# GROUP 14

Abdullah Enes Akdoğan

Alihan Okka

Mehmet Çağrı Kaymak

Mesut Gürlek

# INTRODUCTION

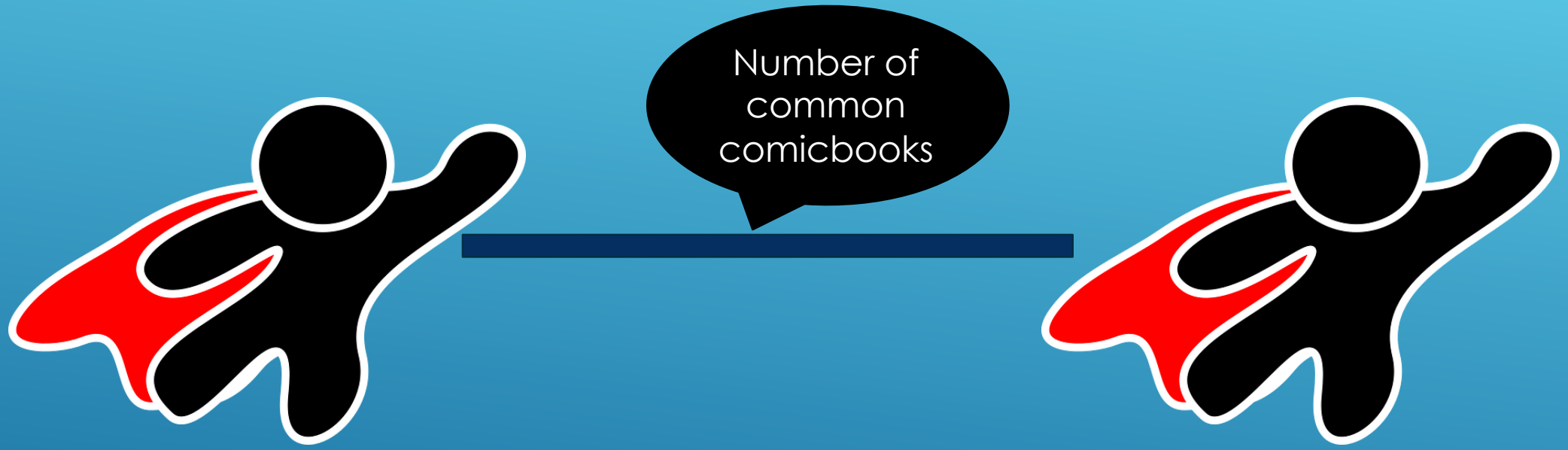
# DATASET

→6486 Superheroes

→12942 Comicbooks

# INITIAL FORMAT OF THE DATA:

# PREPROCESS ON THE DATA:



# PAGERANK ON MARVEL

# WHICH LIBRARIES ARE USED FOR THIS PART?

- NetWorkX
- PageRank on undirected weighted graph
- Teleport Set (Personalized PageRank)

# WHY WE USED PAGERANK?

- To get popular ones among 6486 heroes to validate our algorithms
- To understand our data better for future work
- Also, to interpret Marvel universe

0.0109

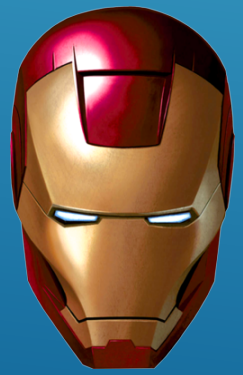
0.007

0.0067

0.0108

0.007

0.0064



0.008

0.007

0.0062

# SIMILARITY RANK WITH SPIDER-MAN

0.0168

0.0109

0.0107

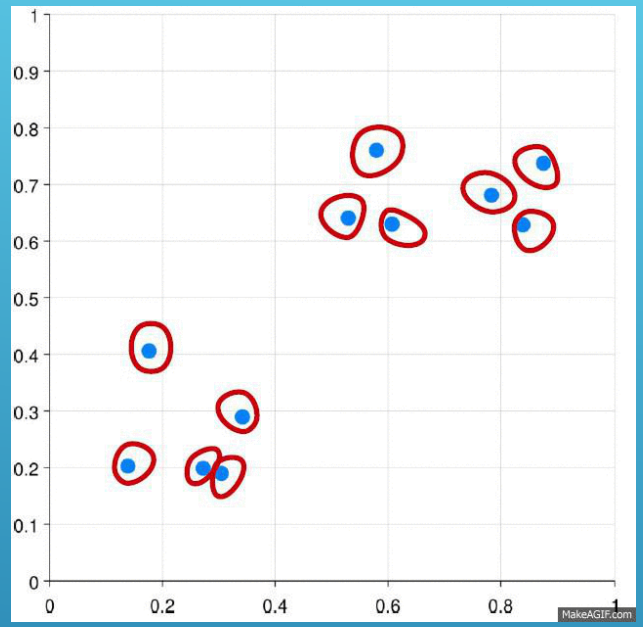
0.0156

0.0101

# HIERARCHICAL CLUSTERING

# AGGLOMERATIVE CLUSTERING

- Bottom-up approach
- No overlapping communities
- Finds local communities bettter



