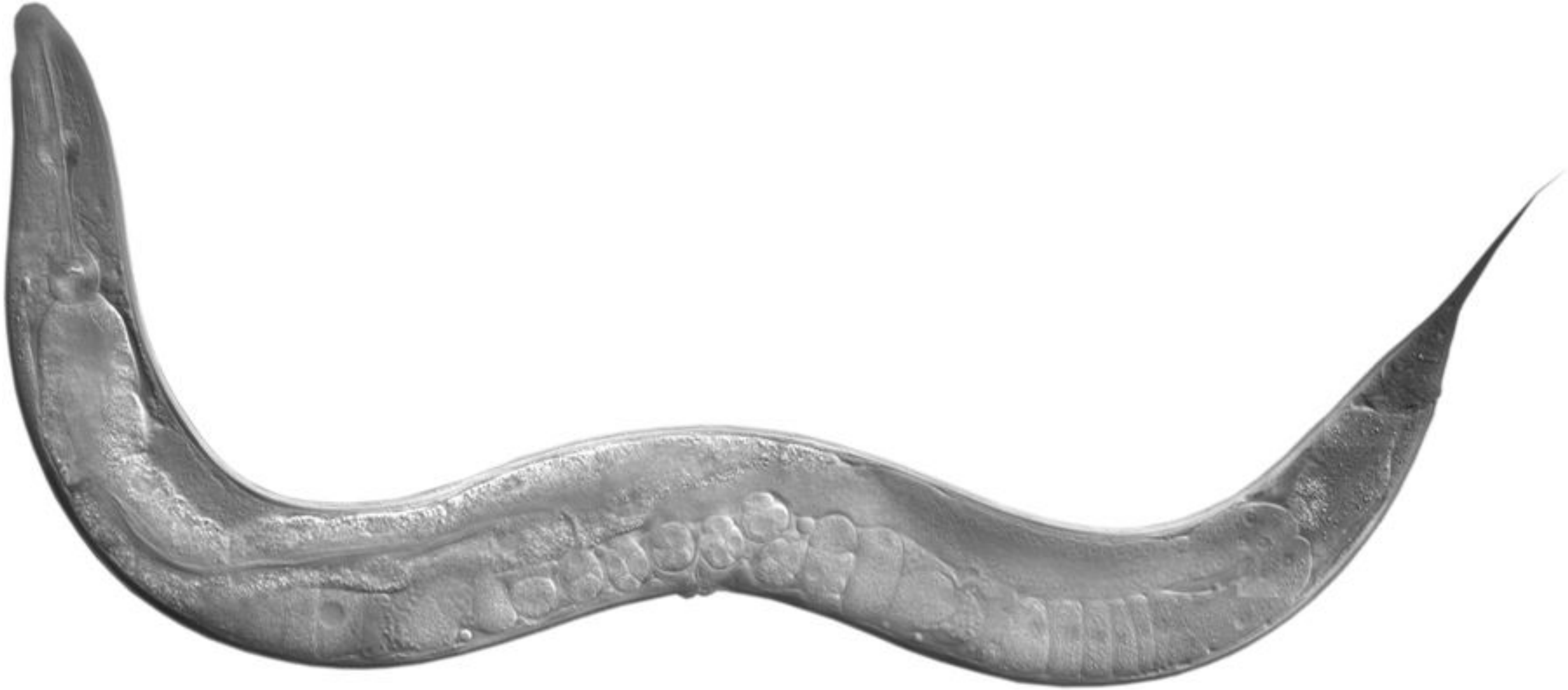
