**인문학술정보 지식그래프 구현**

1. Network format으로 변환

JsonToNetwork.ipynb(py) 파일 아래 항목에 대한 기능 구현.

* 1. 활용 Python 패키지

JSON 파일 로드(json), 네트워크 분석(NetworkX, python-louvain), 시각화(matplotlib, seaborn)

* 1. JSON 파일 로드  
     JSON 파일을 Python의 딕셔너리(dict) 형태로 가져온다.

with open('./Data\_20211122.json', 'r', encoding='utf-8') as file:

    data = json.load(file)

딕셔너리의 key 값이 Node로 시작하면 Node 관련 정보이며, Link로 시작하면 Edge 관련 정보.

* 1. NetworkX로 네트워크 객체 만들기
     1. 네트워크 객체에 Node 정보 투입

Node의 고유 id와 label 등의 속성값(attribute)을 ‘add\_nodes\_from’ 함수를 활용해 투입.

for node\_key in Nodes:

    print(node\_key)

    G.add\_nodes\_from([(N['id'],{k:v for k,v in N.items() if k!='id'}) for N in Nodes[node\_key]])

* + 1. 네트워크 객체에 Edge 정보 투입

Edge의 고유 id와 label 등의 속성값(attribute)을 ‘add\_edges\_from’함수를 활용해 투입

for edge\_key in Edges:

    print(edge\_key)

    G.add\_edges\_from([(E['sourceid'],E['targetid'],{k:v for k,v in E.items() if 'id' not in k}) for E in Edges[edge\_key]])

* 1. Community Detection (커뮤니티 탐지)

Node와 Edge로 구성된 네트워크에서 특정 커뮤니티(그룹)이 있는지 파악한다. Louvain community detection 알고리즘[[1]](#footnote-1)으로 Node의 커뮤니티 아이디를 부여한다. 이때 best\_partition 함수를 활용한다.

partition = community.best\_partition(G)

커뮤니티 탐지 결과를 NetworkX의 kamada\_kawai\_layout[[2]](#footnote-2) 함수를 활용해 시각화한다.

pos = nx.drawing.layout.kamada\_kawai\_layout(G)

# color the nodes according to their partition

cmap = cm.get\_cmap('viridis', max(partition.values()) + 1)

nx.draw\_networkx\_nodes(G, pos, partition.keys(), node\_size=40,

                       cmap=cmap, node\_color=list(partition.values()))

nx.draw\_networkx\_edges(G, pos, alpha=0.5)

plt.show()

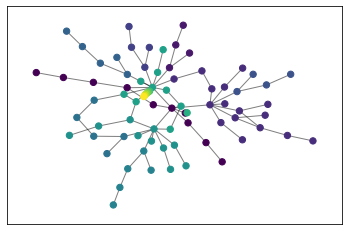


그림 1. 커뮤니티 시각화 결과

* 1. JSON 파일로 네트워크 객체 저장

네트워크 객체를 SigmaJS에서 읽을 수 있도록 JSON 형태로 다시 저장. NetworkX의 json\_graph.node\_link\_data[[3]](#footnote-3) 함수를 활용해 JSON 형태로 변환 가능.

G\_Json = nx.json\_graph.node\_link\_data(G)

이때 위에서 커뮤니티 탐지 결과 등의 추가 속성값(x,y 좌표 등)을 JSON 객체에 추가.

for node in G\_Json['nodes']:

    node['key'] = node.pop('id')

    node['label'] = node.pop('name')

    node['attributes'] = {}

    node['attributes']['group'] = partition[node['key']]

    node['attributes']['x'] = int(kamada\_kawai\_pos[node['key']][0] \* 256)

    node['attributes']['y'] = int(kamada\_kawai\_pos[node['key']][1] \* 256)

    node['attributes']['color'] = "rgba("+ ",".join([str(int(i \* 255)) for i in pal[partition[node['key']]]])+ ")"

1. SigmaJS를 활용한 데이터 시각화

1. <https://en.wikipedia.org/wiki/Louvain_method> [↑](#footnote-ref-1)
2. <https://networkx.org/documentation/stable/reference/generated/networkx.drawing.layout.kamada_kawai_layout.html> [↑](#footnote-ref-2)
3. <https://networkx.org/documentation/stable/reference/readwrite/generated/networkx.readwrite.json_graph.node_link_data.html> [↑](#footnote-ref-3)