Graphs

Lecture Question

Task: Verify a path in a graph

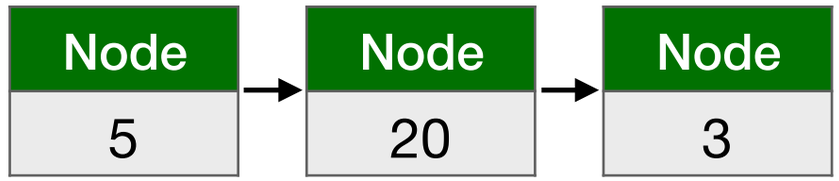
- In the Graph class write a method named isPath that takes a List of node indices as Ints representing a path in the graph
- Return a Boolean indicating whether or not the path is valid
- A path is valid if there is an edge between all adjacent nodes in the List

* This question will be open until midnight

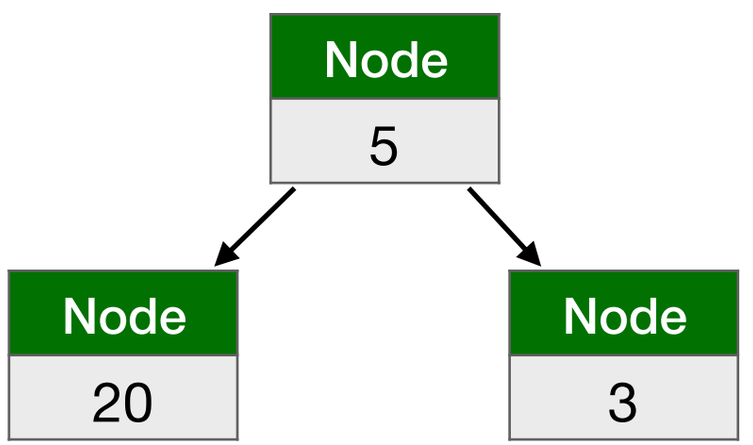
Data Structures: Review

- Sequential Data Structures
  - Elements stored in a specific order
  - Ex: Array, List
- Key-Value Store
  - Stores pairs of elements with no particular order
  - Each key is associated with one value
  - Ex. Map, Dictionary, Object
- Tree
  - Non-linear structure
  - Each element can be associated with multiple other elements

