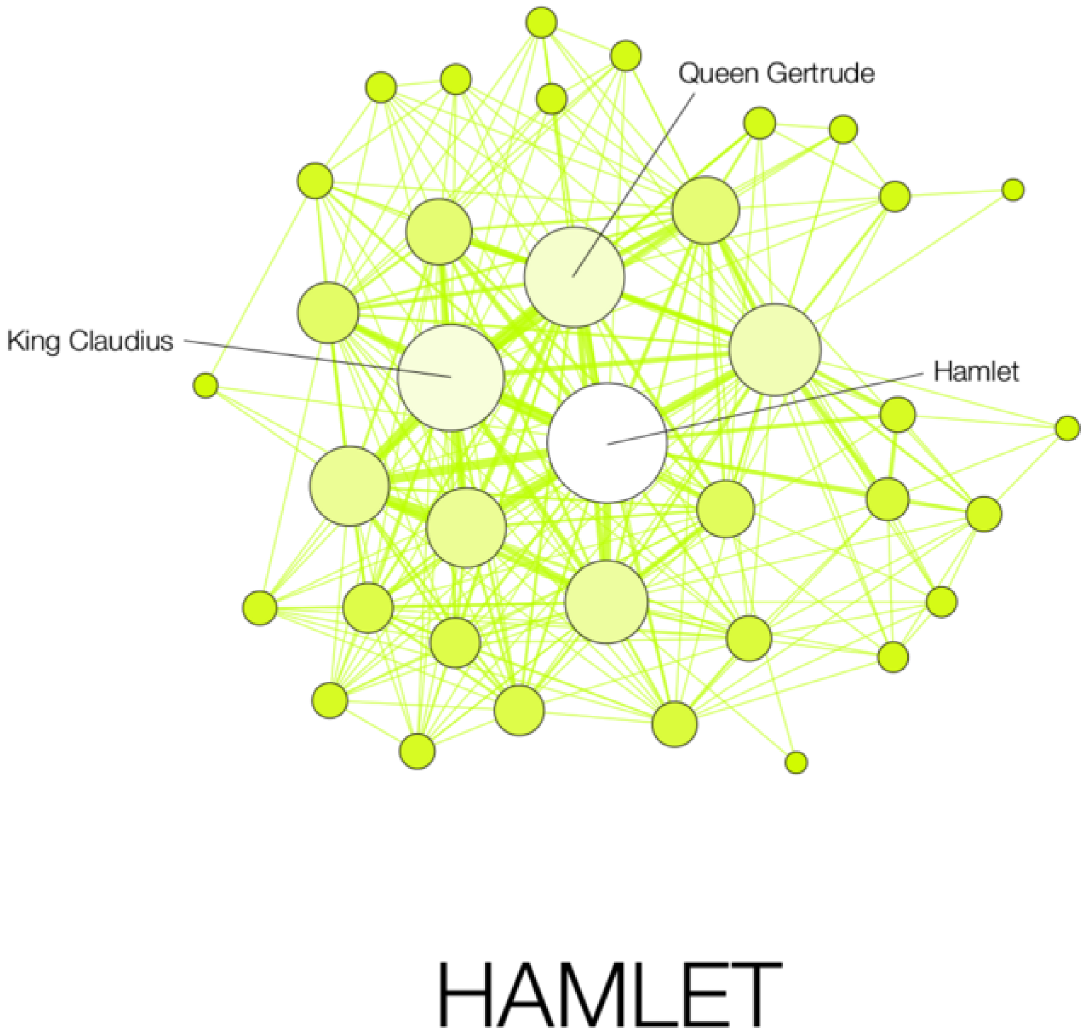
- Doubly Linked List
- Skip List
- Trees
  - BTree
  - Binary Tree
    - Huffman Encoding
    - kd-Tree
    - AVL Tree