Given:

A set of X objects {x1, x2, ....., xn}

A distance dictionary *dist*[(xi, xj)] = jaccard(xi, xj)

Sort *dist* dictionary by distance values

for i = 1 to n

    ci  = Community(xi)

end for

C = {c1,....., cn}

min_dist, centroid1, centroid2 = find_min_dist(*dist*)

while(min_dist < threshold_distance):

    comm = combine_communities(centroid1, centroid2)

    find_centroid(comm)

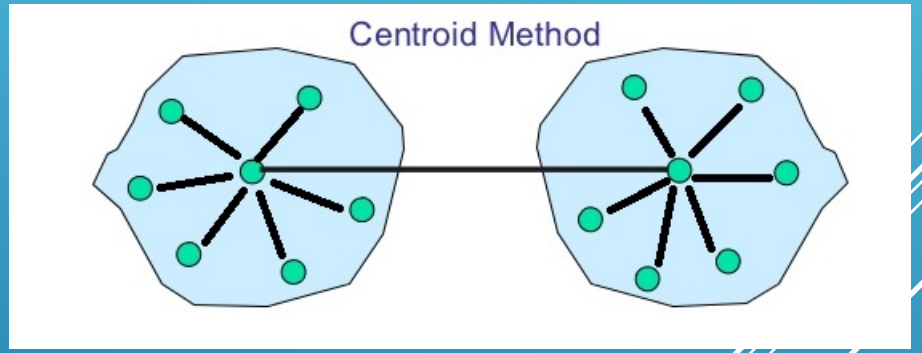
    min_dist, centroid1, centroid2 = find_min_dist(*dist*)

Community:

