Distributed Computing on Core-Periphery Networks: Axiom-Based Design

Chen Avin, Michael Borokhovich, Zvi Lotker, and David Peleg

Department of Communication Systems Engineering, BGU, Israel
Department of Computer Science, The Weizmann Institute, Israel

ICALP 2014

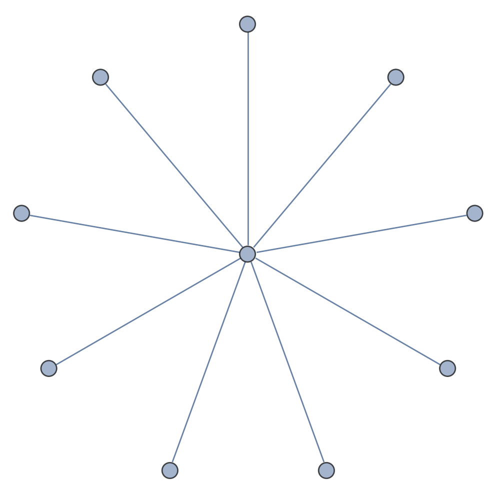
Networks for **Distributed Computing**

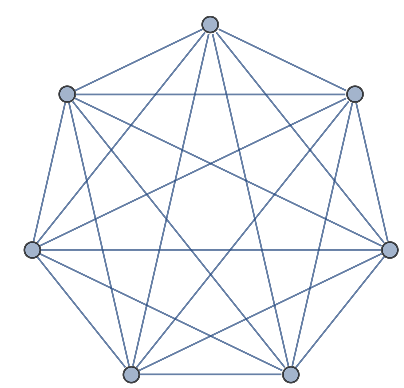
- What do we want?
  - **fast running times**, robust, small diameter, bounded degree?
  - **cost efficient**: communication links, nodes memory

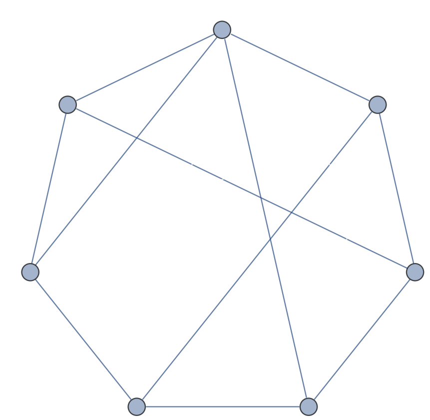
Networks for **Distributed Computing**

- What do we want?
  - **fast running times**, robust, small diameter, bounded degree?
  - **cost efficient**: communication links, nodes memory

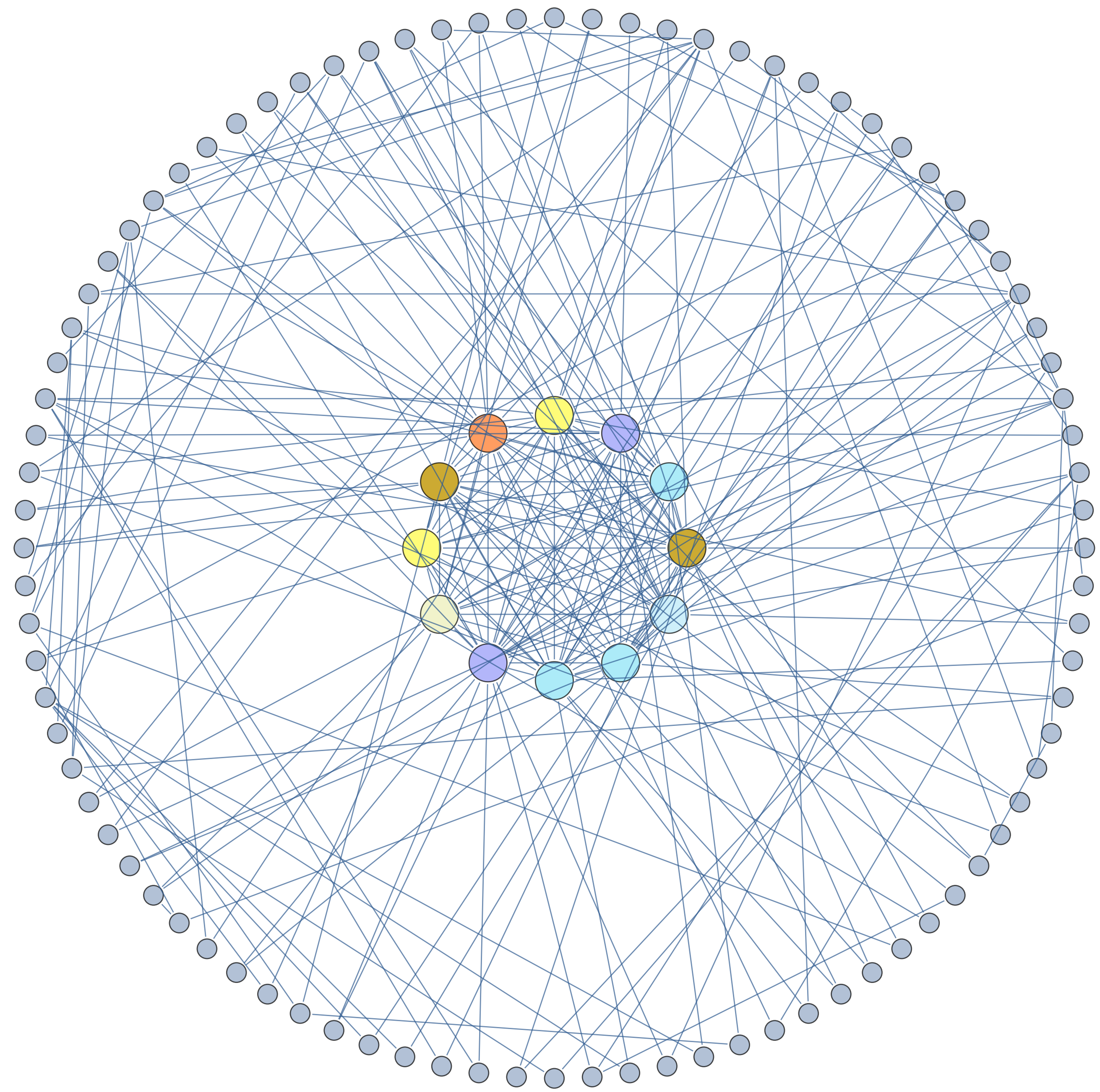
- Classic examples:
  - *Stars*
  - *Cliques*
  - *Bounded degree Expanders*





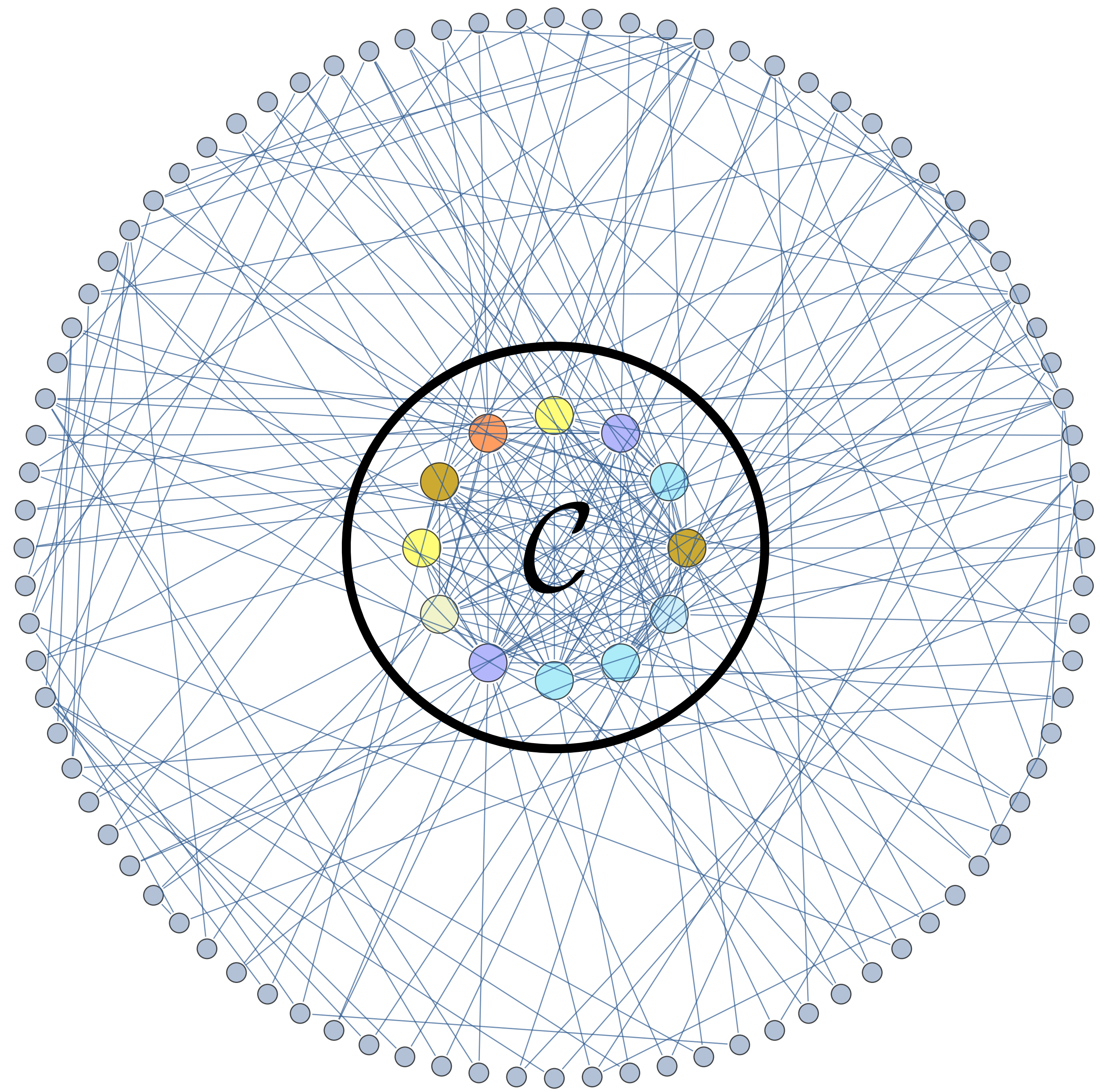


Core-Periphery Structure



Core-Periphery Structure

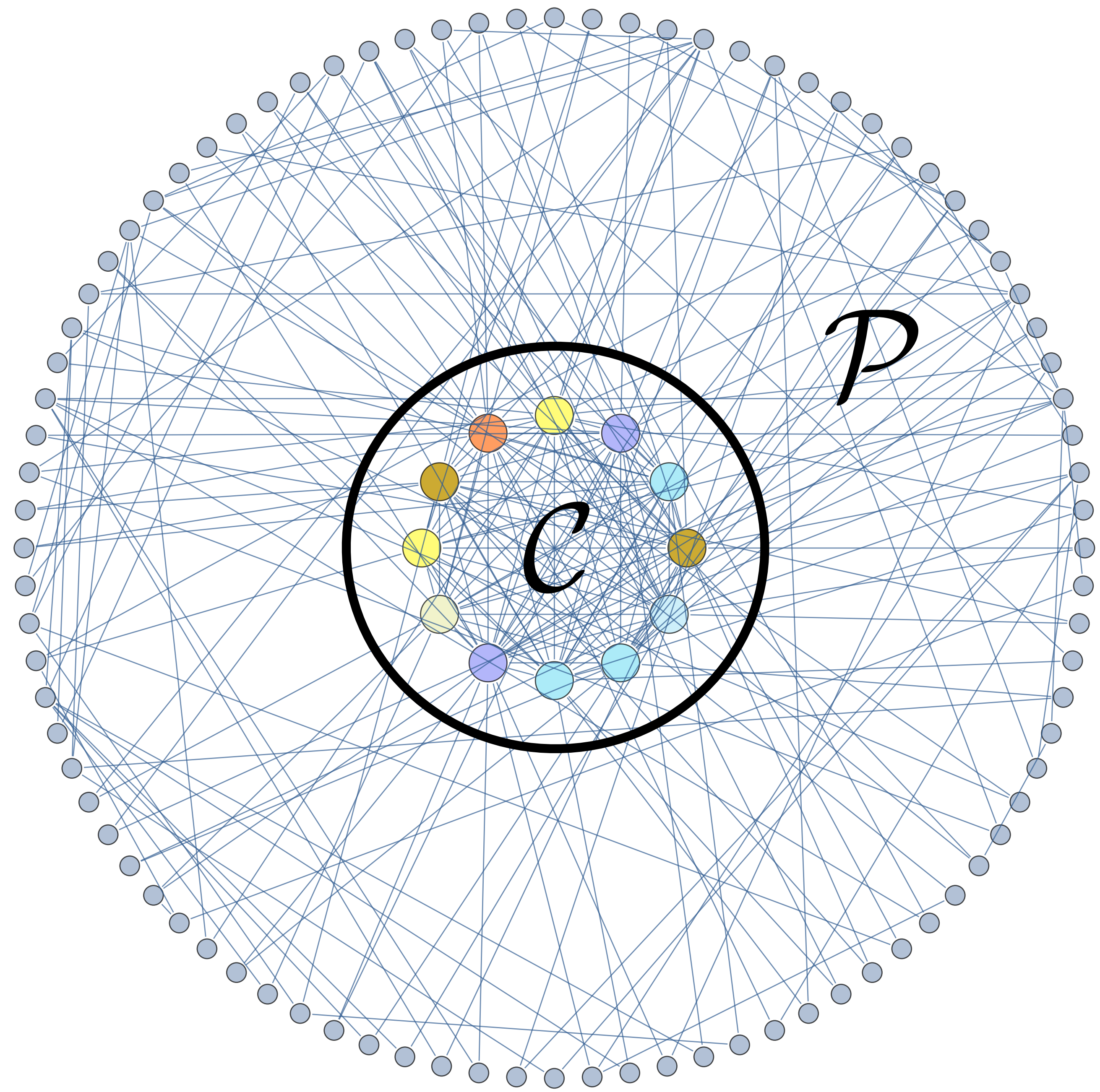
Core:
small, dense



Core-Periphery Structure

Core:
small, dense

Periphery:
large, sparse

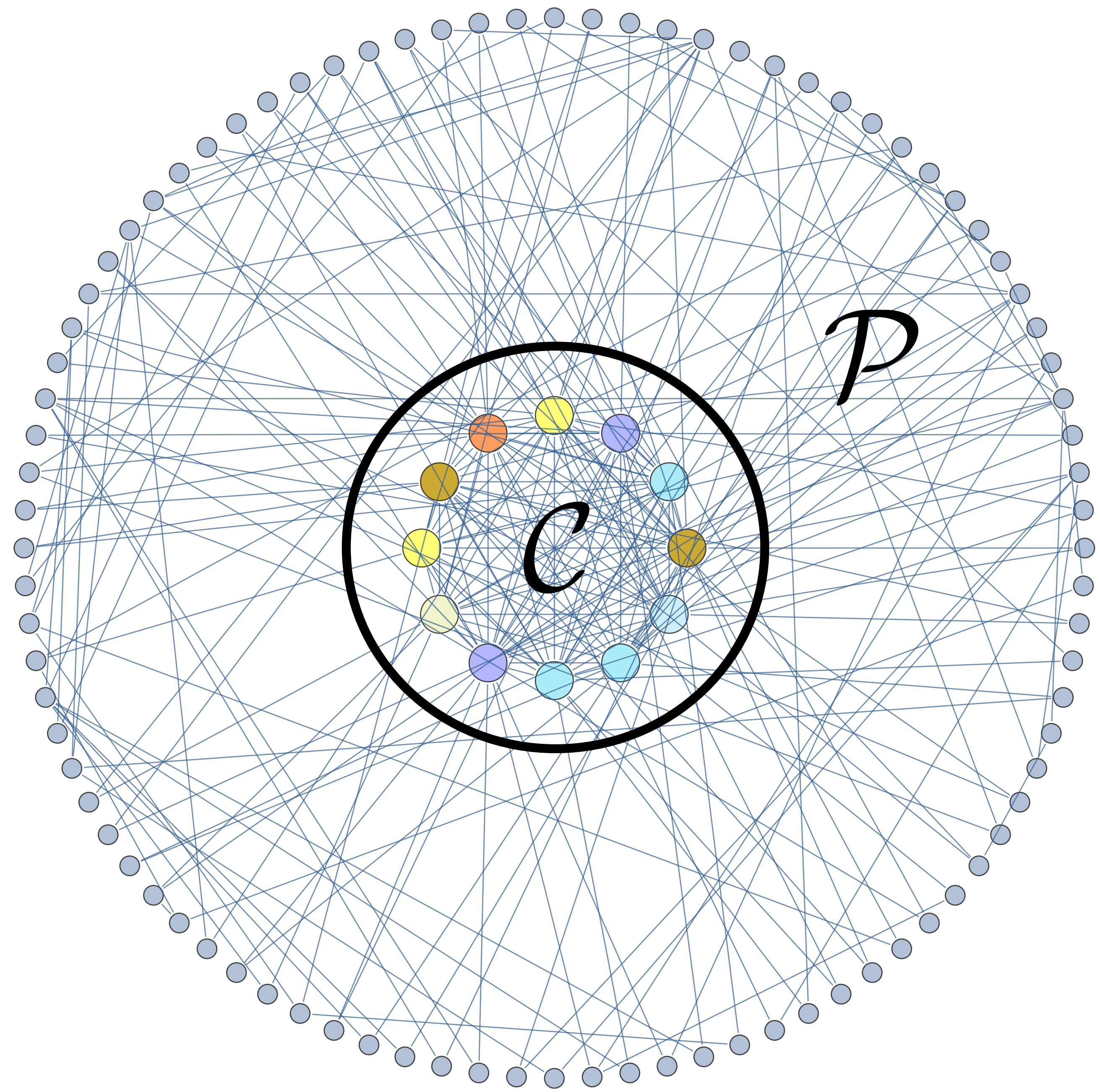


Core-Periphery Structure

Core:
small, dense

Periphery:
large, sparse

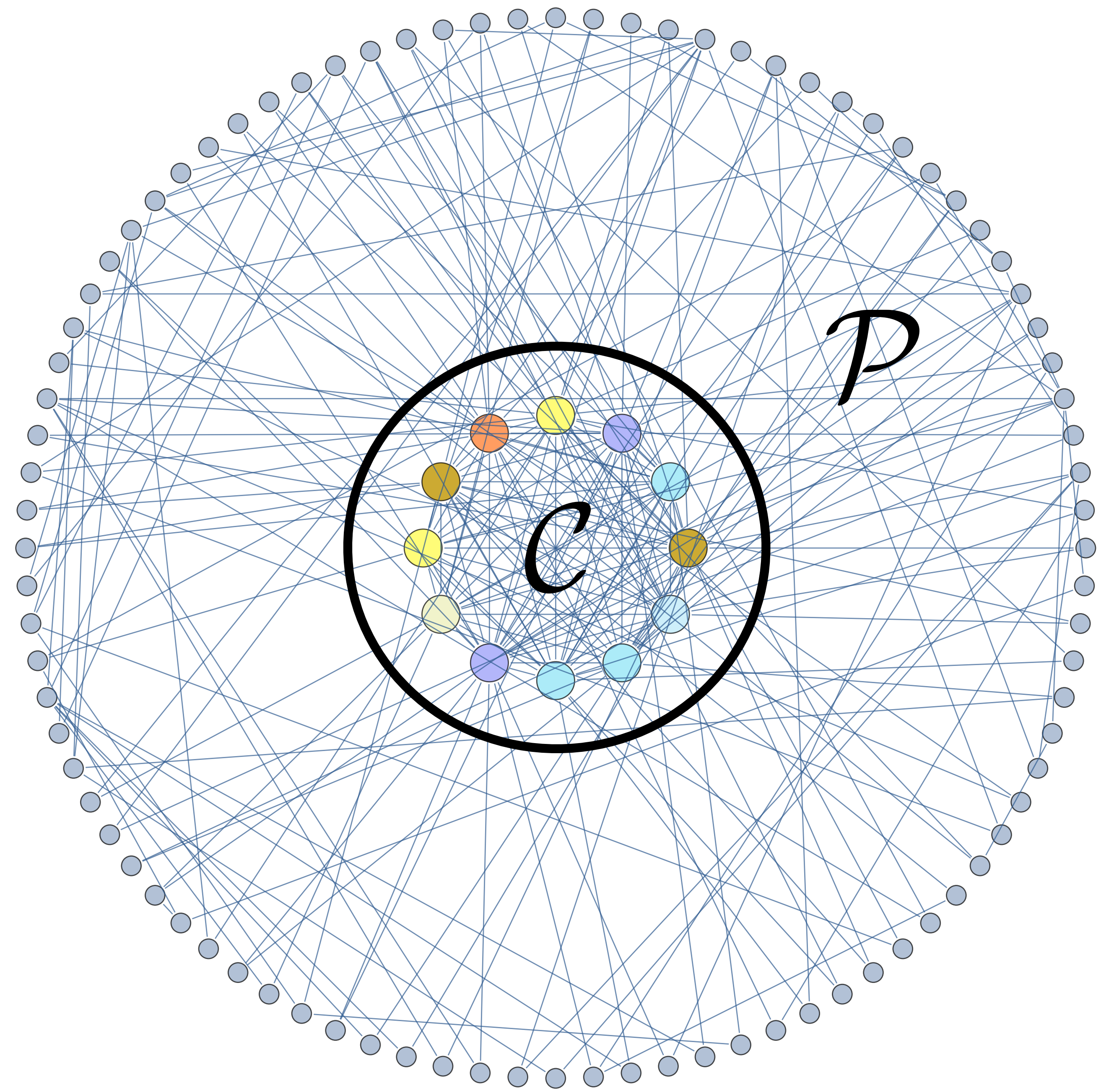
- Social networks structure
  [Avin, Lotker, Pignolet,Turkel 2012]



