Qiskit Mentorship Program

Midterm checkpoint

Project: Good first issues on Retworkx

Mentor: Matthew Treinish

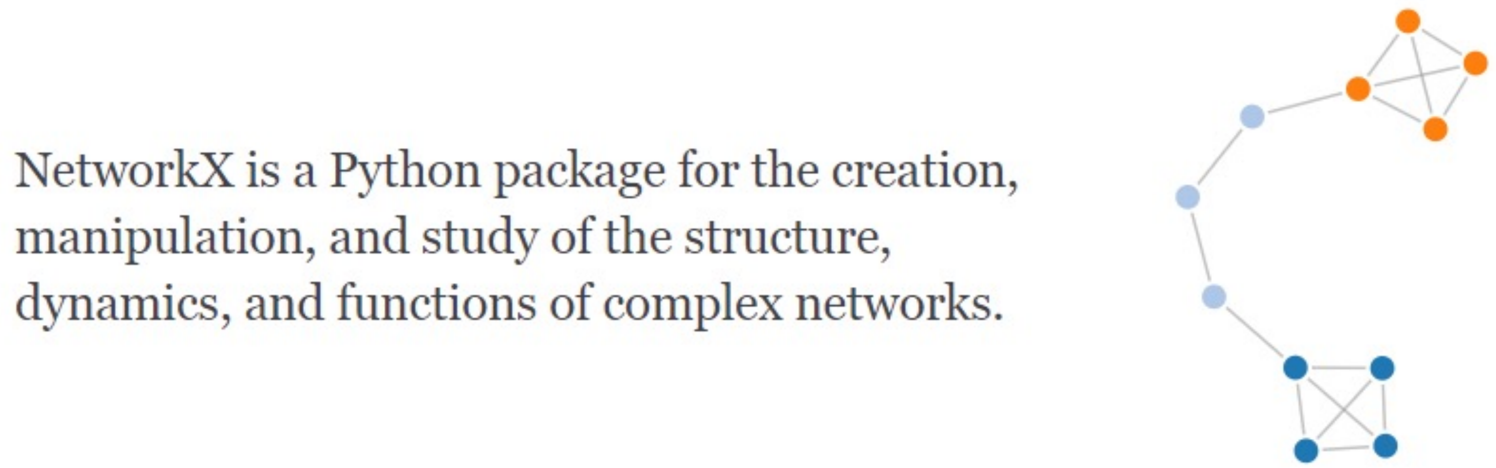
Team: Nahum Rosa Cruz Sá, Georgios Tsilimigkounakis, Chris (Jielun) Chen, Ivan de Avila Ribeiro Carvalho

Presenter: Chris (Jielun) Chen

Disclaimer: this is an individual presentation; the speaker does not represent the whole team.

NetworkX

Network Analysis in Python



NetworkX is a Python package for the creation, manipulation, and study of the structure, dynamics, and functions of complex networks.