self.id

self.community_id

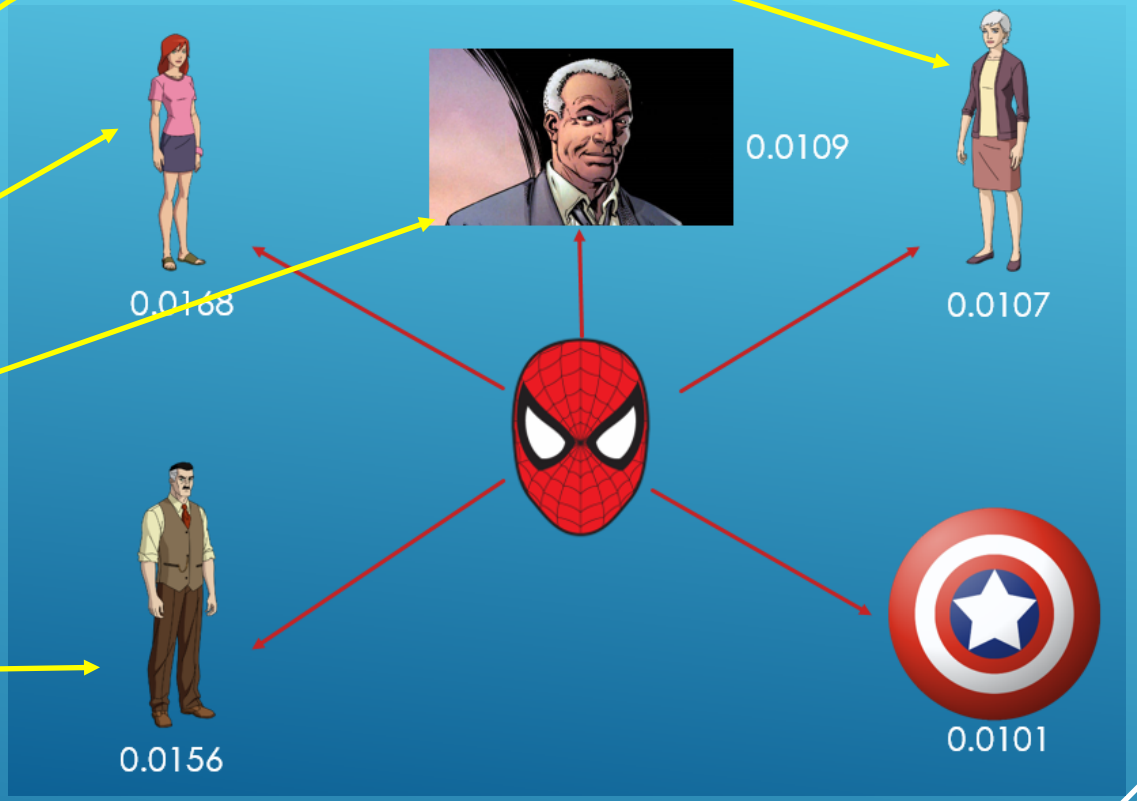
self.isCentroid



## Results of SimRank of Spiderman

**Community id: 2959**

Node 1: WATSON-PARKER, MARY
Node 2: LEEDS, BETTY BRANT
Node 3: PARKER, MAY
Node 4: ROBERTSON, JOE
Node 5: SPIDERMAN
Node 6: THOMPSON, EUGENE FLA
Node 7: JAMESON, J. JONAH



**Community id: 2959**

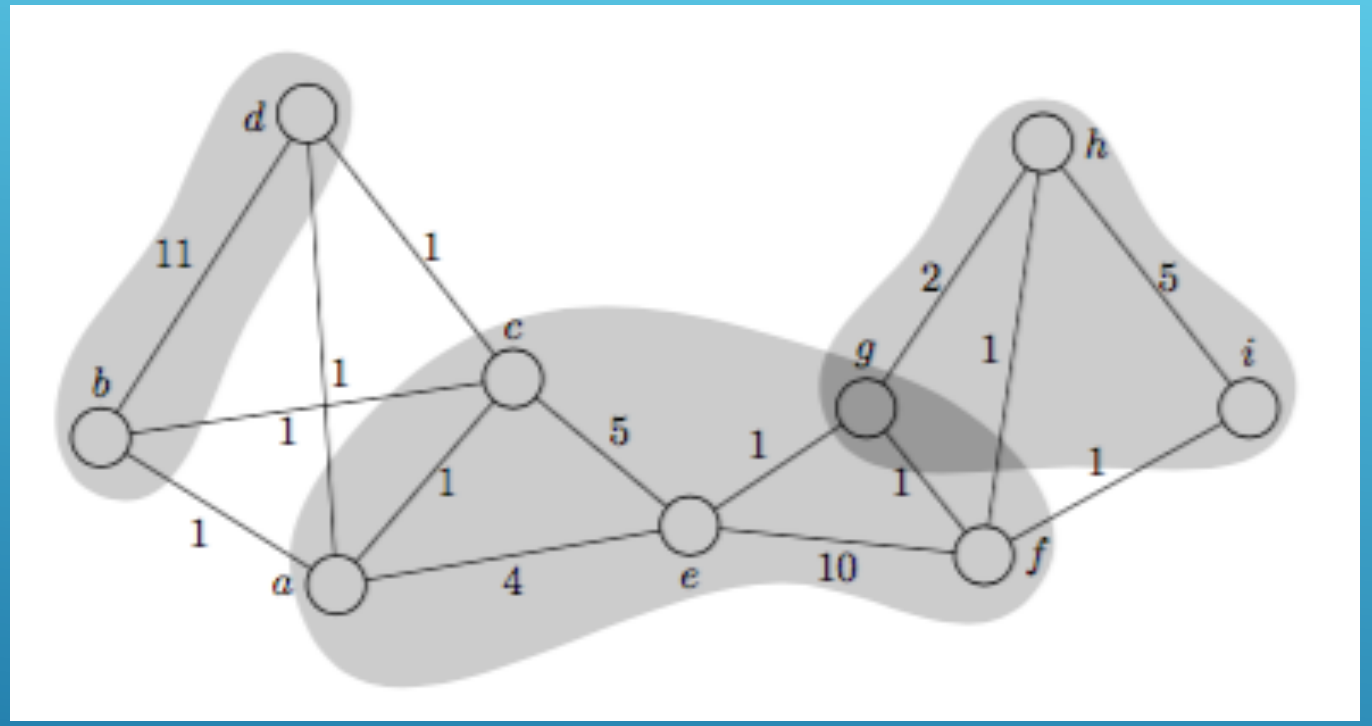
Node 1: IRON MAN/TONY STARK
Node 2: ANT-MAN/DR. HENRY J.
Node 3: CAPTAIN AMERICA
Node 4: QUICKSILVER/PIETRO M
Node 5: SCARLET WITCH/WANDA
Node 6: VISION
Node 7: WASP/JANET VAN DYNE
Node 8: WONDER MAN/SIMON WIL
Node 9: HAWK