Index	0	1	2
Value	5	20	3

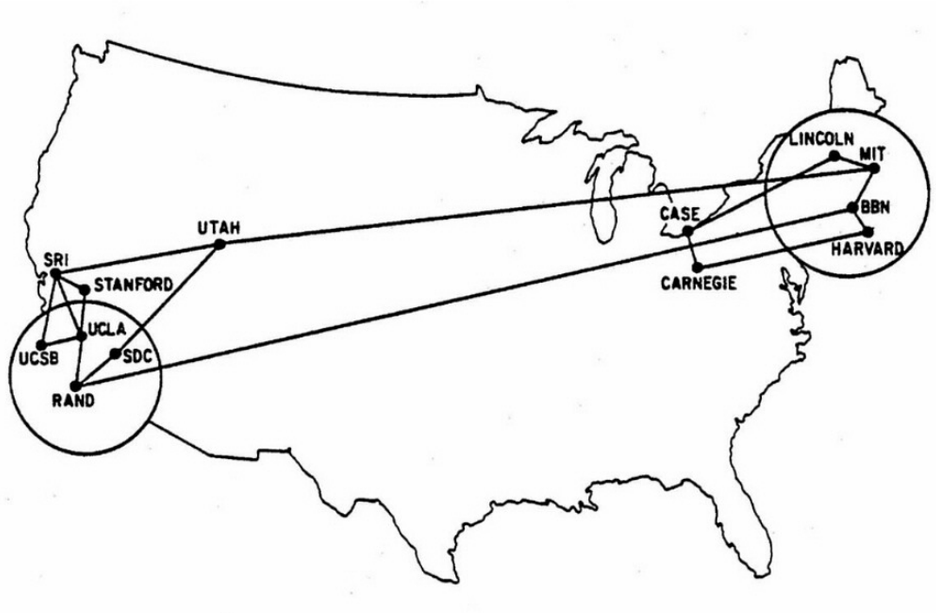


Key	"cse"	"mga"	"geo"
Value	20	3	5



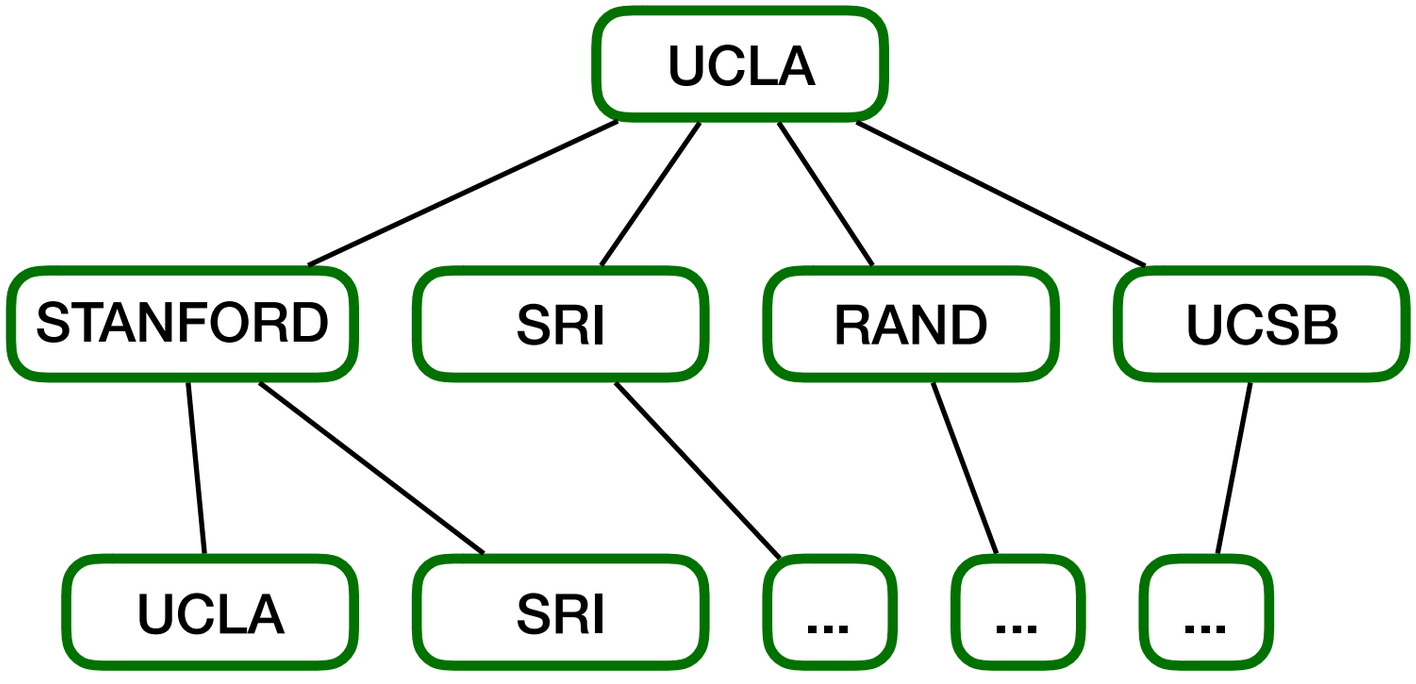
Data Structures

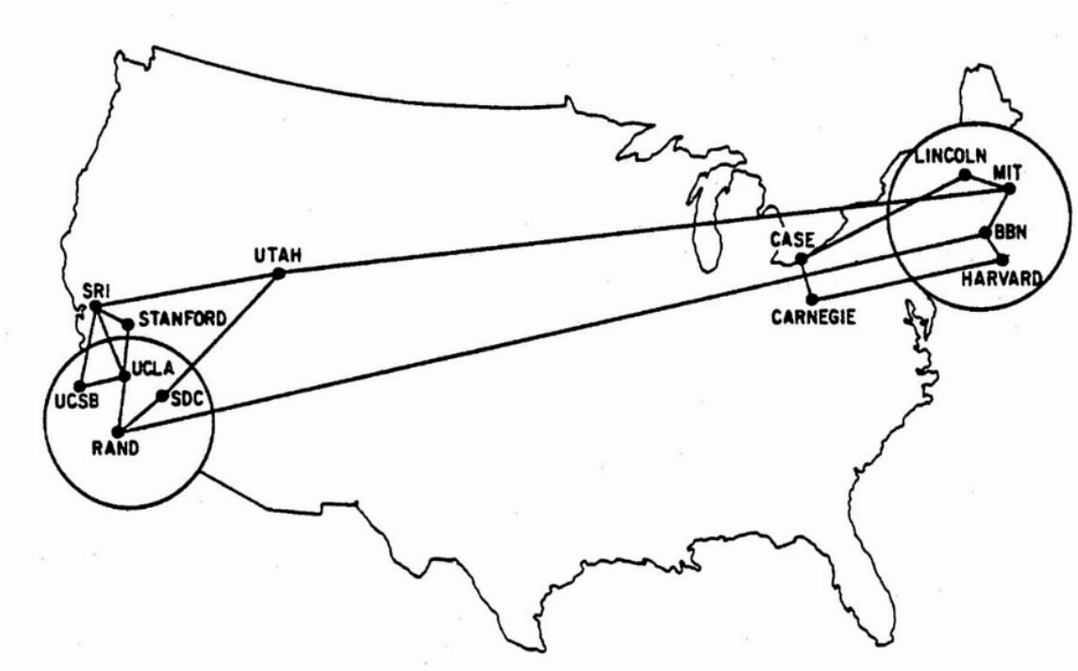
- How do we store data with multiple interconnected associations?
- An [station, intersection, city] can have multiple connections