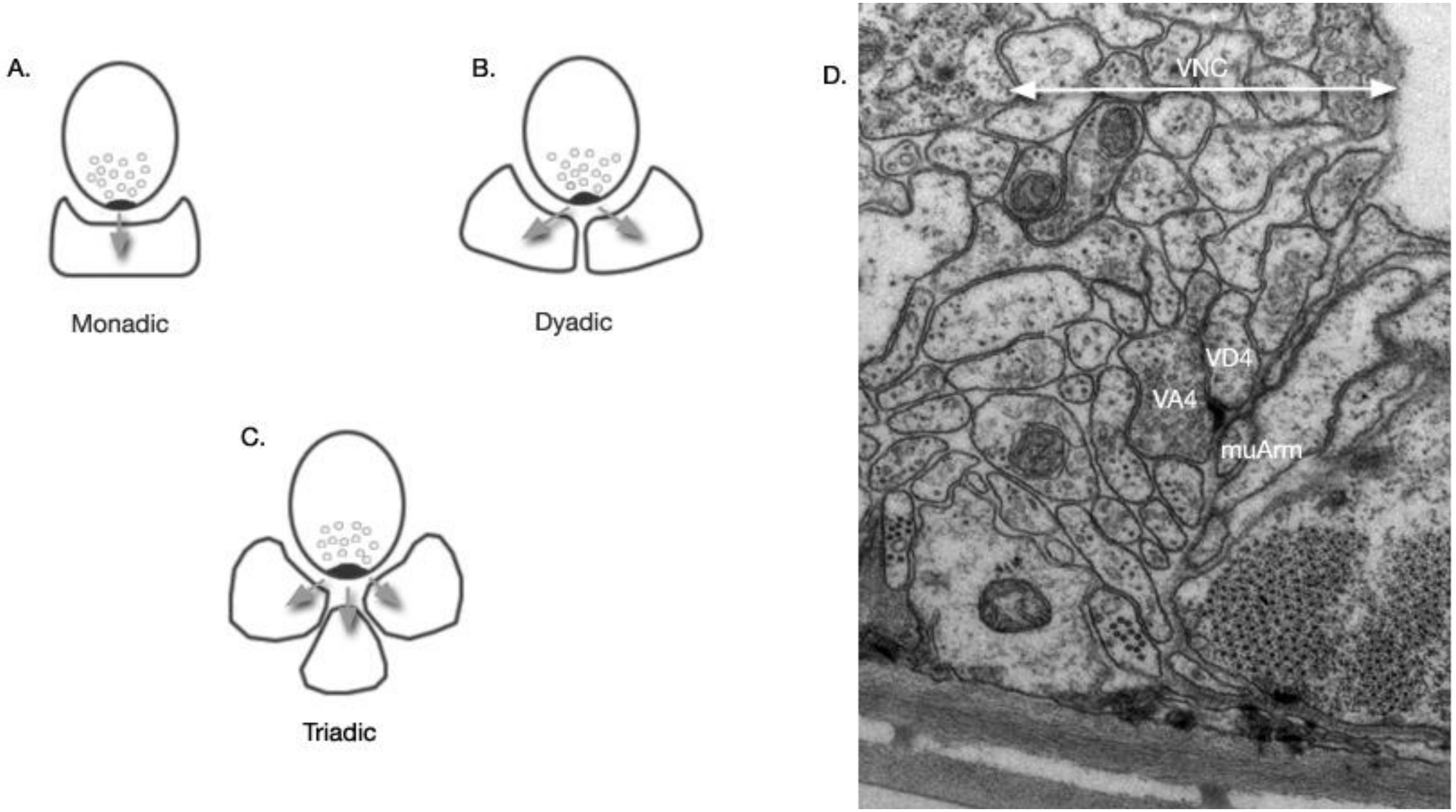