# COMMUNITY DETECTION ALGORITHM

- Communities:
- {b, d}
- {e, f, c, a, g}
- {g, h, i}



# PSEUDOCODE FOR ALGORITHM

**Algorithm 1:** Detection Algorithm

**Input** : $G = (V, E)$
**Output**: $\mathcal{C}$
**Initialize**: $\mathcal{C} = \emptyset$;
**while** $E \neq \emptyset$ **do**
  $C = \{u, v\}$, where $(u, v) = \arg\max_{(u,v) \in E} w_{uv}$;
  **while** $N_C \neq \emptyset$ **do**
    $C' = C \cup \arg\max_{w \in N_C} B(w, C)$;
    **if** $\Phi(C') < \Phi(C)$ **then**
      $C = C'$;
    **else**
      break;
    **end**
  **end**
  $E = E \backslash E_C$;
  $\mathcal{C} = \mathcal{C} \cup C$;
**end**

**Algorithm 1:** Detection Algorithm

**Input** : $G = (V, E)$
**Output**: $\mathcal{C}$
**Initialize**: $\mathcal{C} = \emptyset$;
**while** $E \neq \emptyset$ **do**
  $C = \{u, v\}$, where $(u, v) = \arg\max_{(u,v)\in E} w_{uv}$;
  **while** $N_C \neq \emptyset$ **do**
    $C' = C \cup \arg\max_{w\in N_C} B(w, C)$;
    **if** $\Phi(C') < \Phi(C)$ **then**
      $C = C'$;
    **else**
      break;
    **end**
  **end**
  $E = E \backslash E_C$;
  $\mathcal{C} = \mathcal{C} \cup C$;
**end**