Data Structures

- Let's use trees
- Start with UCLA as the root
- Recursively add all chidden

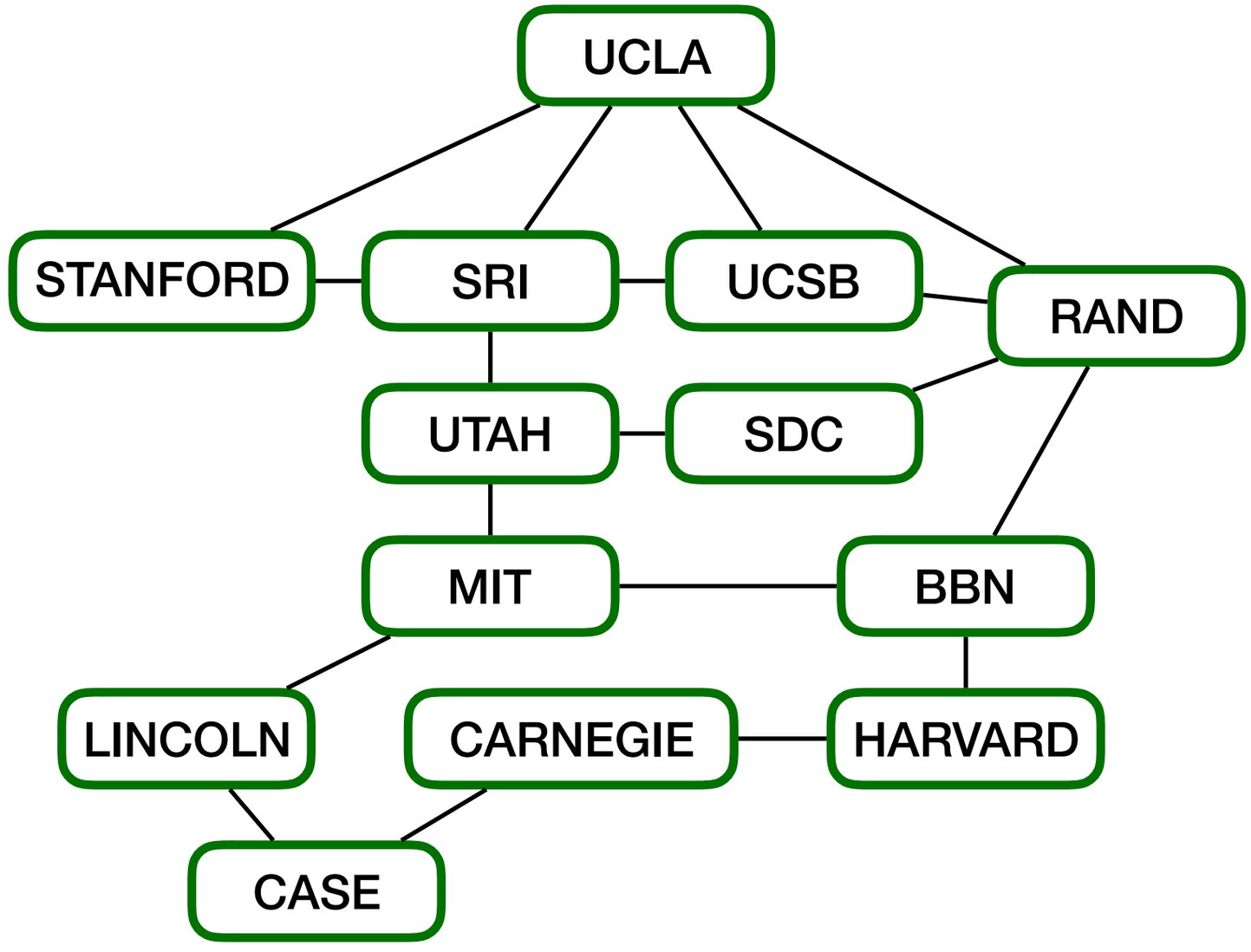


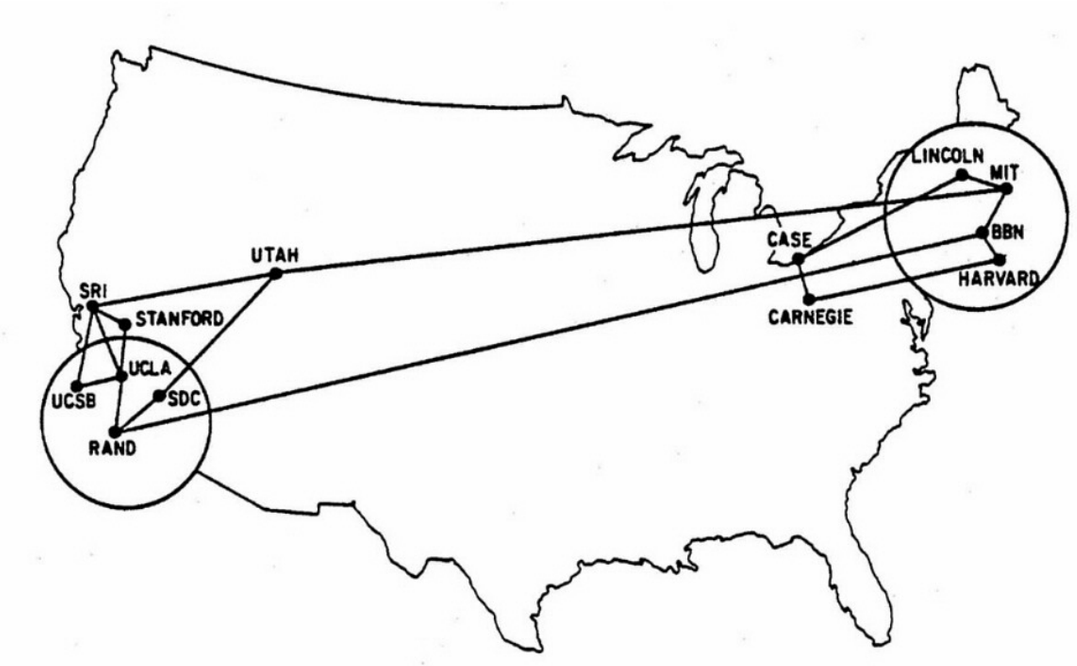


- Oops