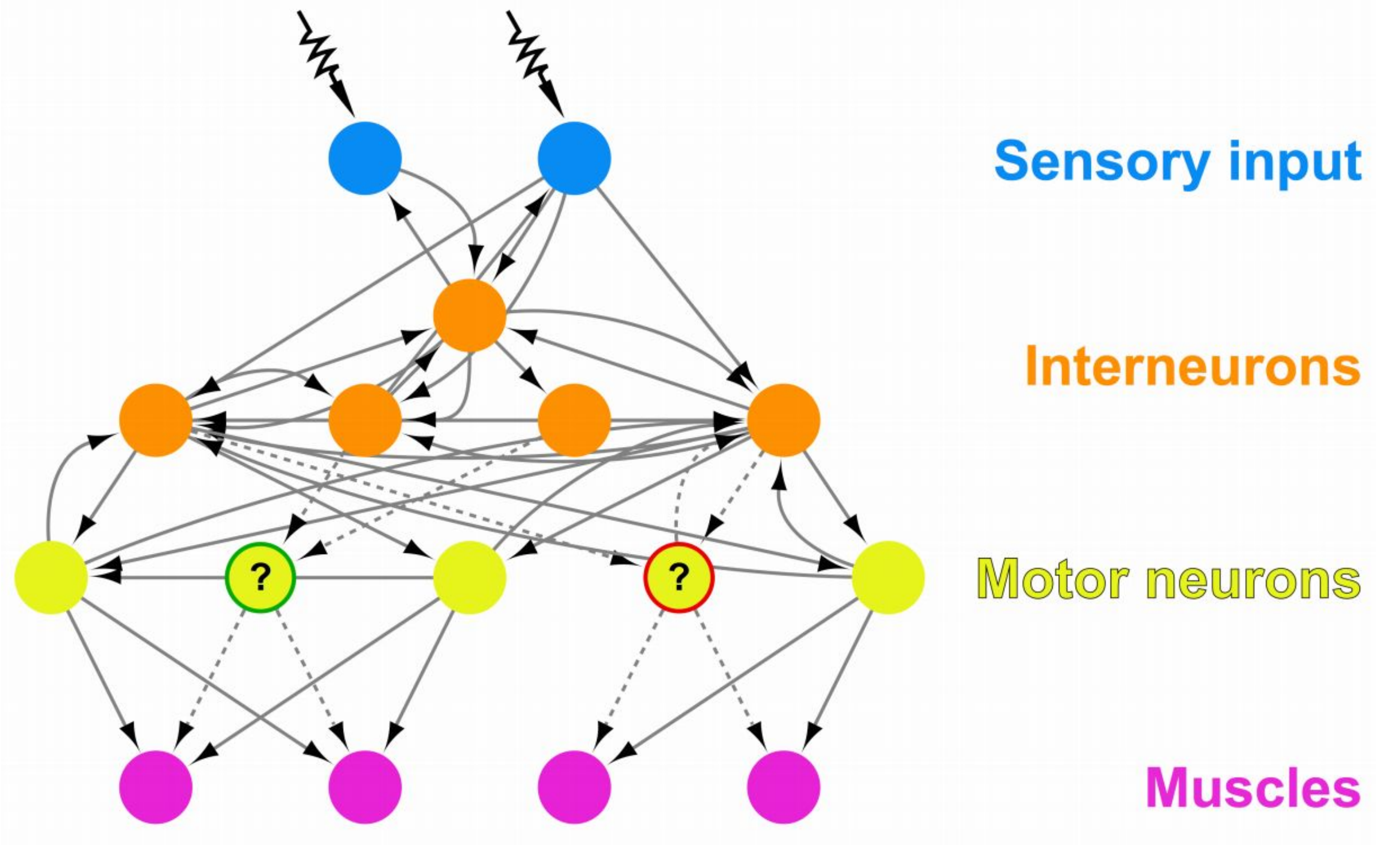
Louvain Community Detection in Connectomes