Find edge with maximum weight

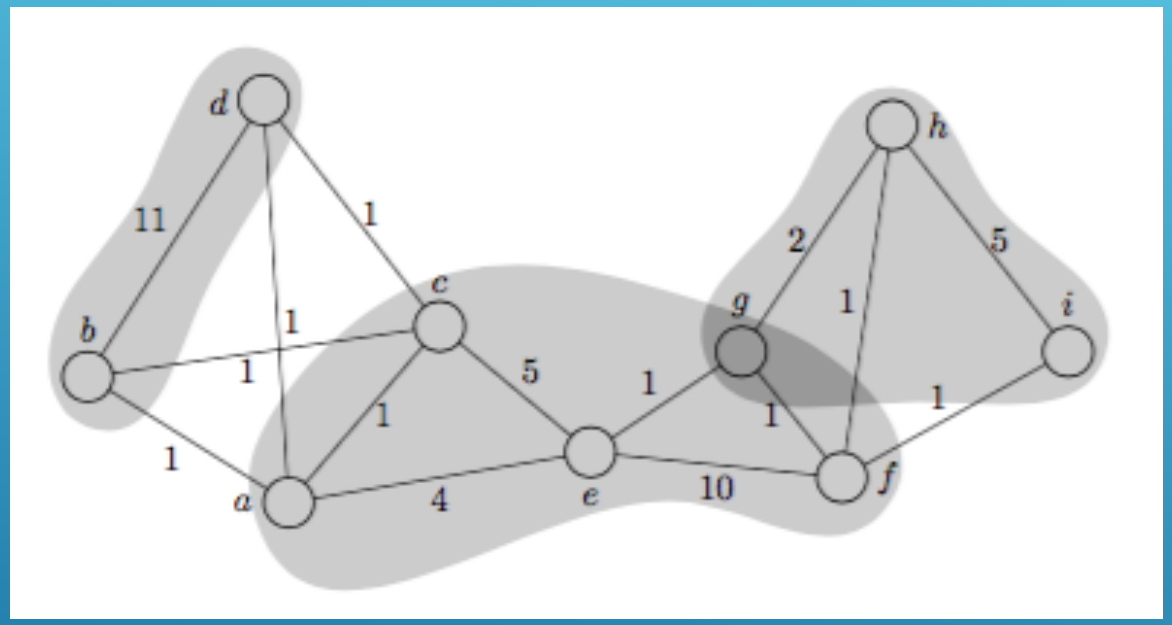
**Algorithm 1:** Detection Algorithm

**Input** : $G = (V, E)$
**Output**: $\mathcal{C}$
**Initialize**: $\mathcal{C} = \emptyset$;
**while** $E \neq \emptyset$ **do**
  $C = \{u, v\}$, where $(u, v) = \arg\max_{(u,v)\in E} w_{uv}$;
  **while** $N_C \neq \emptyset$ **do**
    $C' = C \cup \arg\max_{w\in N_C} B(w, C)$;
    **if** $\Phi(C') < \Phi(C)$ **then**
      $C = C'$;
    **else**
      break;
    **end**
  **end**
  $E = E \backslash E_C$;
  $\mathcal{C} = \mathcal{C} \cup C$;
**end**

Find node with maximum belonging degree to current community

$$B(u,C) = \frac{\sum_{v \in C} w_{uv}}{k_u}.$$

$$k_u = \sum_{v \in N_u} w_{uv}.$$




- B(a, {b, d}) = 2/ 7

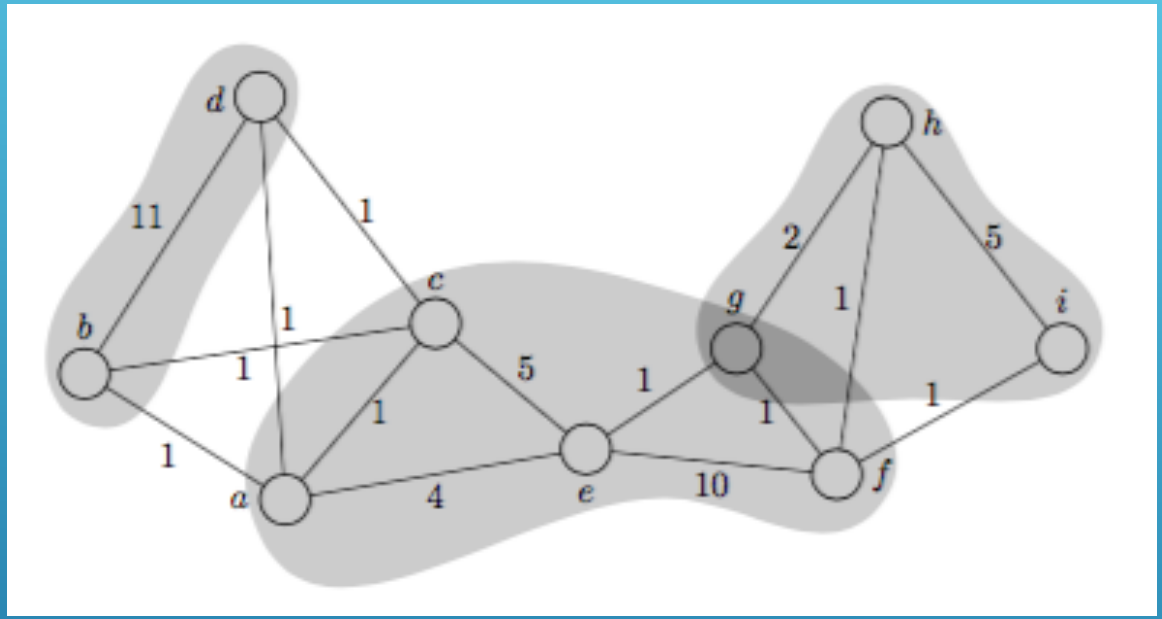
**Algorithm 1:** Detection Algorithm

**Input** : $G = (V, E)$
**Output**: $\mathcal{C}$
**Initialize**: $\mathcal{C} = \emptyset$;
**while** $E \neq \emptyset$ **do**
  $C = \{u, v\}$, where $(u, v) = \arg\max_{(u,v)\in E} w_{uv}$;
  **while** $N_C \neq \emptyset$ **do**
    $C' = C \cup \arg\max_{w\in N_C} B(w, C)$;
    **if** $\Phi(C') < \Phi(C)$ **then**
      $C = C'$;
    **else**
      break;
    **end**
  **end**
  $E = E \backslash E_C$;
  $\mathcal{C} = \mathcal{C} \cup C$;
**end**