Core-Periphery Structure

Core:
small, dense

Periphery:
large, sparse



- Social networks structure
  [Avin, Lotker, Pignolet,Turkel 2012]

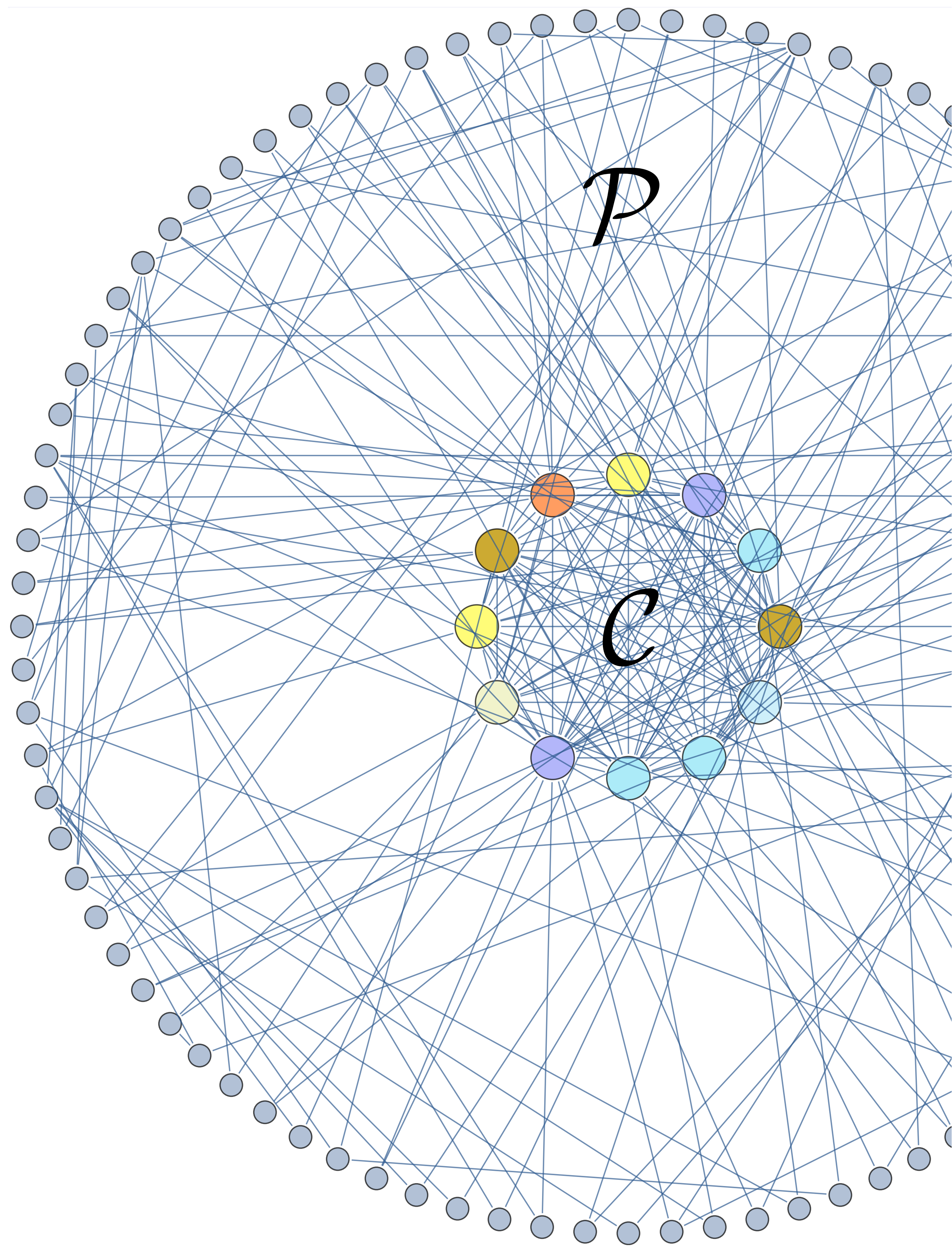
- Global economy - wealthiest countries are well connected with trade and transportation routes

- P2P networks - (e.g., Skype where super nodes are the Core)

Axiomatic Approach

- We define networks using axioms:
  - No concrete generative model
  - Abstract away algorithmic requirements
  - Algorithmic vs structural properties
  - Models that satisfy axioms can be proposed

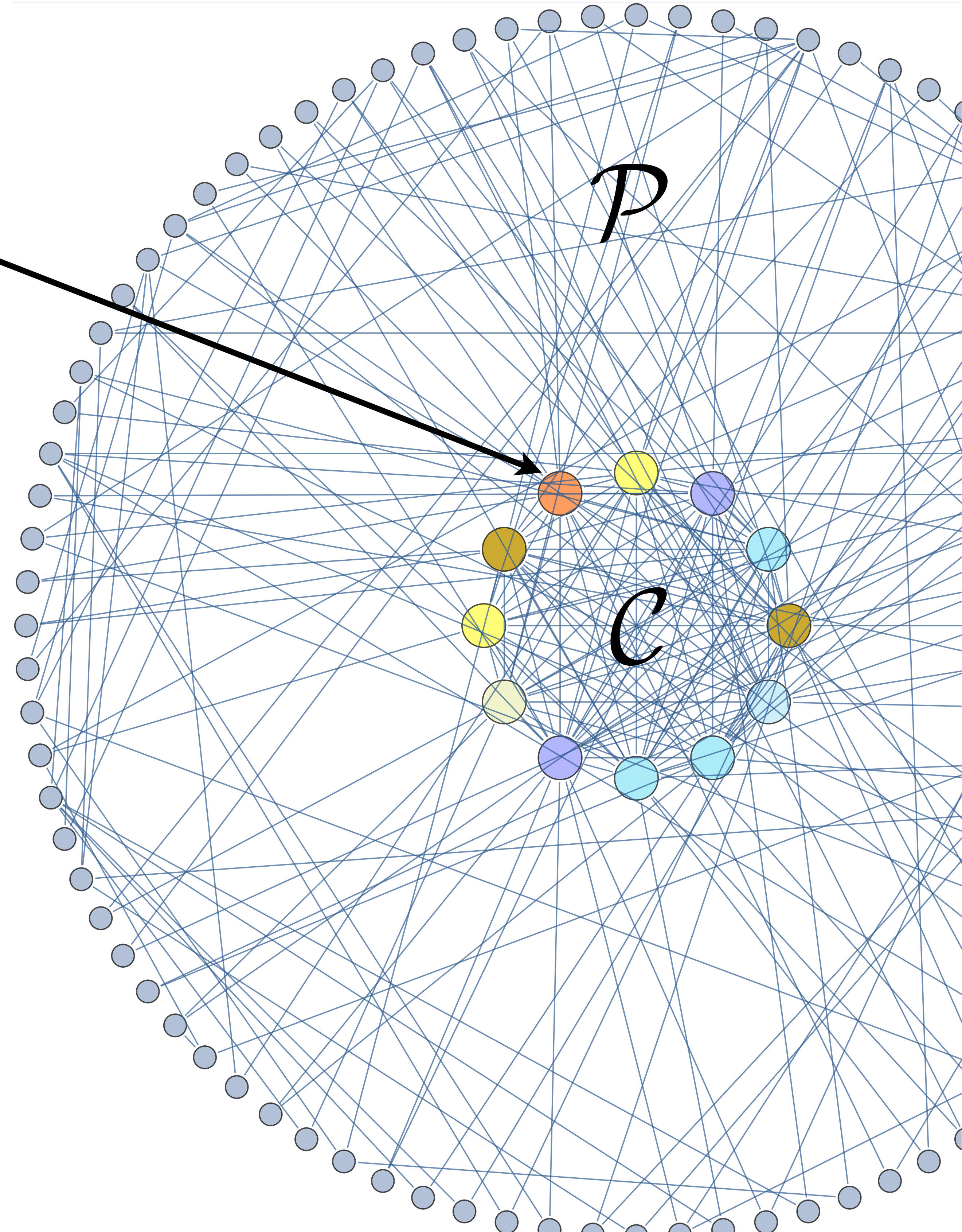
Core-Periphery Axioms: A1, A2, A3



Core-Periphery Axioms: A1, A2, A3

A1: **Balanced *Core* Boundary**

$$\forall v \in \mathcal{C}, \quad \frac{d_{\text{out}}(v)}{d_{\text{in}}(v)} = O(1)$$



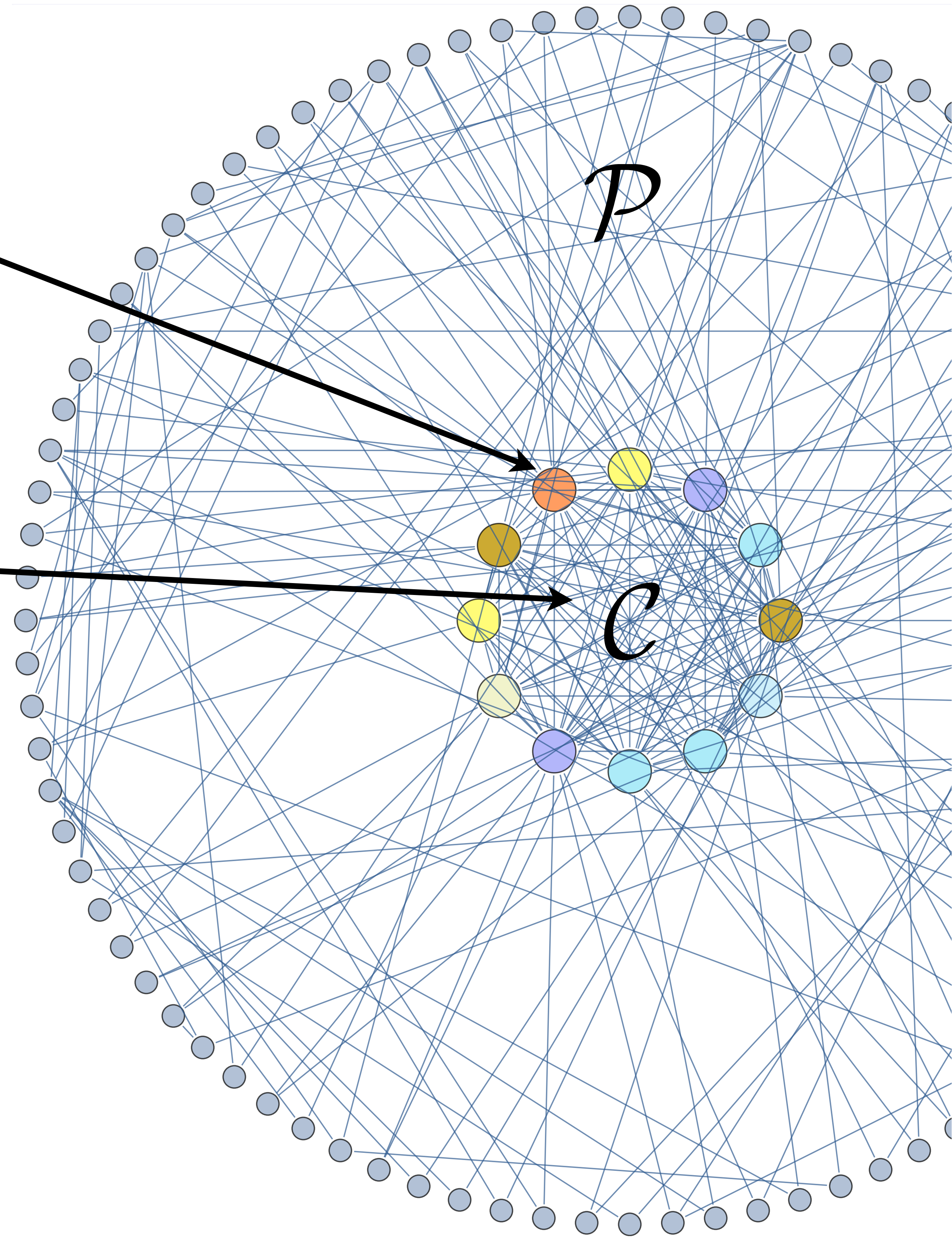
Core-Periphery Axioms: A1, A2, A3

A1: **Balanced *Core* Boundary**

$$\forall v \in \mathcal{C}, \quad \frac{d_{\text{out}}(v)}{d_{\text{in}}(v)} = O(1)$$

A2: ***Core* Clique Emulation**

$$\mathcal{C} \xleftrightarrow{\text{all-to-all}} \mathcal{C}, \text{ in } O(1)$$



Core-Periphery Axioms: A1, A2, A3

A1: **Balanced *Core* Boundary**

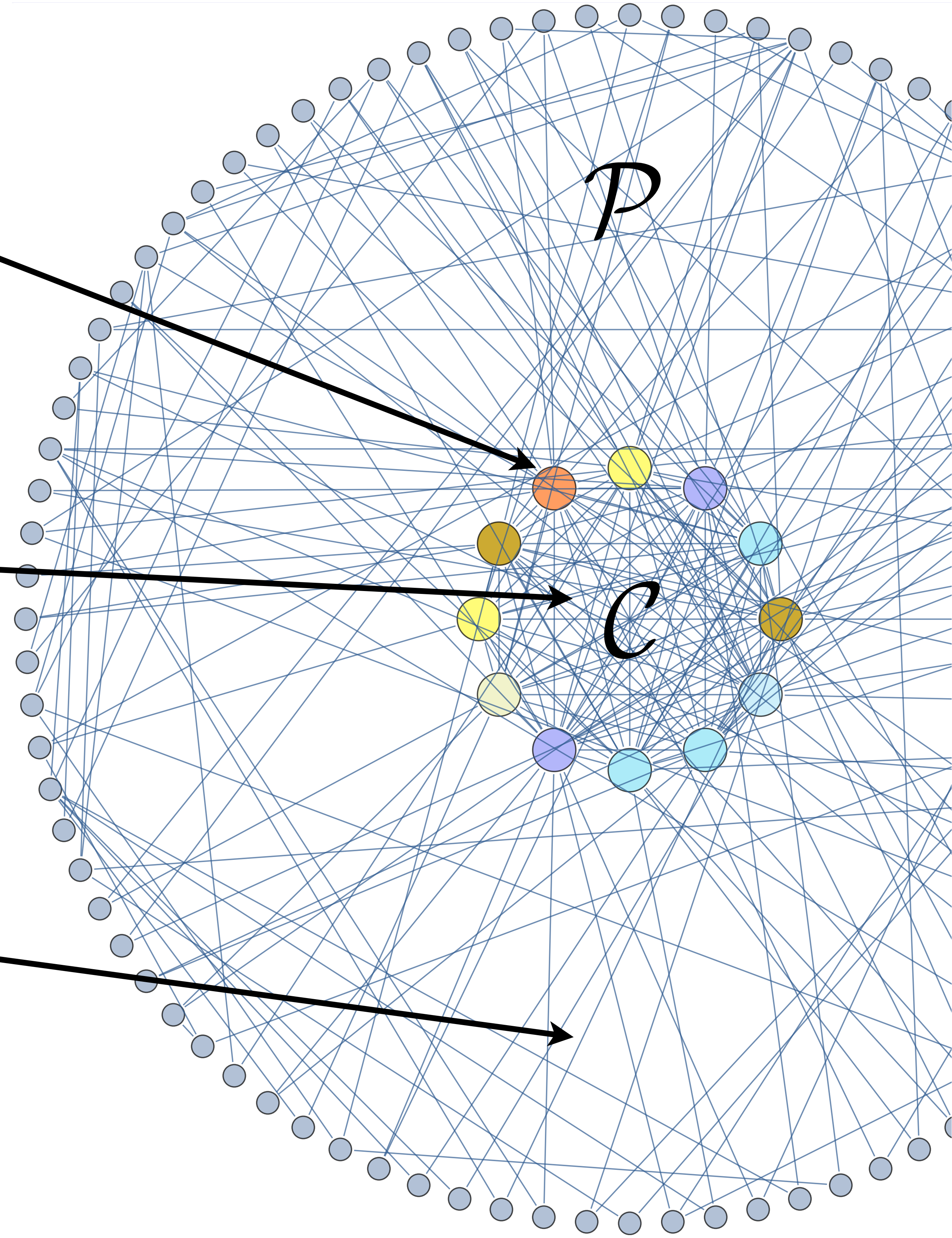
$$\forall v \in \mathcal{C}, \quad \frac{d_{\text{out}}(v)}{d_{\text{in}}(v)} = O(1)$$

A2: ***Core* Clique Emulation**

$$\mathcal{C} \xleftrightarrow{\text{all-to-all}} \mathcal{C}, \ \text{in } O(1)$$