BY MARK HORENI

The Problem



The Problem



The Data

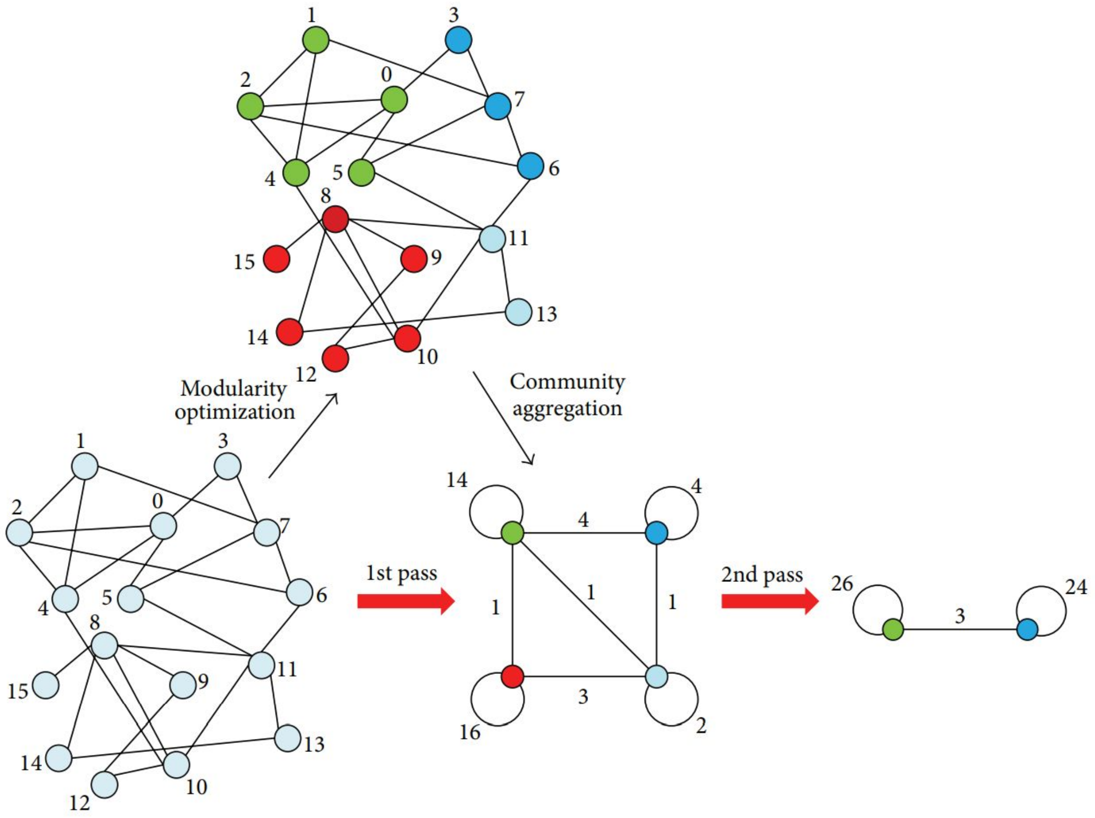
- C. elegans Connectome
  - 279 Nodes
  - 3225 Edges
- Mouse Retina
  - 1123 Nodes
  - 577,350 Edges
- Human Connectome
  - MRI Data
  - 277,345 Nodes
  - 64.4M Edges

Modularity

- Metric to determine densess of communities compared to null model
- Global Property
- Goal: Maximize Modularity
- Suffers from “Resolution Limit”

$$Q = \frac{1}{2m} \sum_{ij} \left[A_{ij} - \frac{k_i k_j}{2m} \right] \delta(c_i, c_j),$$

Louvain Visualized



V: a set of vertices
E: a set of edges
W: a set of weights of edges, initialized to 1
$G \Leftarrow (V, E, W)$
repeat
    $C \Leftarrow \{\{v_i\} | v_i \in G(V)\}$
    calculate current modularity Q_{cur}
    $Q_{new} \Leftarrow Q_{cur}$
    $Q_{old} \Leftarrow Q_{new}$
    repeat
        for $v_i \in V$ **do**
            $Q_{cur} \Leftarrow Q_{new}$
            remove v_i from its current community
            $N_{v_i} \Leftarrow \{c_k | v_i \in G(V), v_j \in c_k, e_{ij} \in G(E)\}$
            find $c_x \in N_{v_i}$ that has $max \Delta Q_{\{v_i\}, c_x} > 0$
            insert v_i into c_x
        end for
        calculate new modularity Q_{new}
    until no membership change or $Q_{new} = Q_{cur}$
    $V' \Leftarrow \{c_i | c_i \in C\}$
    $E' \Leftarrow \{e_{ij} | \forall e_{ij}$ if $v_i \in C_i, v_j \in C_j, and C_i \neq C_j\}$
    $W' \Leftarrow \{w_{ij} | \sum w_{ij}, \forall e_{ij}$ if $v_i \in C_i$ and $v_j \in C_j\}$
    $G \Leftarrow (V', E', W')$
until $Q_{new} = Q_{old}$

