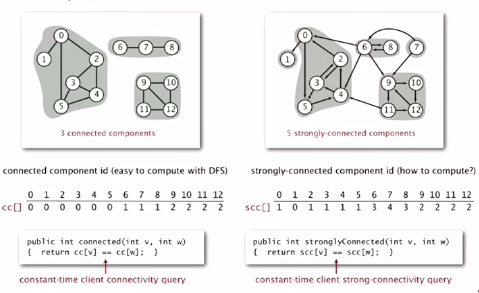
Strong Components

Strongly-connected components: vertices v and w are strongly connected if there is a directed path from v to w and a directed path from w to v.

Key property: strong connectivity is an equivalence relation:

* V is strongly connected to v
* If v is strongly connected to w, then w is strongly connected to v
* If v is strongly connected to w and w to x, then v is strongly connected to x

A strong component is a maximal subset of strongly-connected components.

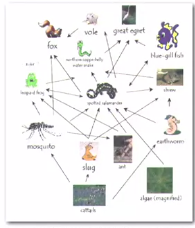


Design pattern is the same

Strongly connected component applications

Ecological food webs (showing which animals eat which)

Food web graph: vertex = species; edge = producer to consumer



Strong component: subset of species with common energy flow

Software module dependency graph

* Vertex = software module; edge = from module to dependency



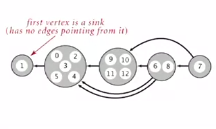
Strong component: subset of mutually interacting modules

Approach 1: package strong components together  
Approach 2: Use to improve design

Kosaraju-Sharir algorithm

Reverse graph: Strong components in G are same as in GR

Kernel DAG: Contract each strong component into a single vertex



Idea:

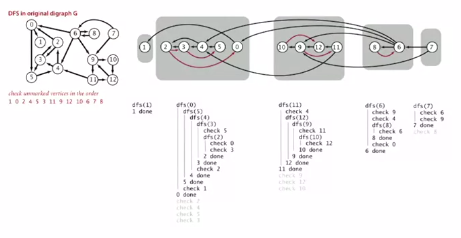
* Compute topological order (reverse postorder) in kernel DAG
* Run DFS, considering vertices in reverse topological order

Kosaraju-Sharir

* Phase 1: Compute reverse postorder in reverse of graph GR
* Phase 2: Run DFS in G, visiting unmarked vertices in reverse postorder of GR

Simple but mysterious algorithm for computing strong components

* Phase 1: run DFS on revered Graph to compute reverse postorder
* Phase 2: run DFS on Graph, considering vertices in order given by fi



Proposition: computs the strong components of a digraph in time proportional to E + V

Proof:

* Running time: bottleneck is running DFS twice (and computing reverse Graph)
* Correctness: tricky… see testbook
* Implementation: easy

Compare implementation of connected components algo in undirected graph (DFS)



Implementation of K-S strongly connected components is VERY SIMILAR



Differences:

* Name (obviously)
* Create new order with a DFS of reverse graph
* Compute DFS in the above order