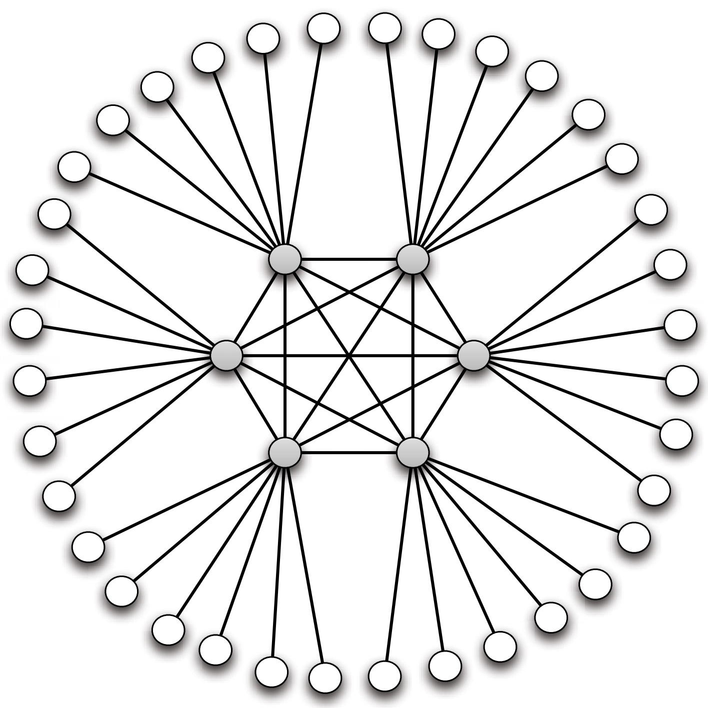
A3: ***Periphery-Core* Convergecast**

$$\mathcal{P} \xrightarrow{\text{all-to-any}} \mathcal{C}, \ \text{in } O(1)$$



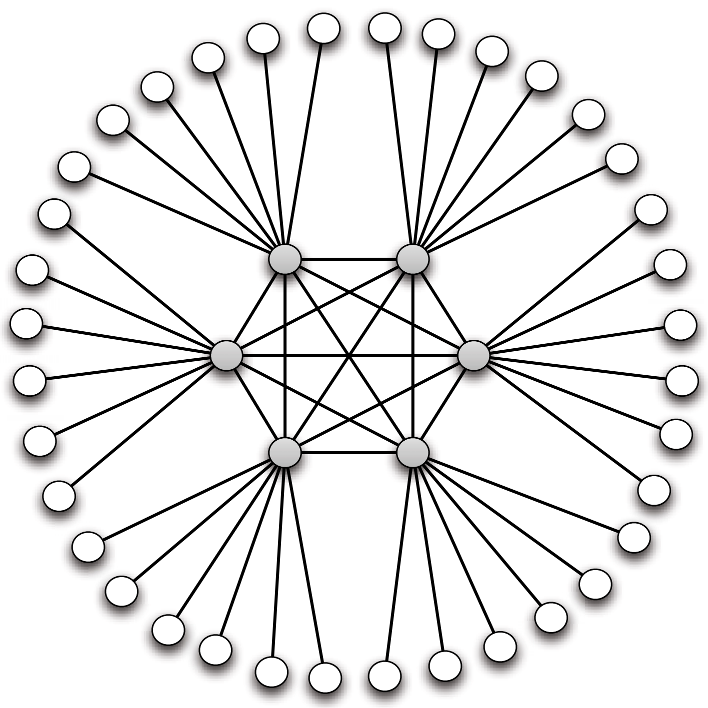
Axiom Independence

Axiom Independence

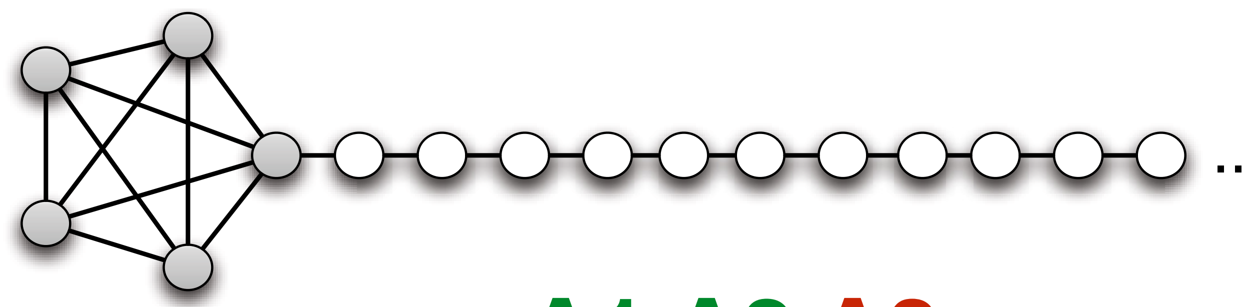


A1 A2 A3

Axiom Independence



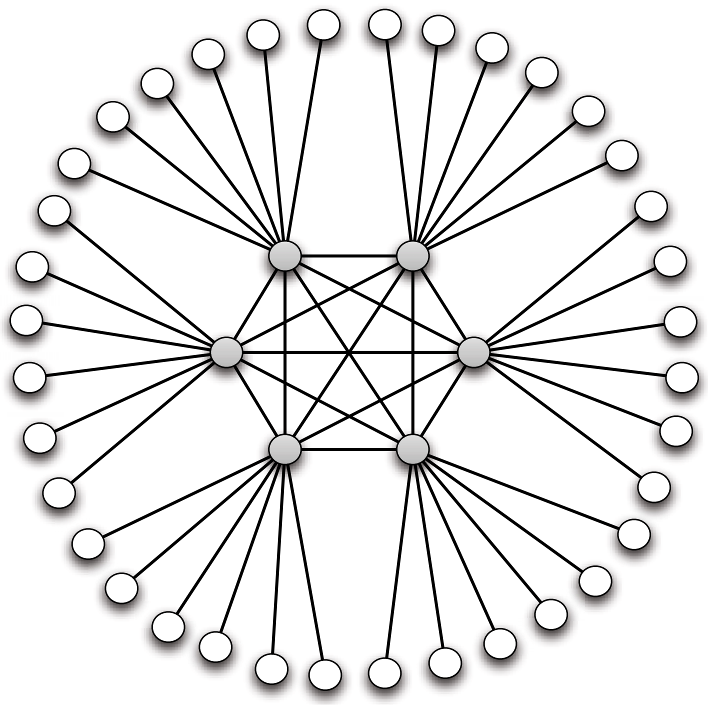
A1 A2 A3



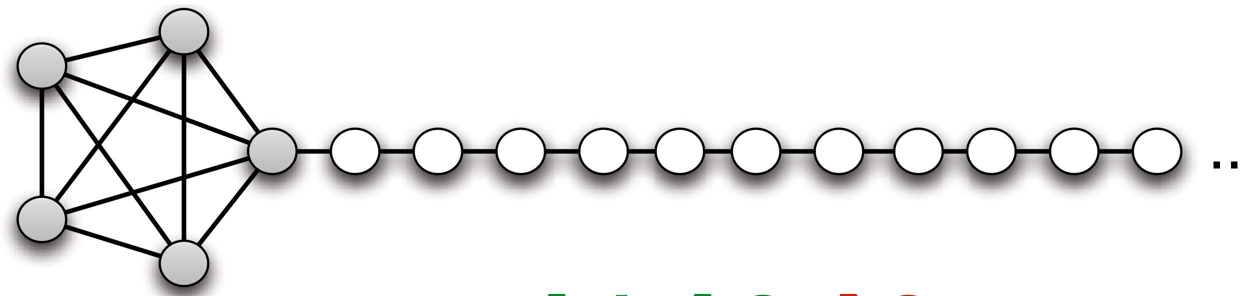
A1 A2 A3

no convergecast

Axiom Independence

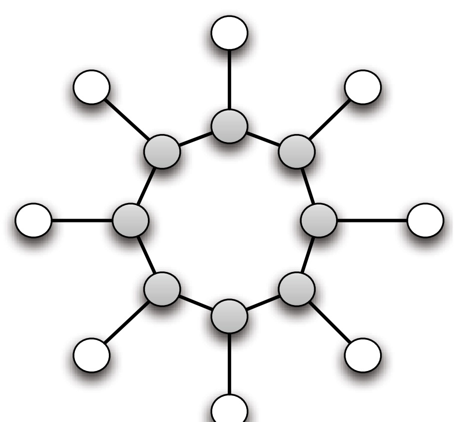


A1 A2 A3



A1 A2 A3

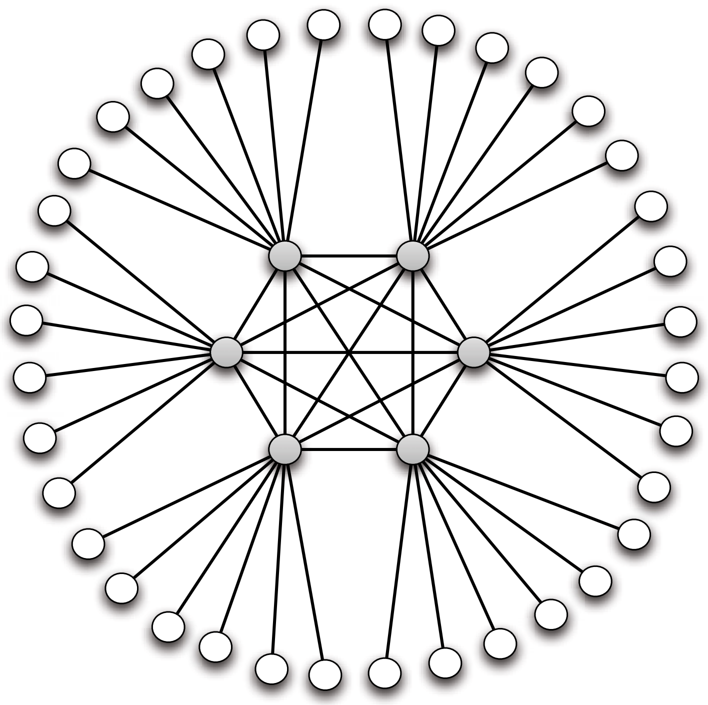
no convergecast



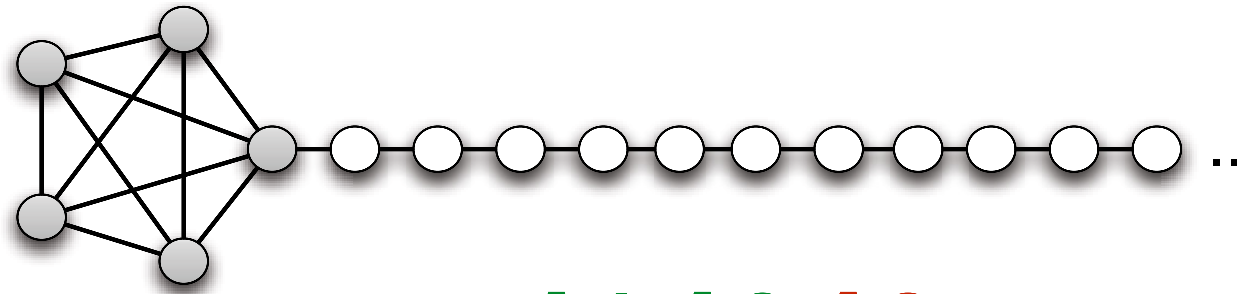
A1 A2 A3

no clique emulation

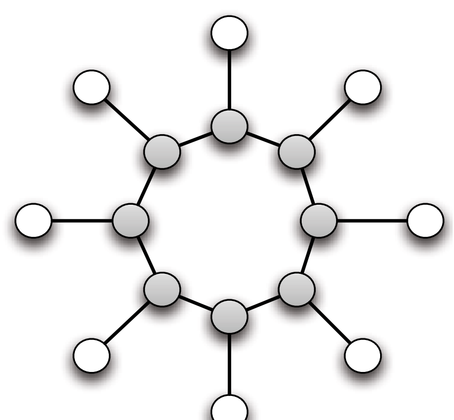
Axiom Independence



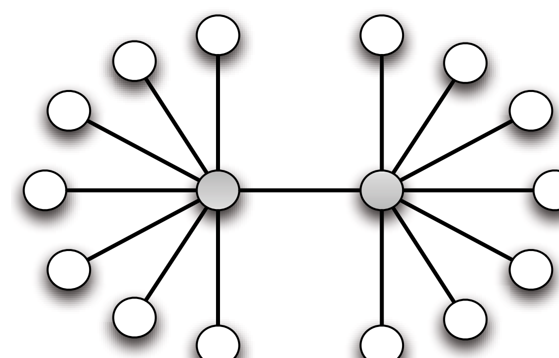
A1 A2 A3



A1 A2 A3
no convergecast

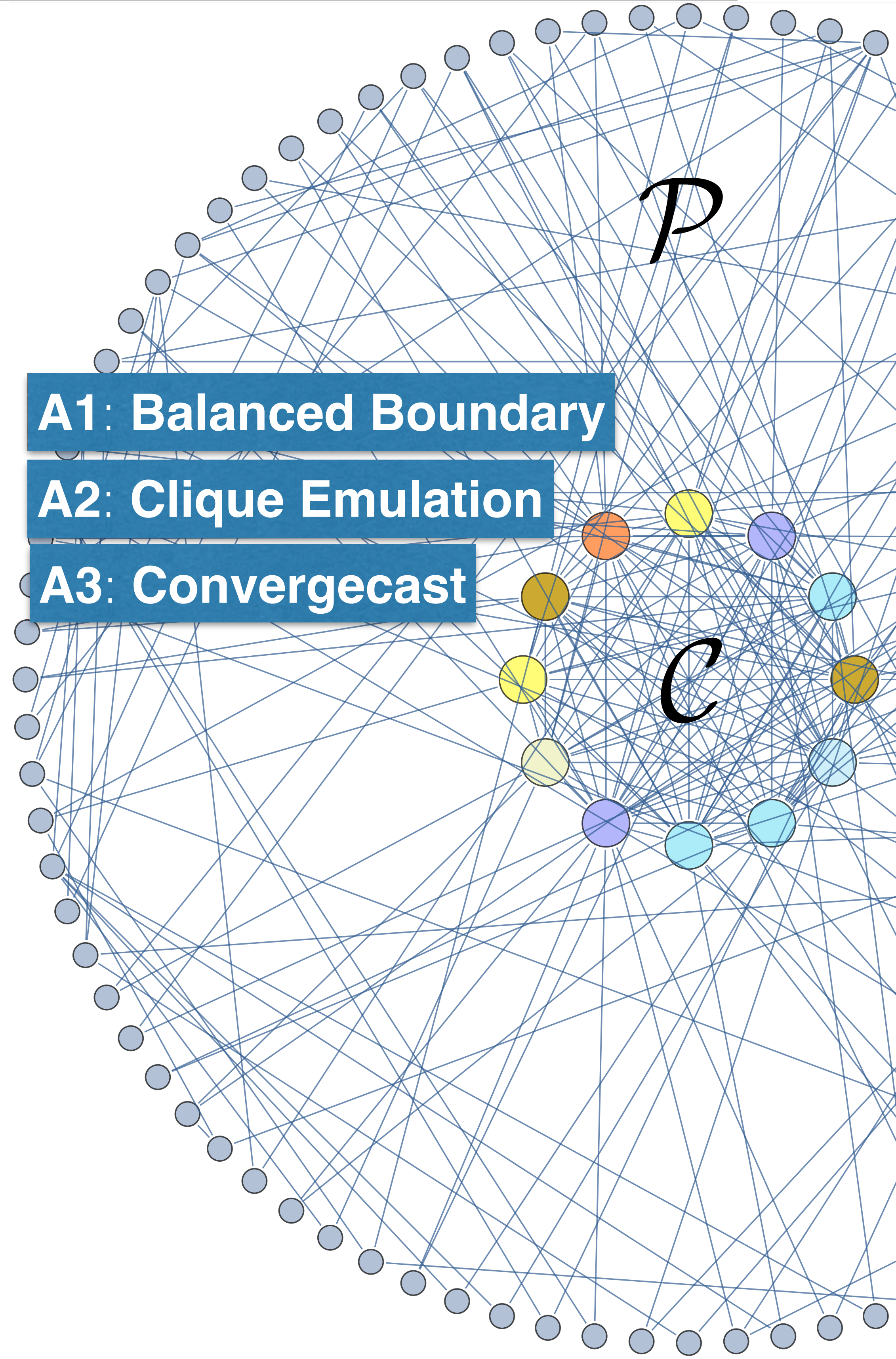


A1 A2 A3
no clique emulation



A1 A2 A3
no balanced boundary

Structural Implications

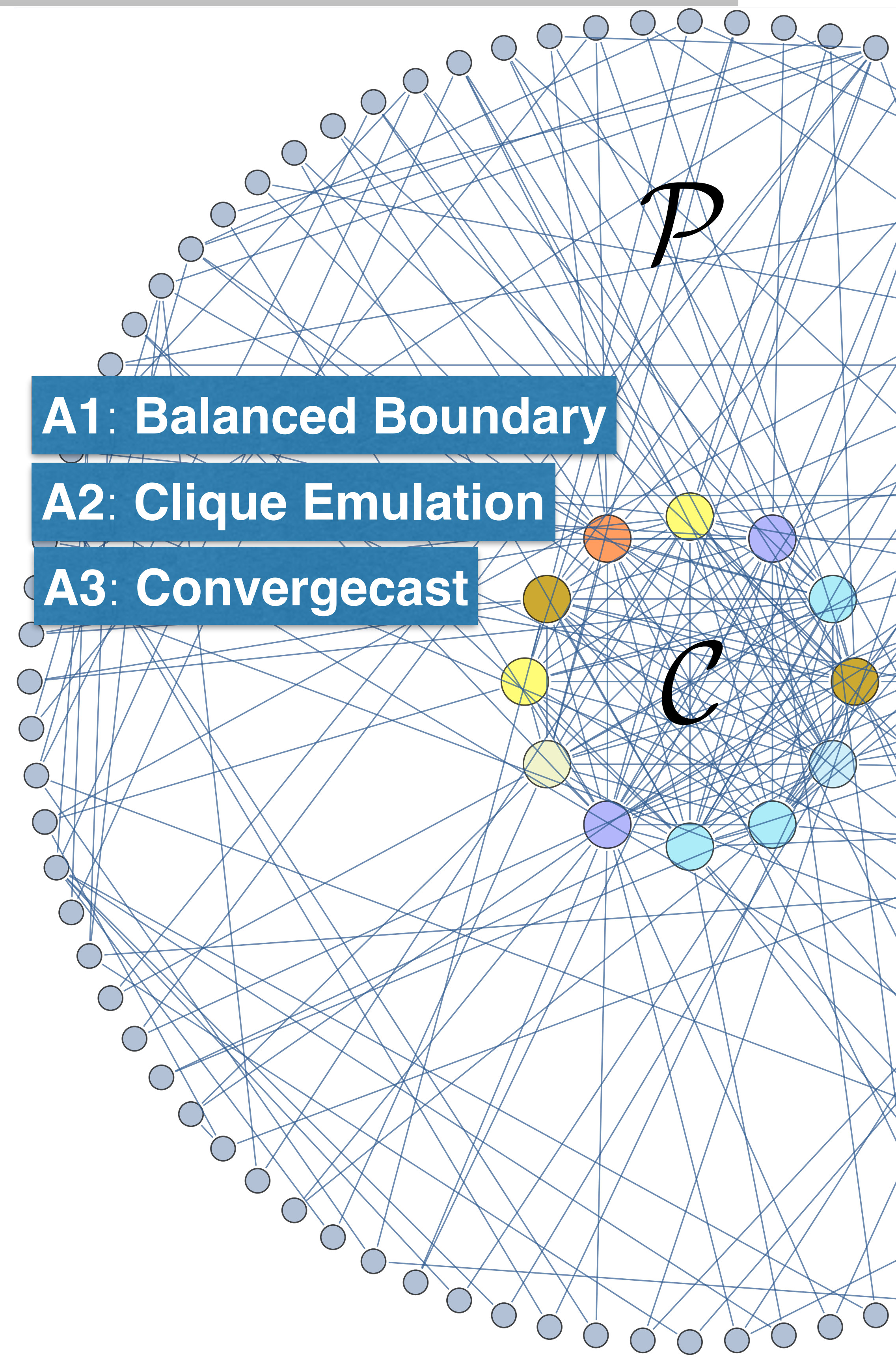


Structural Implications

Balanced boundary

$$\forall v \in \mathcal{C}, d_{\text{out}}(v) = O(n_{\mathcal{C}}) \quad \textbf{A1}$$