  - We have duplicates in our data structure

Data Structures

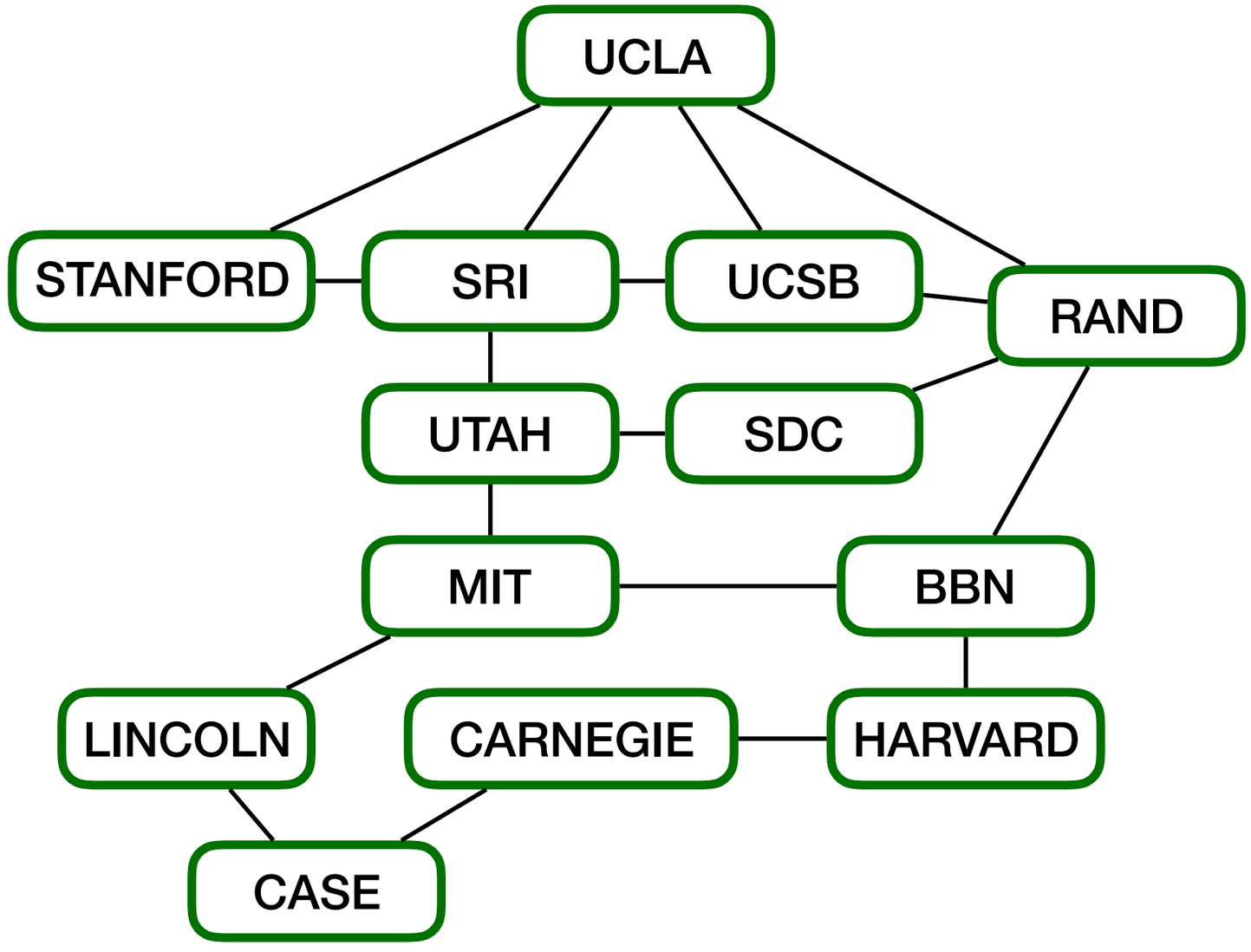
- Let's try again
  - When we try to add a duplicate, add a reference to the existing node