networkx in Rust → retworkx

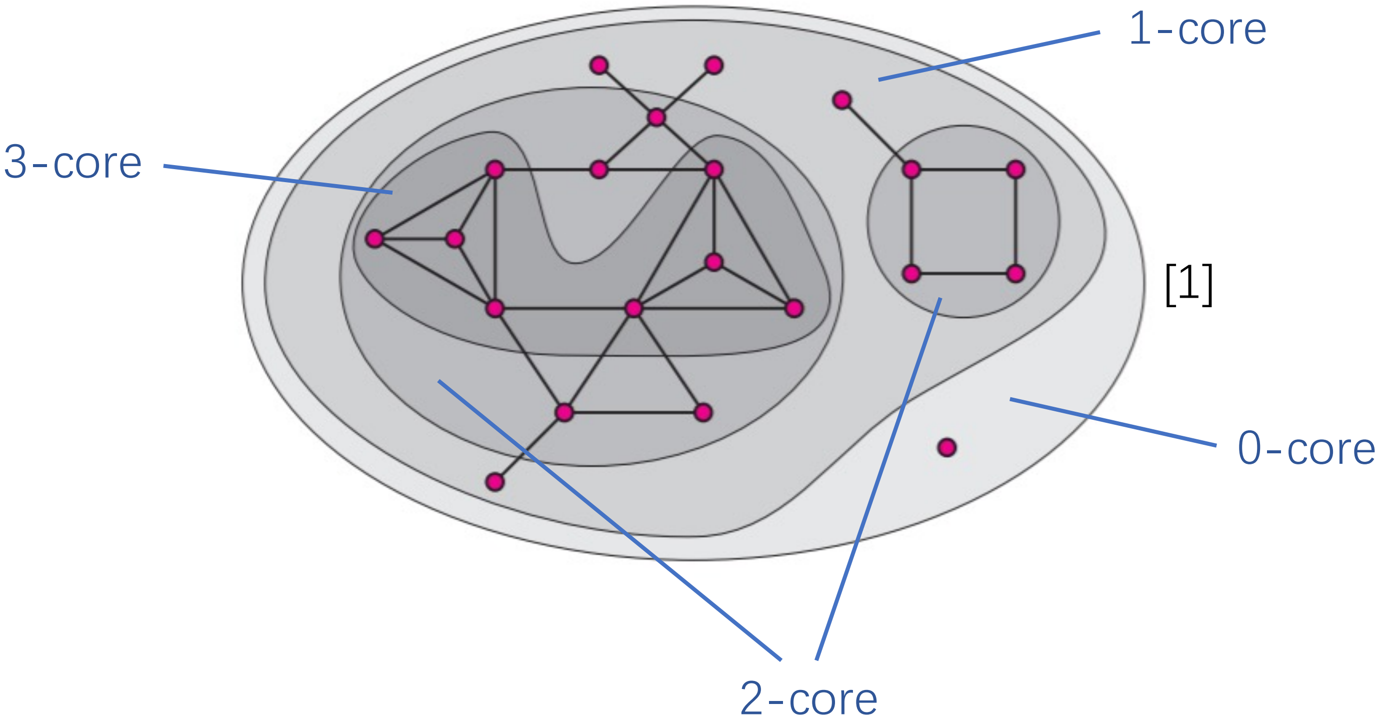
All graph usage in qiskit!

Why Rust

- 1. Memory safety
  - Does not permit null pointers, dangling pointers, or data races
  - Data values can be initialized only through a fixed set of forms
- 2. Memory management
  - Ownership
  - Low-level control
  - No need for garbage collection
- 3. High performance
  - Speed on par with C++; much faster than Python
  - Low-risk parallel computation

Merged issue: **k-core** of a graph

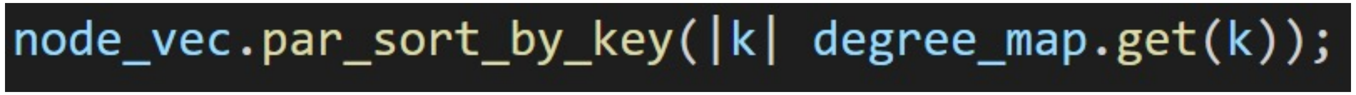
A **k-core** is a maximal subgraph that contains nodes of degree k or more. For directed graphs, the degree is in_degree + out_degree.



