Densest subgraph

Maximilien Danisch

LIP6 – CNRS and Sobonne University

first_name.last_name@lip6.fr

Outline

1. Motivations
2. k-core decomposition
   - Definition
   - Algorithm
   - Properties and applications

Motivations

- Real-world graphs are very large $\longrightarrow$ focusing on a smaller part of the graph
- Finding “interesting” subgraphs
- “Mining” the input graph

Definition: The densest subgraph is the maximum subgraph maximizing the ratio between the number of edges and the number of nodes.

Properties: The densest subgraph can be found in polynomial time. In general, in real-world graphs, it is much “denser” and much smaller than the original graph.

Outline

1. Motivations

2. k-core decomposition
   - Definition
   - Algorithm
   - Properties and applications

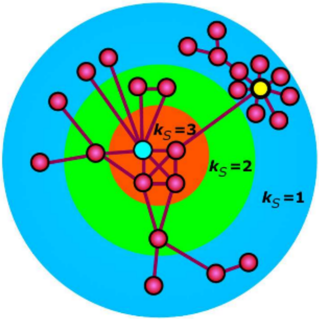
Definition

Definition: The k-core of a graph is the maximum subgraph such that each node has degree k or more.

Definition: The core value of a node u is the maximum number $c(u)$ such that the node u belongs to the $c(u)$-core.

Definition: The core value of a graph is the maximum number c such that a c-core exists.

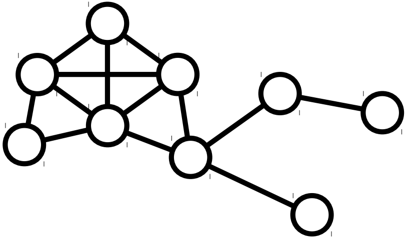
Definition: The k-core decomposition is the collection of nested k-cores for k from 1 to c.



Core decomposition

Algorithm Core decomposition

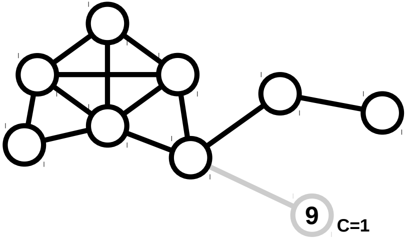
1: $i \leftarrow n$, $c \leftarrow 0$
2: **while** $V(G) \neq \emptyset$ **do**
3: Let v be a node with minimum degree in G
4: $c \leftarrow \max(c, d_G(v))$
5: $V(G) \leftarrow V(G) \setminus \{v\}$
6: $E(G) \leftarrow E(G) \setminus \Delta(v)$
7: $\eta(v) = i$
8: $i \leftarrow i - 1$



Core decomposition

Algorithm Core decomposition

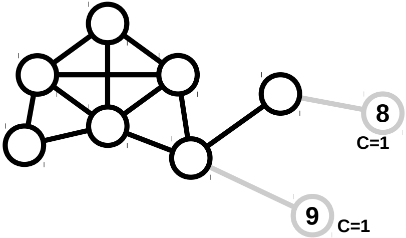
1: $i \leftarrow n, c \leftarrow 0$
2: **while** $V(G) \neq \emptyset$ **do**
3: Let v be a node with minimum degree in G
4: $c \leftarrow \max(c, d_G(v))$
5: $V(G) \leftarrow V(G) \setminus \{v\}$
6: $E(G) \leftarrow E(G) \setminus \Delta(v)$
7: $\eta(v) = i$
8: $i \leftarrow i - 1$



Core decomposition

Algorithm Core decomposition

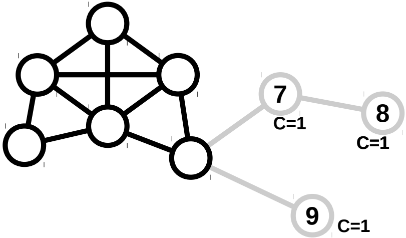
$i \leftarrow n$, $c \leftarrow 0$
while $V(G) \neq \emptyset$ **do**
  Let v be a node with minimum degree in G
  $c \leftarrow \max(c, d_G(v))$
  $V(G) \leftarrow V(G) \setminus \{v\}$
  $E(G) \leftarrow E(G) \setminus \Delta(v)$
  $\eta(v) = i$
  $i \leftarrow i - 1$



Core decomposition

Algorithm Core decomposition

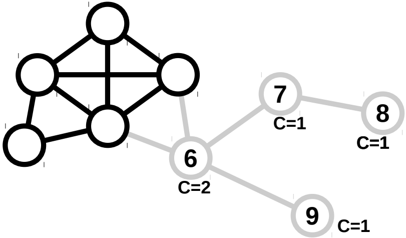
$i \leftarrow n$, $c \leftarrow 0$
while $V(G) \neq \emptyset$ **do**
  Let v be a node with minimum degree in G
  $c \leftarrow \max(c, d_G(v))$
  $V(G) \leftarrow V(G) \setminus \{v\}$
  $E(G) \leftarrow E(G) \setminus \Delta(v)$
  $\eta(v) = i$
  $i \leftarrow i - 1$



