<pre class="code-snippet\_\_js" data-lang="ruby"># -\*- coding: utf-8 -\*-

# <span class="code-snippet\_\_doctag">@Author

: 武辛

#<span class="code-snippet\_\_doctag">@Email

:geo\_data\_analysis@163.com

# <span class="code-snippet\_\_doctag">@Note

: 如有疑问，可加微信"wxid-3ccc"

# <span class="code-snippet\_\_doctag">@All

Rights Reserved!

import sys, copy, time

def

FN

()

:

print(load the network..."

)

network=loadNetwork("network/test.txt"

)

max\_Q = float("-inf"

) partition = None

start\_time = time.time()

while

len(network.Communities)

:

print("left %d communities need to merge, waiting..."

% len(network.Communities))

det\_Q = float("-inf"

)

max\_link = None

for

link in

network.Links.values():

community\_i=link[0

]

community\_j=link[1

]

if

community\_i == community\_j:

continue

# 计算两个community的det\_Q

cur\_Q = cal\_det\_Q(network, community\_i, community\_j)

# 找到合并两个community Q值增加最大的进行合并

if

cur\_Q &gt det\_Q:

det\_Q = cur\_Q

max\_link=link

if

max\_link is None:

break

# 合并两个community，将社区j合并到社区i中

community\_i = max\_link[0

]

community\_j=max\_link[1

]

merge(network, community\_i, community\_j)

# 合并社区j后，更新社区i的e\_ii,a\_i,e\_ij信息

update\_community\_info(network, community\_i, community\_j)

# 删除边ij

delnetwork.Links[max\_link]

# 计算合并社区ij后的模块度

cur\_Q = cal\_Q(network)

if

cur\_Q &gt max\_Q:

max\_Q = cur\_Q

partition = copy.deepcopy(list(network.Communities.values()))

t2 = time.time()

print("Find%dcommunitiesafter%.3fseconds,themaximalQis%.3f"

%(len(partition),t2-start\_time,max\_Q))

if

\_\_name\_\_

== '\_\_main\_\_'

:

FN()

</pre>