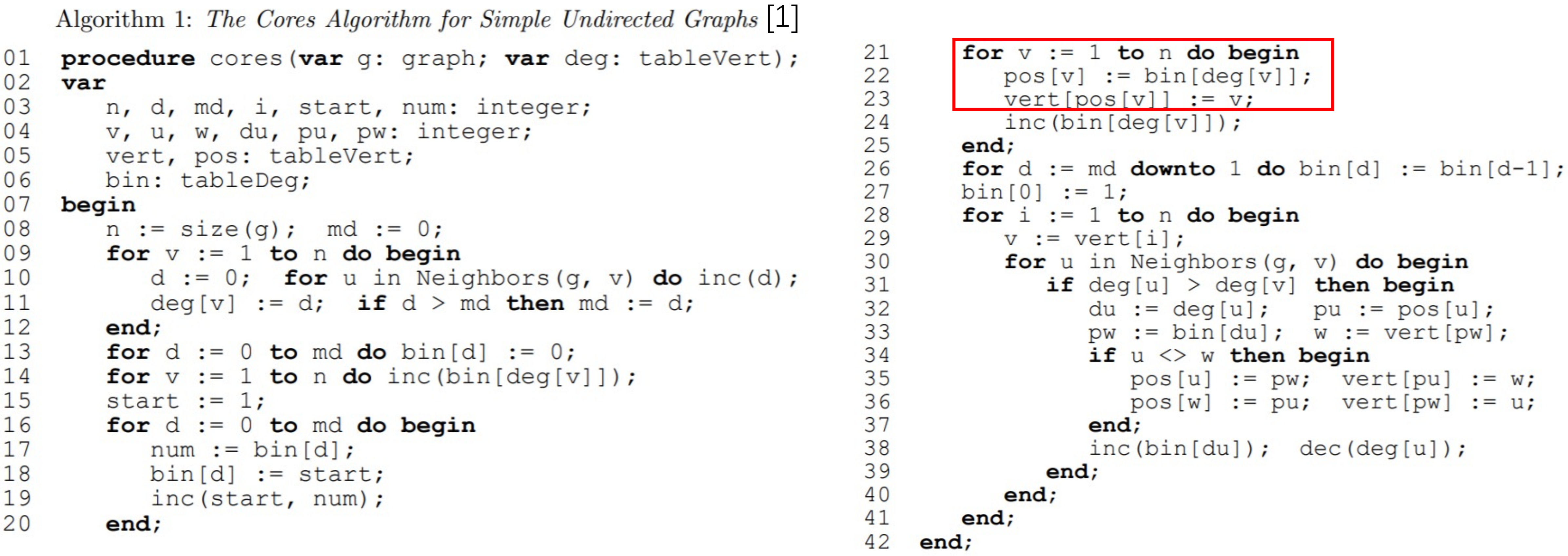
[1] Batagelj, Vladimir & Zaveršnik, Matjaž. (2003). An O(m) Algorithm for Cores Decomposition of Networks. CoRR. cs.DS/0310049.

Merged issue: **k-core** of a graph

rayon::slice::ParallelSliceMut



```
node_vec.par_sort_by_key(|k| degree_map.get(k));
```

