

# Density Based Merging Search of Functional Modules in Protein-Protein Interaction (PPI) Networks

\*  
Jing Wang<sup>1</sup>, Jie Tang<sup>1</sup>, and Jian Pei<sup>2</sup>  
<sup>1</sup>Department of Mathematics, Peking University  
Beijing 100871, China  
jwma@math.pku.edu.cn

**Abstract.** In this paper, we propose a density based merging search algorithm for functional modules in PPI networks. The algorithm is based on the density based clustering algorithm DBSCAN. We first define the density based merging search algorithm. Then we prove that the algorithm can find all functional modules in PPI networks. Finally, we apply the algorithm to the yeast PPI network and find many functional modules. The results show that the algorithm is effective and efficient.

**Keywords:** protein-protein interaction, functional modules, density based merging search

## 1 Introduction

Protein-protein interaction (PPI) networks have become a central focus in systems biology. The study of PPI networks can help us understand the complex biological processes and the underlying mechanisms of various diseases. In this paper, we propose a density based merging search algorithm for functional modules in PPI networks. The algorithm is based on the density based clustering algorithm DBSCAN. We first define the density based merging search algorithm. Then we prove that the algorithm can find all functional modules in PPI networks. Finally, we apply the algorithm to the yeast PPI network and find many functional modules. The results show that the algorithm is effective and efficient.

---

\* This work is supported by the National Natural Science Foundation of China (Grant No. 30725021).

Clustering. HCS ( ) GN the Hierarchical  
 the similarity  
 RSNC ( )  
 E ( )

## 2 The DBMS Algorithm

### 2.1 The Characteristics of a Complex

m m m m  
 m m m m m  
 m m m m m  
 m m m m m  
 m m m m m

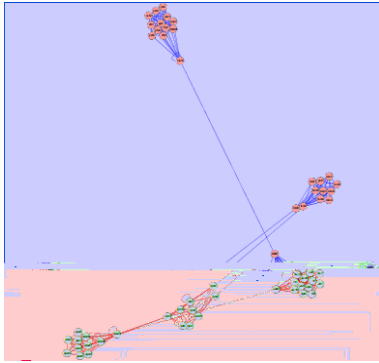


Fig. 1.

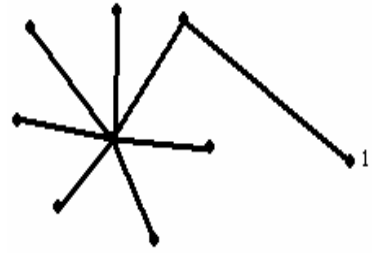


Fig. 2.

$$\rho_p = \left( \sum_{v \in U_p} d_v + d_p \right) d_p \quad (1)$$

$$d_v = \sum_{u \in U_p} d_{uv} \quad (2)$$

$$d_p = \sum_{u \in U_p} d_{pu} \quad (3)$$

$$d_{uv} = \sum_{w \in U_p} d_{uvw} \quad (4)$$

$$d_{pu} = \sum_{w \in U_p} d_{puw} \quad (5)$$

$$d_{uvw} = \sum_{x \in U_p} d_{uvw} \quad (6)$$

$$d_{puw} = \sum_{x \in U_p} d_{puw} \quad (7)$$

## 2.2 The Description of the DBMS Algorithm

The DBMS algorithm is a distributed algorithm for finding the minimum spanning tree of a graph. It is based on the idea of the distributed breadth-first search algorithm. The algorithm starts with each node in the graph having a unique identifier. The nodes then exchange information about their neighbors and their own identifiers. The algorithm then proceeds to build the minimum spanning tree by selecting the edges that connect the nodes with the smallest identifiers.

$$\begin{aligned}
 & \dots, m \dots n \dots S = n \dots d_p = m \dots \\
 & \dots P \in S \dots m \dots \\
 & \dots k \leq n \dots \\
 & k \leq d_p \dots c = k \dots n \dots
 \end{aligned}$$

$$\dots, m \dots P \in S \dots c \geq \frac{b}{n} \dots P \in S$$

$$\frac{k \geq b}{m} \dots \frac{b}{m} \dots n \dots$$

$$\begin{aligned}
 b = m \dots n \dots f \dots d_p \dots n \dots \\
 \dots m \dots d_p \dots m \dots n \dots b = \dots n \dots
 \end{aligned}$$