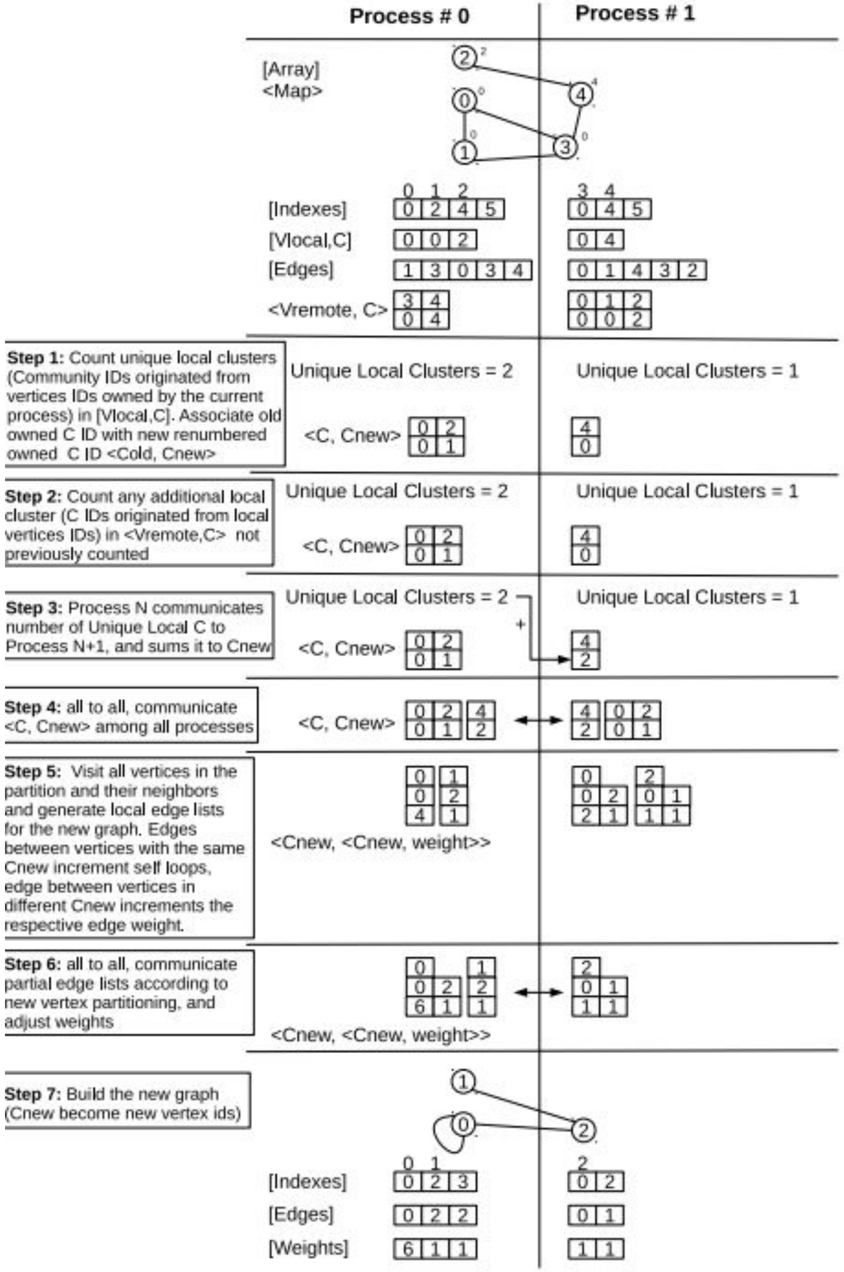
Sequential Pseudocode

"$O(N \text{ LOG} N)$"