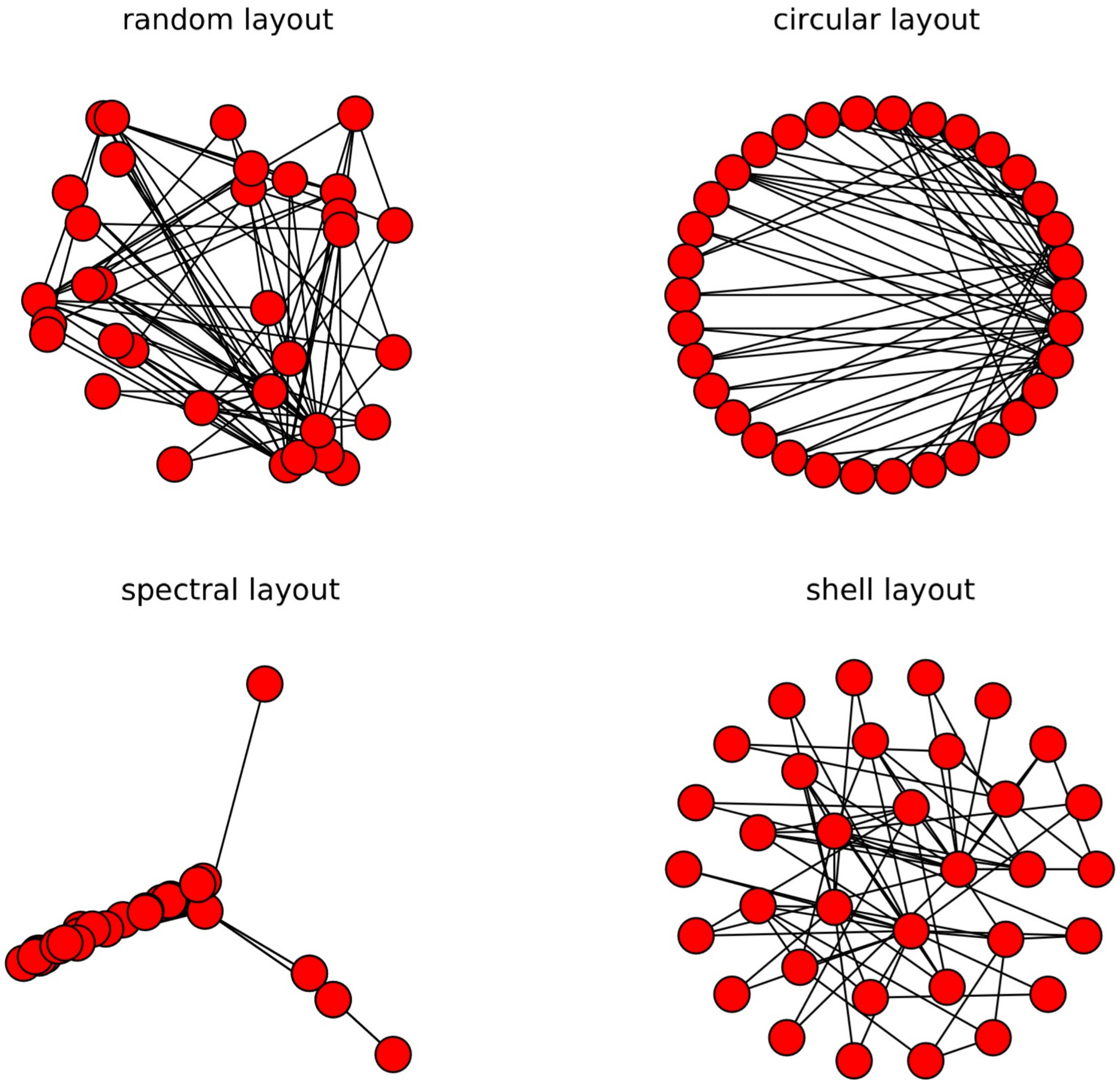


Algorithm 1: *The Cores Algorithm for Simple Undirected Graphs* [1]

```
procedure cores(var g: graph; var deg: tableVert);
var
  n, d, md, i, start, num: integer;
  v, u, w, du, pu, pw: integer;
  vert, pos: tableVert;
  bin: tableDeg;
begin
  n := size(g);  md := 0;
  for v := 1 to n do begin
    d := 0;  for u in Neighbors(g, v) do inc(d);
    deg[v] := d;  if d > md then md := d;
  end;
  for d := 0 to md do bin[d] := 0;
  for v := 1 to n do inc(bin[deg[v]]);
  start := 1;
  for d := 0 to md do begin
    num := bin[d];
    bin[d] := start;
    inc(start, num);
  end;
  for v := 1 to n do begin
    pos[v] := bin[deg[v]];
    vert[pos[v]] := v;
    inc(bin[deg[v]]);
  end;
  for d := md downto 1 do bin[d] := bin[d-1];
  bin[0] := 1;
  for i := 1 to n do begin
    v := vert[i];
    for u in Neighbors(g, v) do begin
      if deg[u] > deg[v] then begin
        du := deg[u];  pu := pos[u];
        pw := bin[du];  w := vert[pw];
        if u <> w then begin
          pos[u] := pw;  vert[pu] := w;
          pos[w] := pu;  vert[pw] := u;
        end;
        inc(bin[du]);  dec(deg[u]);
      end;
    end;
  end;
end;
```

[1] Batagelj, Vladimir & Zaveršnik, Matjaž. (2003). An O(m) Algorithm for Cores Decomposition of Networks. CoRR. cs.DS/0310049.

