# Enumeration of Molecular Clusters

Karthik Mahadevan

Yuji Urano

# Introduction

- Terminologies
  - Floppy cluster
  - Rigid cluster
- Why enumerate?
  - Scientific importance
- How to enumerate?
  - Start by breaking a bond

# Interesting question

- How many rigid clusters are there for a particular n (n: number of particles)?
- The number of rigid clusters increases dramatically

up to n=5, there is one

n=6, there are 2          n=7, there are 5

n=8, there are 13        n=9, there are 52

n=10, there are 263   n=11, there are 1657, etc

# Representation

- Rigid cluster, N = 6

$$
N \quad N \begin{bmatrix} 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 \end{bmatrix}
$$

# Algorithms

- Algorithm 1
  - Floppy to rigid
- Algorithm 2
  - One rigid cluster to all possible enumerations

# Algorithm 1(Floppy to Rigid)

- Start with a particle and a matrix M of bond constraints
- Compute the Null space of M
- Take a step in the direction of the vector in the tangent space
- Get back to manifold by solving the nonlinear system of equations by Newton's method
- Repeat until new bond is formed

# Algorithm 2(Permute and find all packings)

- Break a bond, then you get a floppy particle(s)
- Use Algorithm 1 to find a rigid cluster
- The new rigid cluster might be different
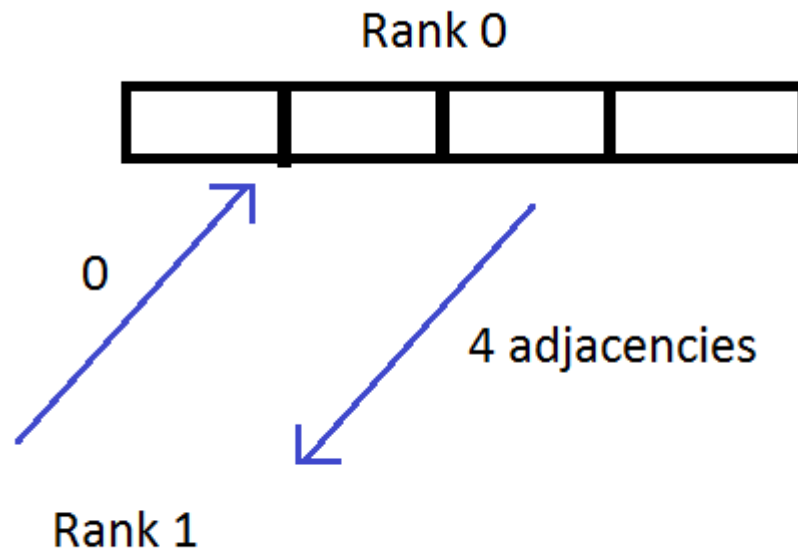- You have to do this for all bonds from all rigid clusters

# Parallelization

- You have to break all the bonds from all rigid clusters

- You can do it independently

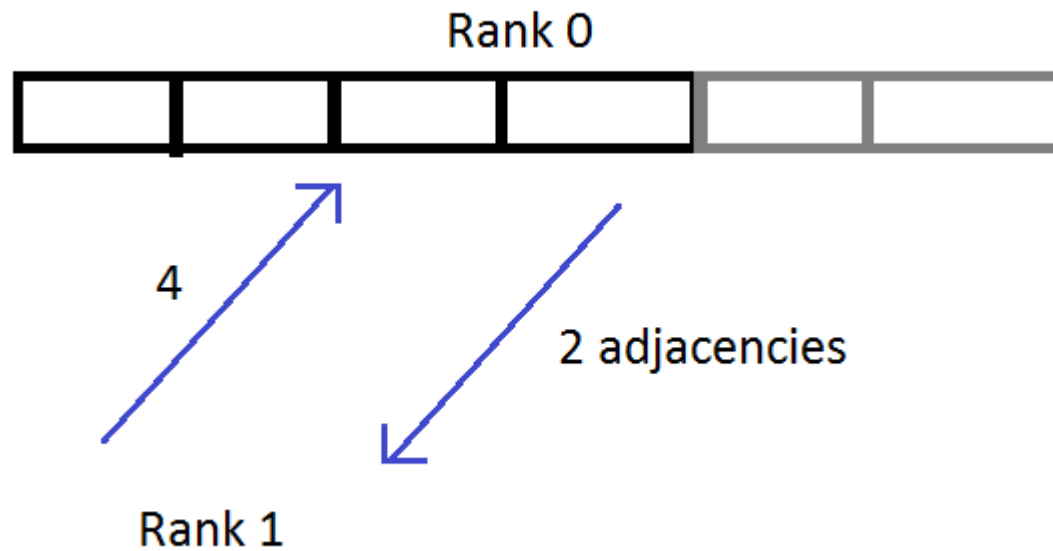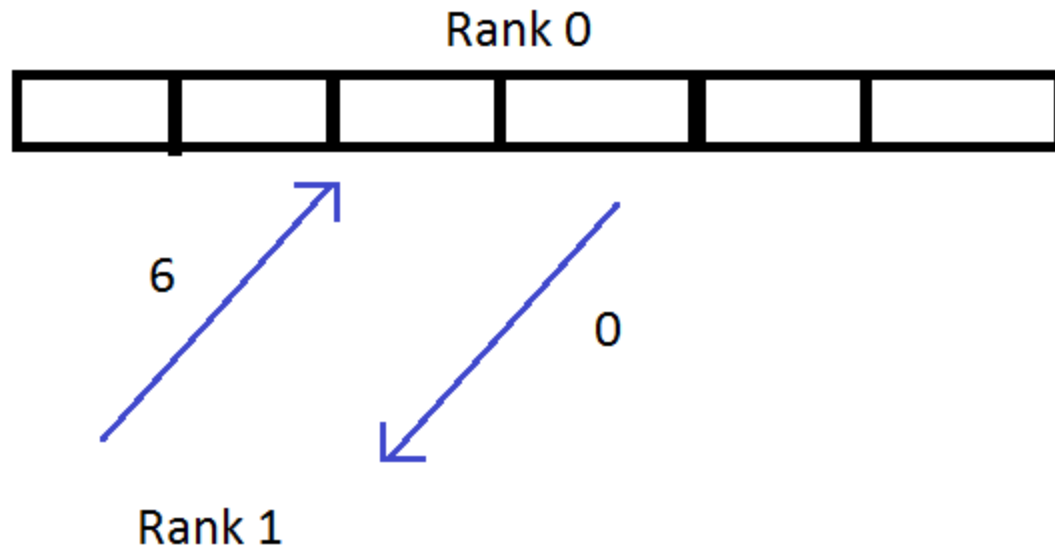- This operation is parallelizable!