Could be $O(N^2)$

Enhanced Kernel

- Every process gets its own set of vertices
- Consists of 2 Parts
  - Iterations of Louvain Locally
    - Maximizes Modularity in local communities
    - Aggregates to find a global moduality
    - Uses Ghost Vertices for interprocess communication
  - Building a new Graph
    - Communicate with other processes to construct new communities



Building the graph

Enhanced Psudocode

Algorithm 2: Parallel Louvain Algorithm (at rank i).
Input: Local portion $G_i = (V_i, E_i)$ of the graph $G = (V, E)$
Input: Threshold, τ (default: 10^{-6})

$C_{curr} \leftarrow \{\{u\} | \forall u \in V\}$
$\{currMod, prevMod\} \leftarrow 0$
while true **do**
  $currMod \leftarrow LouvainIteration(G_i, C_{curr})$
  if $currMod - prevMod \leq \tau$ **then**
    break and output the final set of communities
  $BuildNextPhaseGraph(G_i, C_{curr})$
  $prevMod \leftarrow currMod$

Algorithm 3: Algorithm for the Louvain iterations of a phase at rank i.
Output: Modularity at the end of the phase.

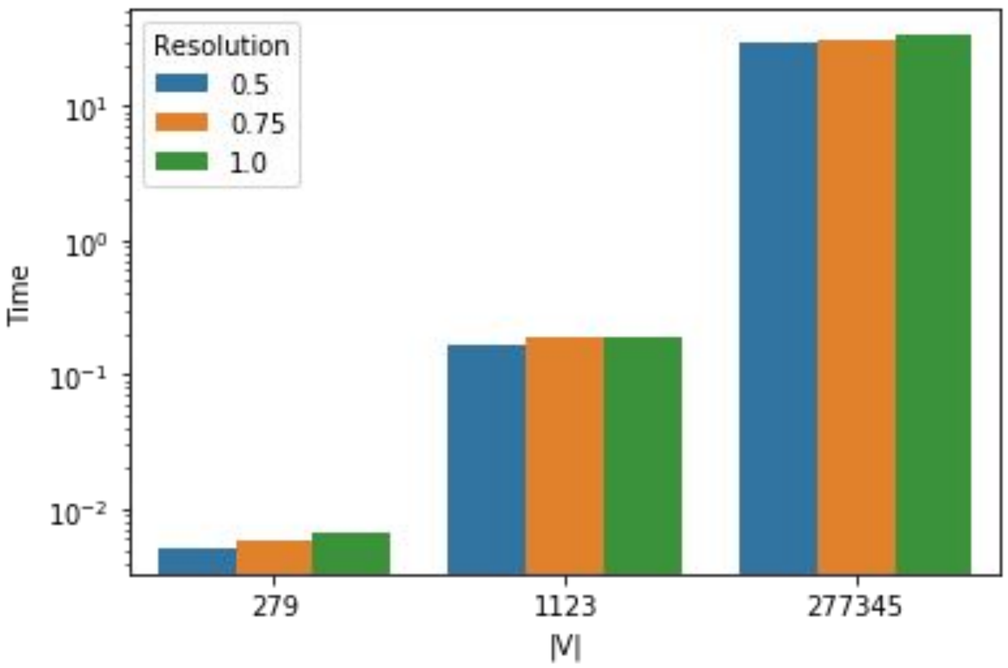
function LOUVAINITERATION(G_i, C_{curr})
$V_g \leftarrow ExchangeGhostVertices(G_i)$
while true **do**
  `send` latest information on those local vertices that are stored as ghost vertices on remote processes
  `receive` latest information on all ghost vertices
  for $v \in V_i$ **do**
    Compute ΔQ that can be achieved by moving v to each of its neighboring communities
    Determine target community for v based on the migration that maximizes ΔQ
    Update community information for both the source and destination communities of v
  `send` updated information on ghost communities to owner processes
  $C_{info} \leftarrow$ `receive` and update information on local communities
  $currMod_i \leftarrow$ Compute modularity based on G_i and C_{info}
  $currMod \leftarrow$ `all-reduce`: $\sum_{\forall i} currMod_i$
  if $currMod - prevMod \leq \tau$ **then**
    break
  $prevMod \leftarrow currMod$
return $prevMod$