Compare current conductance and new conductance

$$\Phi(C) = \frac{cut(C, G \backslash C)}{w_C},$$




- Cut({b,d}, {a, c, e, f, g, h, i}) = 4
- Conductance of C = 11/ 15

**Algorithm 1:** Detection Algorithm

**Input** : $G = (V, E)$
**Output**: $\mathcal{C}$
**Initialize**: $\mathcal{C} = \emptyset$;
**while** $E \neq \emptyset$ **do**
  $C = \{u, v\}$, where $(u, v) = \arg\max_{(u,v)\in E} w_{uv}$;
  **while** $N_C \neq \emptyset$ **do**
    $C' = C \cup \arg\max_{w\in N_C} B(w, C)$;
    **if** $\Phi(C') < \Phi(C)$ **then**
      $C = C'$;
    **else**
      break;
    **end**
  **end**
  $E = E \backslash E_C$;
  $\mathcal{C} = \mathcal{C} \cup C$;
**end**

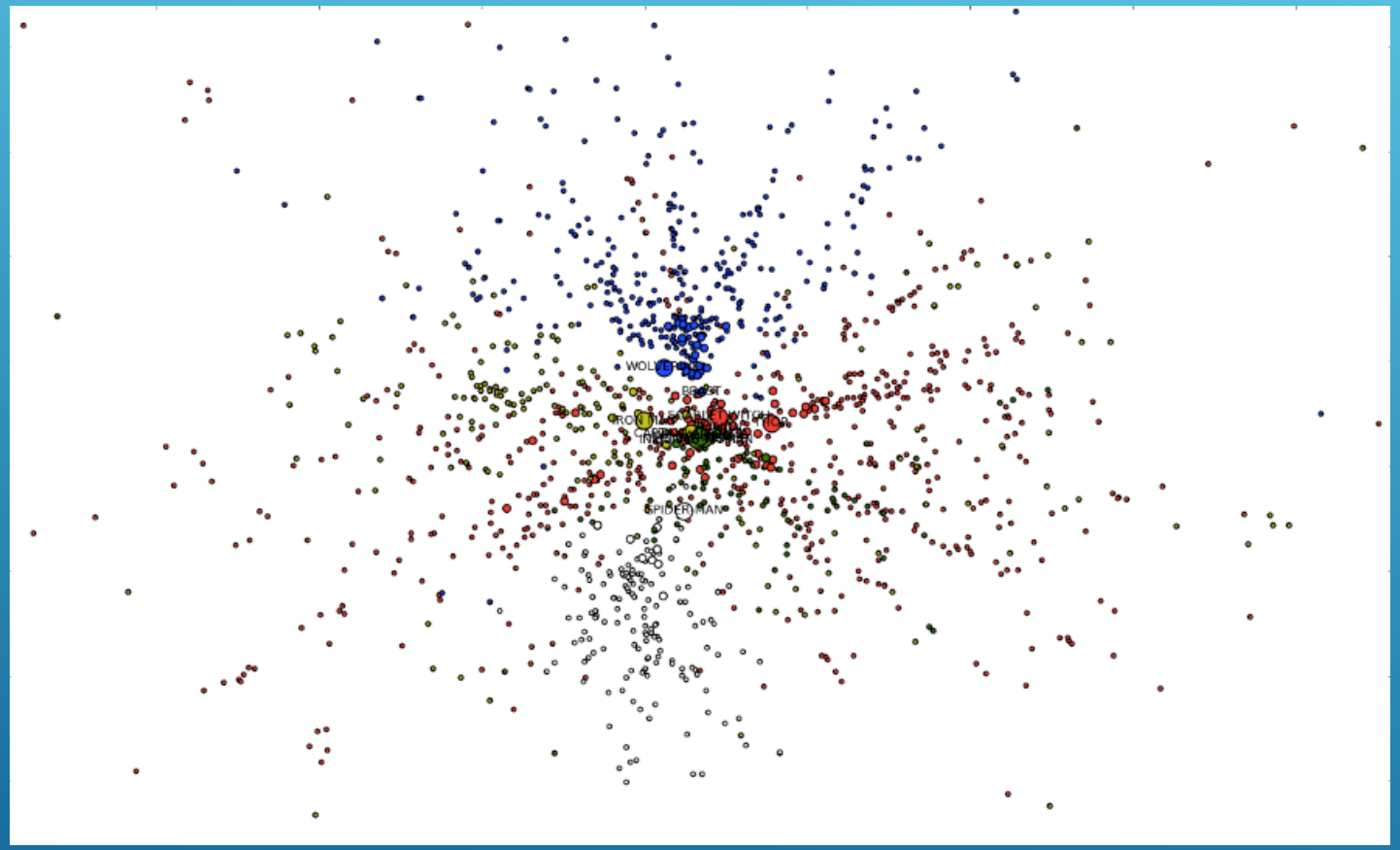
If new conductance is smaller then update community

