

# Optimal graph for decentralized search problem

I. Utkina

Faculty of Business Informatics and Applied Mathematics

Workshop on Clustering and Search techniques for large scale  
networks, 2015

# Outline

## 1 Decentralized search

- Variables
- Example

## 2 Algorithm

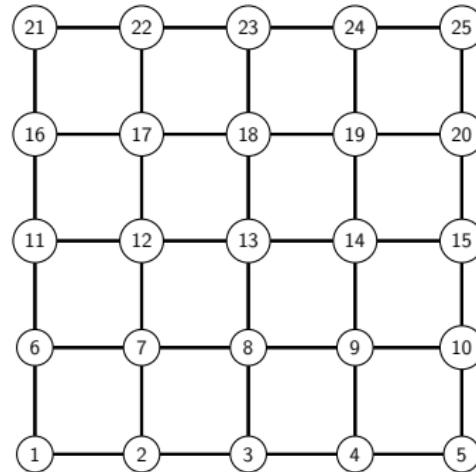
- Goal
- Examples
- Definitions

## 3 Result

# Decentralized search

## Variables

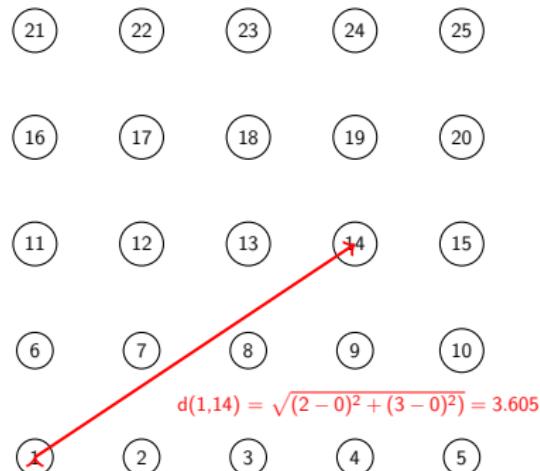
- $G(V,E)$  - simple graph



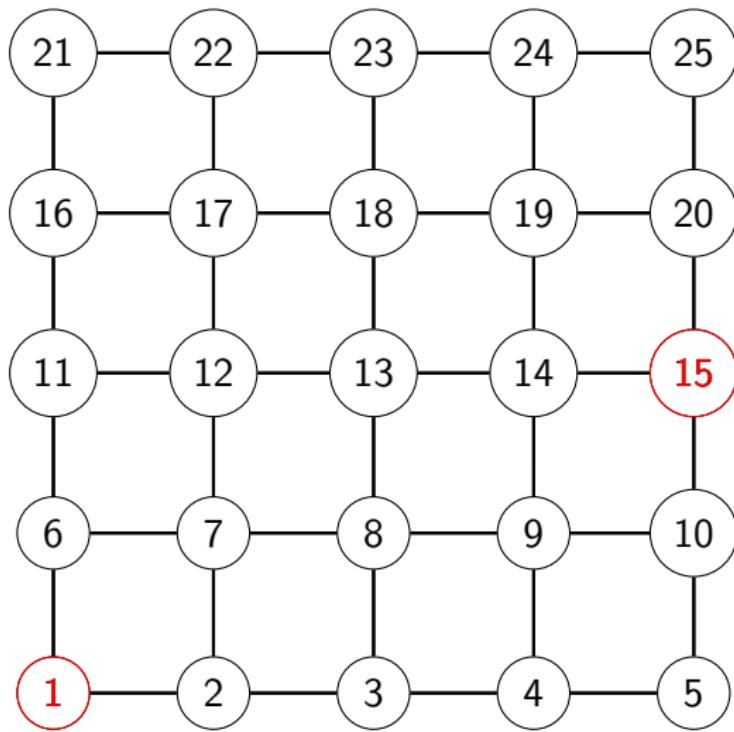
# Decentralized search

## Variables

- $G(V, E)$  - simple graph
- $d(a, b)$  - distance function
- $d(a, b) = \sqrt{(x_a - y_a)^2 + (x_b - y_b)^2}$ ,  
where  $(x_a, y_a)$  and  $(x_b, y_b)$  - coordinates of nodes  $a$  and  $b$ .  
Node 1 has coordinates  $(0, 0)$ .  
Node 25 has coordinates  $(4, 4)$ .