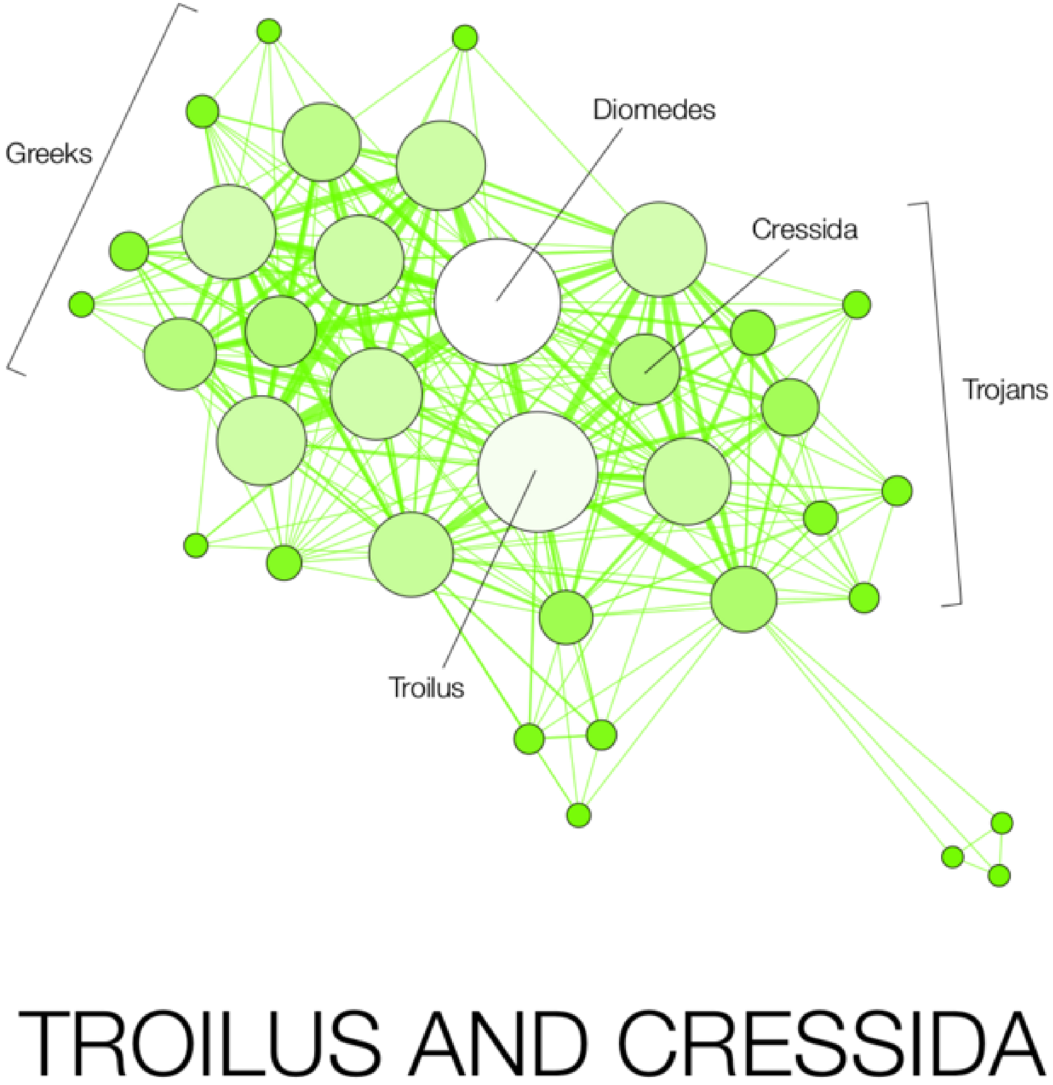


The Internet 2003

The OPTE Project (2003)

Map of the entire internet; nodes are routers; edges are connections.