Enhanced Implementation (Grappolo)

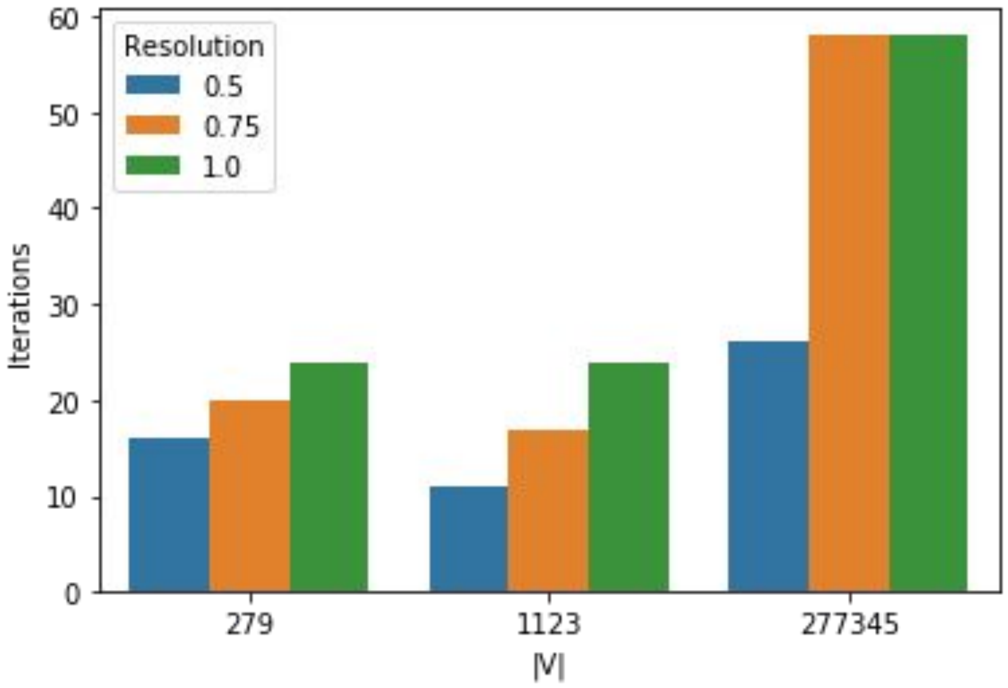
- Public Library in C++
- ~500 Lines
- Uses MPI for interprocess Communication
- CSR to store Graphs
- Looked at time difference in “Resolution” Value