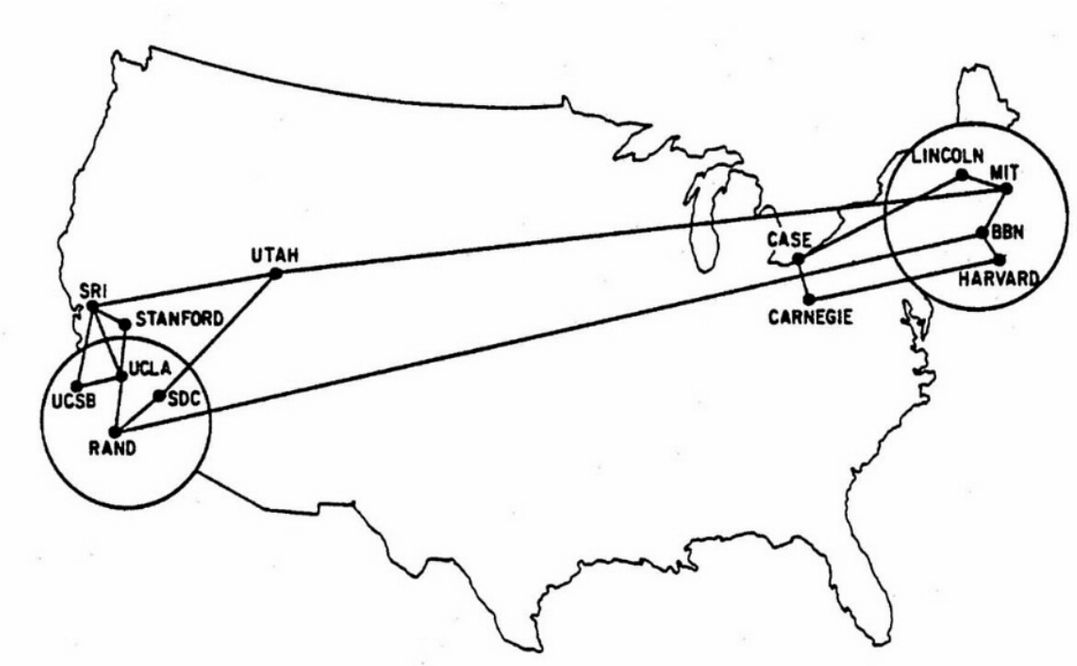




Graphs

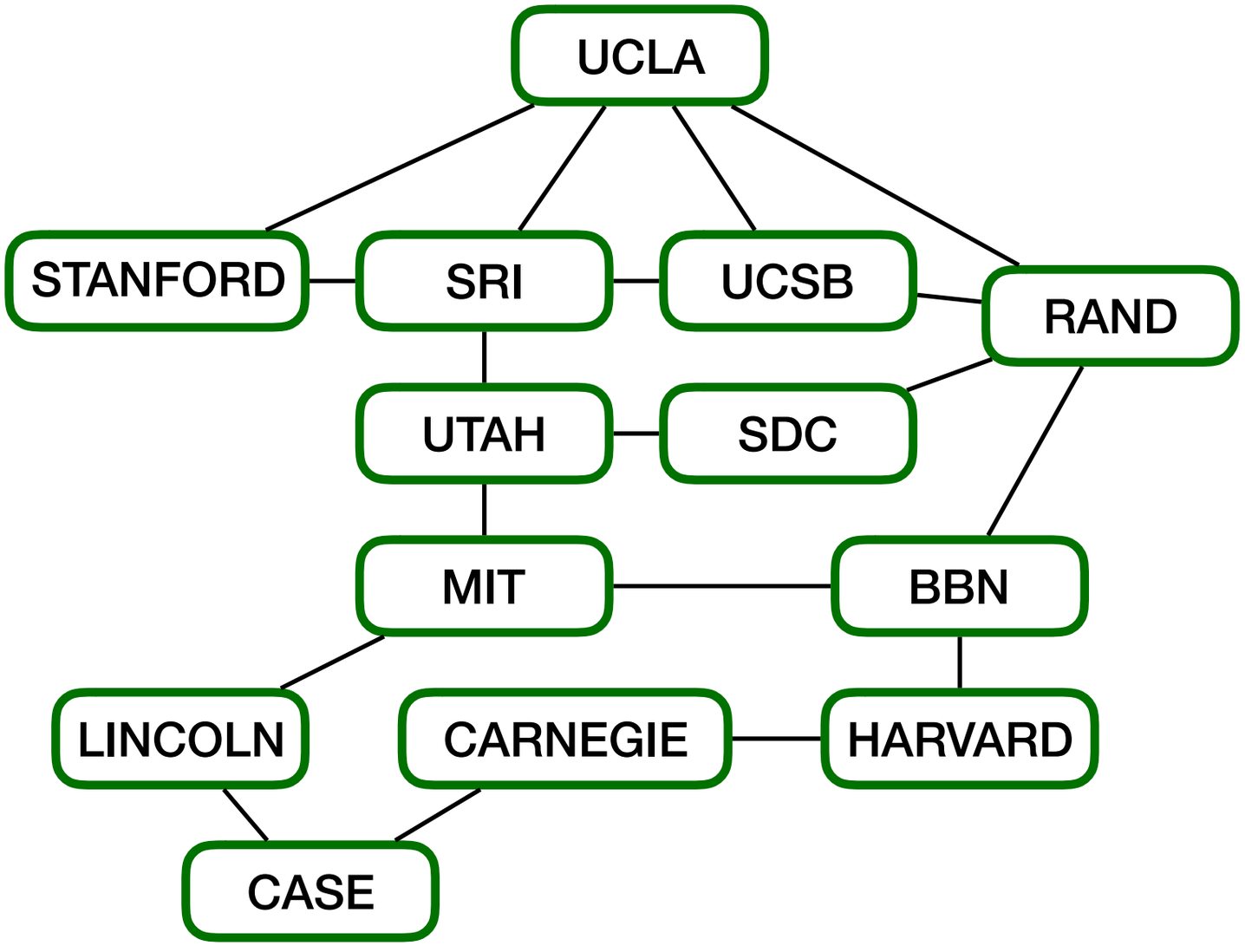
- This is a graph
- We can represent this data, but our tree traversals will get stuck in infinite recursion (Graphs can have cycles)
  - No leaves to terminate the recursion