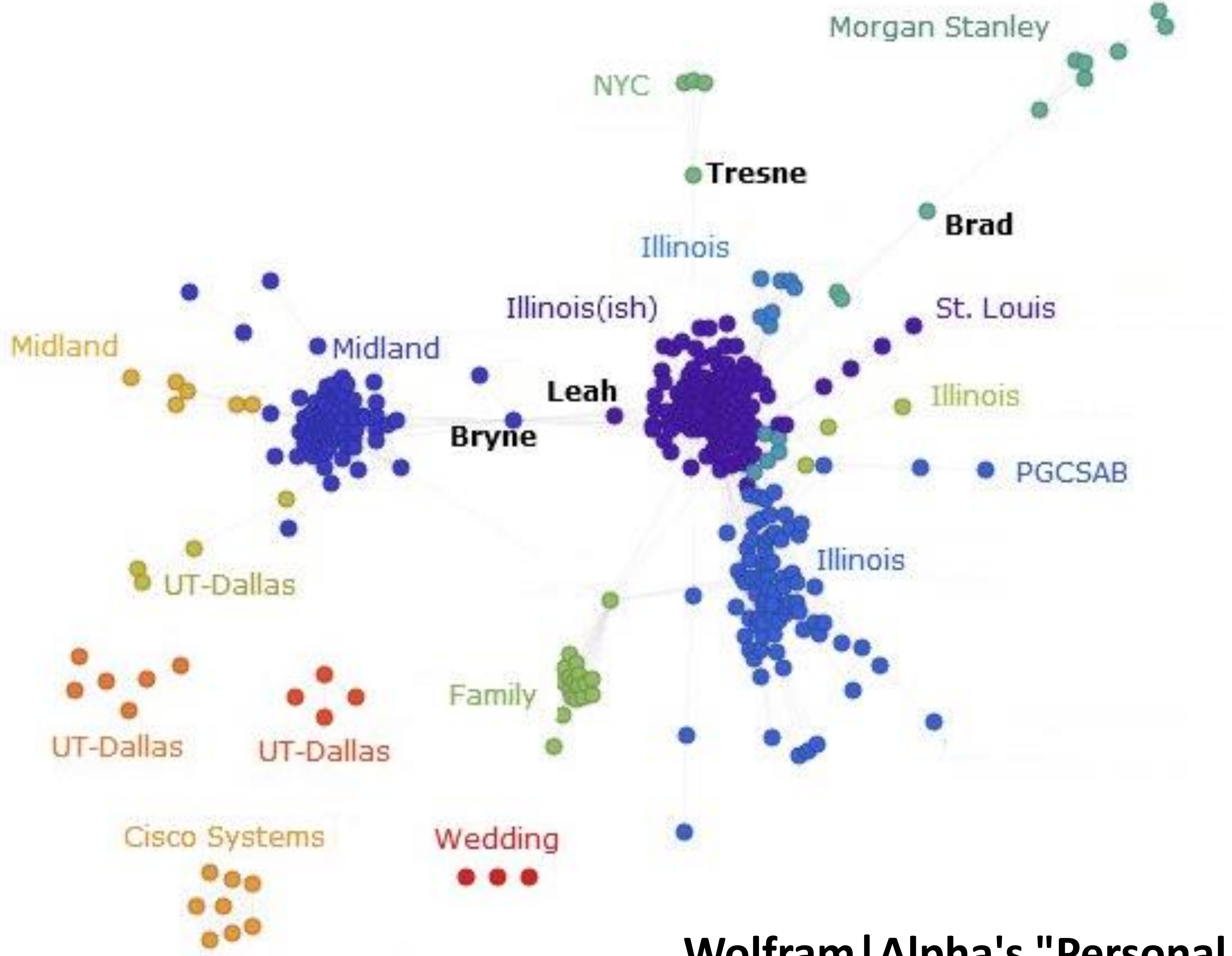




Who’s the real main character in Shakespearean tragedies?

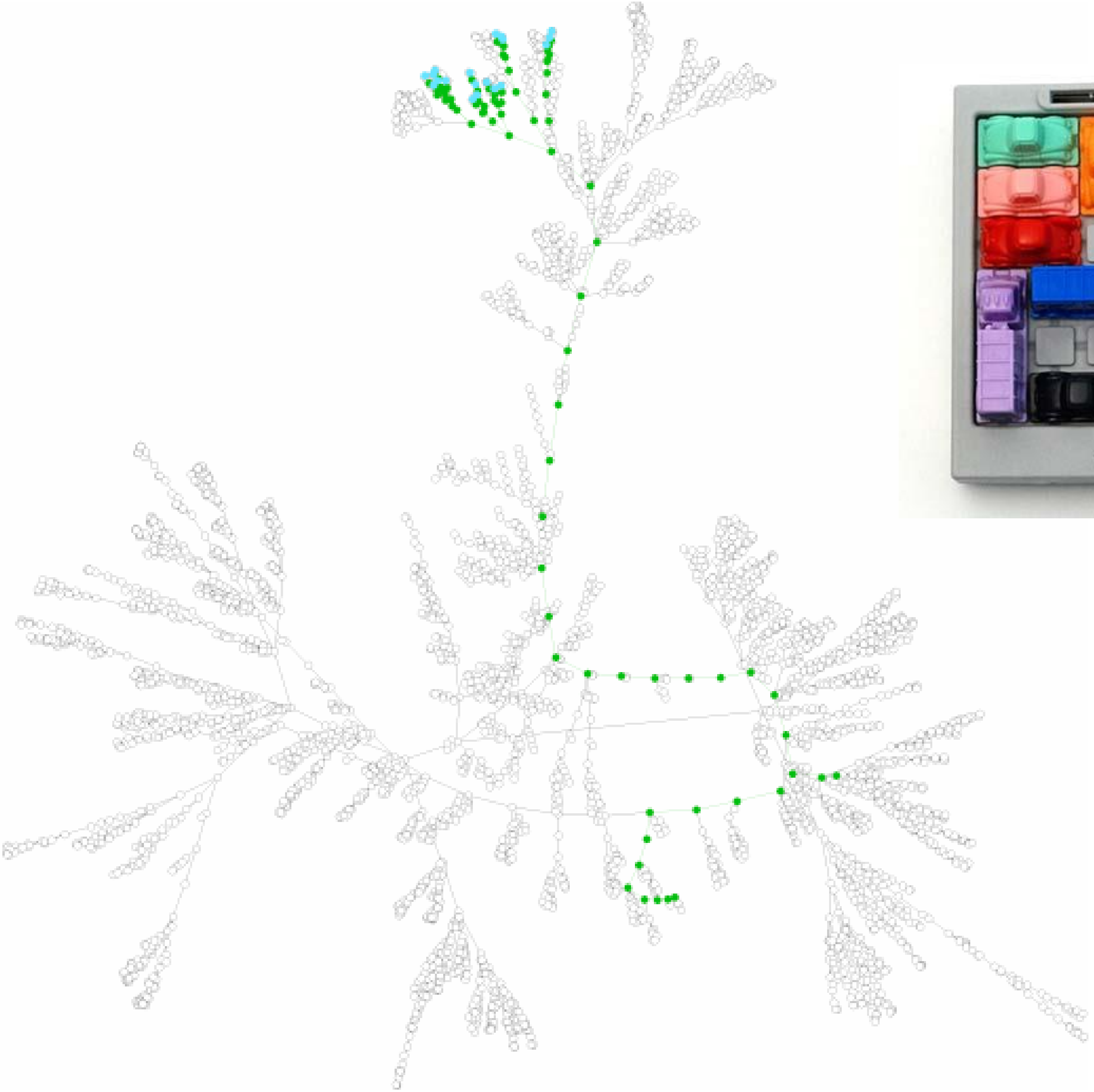
Martin Grandjean (2016)

https://www.pbs.org/newshour/arts/whos-the-real-main-character-in-shakespearen-tragedies-heres-what-the-data-say