Results

Wall Clock Time

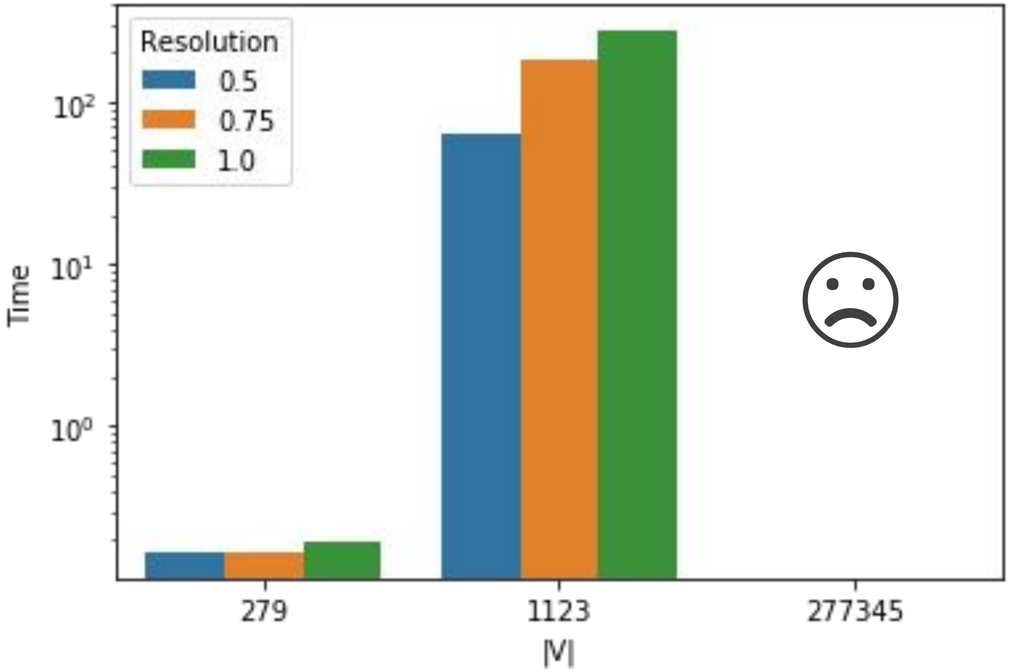


Number of Iterations

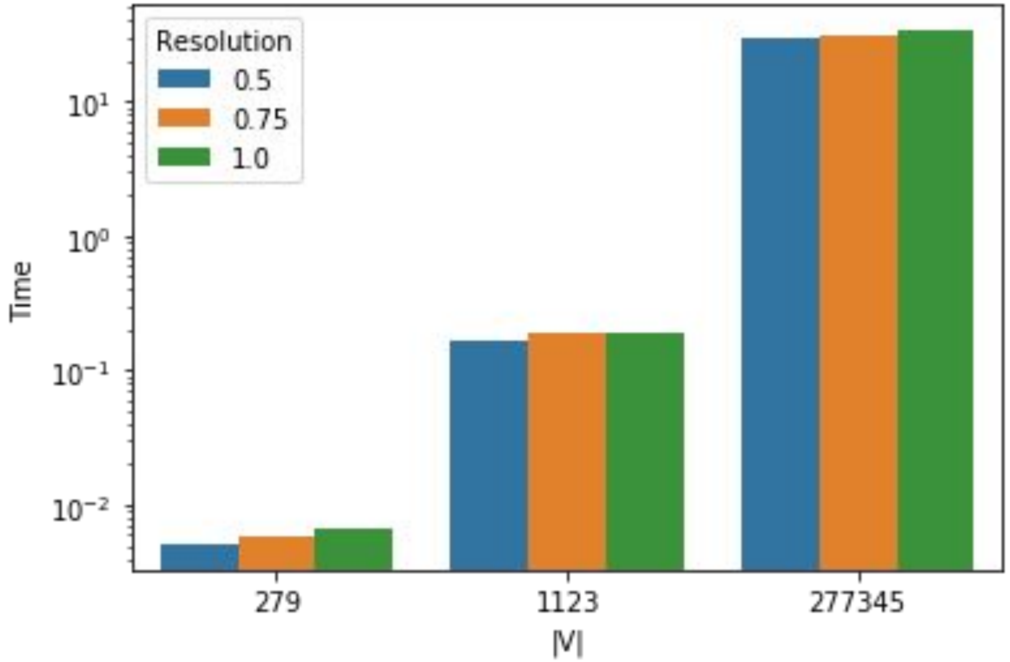


Compared to Sequential

Sequential



Parallel



What I’ve Learned

- Louvain is really fast, especially when parallelized
- Python not as fast
- Modularity may not mean anything in real life
- Will use it as a baseline since it’s so fast

Questions?