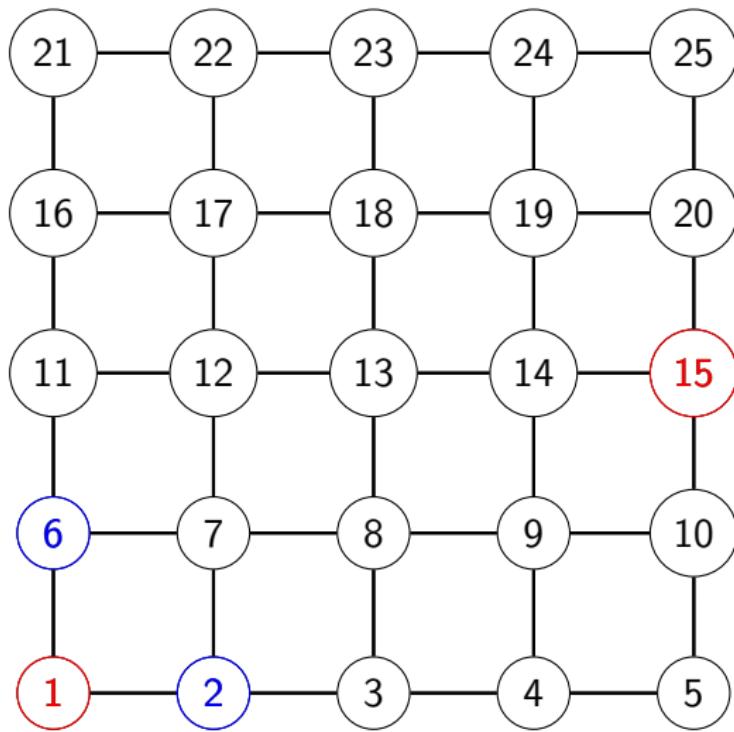
## Example of decentralized search



$$d(1, 15) = \sqrt{16 + 4} = \sqrt{20}$$

*operations<sub>+</sub>* = 1 = 1

## Example of decentralized search

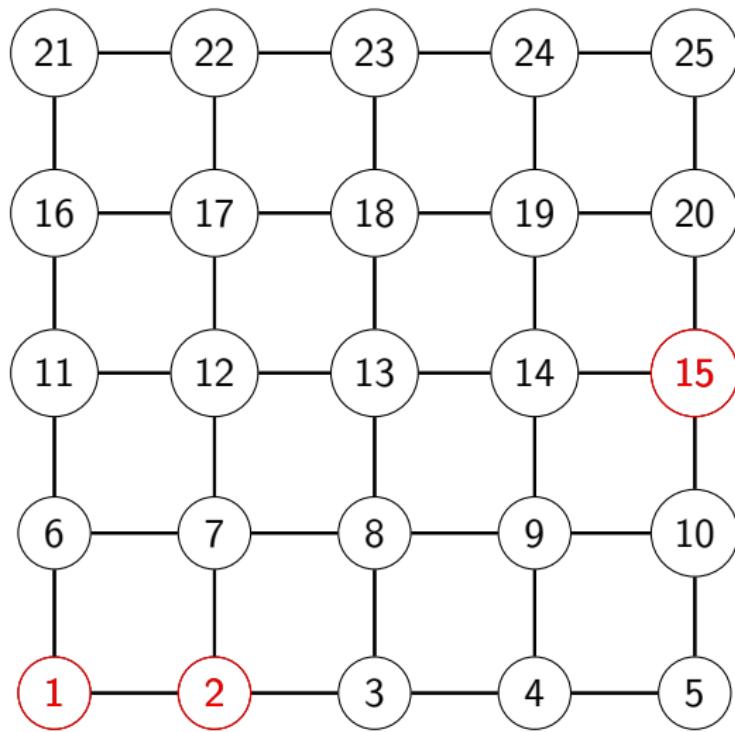


$$d(6, 15) = \sqrt{17}$$

$$d(2, 15) = \sqrt{13}$$

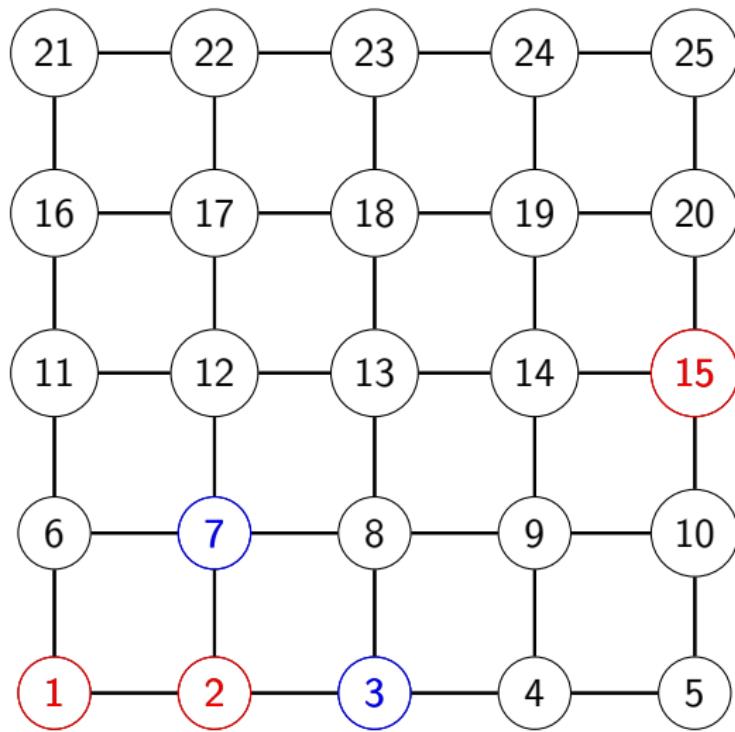
*operations+ = 2 = 3*