Wolfram|Alpha's "Personal Analytics“ for Facebook

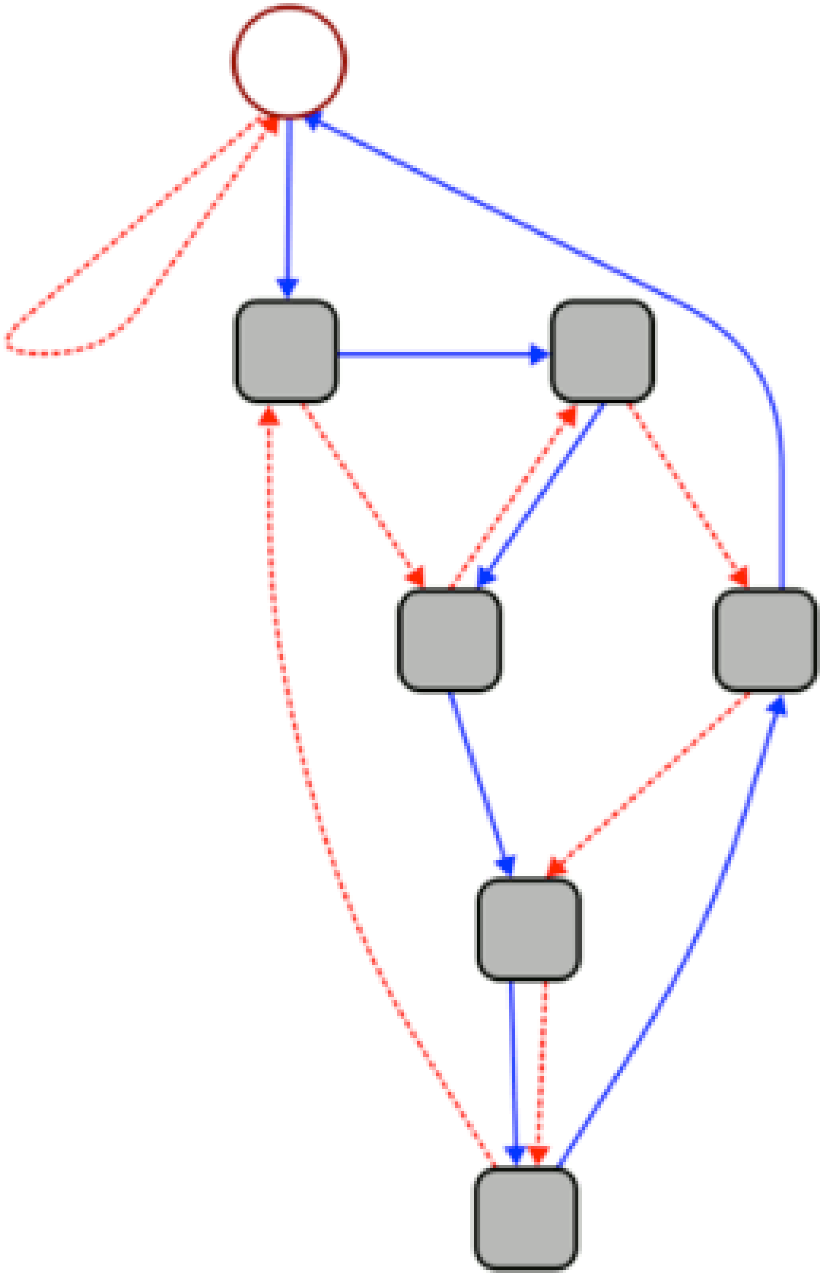
Generated: April 2013 using Wade Fagen-Ulmschneider’s Profile Data



