# Union Find

## Dynamic Connectivity

It automatically establishes ad-hoc network connections between nodes on a network.

Given two nodes, p, and q, in a network, the two nodes have an equivalence relation.

* Reflexive: p is connected to p
* Symmetric: if p is connected to q, then q is connected to p
* Transitive: if p is connected to q and q is connected to r, then p is connected to r

Connect through other nodes. If not connected, do not connect directly.

## Examples

p and q represent two brain regions, and the pair represents a network connection between these two brain regions connected.

## Terminology

Objects (nodes) are called **sites**, node pairs are **connections,** and the equivalence classes are the connected components.

Components can be represented by one of the sites (nodes)

## Implementation

### General idea

In the “Algorithm” book, they considered three different implementations for Union Find, all based on the array data structure. The book abstracts the implementation of Union and Find routines in a base class called UnionFind and defines it in three different implementations. In our python implementation, we mimicked this behavior.   
Data Structure  
Arrays: where the index of an array represents a node, and the value on that index is the connected component.