Core decomposition

Algorithm Core decomposition

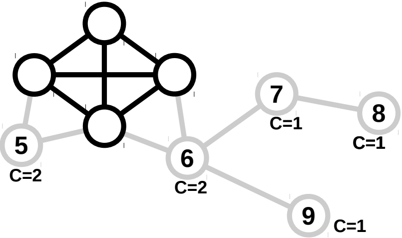
$i \leftarrow n, c \leftarrow 0$
while $V(G) \neq \emptyset$ **do**
  Let v be a node with minimum degree in G
  $c \leftarrow \max(c, d_G(v))$
  $V(G) \leftarrow V(G) \setminus \{v\}$
  $E(G) \leftarrow E(G) \setminus \Delta(v)$
  $\eta(v) = i$
  $i \leftarrow i - 1$



Core decomposition

Algorithm Core decomposition

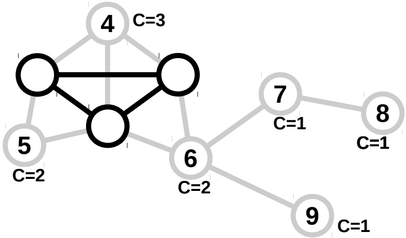
$i \leftarrow n$, $c \leftarrow 0$
while $V(G) \neq \emptyset$ **do**
  Let v be a node with minimum degree in G
  $c \leftarrow \max(c, d_G(v))$
  $V(G) \leftarrow V(G) \setminus \{v\}$
  $E(G) \leftarrow E(G) \setminus \Delta(v)$
  $\eta(v) = i$
  $i \leftarrow i - 1$



Core decomposition

Algorithm Core decomposition

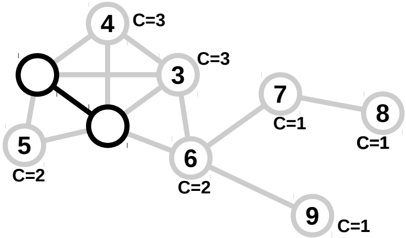
1: $i \leftarrow n$, $c \leftarrow 0$
2: **while** $V(G) \neq \emptyset$ **do**
3: Let v be a node with minimum degree in G
4: $c \leftarrow \max(c, d_G(v))$
5: $V(G) \leftarrow V(G) \setminus \{v\}$
6: $E(G) \leftarrow E(G) \setminus \Delta(v)$
7: $\eta(v) = i$
8: $i \leftarrow i - 1$



Core decomposition

Algorithm Core decomposition

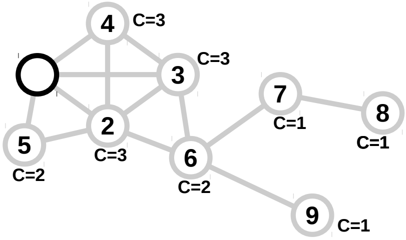
1: $i \leftarrow n$, $c \leftarrow 0$
2: **while** $V(G) \neq \emptyset$ **do**
3: Let v be a node with minimum degree in G
4: $c \leftarrow \max(c, d_G(v))$
5: $V(G) \leftarrow V(G) \setminus \{v\}$
6: $E(G) \leftarrow E(G) \setminus \Delta(v)$
7: $\eta(v) = i$
8: $i \leftarrow i - 1$



Core decomposition

Algorithm Core decomposition

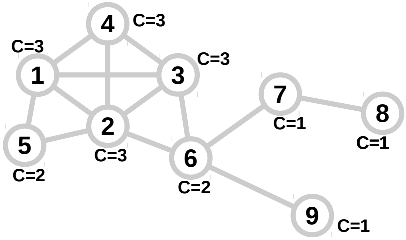
$i \leftarrow n, c \leftarrow 0$
while $V(G) \neq \emptyset$ **do**
  Let v be a node with minimum degree in G
  $c \leftarrow \max(c, d_G(v))$
  $V(G) \leftarrow V(G) \setminus \{v\}$
  $E(G) \leftarrow E(G) \setminus \Delta(v)$
  $\eta(v) = i$
  $i \leftarrow i - 1$