“Rush Hour” Solution

Unknown Source

Presented by Cinda Heeren, 2016



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

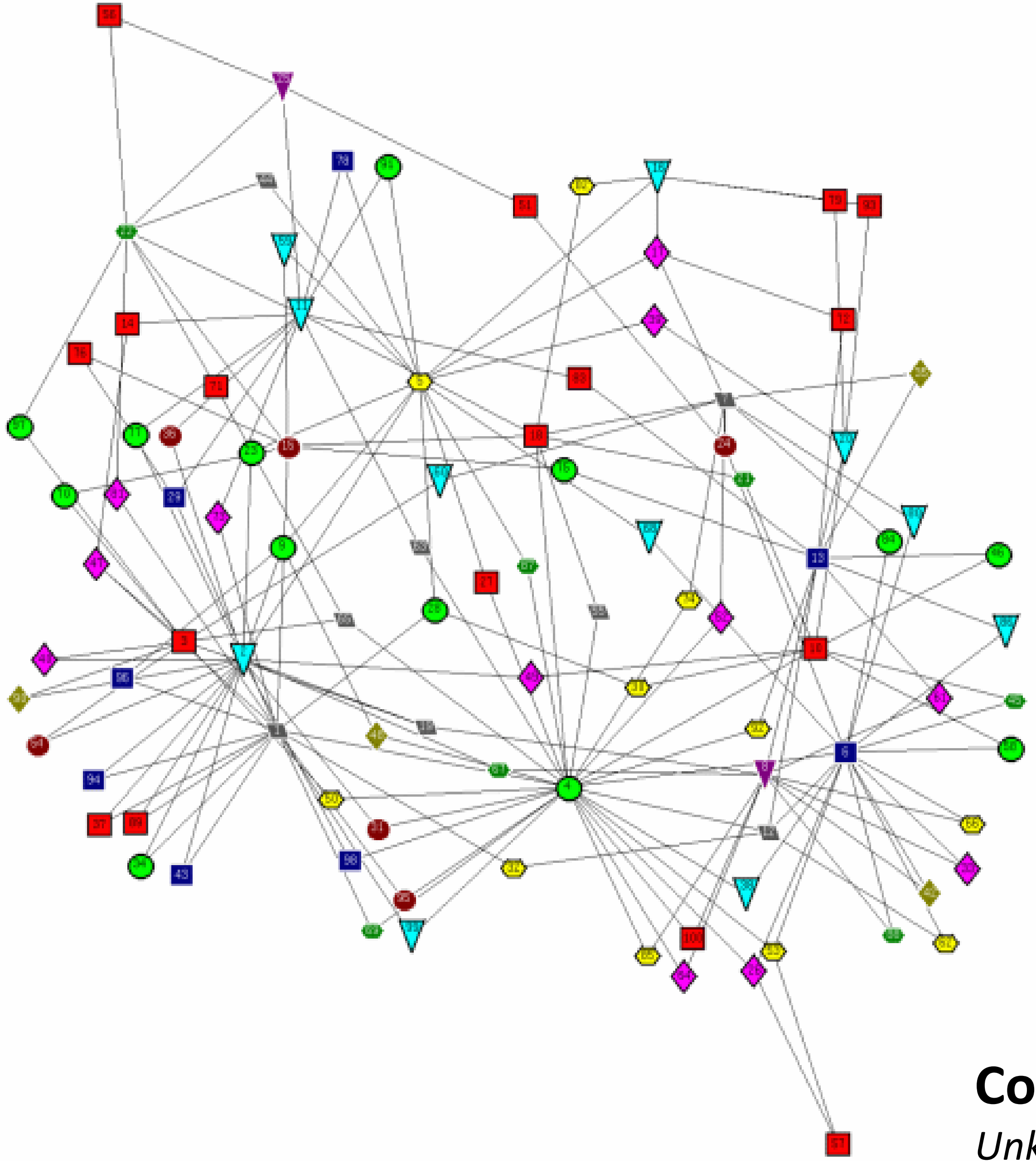
1. Start at the circle node at the top.
2. For each digit **d** in the given number, follow **d** blue (solid) edges in succession. As you move from one digit to the next, follow **1** red (dashed) edge.
3. If you end up back at the circle node, your number is divisible by 7.

3703

"Rule of 7"