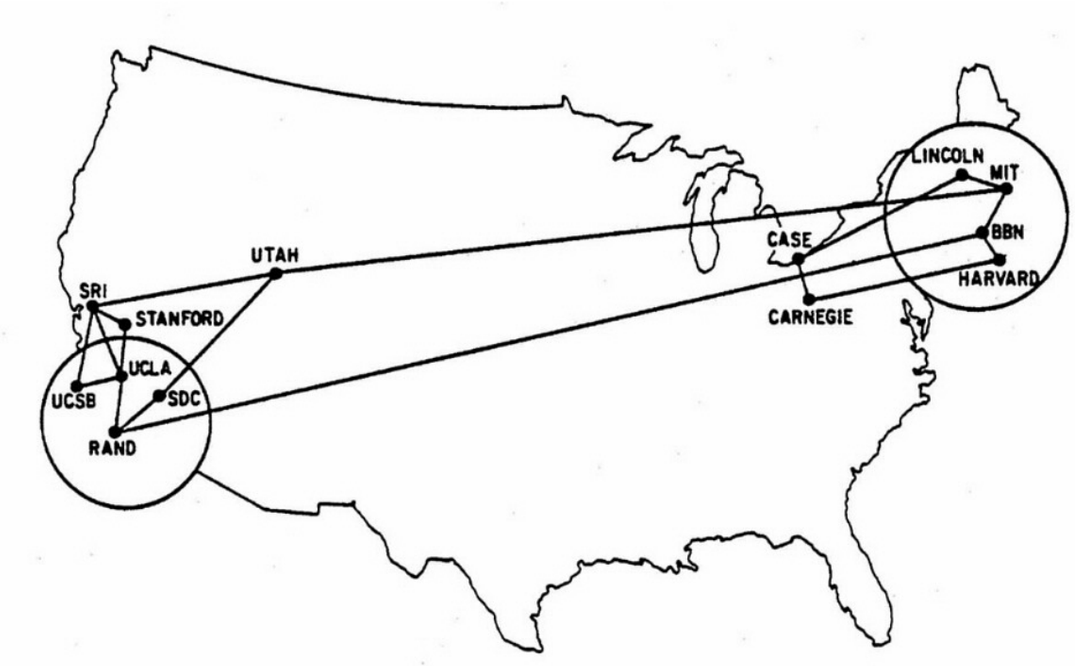




Graphs

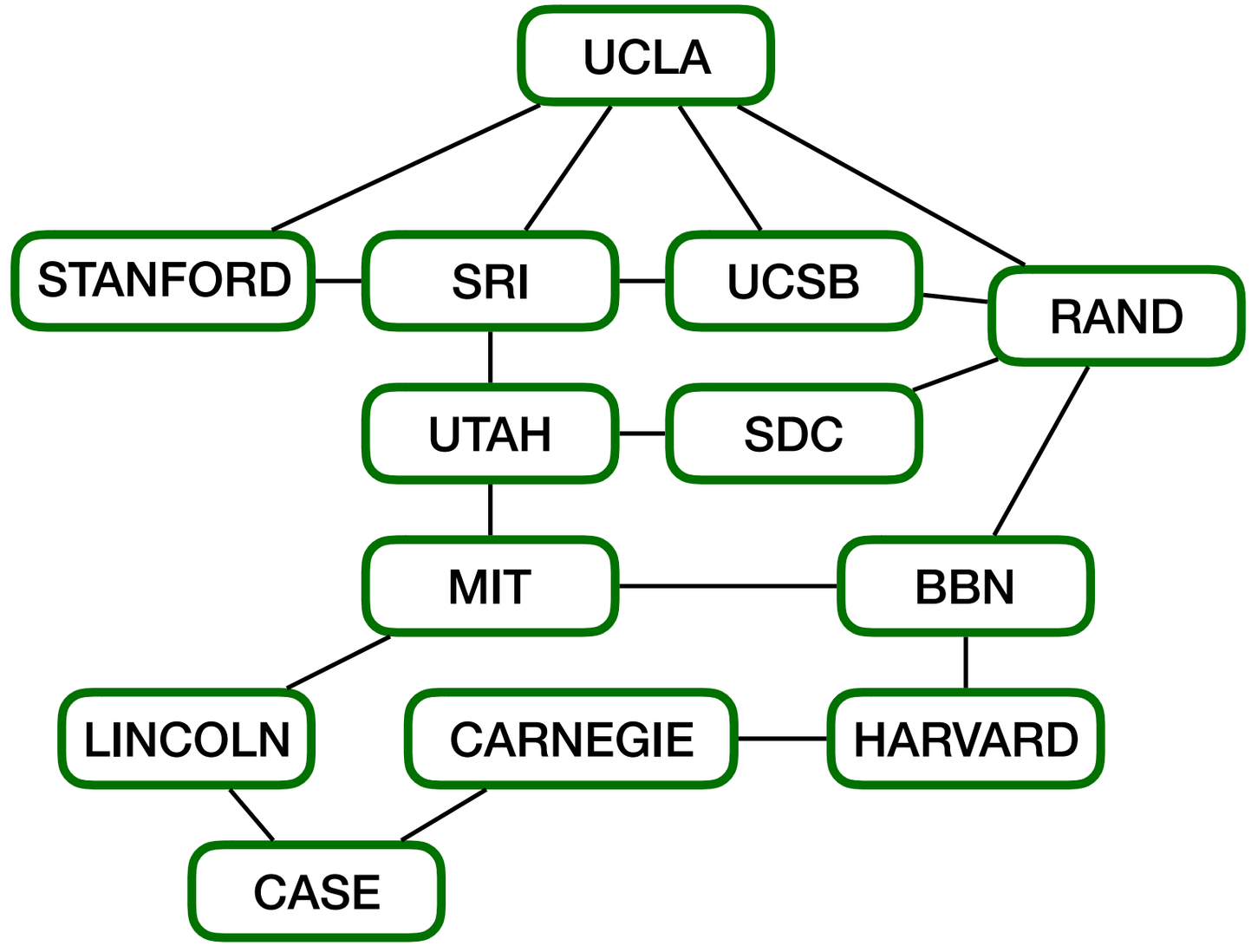
- We'll need a new way of representing this data structure and new algorithms to work with the data
- Store the node and edges