$$d_{\text{in}}(v) = \Theta(n_{\mathcal{C}}) \quad \textbf{A2}$$



Structural Implications

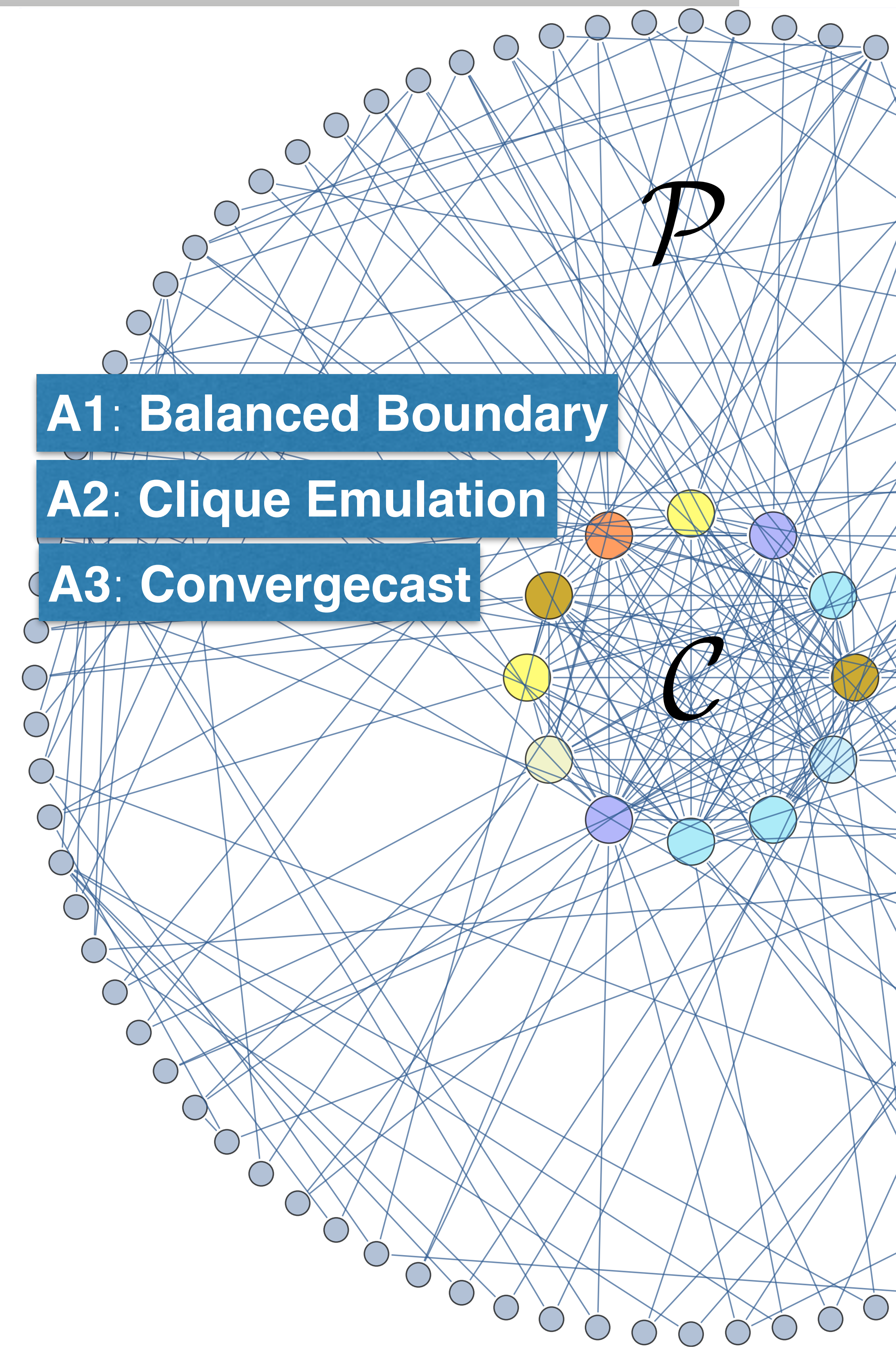
Balanced boundary

$$\forall v \in \mathcal{C}, d_{\text{out}}(v) = O(n_{\mathcal{C}})$$ A1

$$d_{\text{in}}(v) = \Theta(n_{\mathcal{C}})$$ A2

***Core* is dense** A2

$$m_{\mathcal{C}} = \Theta(n_{\mathcal{C}}^2)$$



Structural Implications

Balanced boundary

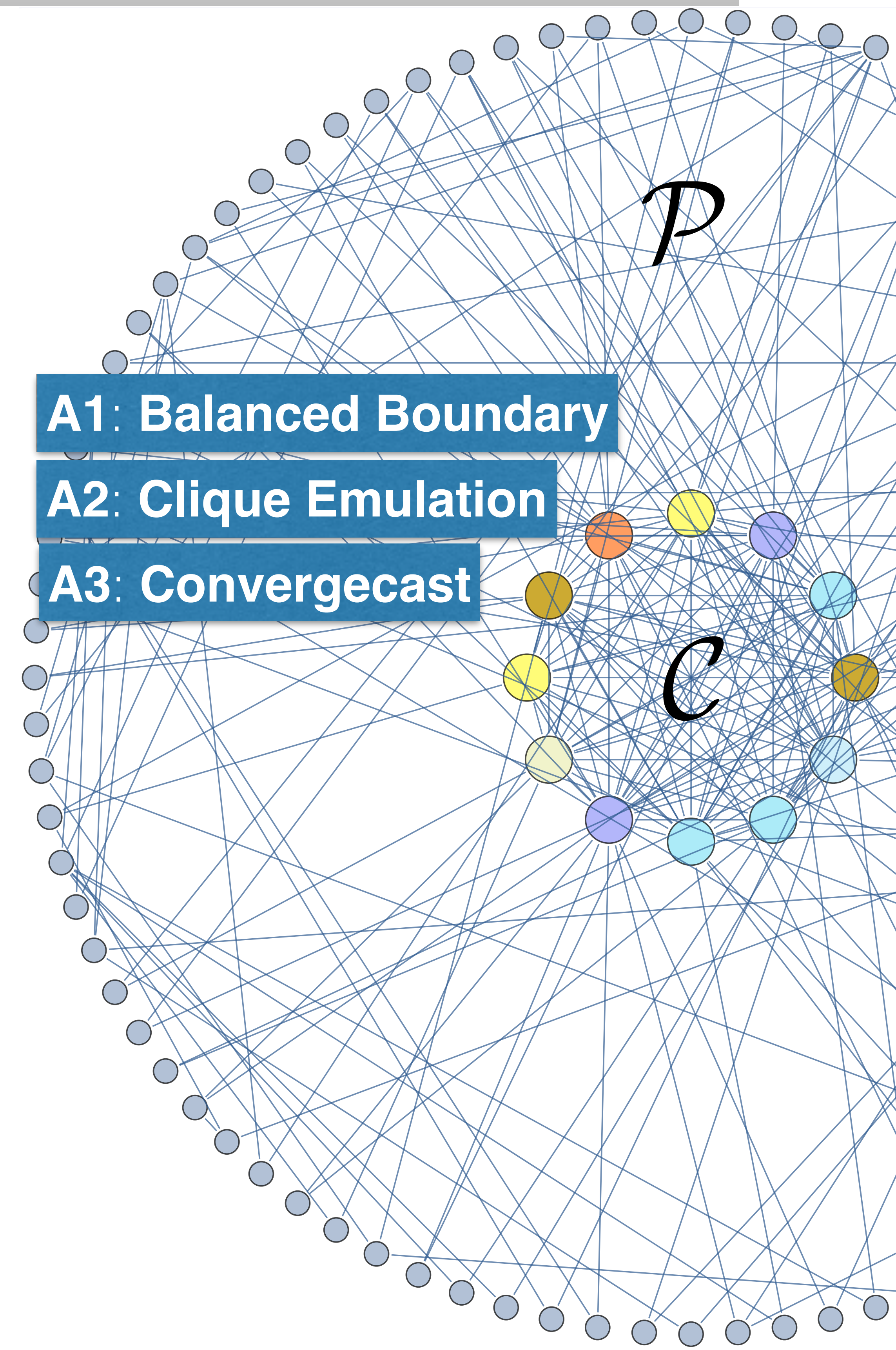
$$\forall v \in \mathcal{C}, d_{\text{out}}(v) = O(n_{\mathcal{C}})$$ **A1**