Unknown Source

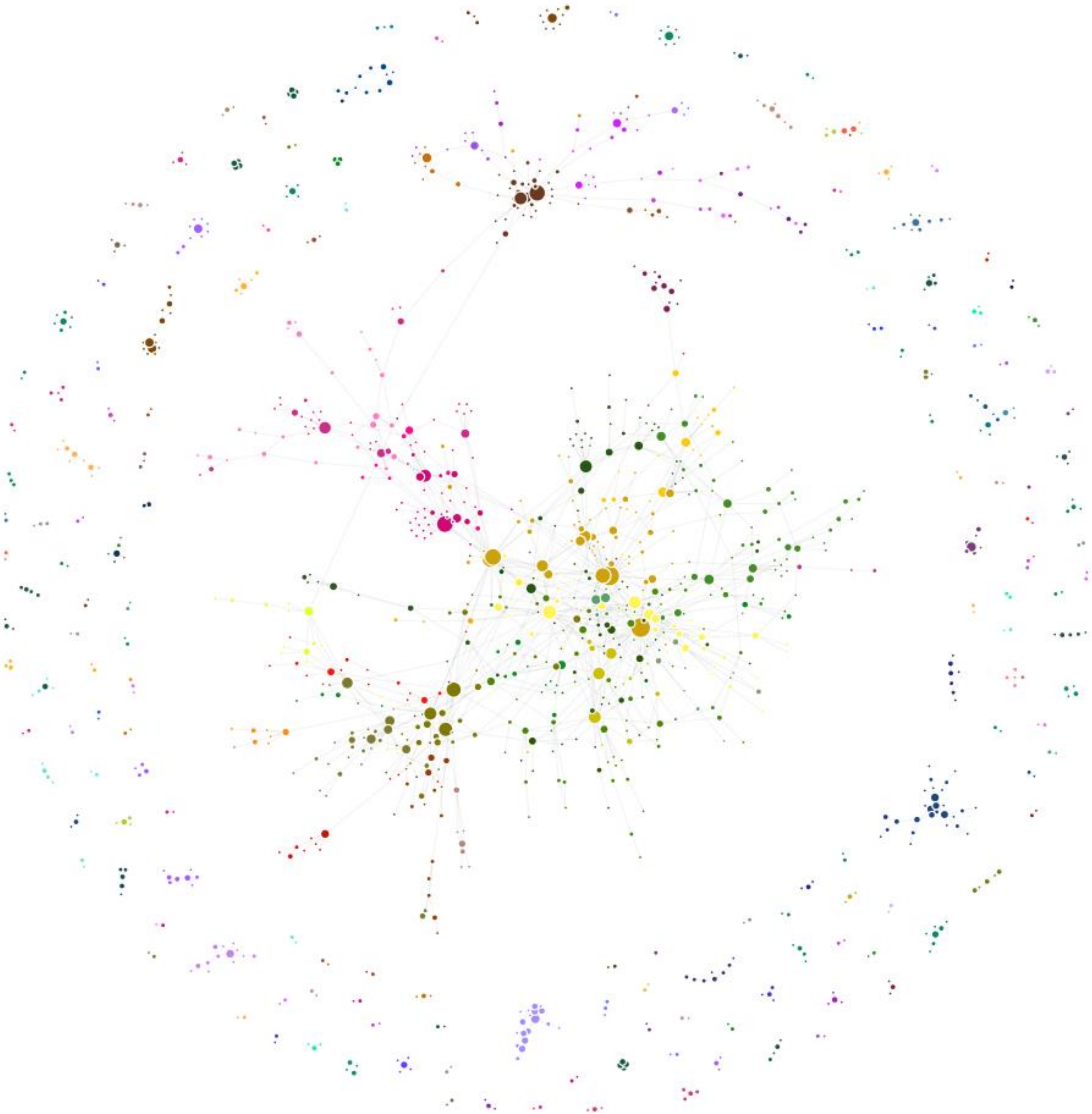
Presented by Cinda Heeren, 2016



Conflict-Free Final Exam Scheduling Graph

Unknown Source

Presented by Cinda Heeren, 2016

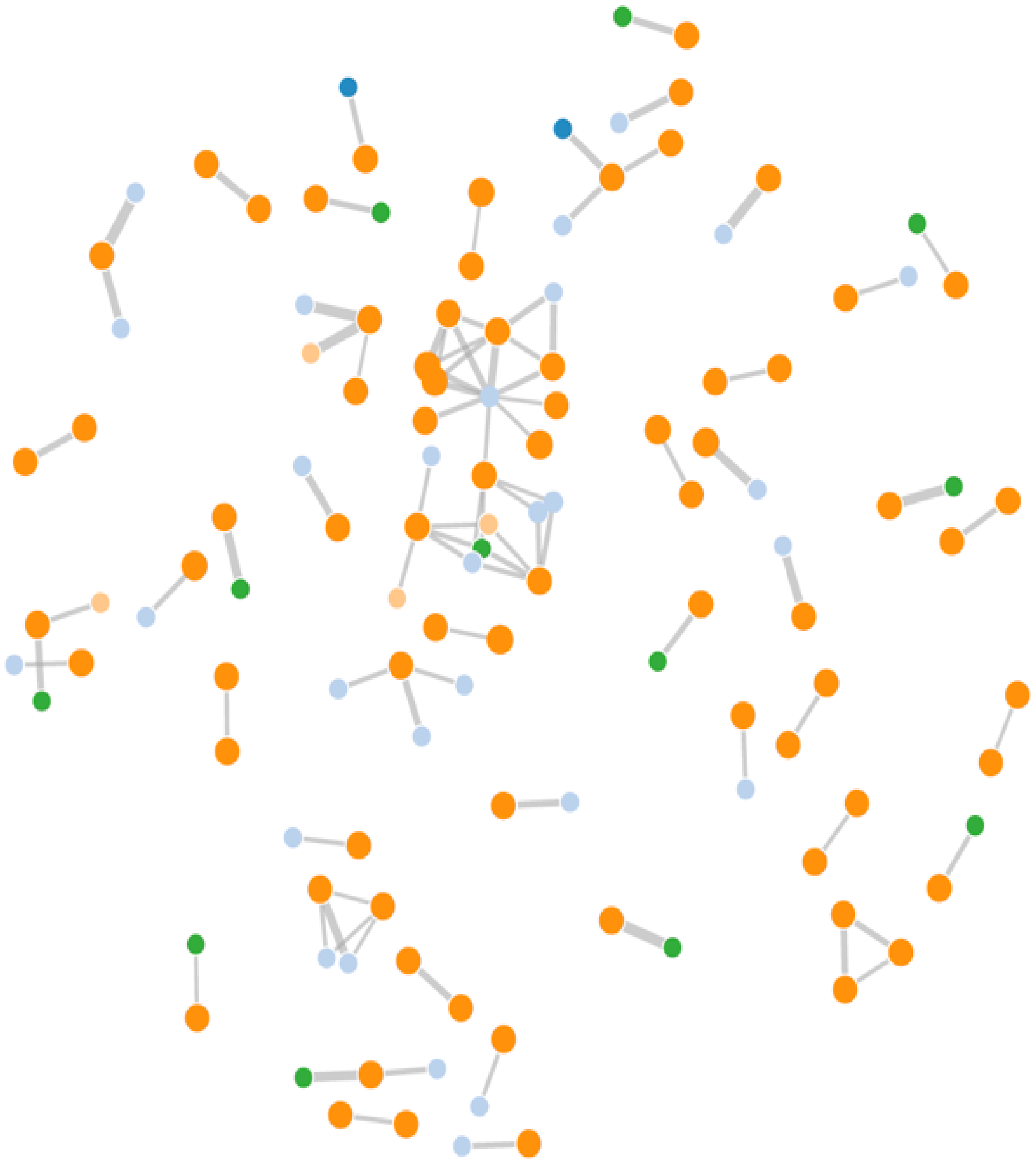


Class Hierarchy At University of Illinois Urbana-Champaign

A. Mori, W. Fagen-Ulmschneider, C. Heeren