### 3 Experiment Results

#### 3.1 The PPI Datasets

$\lambda = \frac{1}{\sum_{i=1}^n \frac{1}{d_i}}$

$f = \frac{1}{\sum_{i=1}^n \frac{1}{d_i^2}}$

$d_p = \frac{1}{\sum_{i=1}^n \frac{1}{d_i^p}}$

$E = \frac{1}{\sum_{i=1}^n \frac{1}{d_i^2}}$

#### 3.2 Simulation Results

$\lambda = \frac{1}{\sum_{i=1}^n \frac{1}{d_i}}$

$f = \frac{1}{\sum_{i=1}^n \frac{1}{d_i^2}}$

$d_p = \frac{1}{\sum_{i=1}^n \frac{1}{d_i^p}}$

$E = \frac{1}{\sum_{i=1}^n \frac{1}{d_i^2}}$

#### 3.3 Experimental Results on the Real-World PPI Datasets

$\lambda = \frac{1}{\sum_{i=1}^n \frac{1}{d_i}}$

$f = \frac{1}{\sum_{i=1}^n \frac{1}{d_i^2}}$

$d_p = \frac{1}{\sum_{i=1}^n \frac{1}{d_i^p}}$

$E = \frac{1}{\sum_{i=1}^n \frac{1}{d_i^2}}$

**Table 1.**

m	m	m	E	m	m
m	m	E	m	m	m
m	E	m	m	m	m

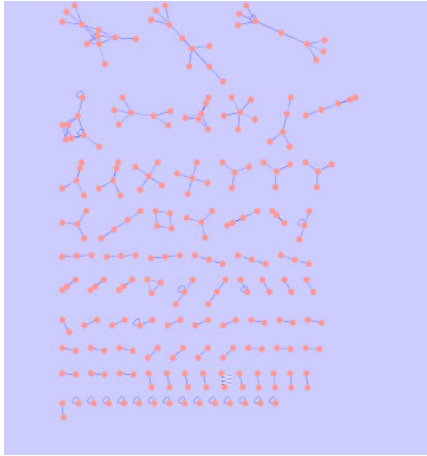


Fig. 3.

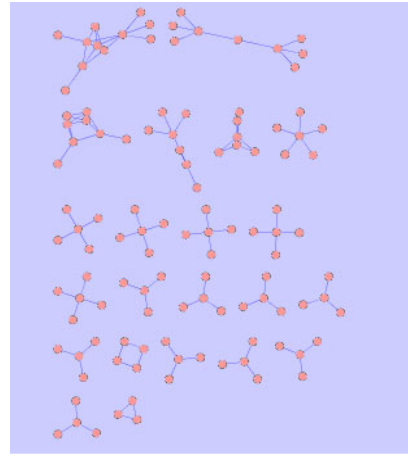


Fig. 4.

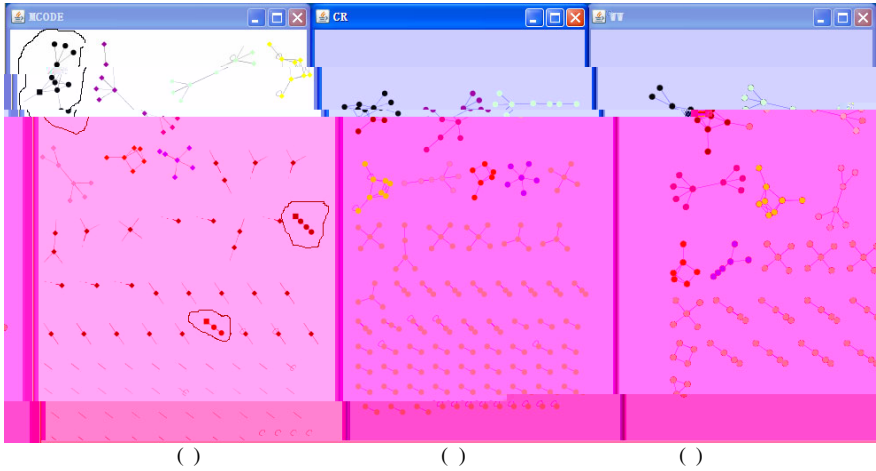


Fig. 5.