$$d_{\text{in}}(v) = \Theta(n_{\mathcal{C}})$$ **A2**

Core is dense **A2**

$$m_{\mathcal{C}} = \Theta(n_{\mathcal{C}}^2)$$

Constant Diameter **A2,A3**



Structural Implications

Balanced boundary

$$\forall v \in \mathcal{C}, d_{\text{out}}(v) = O(n_{\mathcal{C}})$$ A1

$$d_{\text{in}}(v) = \Theta(n_{\mathcal{C}})$$ A2

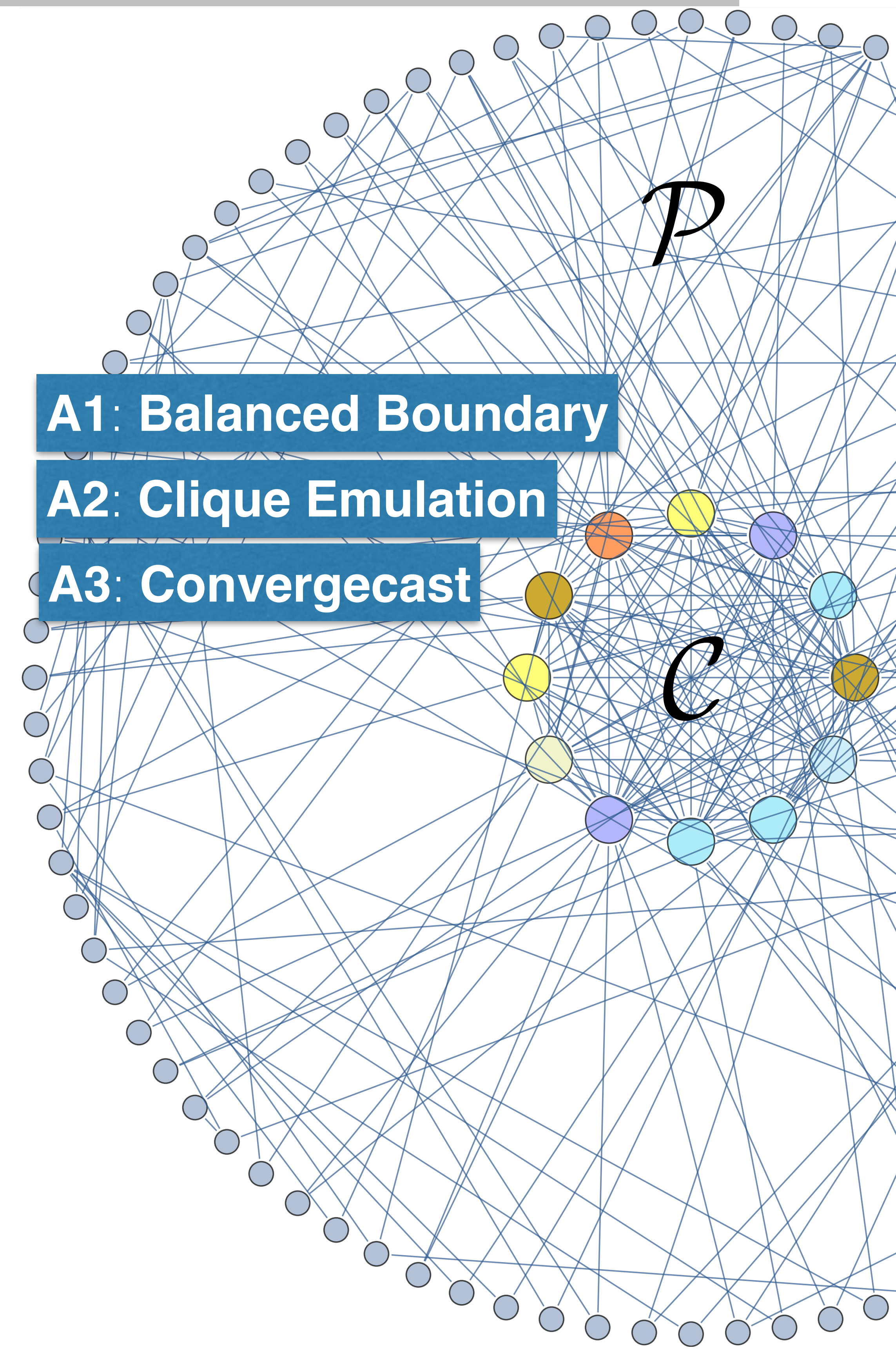
***Core* is dense** A2

$$m_{\mathcal{C}} = \Theta(n_{\mathcal{C}}^2)$$

Constant Diameter A2,A3

Size of the *Core*

$$\Omega(\sqrt{n}) \leq n_{\mathcal{C}} \leq O(\sqrt{m})$$



Structural Implications

Balanced boundary

$$\forall v \in \mathcal{C}, d_{\text{out}}(v) = O(n_{\mathcal{C}})$$ A1

$$d_{\text{in}}(v) = \Theta(n_{\mathcal{C}})$$ A2

Core is dense

A2

$$m_{\mathcal{C}} = \Theta(n_{\mathcal{C}}^2)$$

Constant Diameter

A2,A3

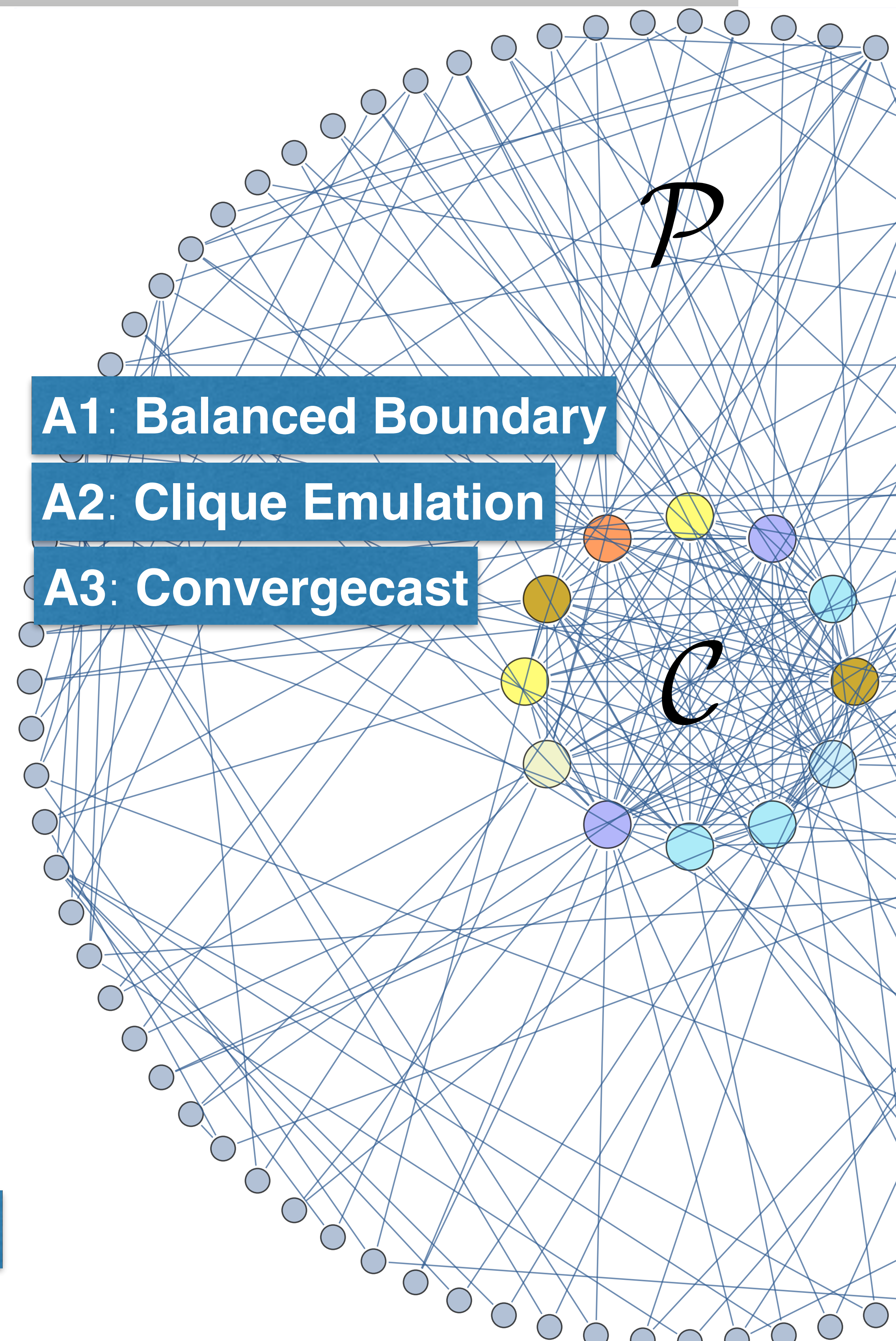
Size of the *Core*

$$\Omega(\sqrt{n}) \leq n_{\mathcal{C}} \leq O(\sqrt{m})$$

$$\sum_{v \in \mathcal{C}} d_{\text{in}}(v) \leq 2m$$

$$n_{\mathcal{C}}^2 \leq 2m$$ A2

$$n_{\mathcal{C}} \leq O(\sqrt{m})$$



Structural Implications

Balanced boundary

$$\forall v \in \mathcal{C}, d_{\text{out}}(v) = O(n_{\mathcal{C}})$$ A1

$$d_{\text{in}}(v) = \Theta(n_{\mathcal{C}})$$ A2

Core is dense

A2

$$m_{\mathcal{C}} = \Theta(n_{\mathcal{C}}^2)$$

Constant Diameter

A2,A3

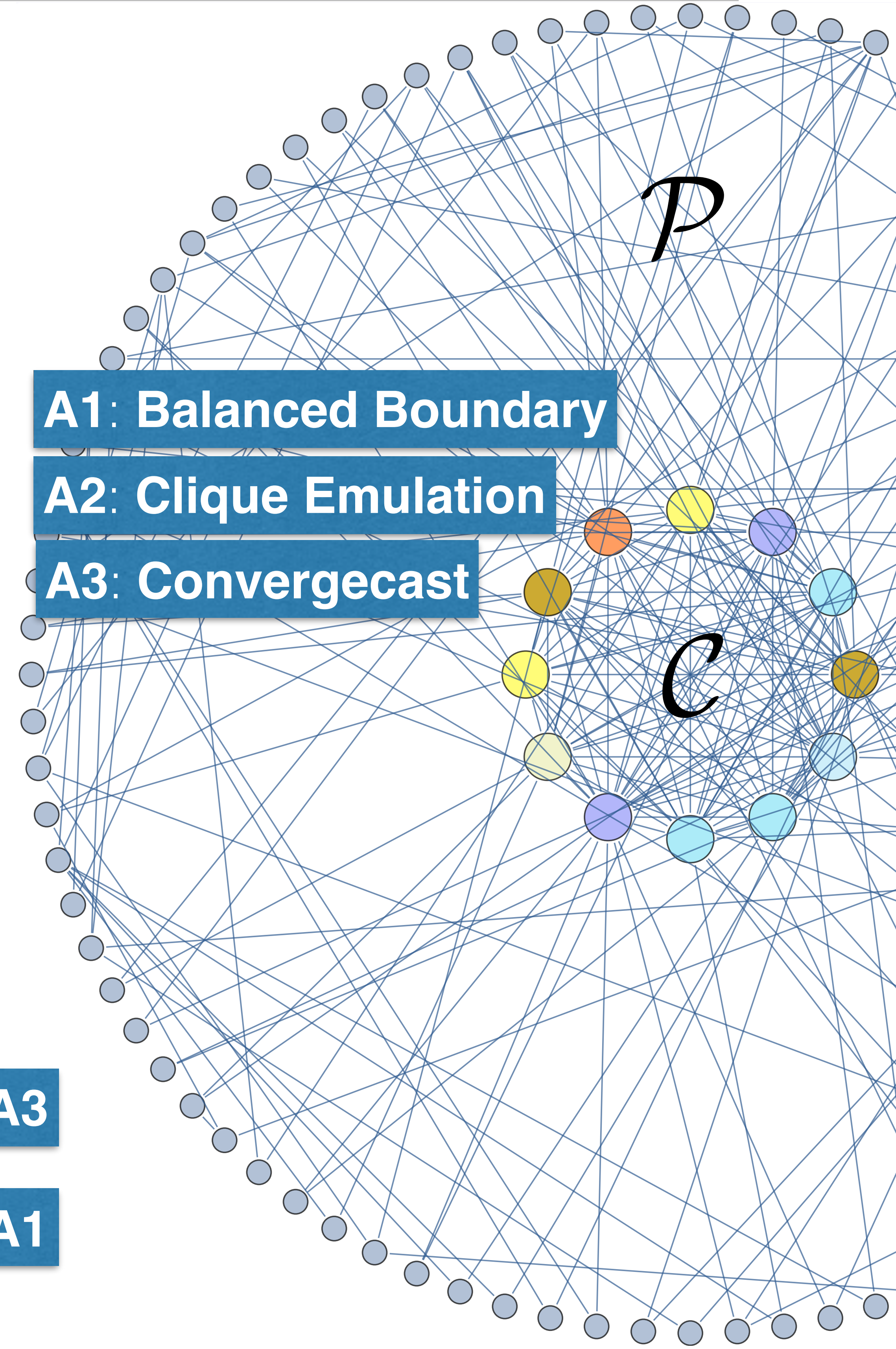
Size of the *Core*

$$\Omega(\sqrt{n}) \leq n_{\mathcal{C}} \leq O(\sqrt{m})$$

$$\sum_{v \in \mathcal{C}} d_{\text{out}}(v) \geq n - n_{\mathcal{C}}$$ A3

$$n_{\mathcal{C}}^2 \geq n - n_{\mathcal{C}}$$ A1

$$n_{\mathcal{C}} \geq \Omega(\sqrt{n})$$



Structural Implications

Balanced boundary

$\forall v \in \mathcal{C}, d_{\text{out}}(v) = O(n_{\mathcal{C}})$ **A1**

$d_{\text{in}}(v) = \Theta(n_{\mathcal{C}})$ **A2**

***Core* is dense** **A2**

$m_{\mathcal{C}} = \Theta(n_{\mathcal{C}}^2)$

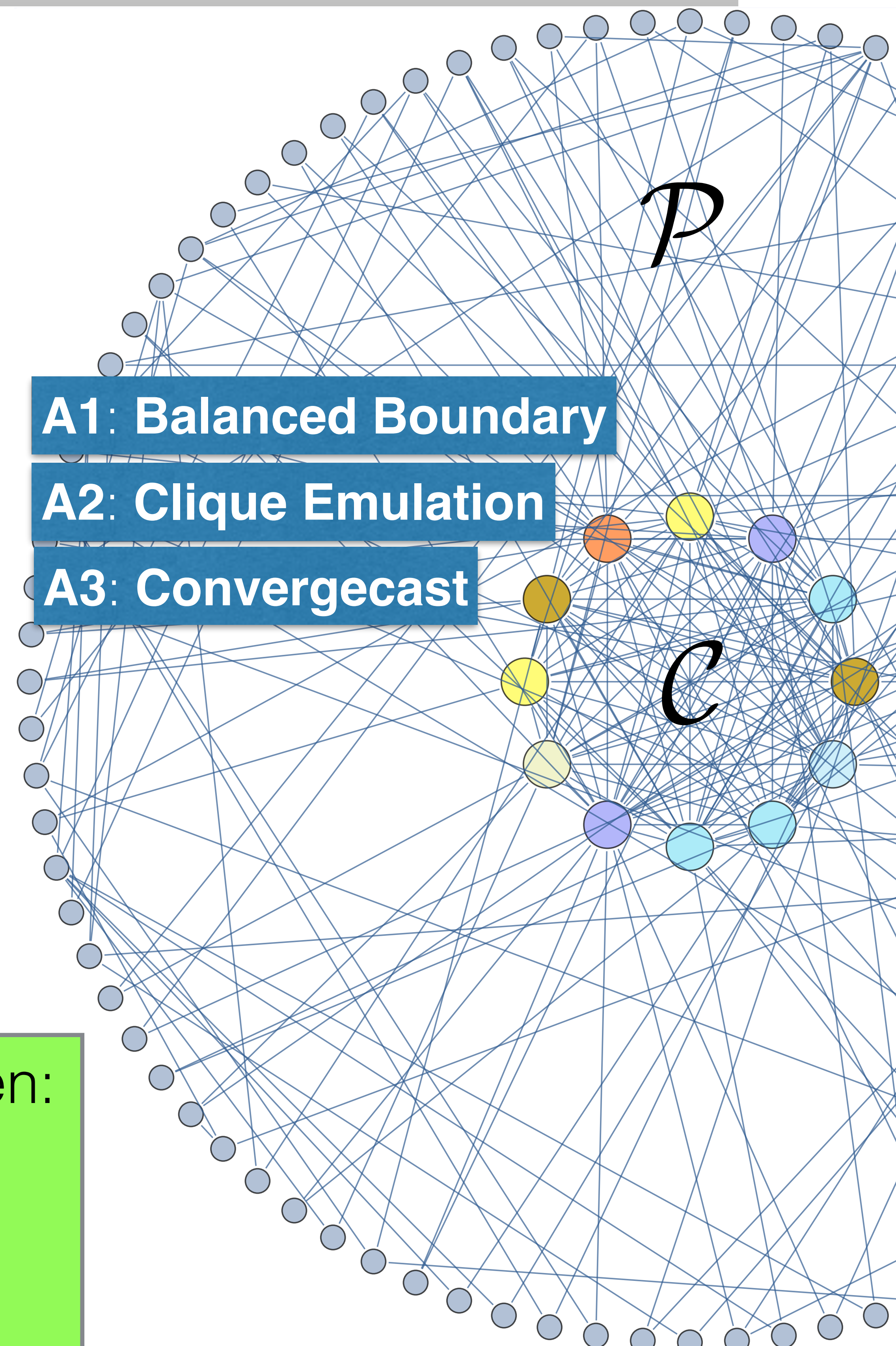
Constant Diameter **A2,A3**

Size of the *Core*

$\Omega(\sqrt{n}) \leq n_{\mathcal{C}} \leq O(\sqrt{m})$

if *m* is linear, then:

$$n_{\mathcal{C}} = \Theta(\sqrt{n})$$
$$m_{\mathcal{C}} = \Theta(m)$$



Algorithms for Core-Periphery Networks

Matrix Operations

$A, B \in \mathbb{Z}^{n \times n}$, $O(k)$ sparse

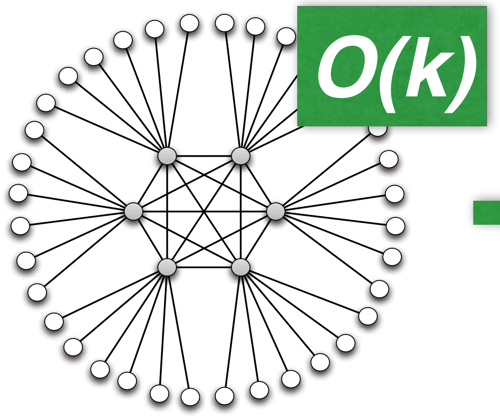
$s \in \mathbb{Z}^n$

Algorithms for Core-Periphery Networks

Matrix Operations

$A, B \in \mathbb{Z}^{n \times n}$, $O(k)$ sparse

$s \in \mathbb{Z}^n$

A →  → A^T

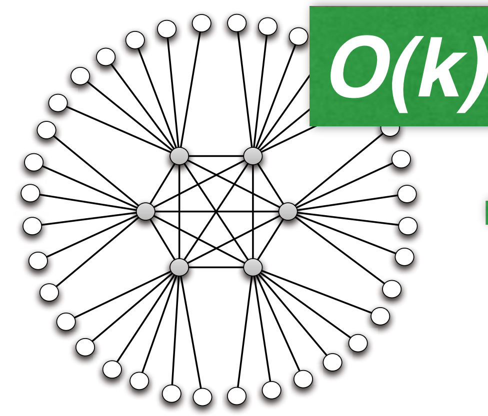
Algorithms for Core-Periphery Networks

Matrix Operations

$A, B \in \mathbb{Z}^{n \times n}$, $O(k)$ sparse

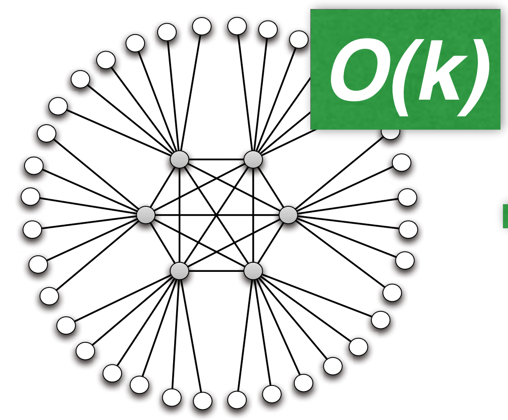
$s \in \mathbb{Z}^n$

A



A^T

s, A



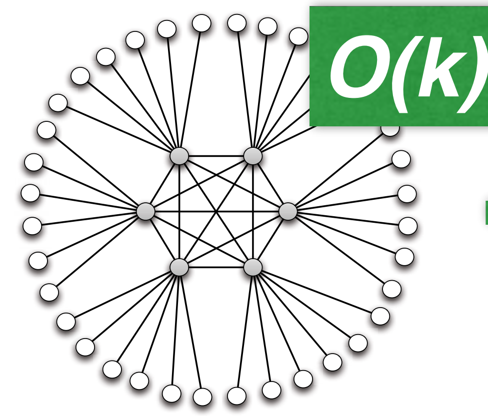
$s' = sA$

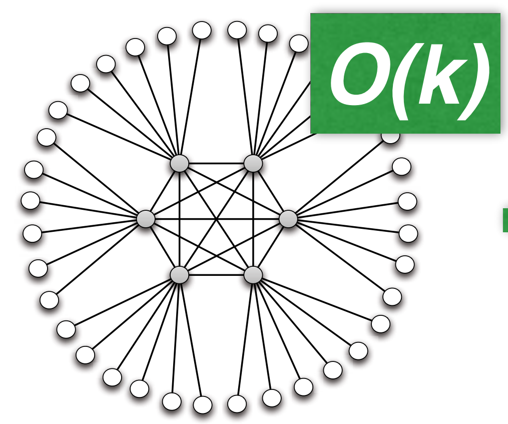
Algorithms for Core-Periphery Networks

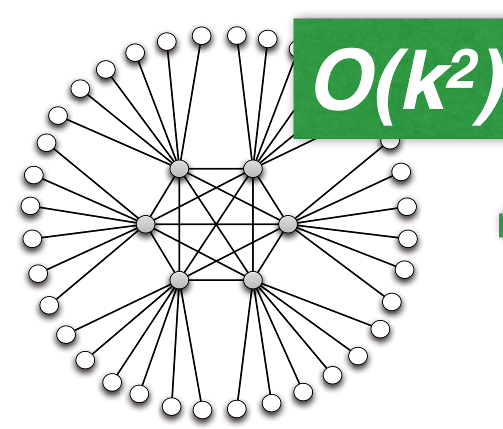
Matrix Operations

$A, B \in \mathbb{Z}^{n \times n}$, $O(k)$ sparse

$s \in \mathbb{Z}^n$

A →  → A^T

s, A →  → $s' = sA$

A, B →  → $C = AB$

Algorithms for Core-Periphery Networks

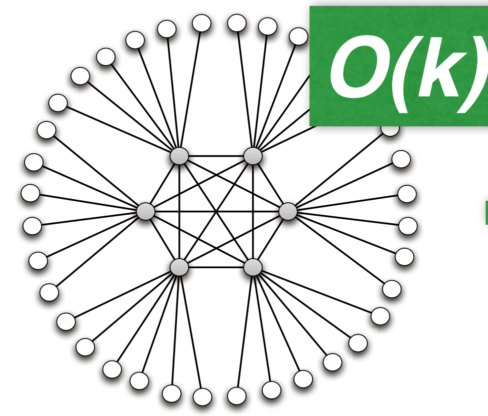
Matrix Operations

$A, B \in \mathbb{Z}^{n \times n}$, $O(k)$ sparse

$s \in \mathbb{Z}^n$

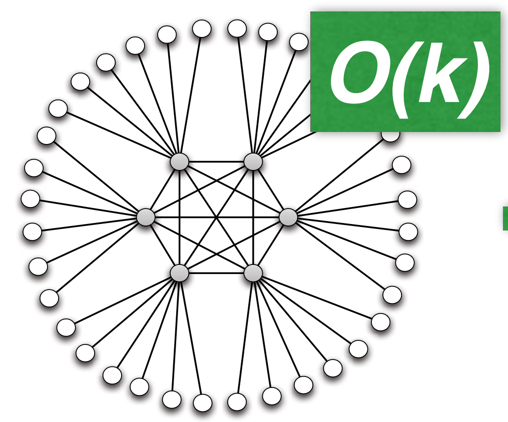
Memory per node - $O(\sqrt{n})$

A →



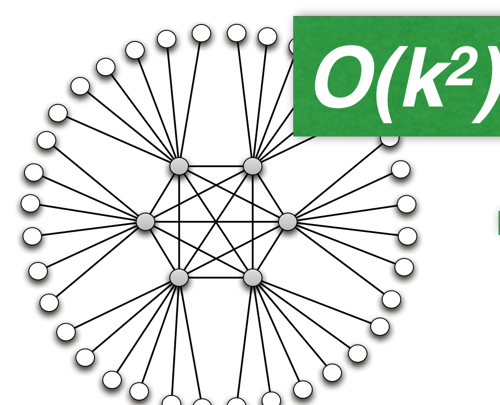
→ A^T

s, A →



→ $s' = sA$

A, B →



→ $C = AB$

Algorithms for Core-Periphery Networks

Matrix Operations

$A, B \in \mathbb{Z}^{n \times n}$, $O(k)$ sparse

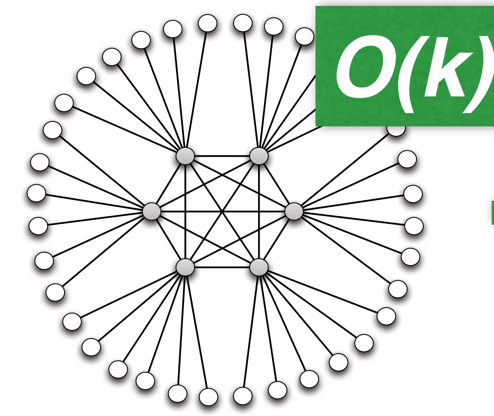
$s \in \mathbb{Z}^n$

Memory per node - $O(\sqrt{n})$

Based on fast info spreading in clique

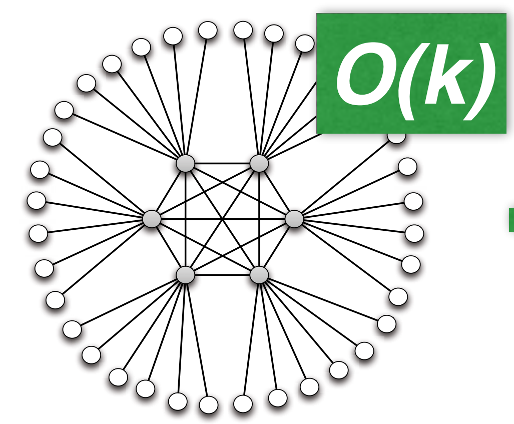
[Lenzen, Wattenhofer, 2011]

$A \rightarrow$



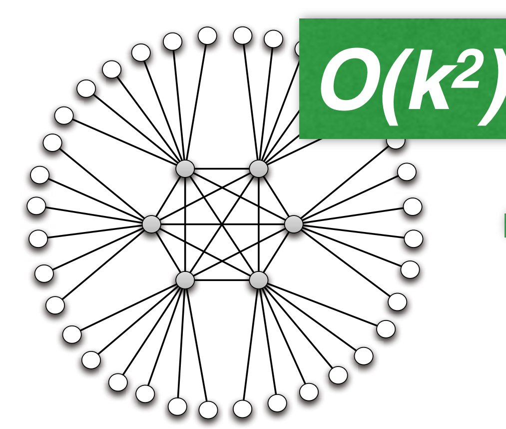
$\rightarrow A^T$

$s, A \rightarrow$



$\rightarrow s' = sA$

$A, B \rightarrow$



$\rightarrow C = AB$

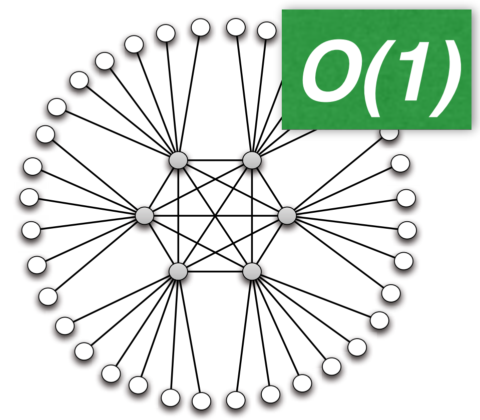
Algorithms for Core-Periphery Networks

Aggregate Functions

Algorithms for Core-Periphery Networks

Aggregate Functions

value for each node



each node knows:

rank of its value

mode of the values

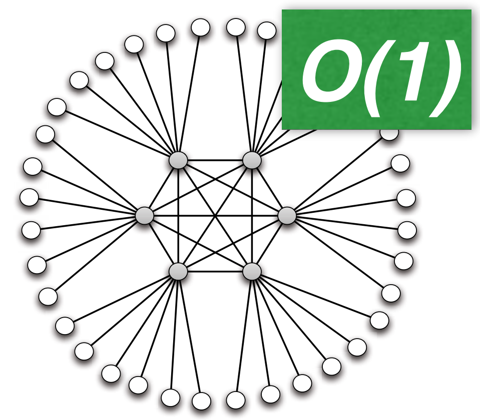
num of distinct values

median

Algorithms for Core-Periphery Networks

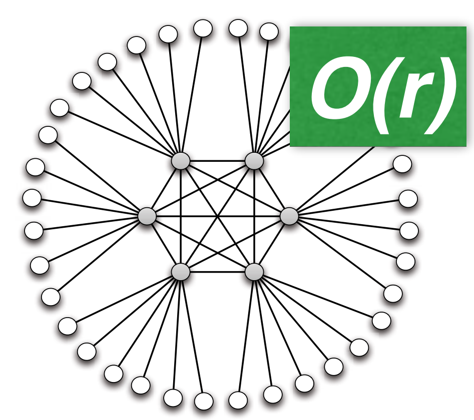
Aggregate Functions

value for each node



each node knows:
rank of its value
mode of the values
num of distinct values
median

value for each node
each value belongs to
a specific group/area

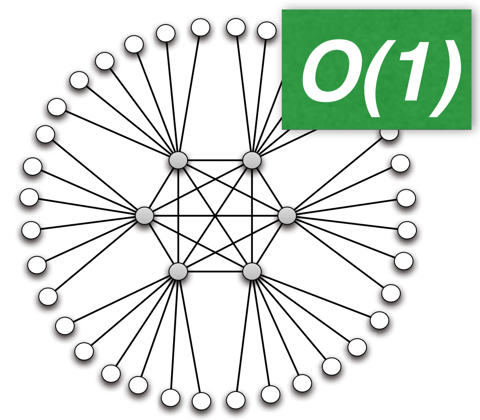


each node knows
top *r* of each area

Algorithms for Core-Periphery Networks

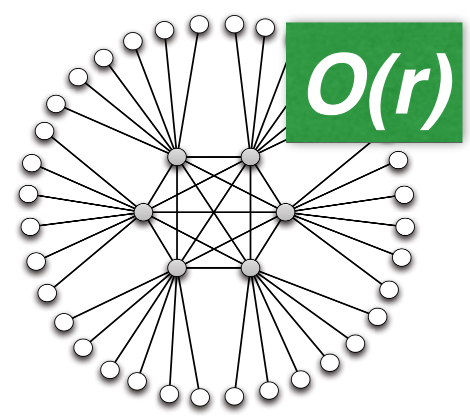
Aggregate Functions

value for each node



each node knows:
rank of its value
mode of the values
num of distinct values
median