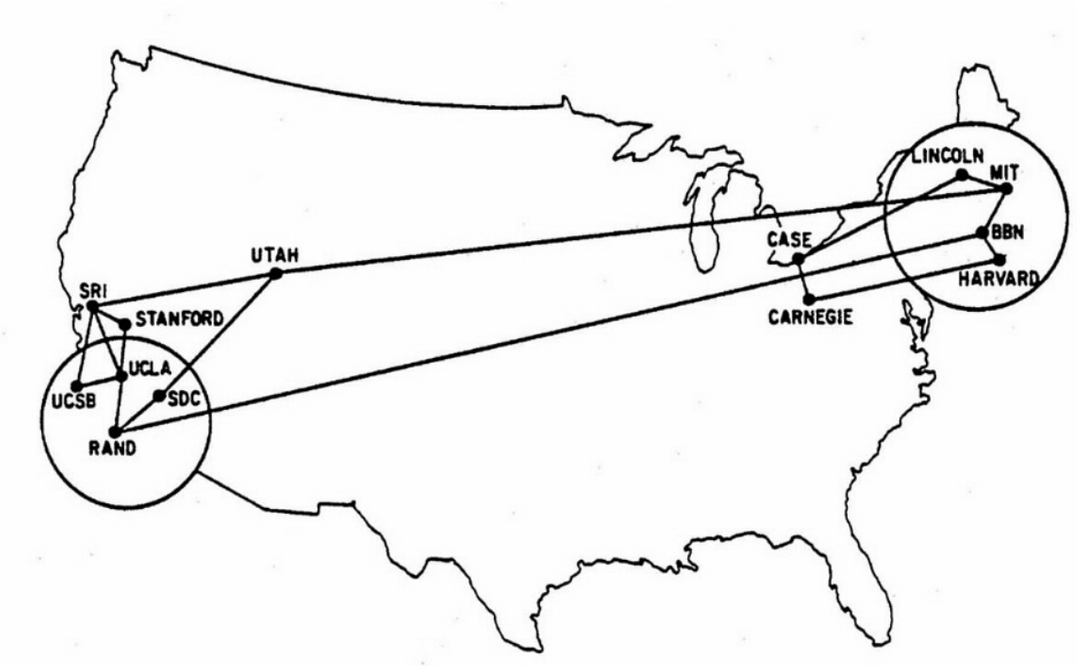




Graphs - Nodes and Edges

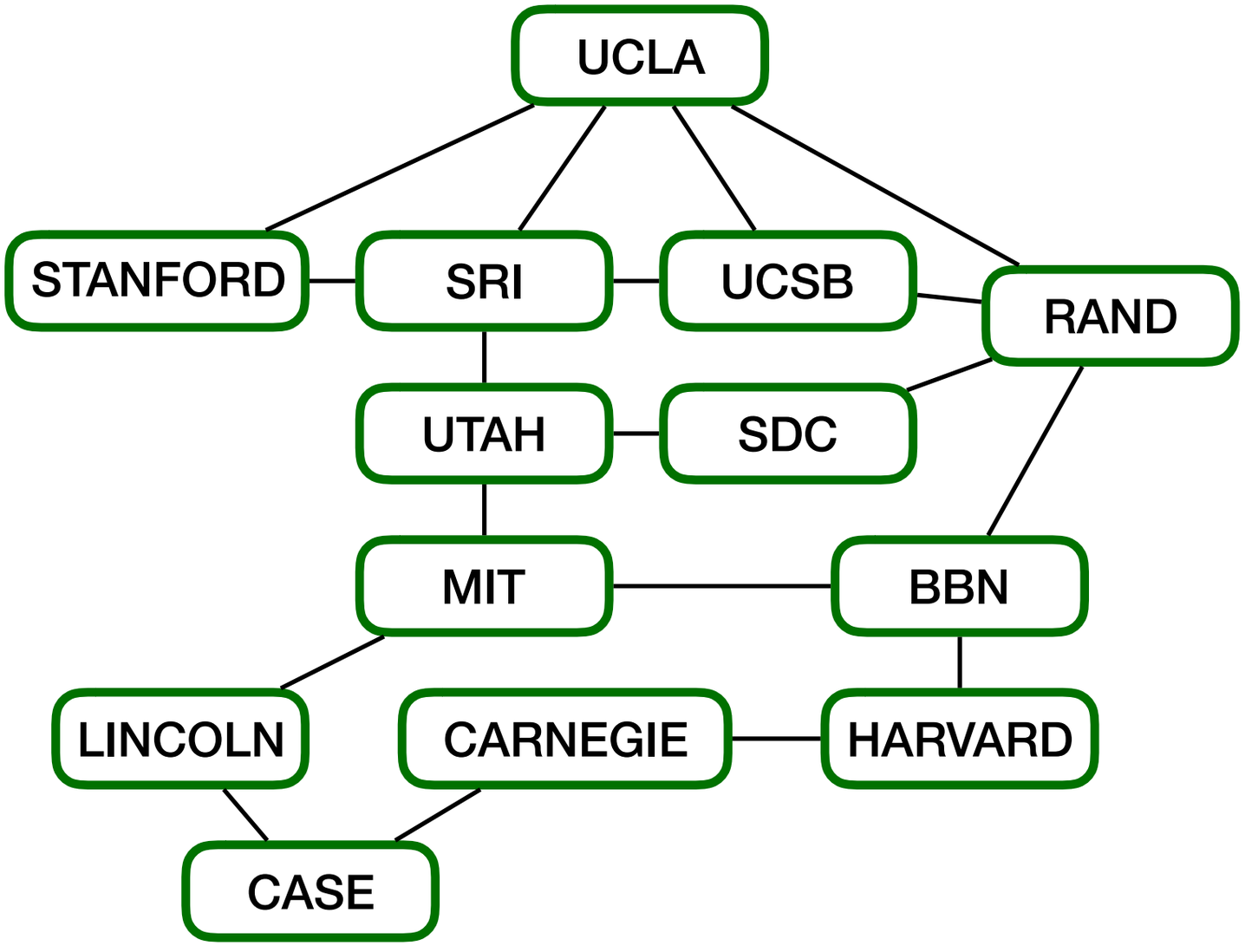
- Node: Each data element is stored in a node, similar to linked list and tree
- Edge: A connection between two nodes