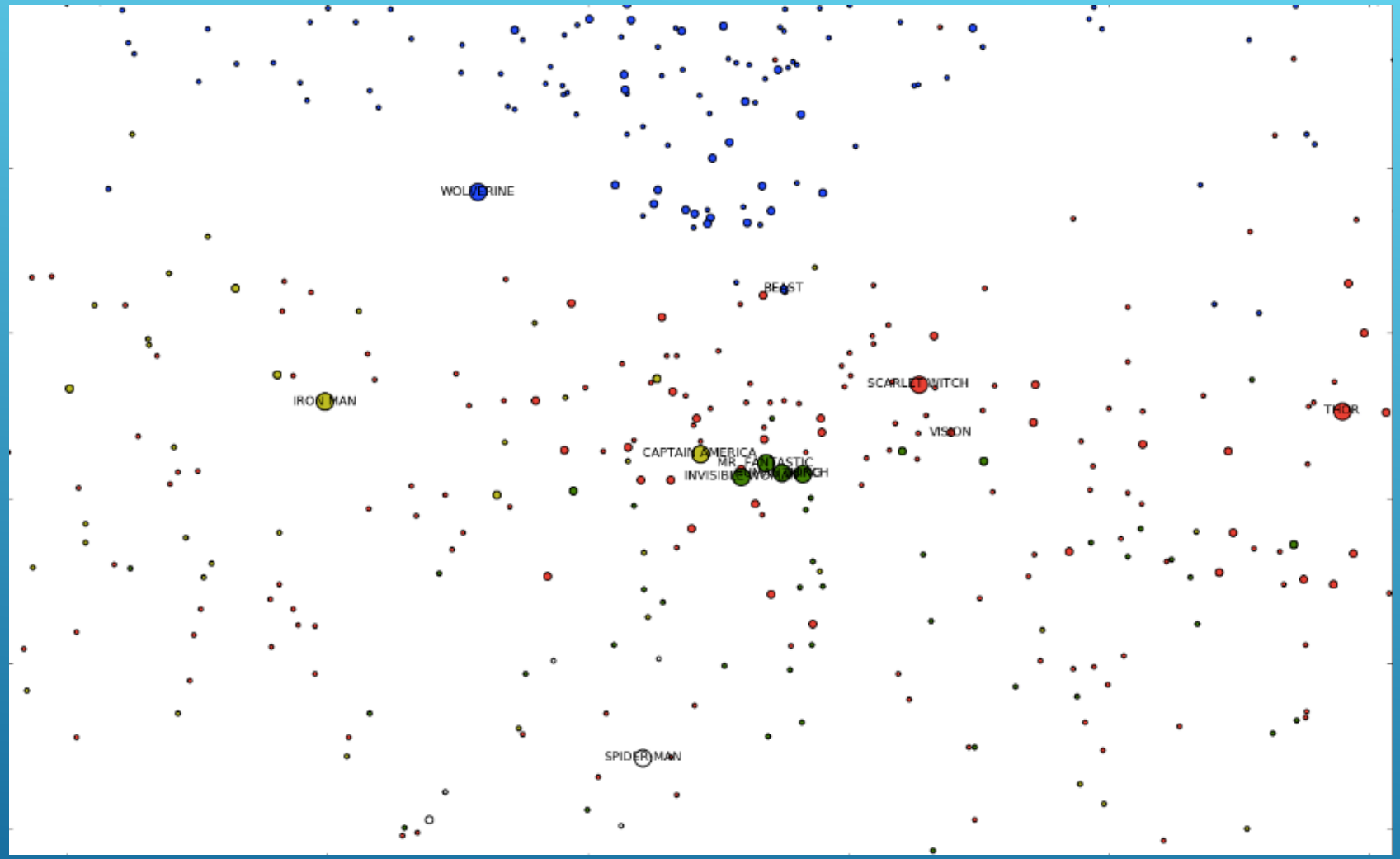
**Algorithm 1:** Detection Algorithm

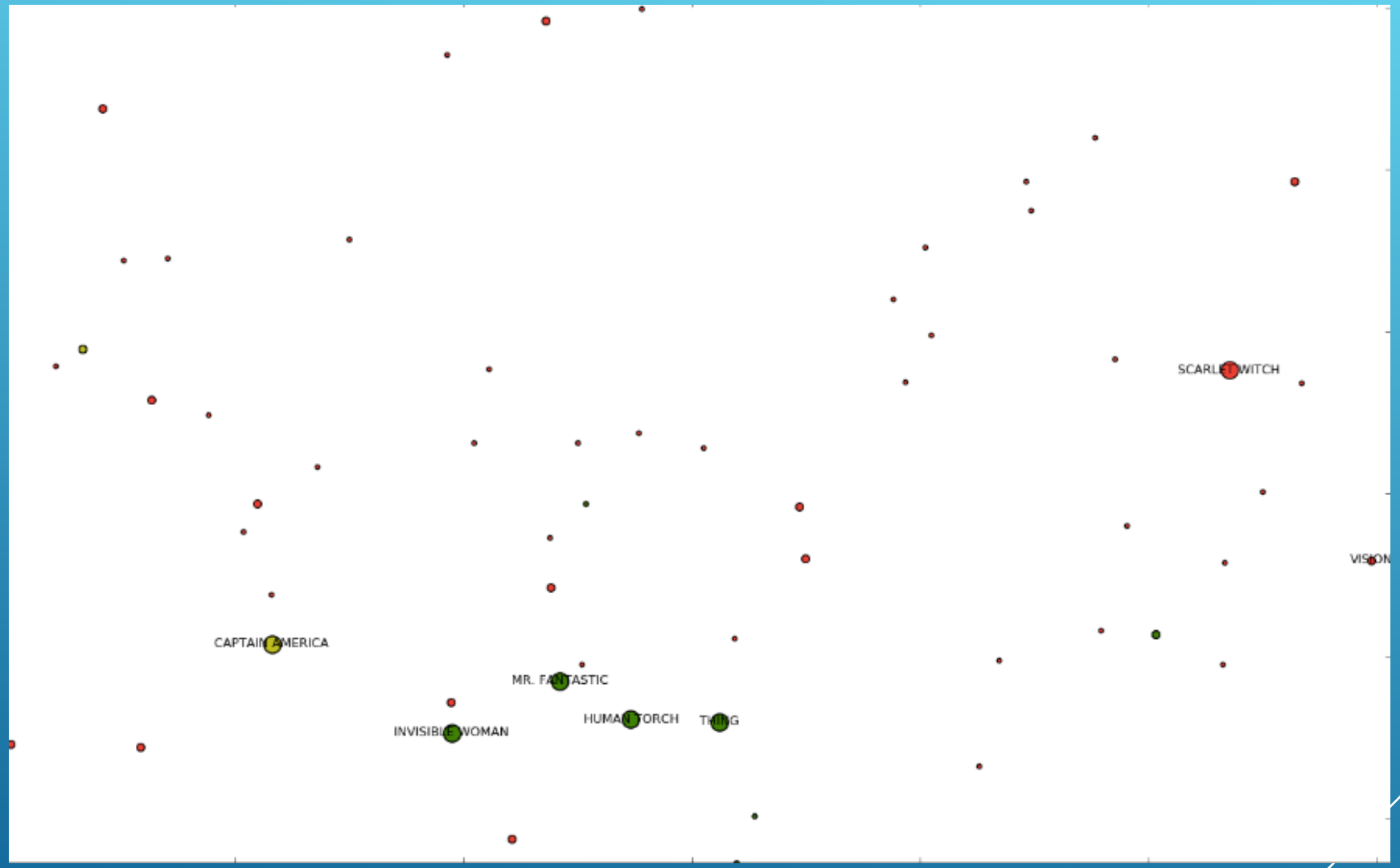
**Input** : $G = (V, E)$
**Output**: $\mathcal{C}$
**Initialize**: $\mathcal{C} = \emptyset$;
**while** $E \neq \emptyset$ **do**
  $C = \{u, v\}$, where $(u, v) = \arg\max_{(u,v)\in E} w_{uv}$;
  **while** $N_C \neq \emptyset$ **do**
    $C' = C \cup \arg\max_{w\in N_C} B(w, C)$;
    **if** $\Phi(C') < \Phi(C)$ **then**
      $C = C'$;
    **else**
      break;
    **end**
  **end**
  $E = E \backslash E_C$;
  $\mathcal{C} = \mathcal{C} \cup C$;
**end**

Else finish extending community

# RESULTS ON MARVEL CHARACTERS




WOLVERINE
BEAST
IRON MAN
SCARLET WITCH
THOR
VISION
CAPTAIN AMERICA
MR. FANTASTIC
SPIDER-MAN


SCARLET WITCH
CAPTAIN AMERICA
MR. FANTASTIC
HUMAN TORCH
THING
INVISIBLE WOMAN


CLAP
CLAP
CLAP
THAT. WAS.
AWESOME.