value for each node
each value belongs to
a specific group/area



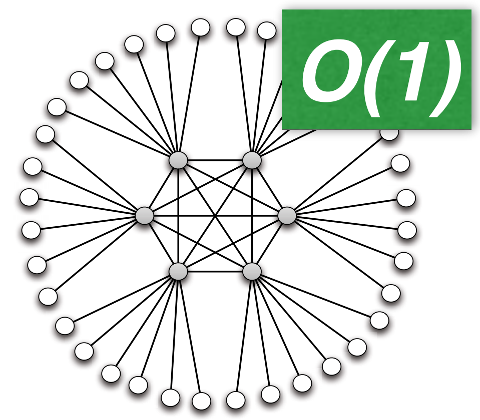
each node knows
top *r* of each area

Memory per node - $O(\sqrt{n})$

Algorithms for Core-Periphery Networks

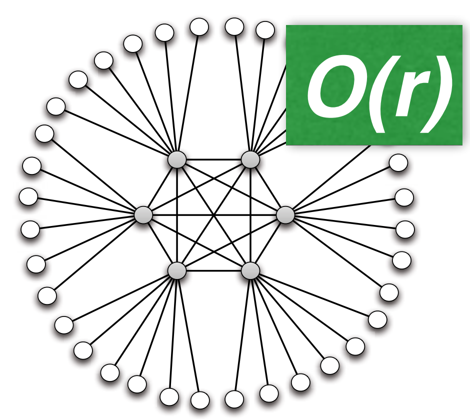
Aggregate Functions

value for each node



each node knows:
rank of its value
mode of the values
num of distinct values
median

value for each node
each value belongs to
a specific group/area

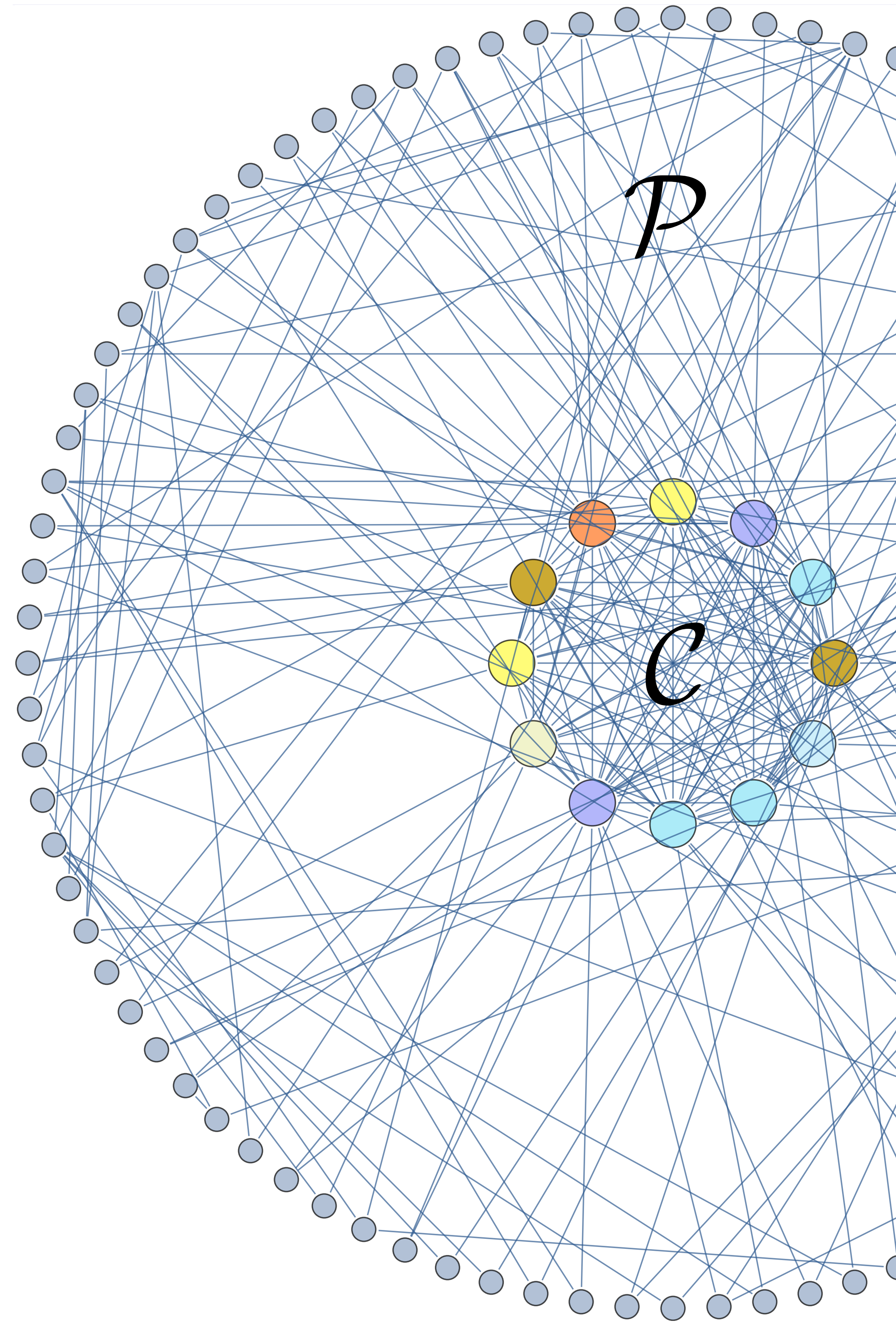


each node knows
top *r* of each area

Memory per node - $O(\sqrt{n})$

Based on fast sorting in clique
[Lenzen, 2013]

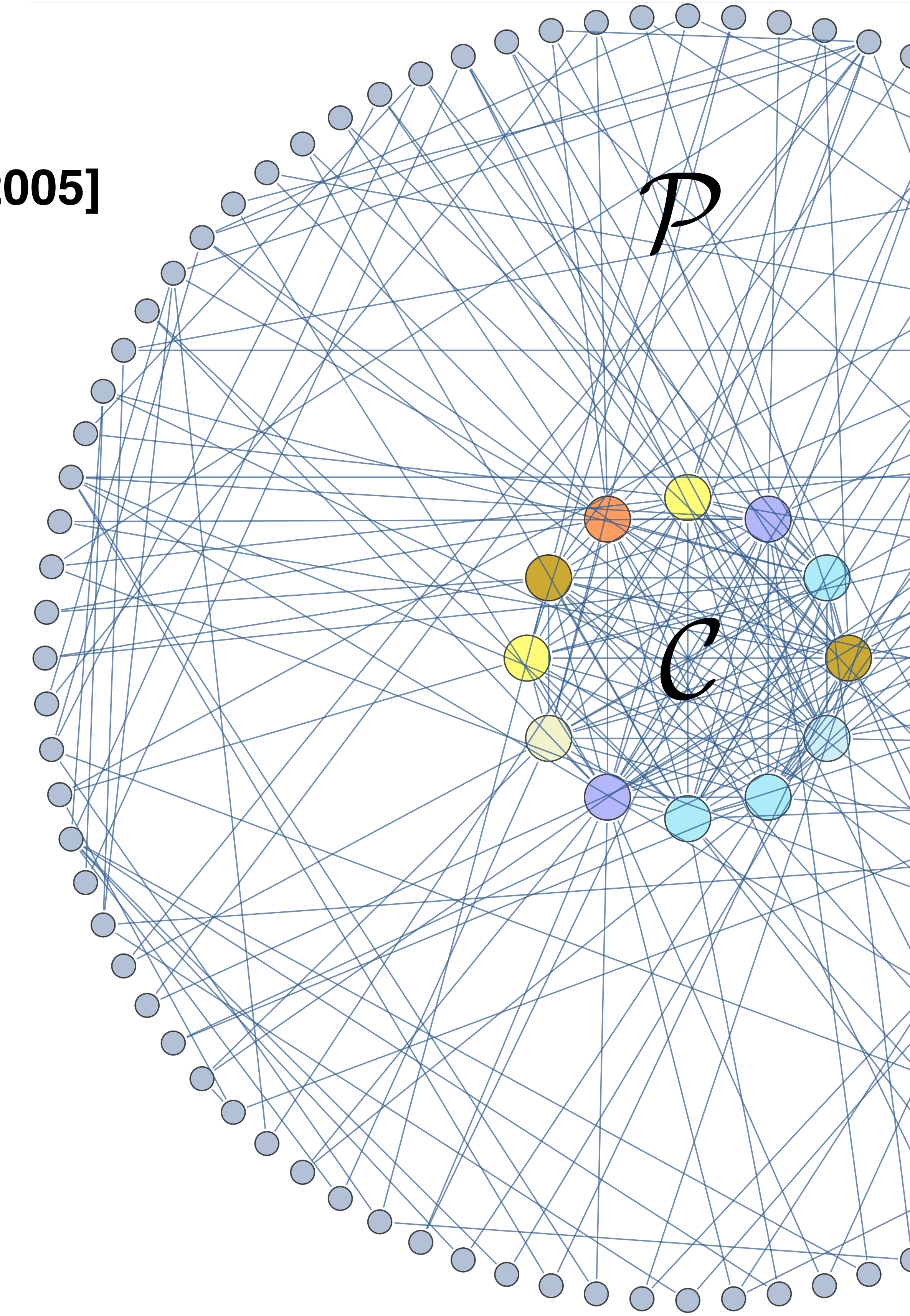
Distributed Minimum Spanning Tree (*CONGEST* Model)



Distributed Minimum Spanning Tree (*CONGEST* Model)

$D = 1 : \quad O(\log \log n)$

[Z. Lotker, B. Patt-Shamir, E. Pavlov, D. Peleg 2005]



Distributed Minimum Spanning Tree (*CONGEST* Model)

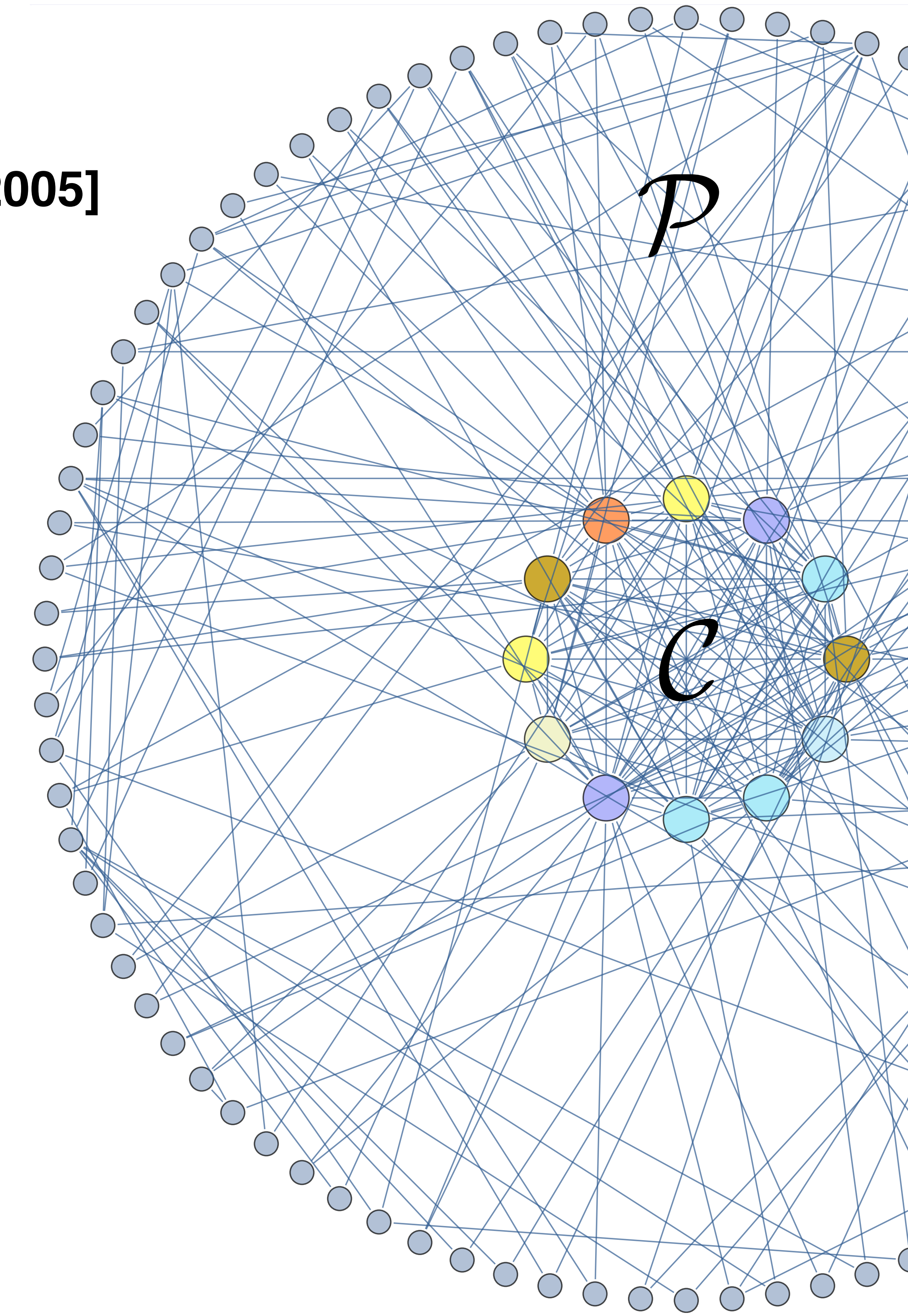
$D = 1 : \quad O(\log \log n)$

[Z. Lotker, B. Patt-Shamir, E. Pavlov, D. Peleg 2005]

$D = 2 : \quad O(\log n)$

$D \geq 3 : \quad \Omega(\sqrt[3]{n})$

[Z. Lotker, B. Patt-Shamir, D. Peleg 2006]



Distributed Minimum Spanning Tree (*CONGEST* Model)

$D = 1: \quad O(\log \log n)$

[Z. Lotker, B. Patt-Shamir, E. Pavlov, D. Peleg 2005]

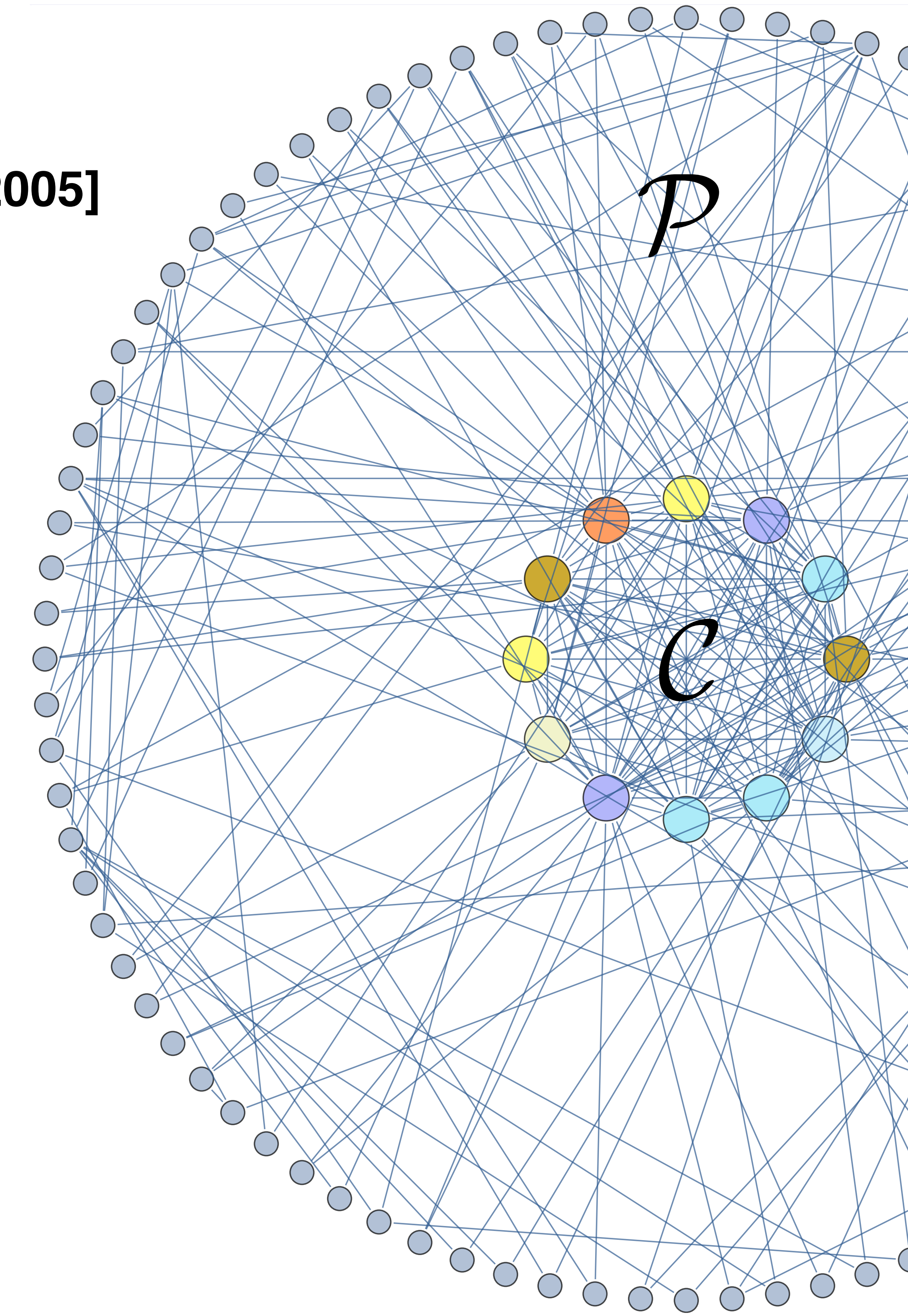
$D = 2: \quad O(\log n)$

$D \geq 3: \quad \Omega(\sqrt[3]{n})$

[Z. Lotker, B. Patt-Shamir, D. Peleg 2006]

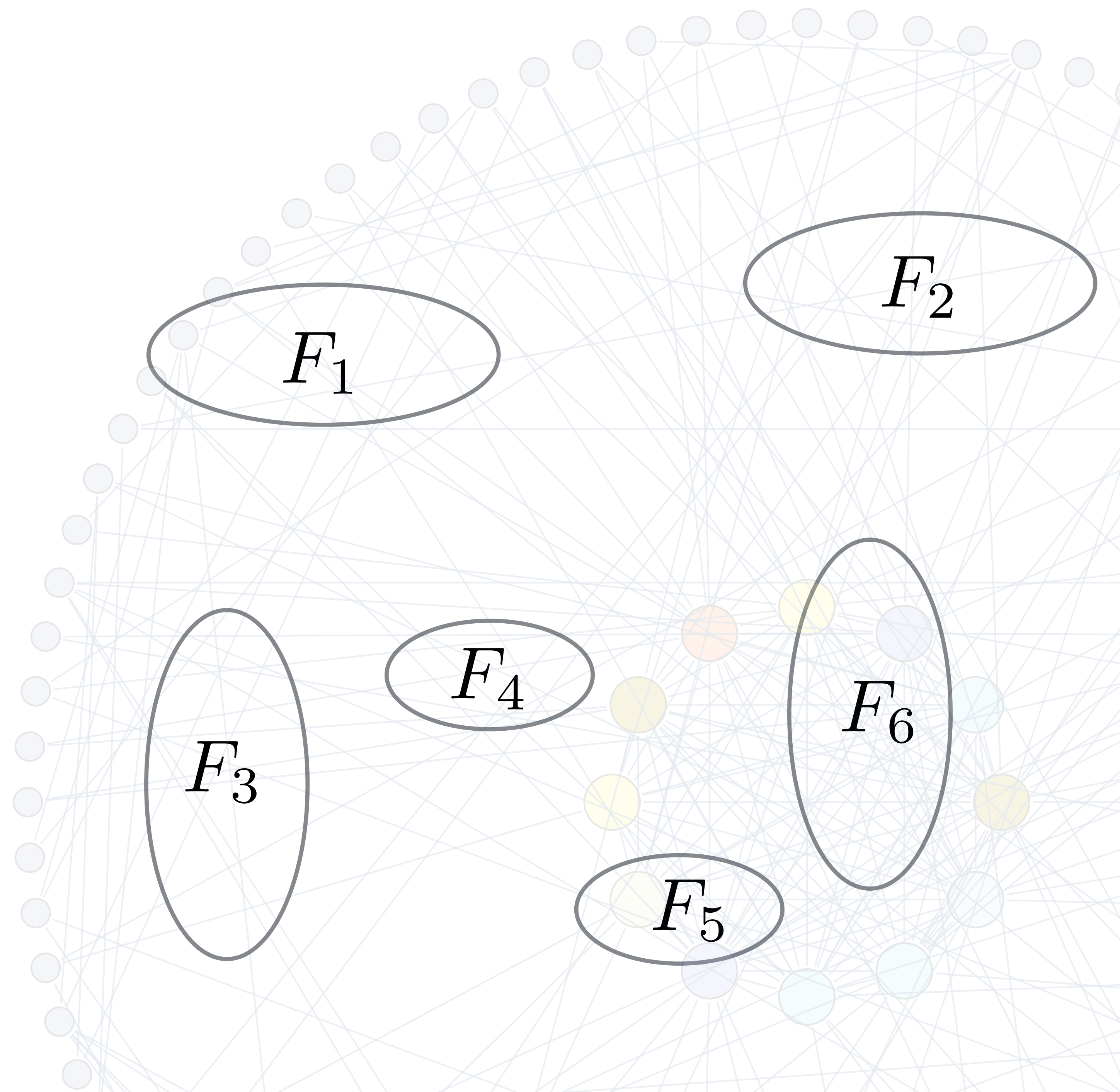
We prove:

$\mathcal{CP}$-MST algorithm: $\quad O(\log^2 n)$



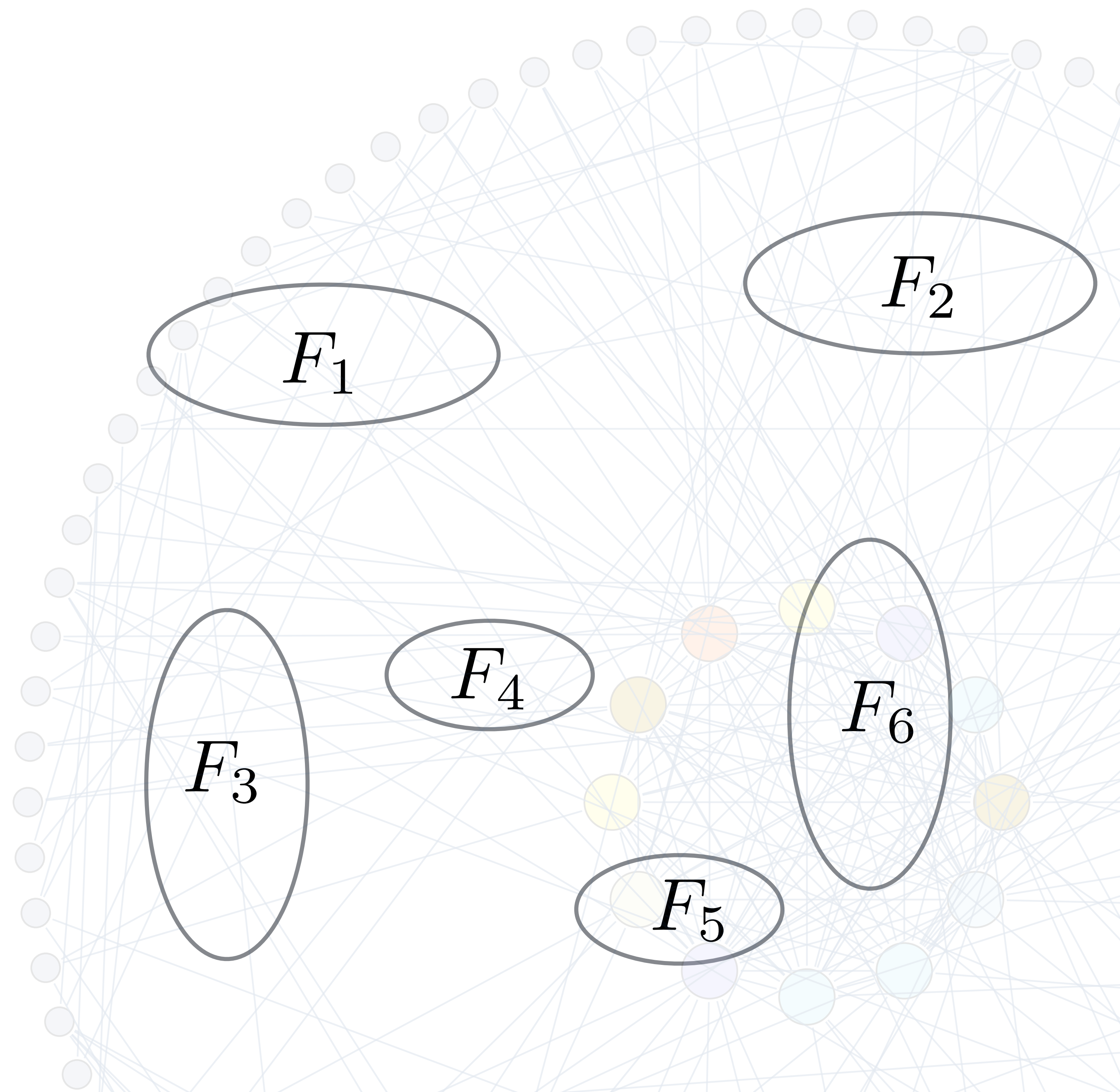
Distributed MST for Core-Periphery Networks

- Based on Boruvka's algorithm - *log n* phases



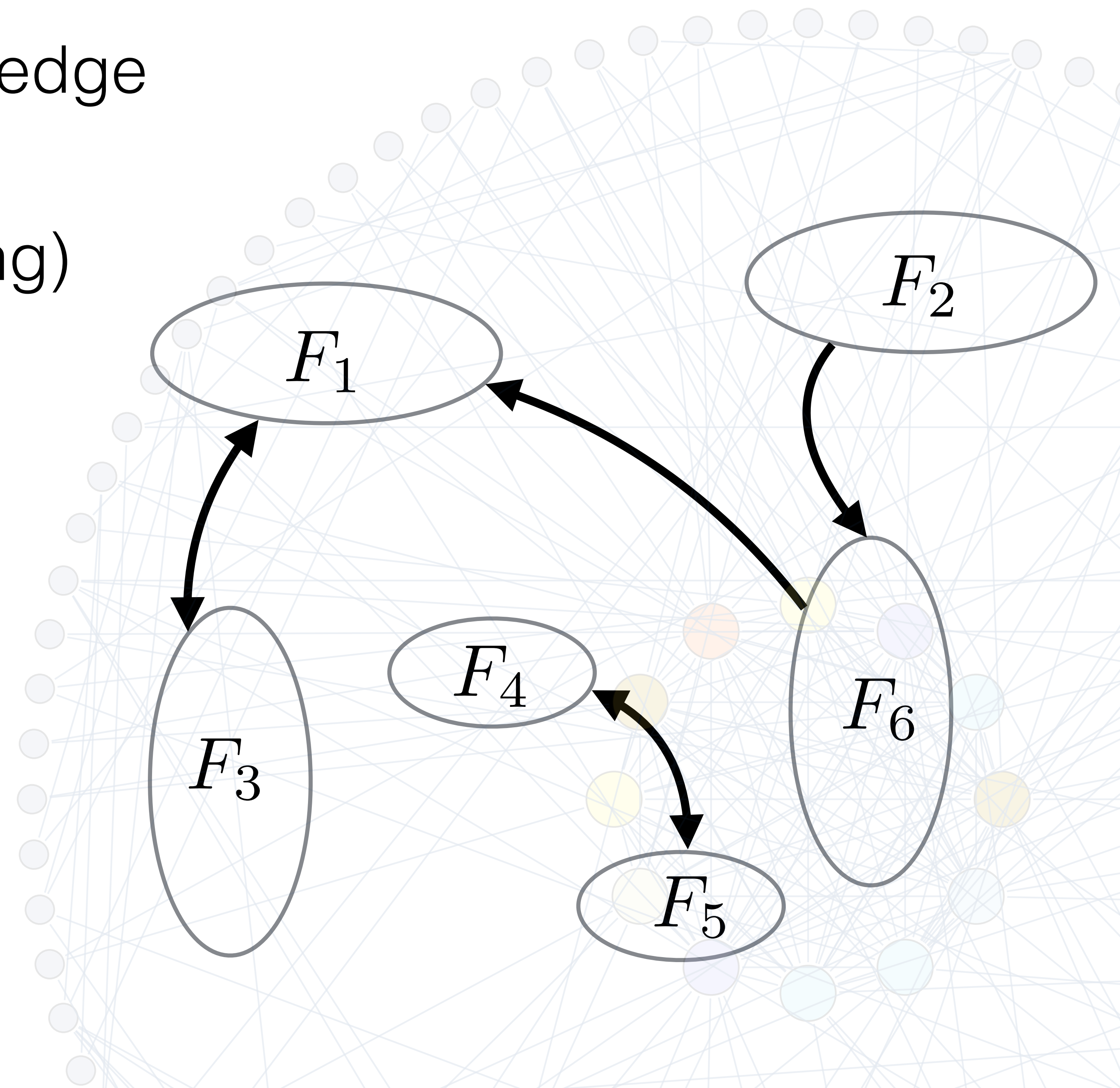
Distributed MST for Core-Periphery Networks

- Based on Boruvka's algorithm - *log n* phases
- On each phase need to exchange many messages:



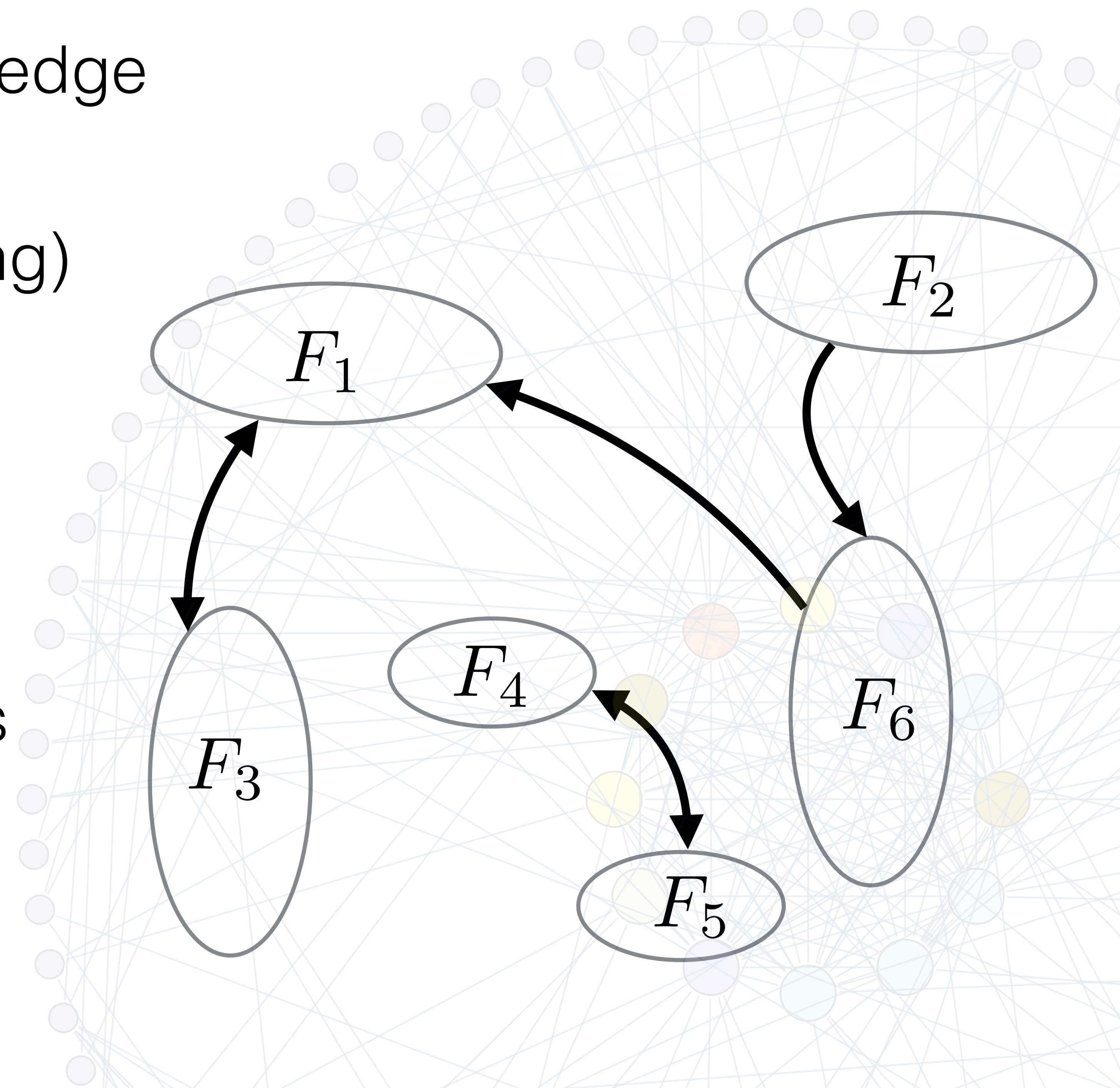
Distributed MST for Core-Periphery Networks

- Based on Boruvka's algorithm - *log n* phases
- On each phase need to exchange many messages:
  - select min-weight outgoing edge
  - merge fragments
    - (amortized pointer jumping)



Distributed MST for Core-Periphery Networks

- Based on Boruvka's algorithm - *log n* phases
- On each phase need to exchange many messages:
  - select min-weight outgoing edge
  - merge fragments
    - (amortized pointer jumping)