Graphs - Adjacency List

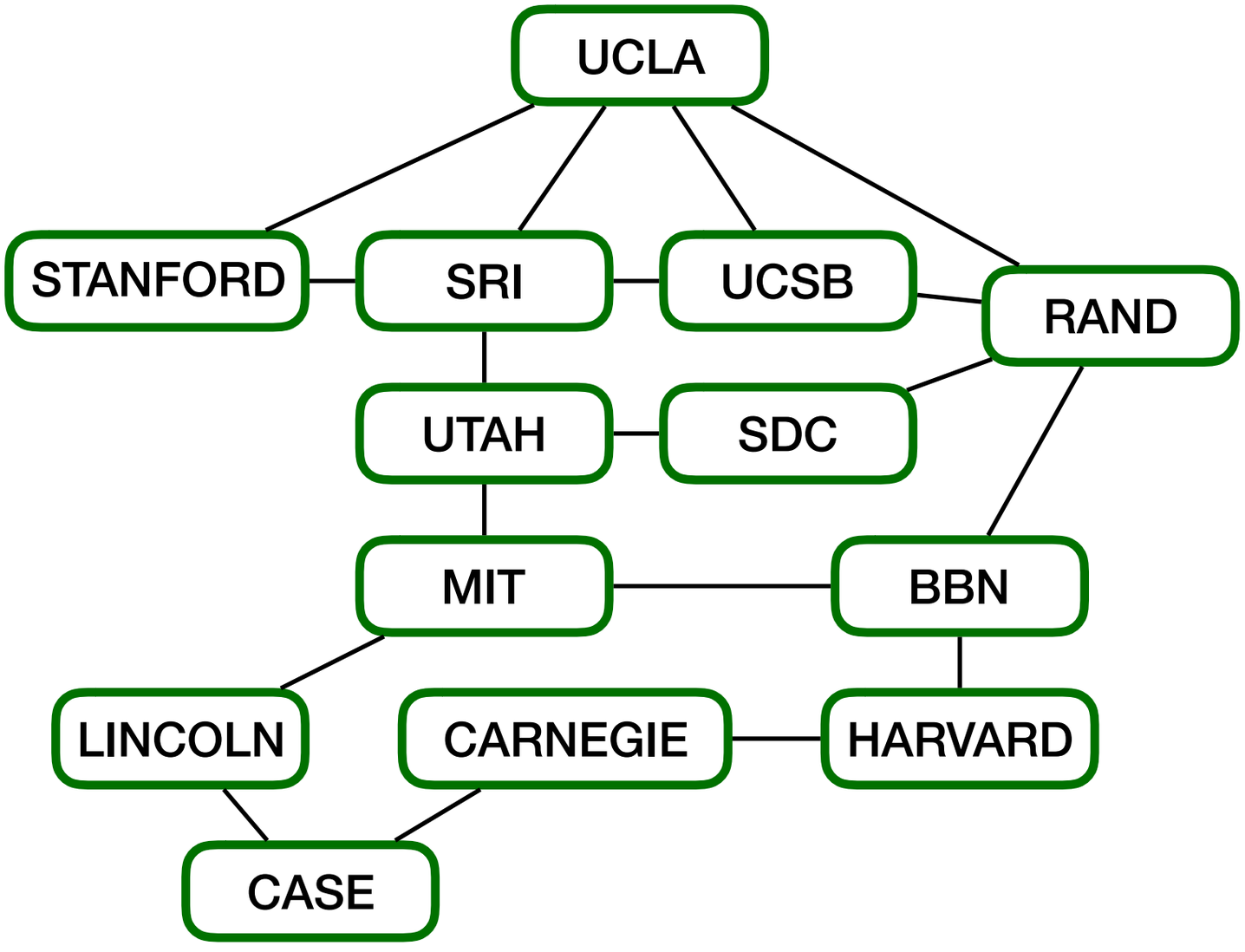
- A map of nodes to all nodes connected to it through an edge
- This is how we'll represent graphs



UCLA	STANFORD, SRI, UCSB, RAND
STANFORD	UCLA, SRI
SRI	UCLA, UCSB, UTAH, STANFORD
UCSB	UCLA, SRI, RAND
RAND	UCLA, UCSB, SDC, BBN
UTAH	SRI, SDC, MIT
SDC	UTAH, RAND
MIT	UTAH, BBN, LINCOLN
BBN	RAND, MIT, HAVARD
LINCOLN	MIT, CASE
CARNEGIE	CASE, HARVARD
HARVARD	BBN, CARNEGIE
CASE	LINCOLN, CARNEGIE

Paths

- A path is a sequence of nodes where each pair of adjacent nodes is connected by an edge
- ["UCLA", "SRI", "UTAH", "MIT", "BBN", "RAND"] is a path in this graph
- ["SRI", "UTAH", "BBN"] is not a path since UTAH and BBN are not connected by an edge



UCLA	STANFORD, SRI, UCSB, RAND
STANFORD	UCLA, SRI
SRI	UCLA, UCSB, UTAH, STANFORD
UCSB	UCLA, SRI, RAND
RAND	UCLA, UCSB, SDC, BBN
UTAH	SRI, SDC, MIT
SDC	UTAH, RAND
MIT	UTAH, BBN, LINCOLN
BBN	RAND, MIT, HAVARD
LINCOLN	MIT, CASE
CARNEGIE	CASE, HARVARD
HARVARD	BBN, CARNEGIE
CASE	LINCOLN, CARNEGIE

Lecture Question

Task: Verify a path in a graph

- In the Graph class write a method named isPath that takes a List of node indices as Ints representing a path in the graph
- Return a Boolean indicating whether or not the path is valid
- A path is valid if there is an edge between all adjacent nodes in the List

* This question will be open until midnight