补充材料 (2025 *Acta Phys. Sin.* **74** 166401)

《基于谱图理论的大规模复杂网络重要节点组挖掘算法》的补充材料

邢梓涵[1)] 刘丝语[1)] 刘慧[1)2)3)†] 陈凌霄[1)]

1)(华中科技大学人工智能与自动化学院，武汉 430074)

2)(华中科技大学图像信息处理与智能控制教育部重点实验室，武汉 430074)

3)(华中科技大学类脑智能系统湖北省重点实验室，武汉 430074)

MFG 算法伪代码如下所示

INPUT：网络邻接矩阵 $\boldsymbol{A}$，受控节点个数 k，子集数 m，每子集节点数 p，搜索步数 s.

OUTPUT：目标筛选节点组 Γ

```
global_attributes = ComputeGlobalAttributes(A);
node_features_list = ComputeNodeFeatures(A);
normalized_features_list = NormalizeFeatures(node_features_list, global_attributes);
fusion_metric = ComputeFusionMetric(normalized_features_list);
subsets_list = PartitionNodeSet(A, m);
S = [];
for each subset in subsets_list:
    top_p_nodes = SelectTopNodes(subset, fusion_metric, p);
    S. extend(top_p_nodes)
S = RemoveDuplicatesAndSort (S, fusion_metric);
Γ = set( );
Count=0;
while |Γ| < k:
    best_node = None;
    current_min_eigenvalue = FLOAT_MAX;
    for each node v in S:
        if v in Γ:
            continue;
        Γ_temp = Γ + {v};
        Count=Count+1;
        λ1 =ComputeMinEigenvalue(A, Γ_temp);
        if λ1 < current_min_eigenvalue:
            current_min_eigenvalue = λ1;
            best_node = v;
    Γ. add(best_node);
    second_hop_neighbors = GetNeighbors (A, best_node)
    candidate_nodes = [u for u in second_hop_neighbors if u not in S and u not in Γ]
```

u = SelectNodeWithMinResistanceDistance(candidate_nodes);

λ_u =ComputeMinEigenvalue ($\boldsymbol{A}, \Gamma$-{best_node} + $\{u\}$);

$\lambda_{\text{bestnode}}$ = ComputeMinEigenvalue ($\boldsymbol{A}, \Gamma$);

if $\lambda_u > \lambda_{\text{bestnode}}$:

  Γ. remove(best_node);

  Γ. add(u);

if Count%=0:

  nodes_not_in_Γ = [e for e in S if e not in Γ]

  if nodes_not_in_Γ is empty:

    break;

  else:

    e = SelectRandomNode (nodes_not_in_Γ);

    g = SelectRandomNode (Γ);

    λ_{replace} = ComputeMinEigenvalue ($\boldsymbol{A}, \Gamma - \{g\} + \{e\}$);

    λ_{origin} =ComputeMinEigenvalue ($\boldsymbol{A}, \Gamma$);

    if $\lambda_{\text{replace}} > \lambda_{\text{origin}}$:

      Γ. remove(g);

      Γ. add(e);

return Γ.