# Process Communication

# Isomorphism Check

Rank 0

0

4 adjacencies

Rank 1
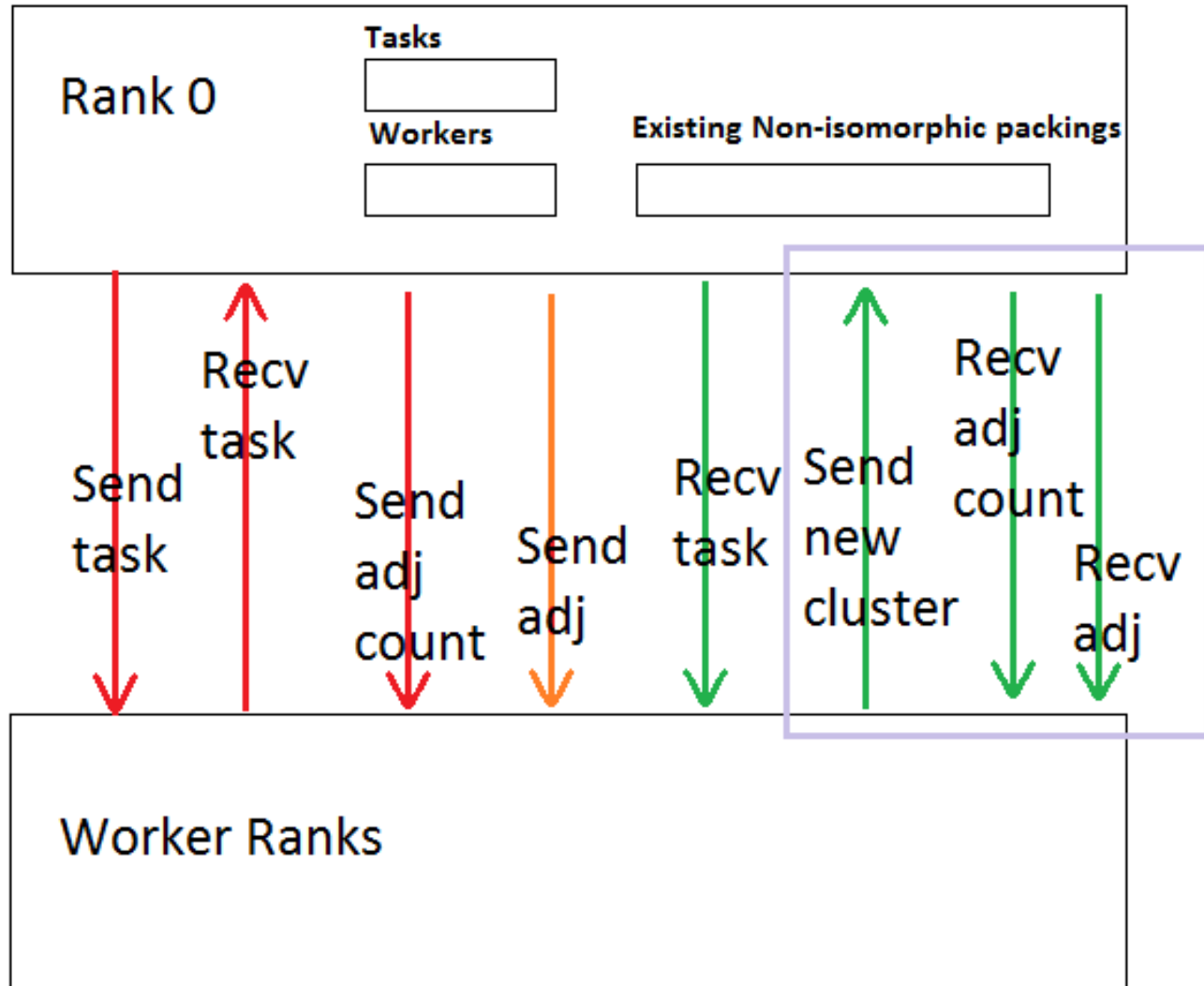
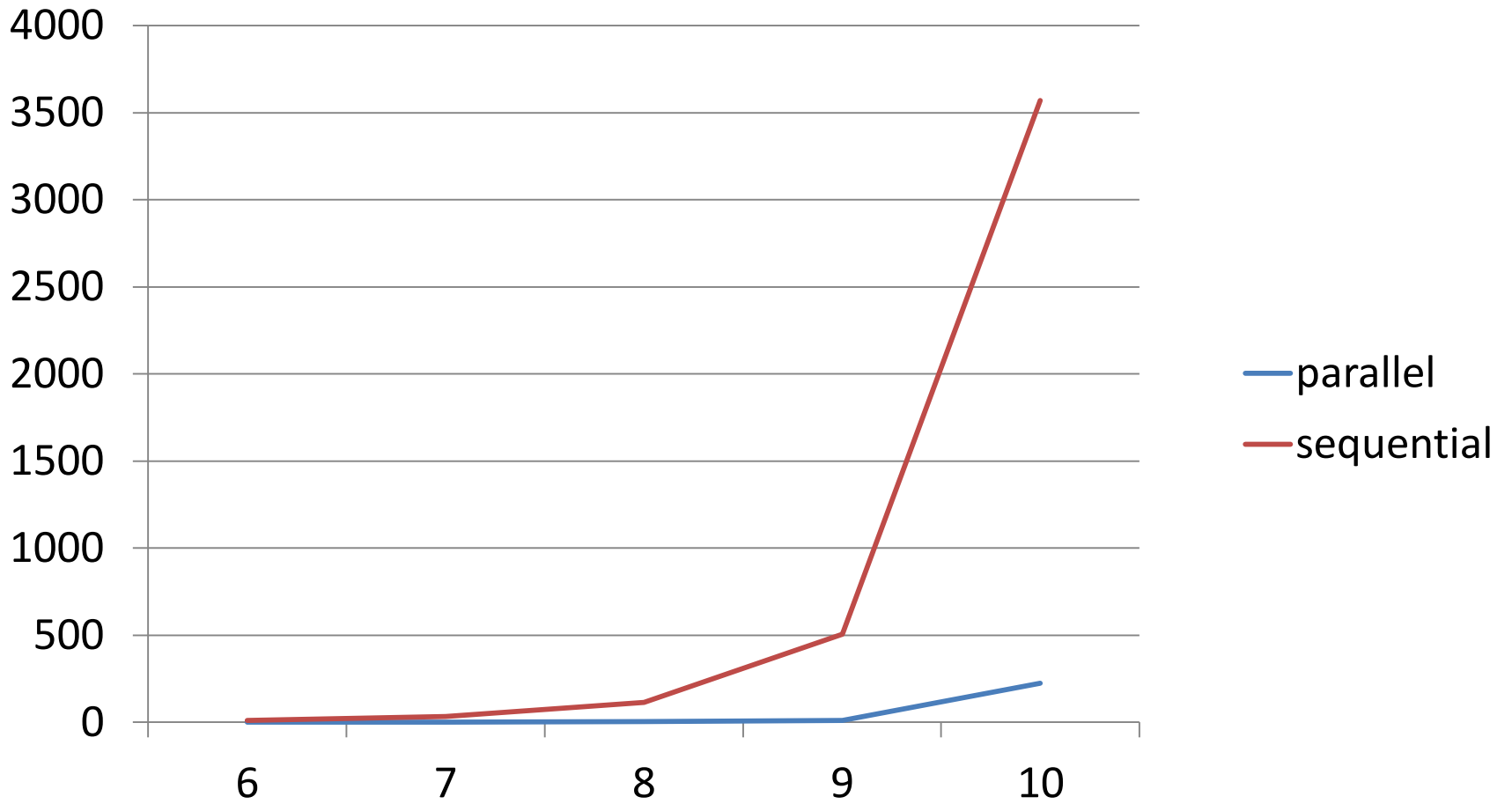# Isomorphism Check

# Isomorphism Check

# Optimization

- Adjacency check is done in parallel by OpenMP threads
- All send messages are non-blocking

# Timing (seconds)

| n | parallel (no. ranks) | sequential |
|---|---|---|
| 6 | 1.14 (24) | 11 |
| 7 | 1.23 (40) | 33 |
| 8 | 3.12 (80) | 115 |
| 9 | 10.02 (64) | 505 |
| 10 | 223.6 (104) | 3570 |

# Timing

parallel
sequential

# Thank You