Core decomposition

Algorithm Core decomposition

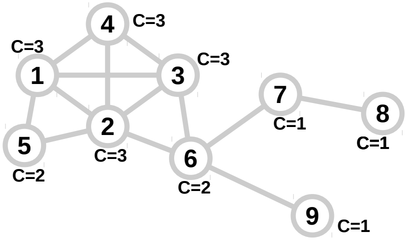
$i \leftarrow n$, $c \leftarrow 0$
while $V(G) \neq \emptyset$ **do**
  Let v be a node with minimum degree in G
  $c \leftarrow \max(c, d_G(v))$
  $V(G) \leftarrow V(G) \setminus \{v\}$
  $E(G) \leftarrow E(G) \setminus \Delta(v)$
  $\eta(v) = i$
  $i \leftarrow i - 1$



Core decomposition

Algorithm Core decomposition

$i \leftarrow n$, $c \leftarrow 0$
while $V(G) \neq \emptyset$ **do**
  Let v be a node with minimum degree in G
  $c \leftarrow \max(c, d_G(v))$
  $V(G) \leftarrow V(G) \setminus \{v\}$
  $E(G) \leftarrow E(G) \setminus \Delta(v)$
  $\eta(v) = i$
  $i \leftarrow i - 1$



Exercise: Which datastructures should be used? And what is the complexity of the Algorithm?

Relation to densest subgraph

Exercise: Try to guess some relation between the densest subgraph and the k-core ordering

Relation to densest subgraph

Exercise: Try to guess some relation between the densest subgraph and the k-core ordering

Theorem (not proven here)**:** A densest prefix is a 2-approximation of the densest subgraph.

Relation to densest subgraph

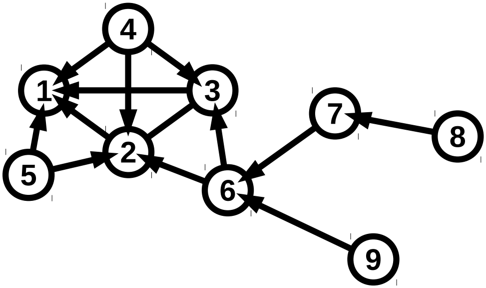
Exercise: Try to guess some relation between the densest subgraph and the k-core ordering

Theorem (not proven here)**:** A densest prefix is a 2-approximation of the densest subgraph.

Exercise: Given an ordering of the nodes, give an efficient algorithm to compute a densest prefix.

Making faster algorithms: induced DAG

DAG stands for Directed Acyclic Graph:

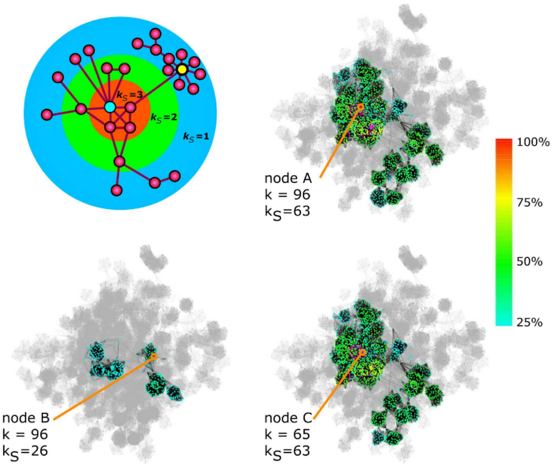


Exercise: What is the maximum out-degree of such a DAG?

Exercise: What is the running time of our triangle-listing algorithm (c.f. course 2) if the core ordering is used? Note that, in general, in real-world graphs $c << n$.

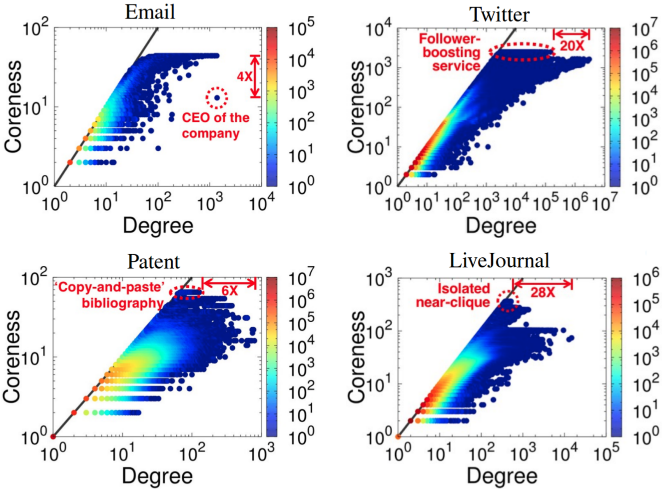
Finding best spreaders

Identification of influential spreaders in complex networks - *Kitsak et al. 2010*



Finding anomalous nodes

CoreScope: Graph Mining Using k-Core Analysis - *Shin et al. 2016*



For more on k-core check the tutorial at:

http://fragkiskos.me/papers/Tutorial_Slides_ICDM_2016.pdf