Graph of every course at UIUC; nodes are courses, edges are prerequisites

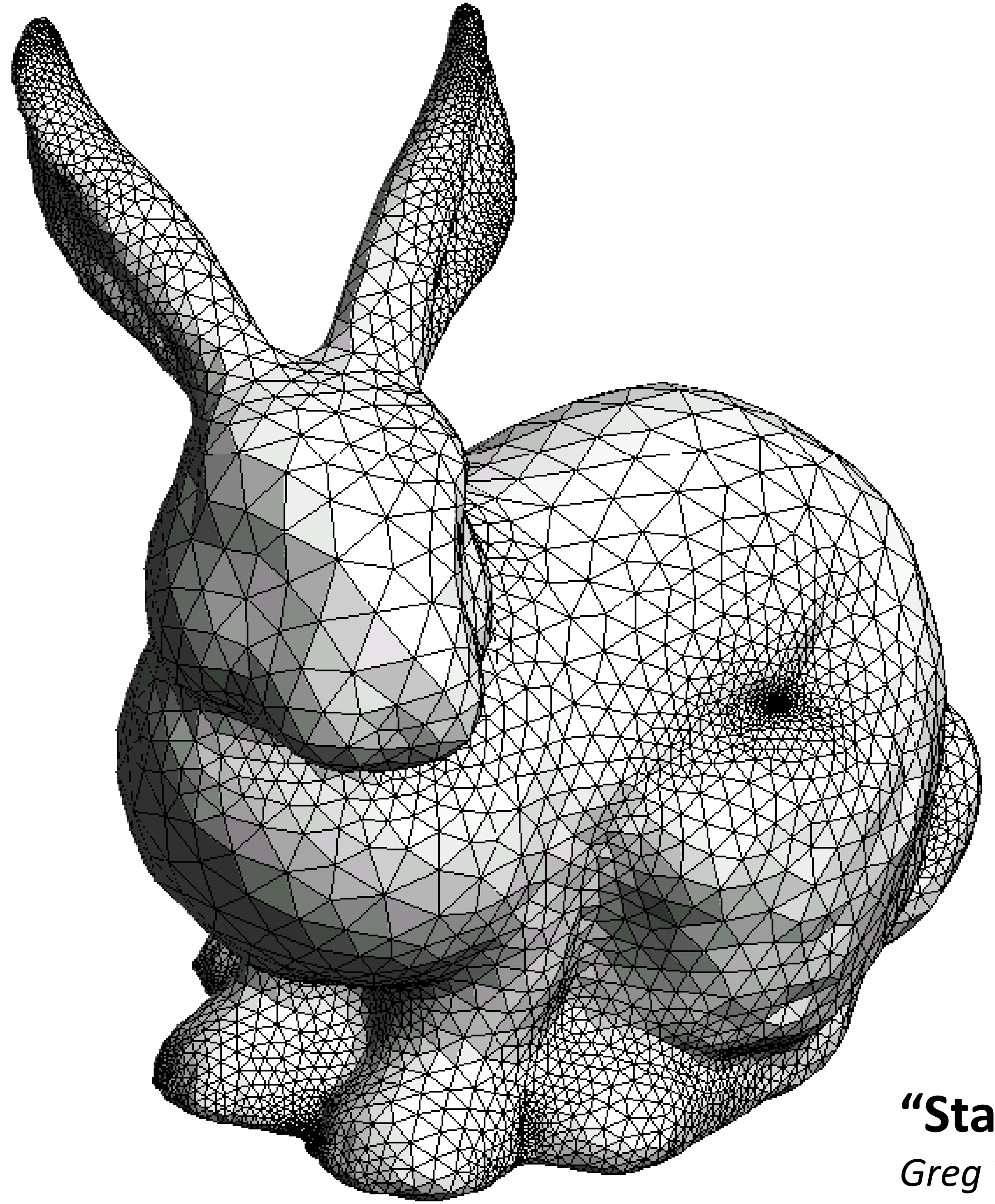
http://waf.cs.illinois.edu/discovery/class_hierarchy_at_illinois/



MP Collaborations in CS 225

Unknown Source

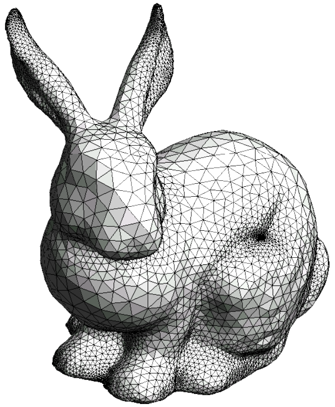
Presented by Cinda Heeren, 2016



“Stanford Bunny”