Issue in progress: **graph layouts**



random layout

circular layout

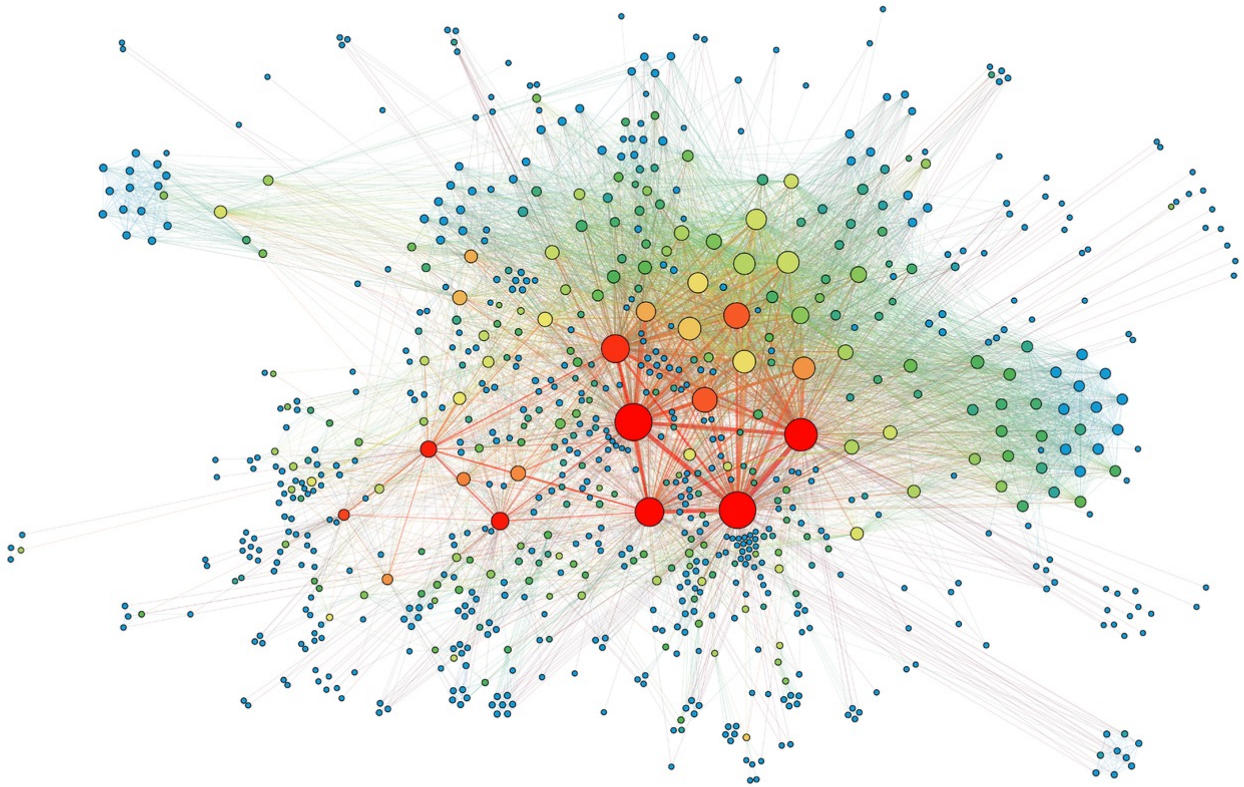
spectral layout

shell layout

Issue in progress: **graph layouts**

The algorithm simulates a force-directed representation of the network treating edges as springs holding nodes close, while treating nodes as repelling objects, sometimes called an anti-gravity force. Simulation continues until the positions are close to an equilibrium.

Fruchterman-Reingold force-directed algorithm.



@georgios-ts is working on the same issue.