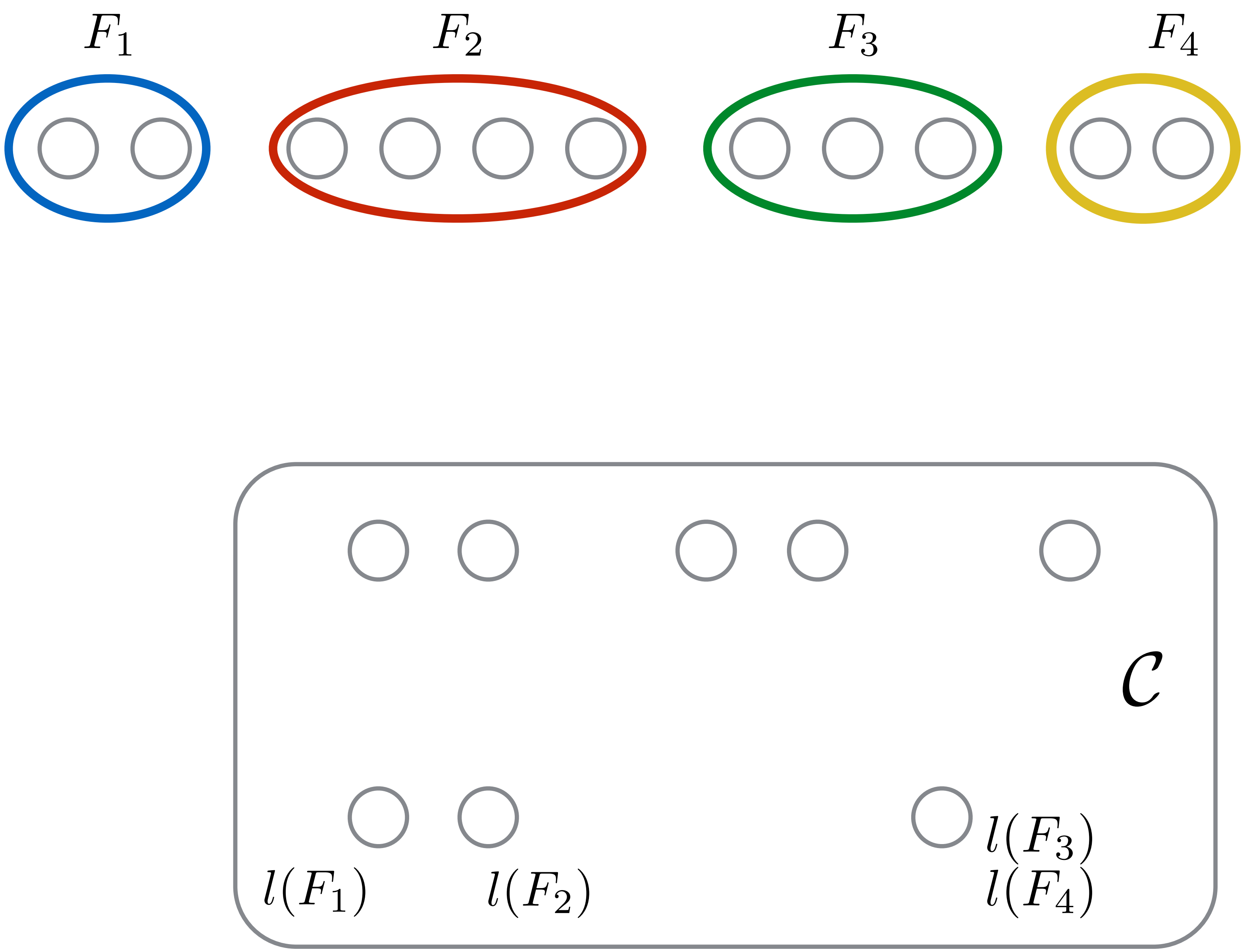


- We want:
  - each phase *O(log n)* rounds

Distributed MST for Core-Periphery Networks

Each node in ***V*** has two ***officials*** in ***C:***

1. Node's **Representative** - fixed
2. Fragment's **Leader** - may migrate in each phase

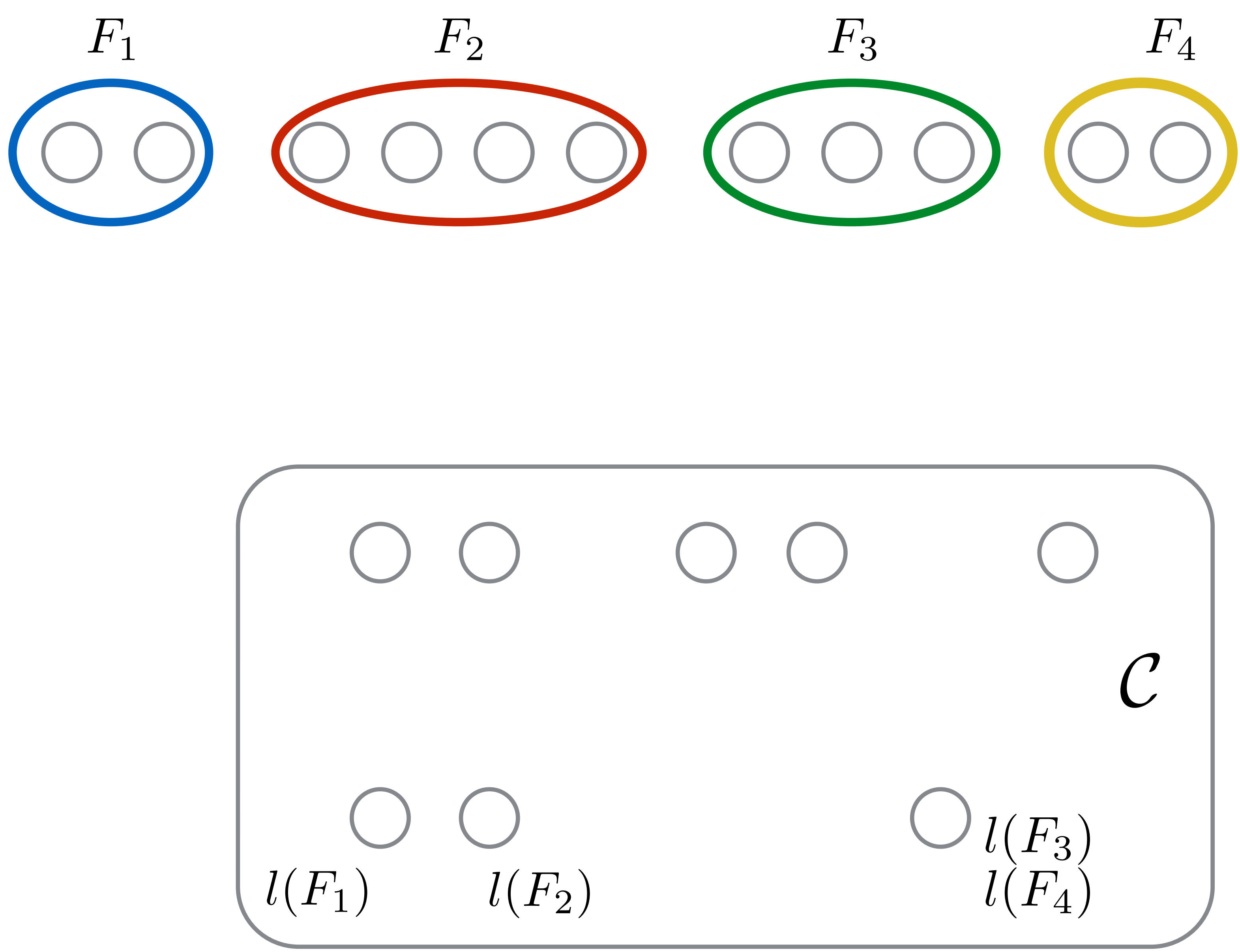


Distributed MST for Core-Periphery Networks

Each node in ***V*** has two ***officials*** in ***C:***

1. Node's **Representative** - fixed
2. Fragment's **Leader** - may migrate in each phase

Each node in ***V*** finds its **mwoe**



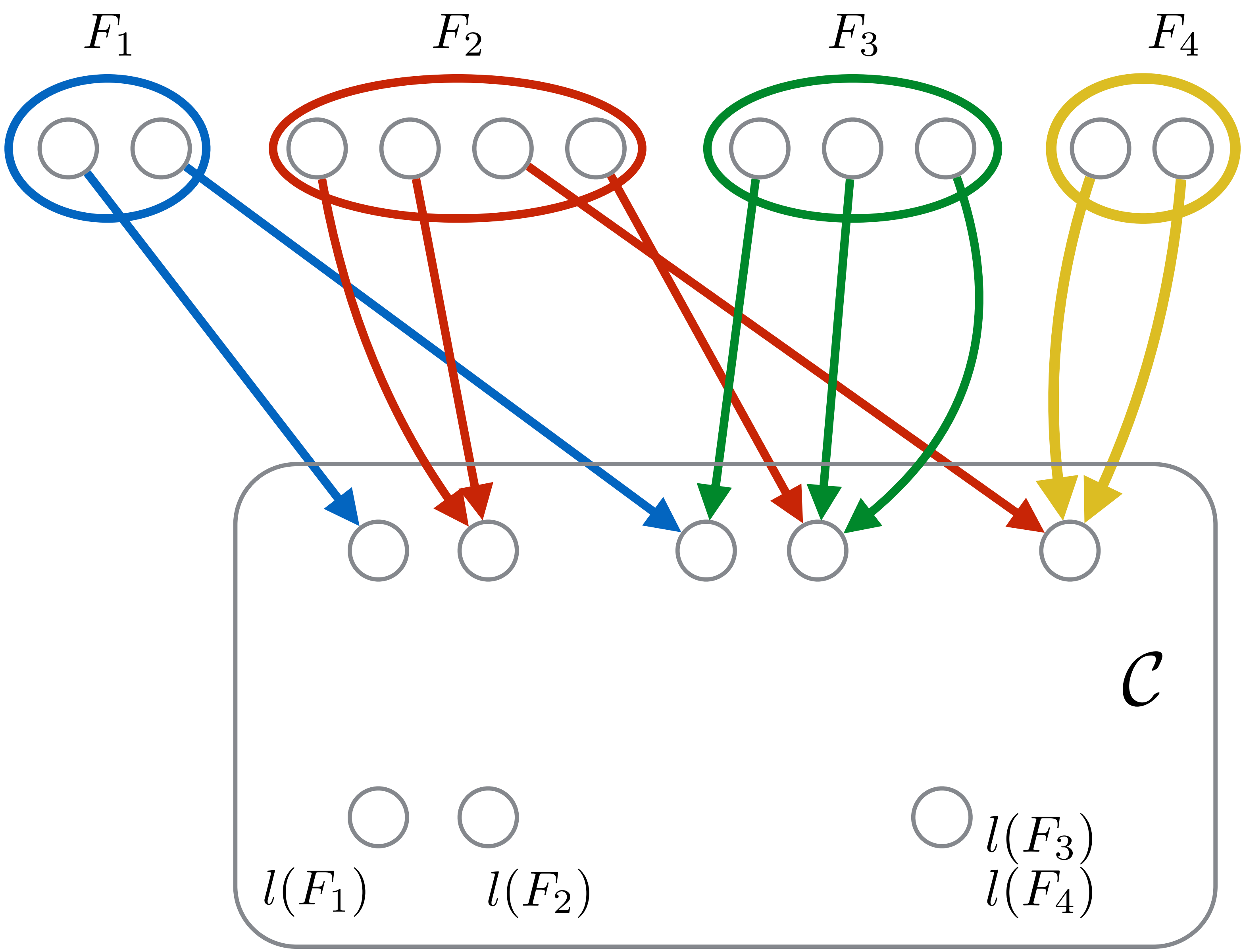
Distributed MST for Core-Periphery Networks

Each node in ***V*** has two ***officials*** in ***C*:**

1. Node's **Representative** - fixed
2. Fragment's **Leader** - may migrate in each phase

Each node in ***V*** finds its **mwoe**

All **mwoe** are delivered to ***C***



Distributed MST for Core-Periphery Networks

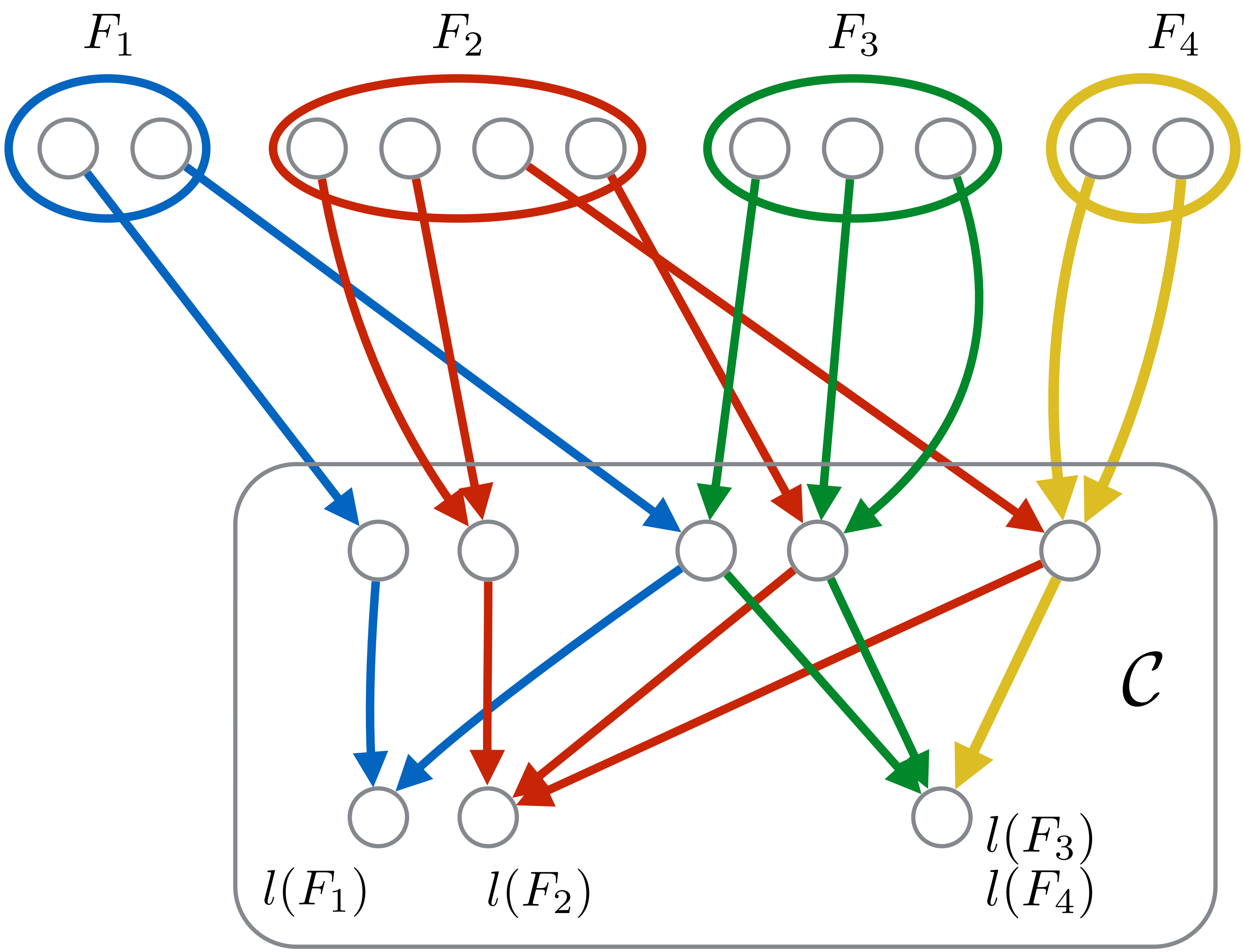
Each node in ***V*** has two ***officials*** in ***C:***

1. Node's **Representative** - fixed
2. Fragment's **Leader** - may migrate in each phase

Each node in ***V*** finds its **mwoe**

All **mwoe** are delivered to ***C***

Representatives send to **leaders**



Distributed MST for Core-Periphery Networks

Each node in ***V*** has two ***officials*** in ***C*:**

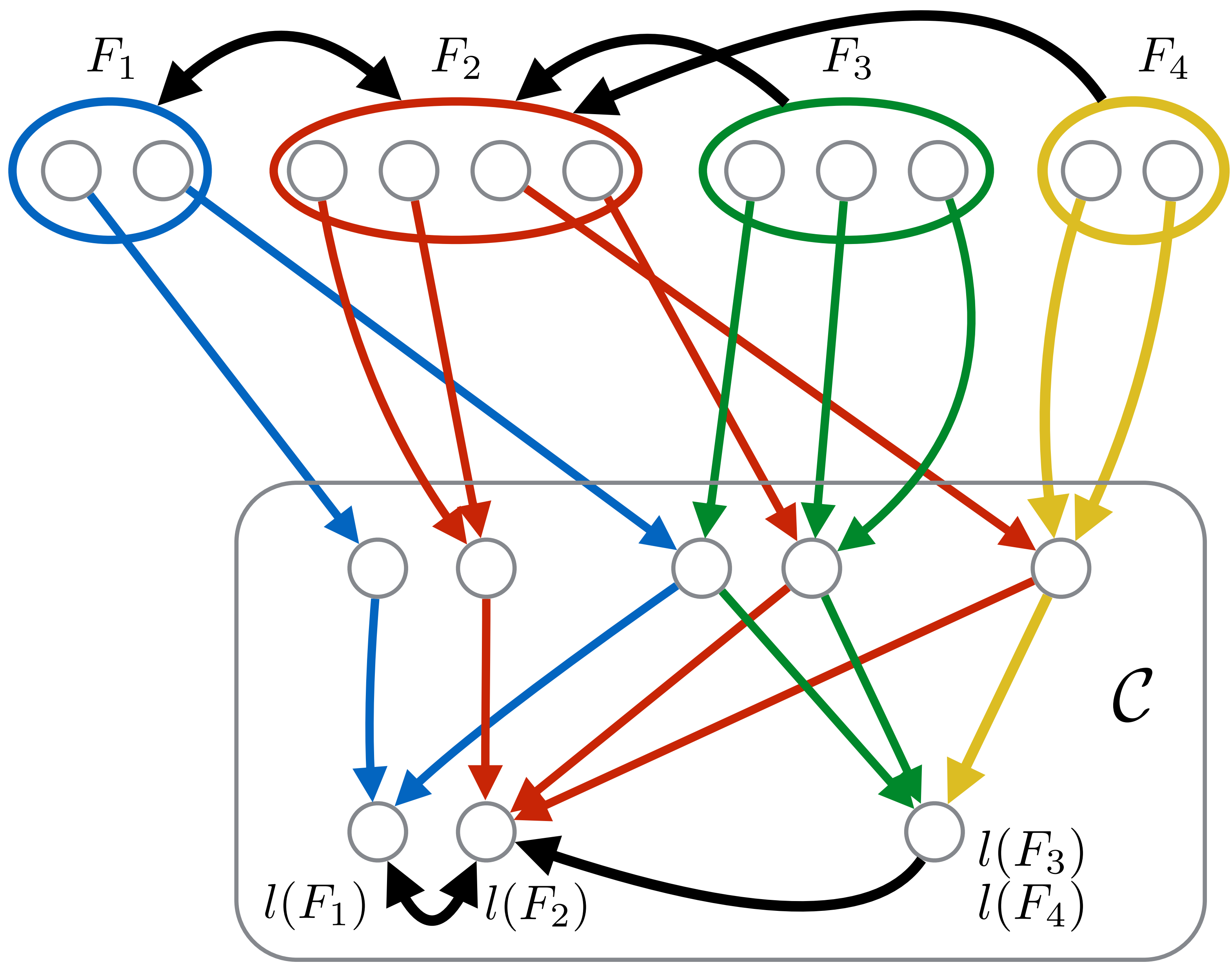
1. Node's **Representative** - fixed
2. Fragment's **Leader** - may migrate in each phase

Each node in ***V*** finds its **mwoe**

All **mwoe** are delivered to ***C***

Representatives send to **leaders**

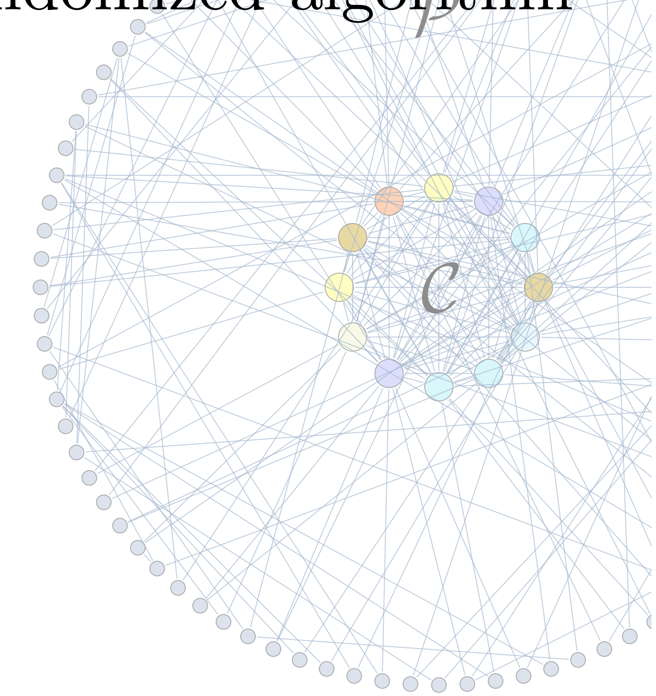
Leaders decide on merging



Algorithms for Core-Periphery Networks

Task	Running time on $\mathcal{CP}$ networks	Lower bounds	
		All Axioms	Any 2 Axioms
MST *	$O(\log^2 n)$	$\Omega(1)$	$\tilde{\Omega}(\sqrt[4]{n})$
Matrix transposition	$O(k)$	$\Omega(k)$	$\Omega(n)$
Vector by matrix multiplication	$O(k)$	$\Omega(k/\log n)$	$\Omega(n/\log n)$
Matrix multiplication	$O(k^2)$	$\Omega(k^2)$	$\Omega(n/\log n)$
Find my rank	$O(1)$	$\Omega(1)$	$\Omega(n)$
Find median	$O(1)$	$\Omega(1)$	$\Omega(\log n)$
Find mode	$O(1)$	$\Omega(1)$	$\Omega(n/\log n)$
Find number of distinct values	$O(1)$	$\Omega(1)$	$\Omega(n/\log n)$
Top r ranked by areas	$O(r)$	$\Omega(r)$	$\Omega(r\sqrt{n})$

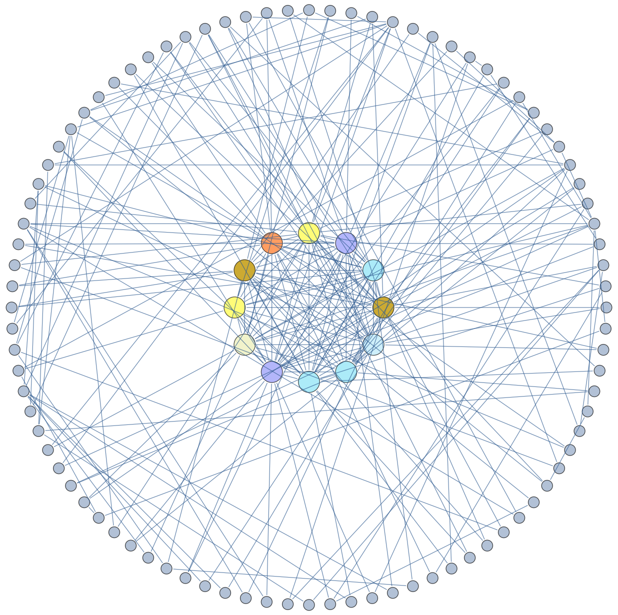
k - maximum number of nonzero entries in a row or column. * - randomized algorithm



Summary

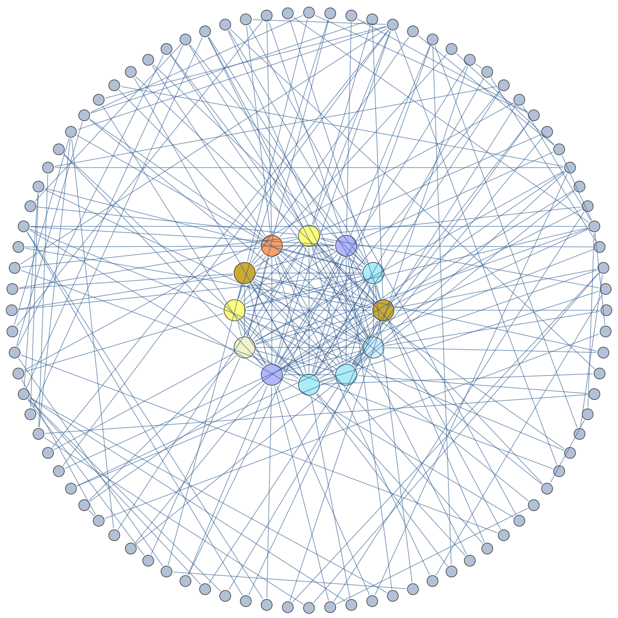
Summary

- Axiomatic approach
- Core-periphery is fast and cost efficient network architecture
- Simple algorithms design



Summary

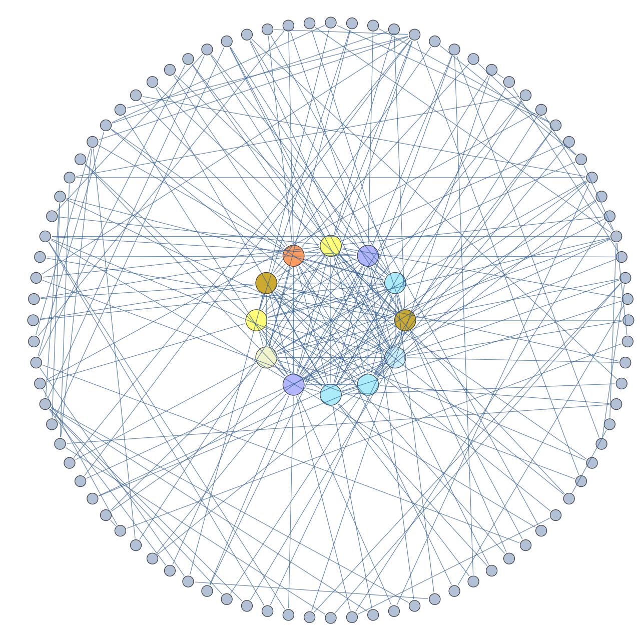
- Axiomatic approach
- Core-periphery is fast and cost efficient network architecture
- Simple algorithms design



- Future work:
  - What else can be computed efficiently?
  - What are the basic building blocks in distributed computing?

Summary

- Axiomatic approach
- Core-periphery is fast and cost efficient network architecture
- Simple algorithms design



- Future work:
  - What else can be computed efficiently?
  - What are the basic building blocks in distributed computing?

Thank You!