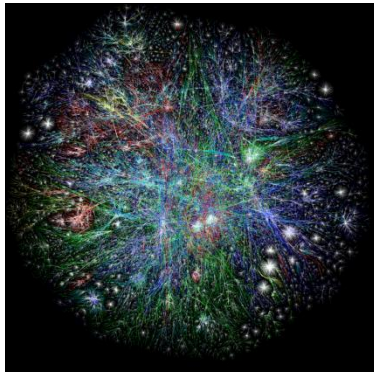
Greg Turk and Mark Levoy (1994)

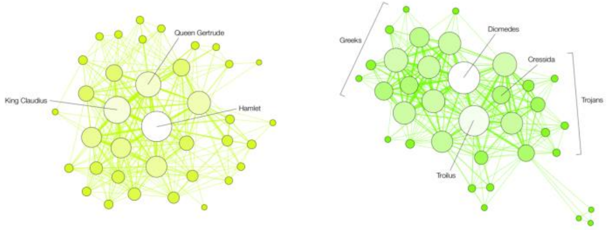
Graphs



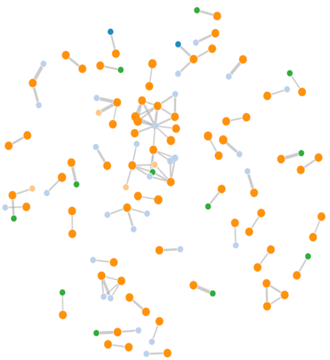
To study all of these structures:

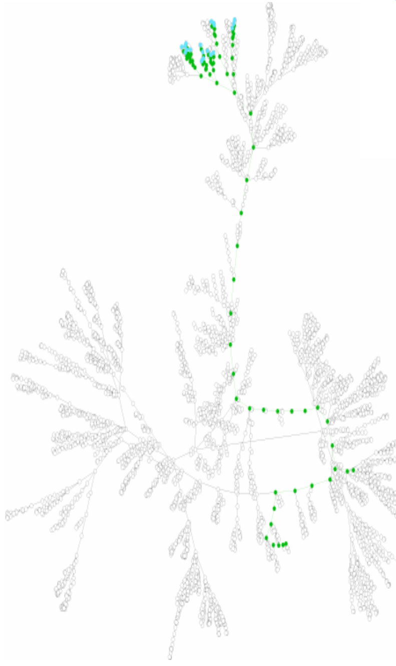
1. A common vocabulary
2. Graph implementations
3. Graph traversals
4. Graph algorithms

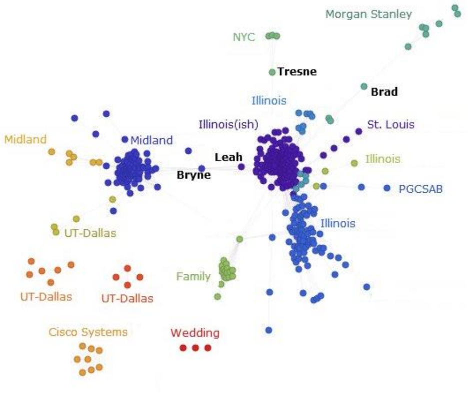


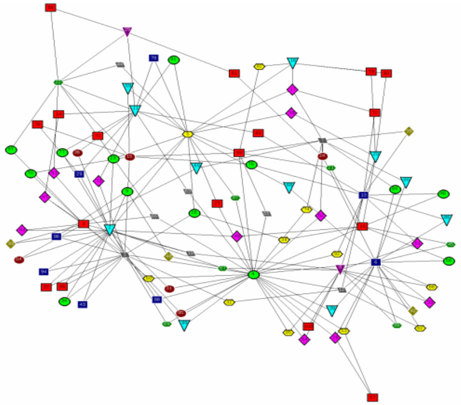


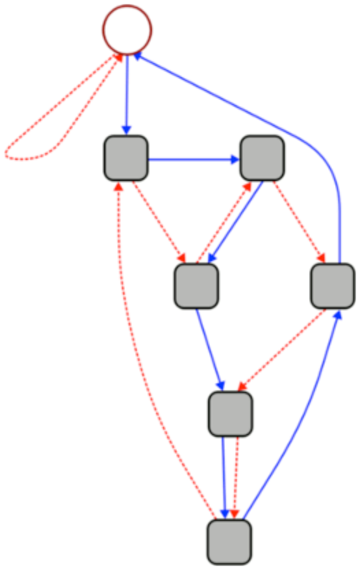
HAMLET TROILUS AND CRESSIDA











CS 225 – Things To Be Doing

Exam 10 (programming) is ongoing!

More Info: https://courses.engr.illinois.edu/cs225/fa2017/exams/

MP6: A one week reflection MP!

Due: Friday, Nov. 17 at 11:59pm

Lab: lab_dict released on Wednesday

Due: Wednesday, Nov. 29 @ 7pm (Before the first lab after break!)

POTD

Worth +1 Extra Credit /problem (up to +40 total)