## Example of decentralized search



$$d(2, 15) = \sqrt{13}$$

## Example of decentralized search

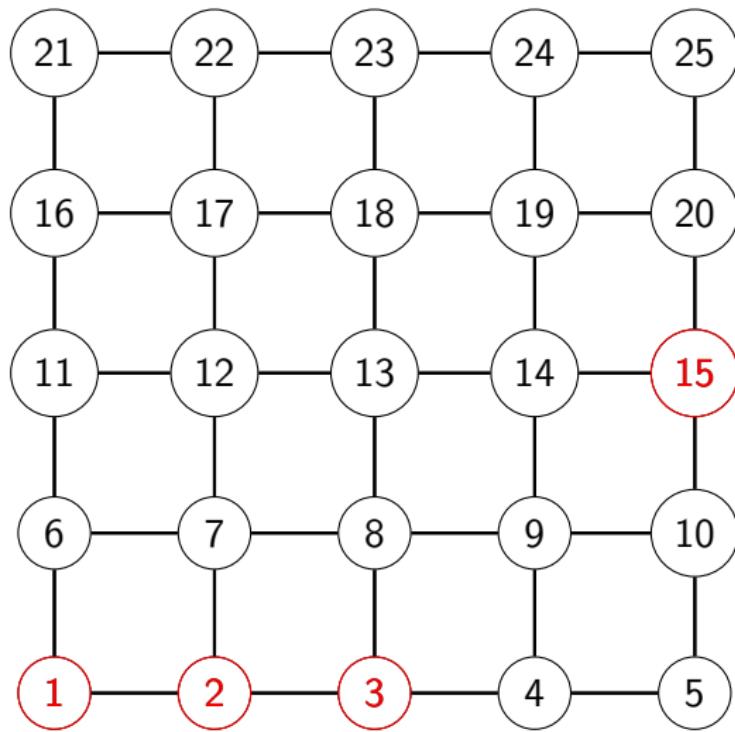


$$d(7, 15) = \sqrt{10}$$

$$d(3, 15) = \sqrt{8}$$

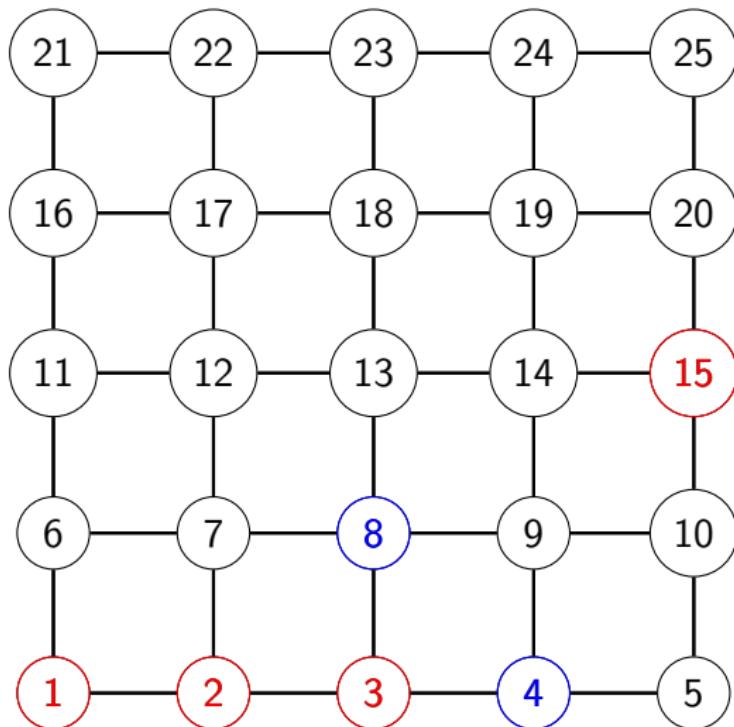
$$\text{operations}+ = 2 = 5$$

## Example of decentralized search



$$d(3, 15) = \sqrt{8}$$

## Example of decentralized search

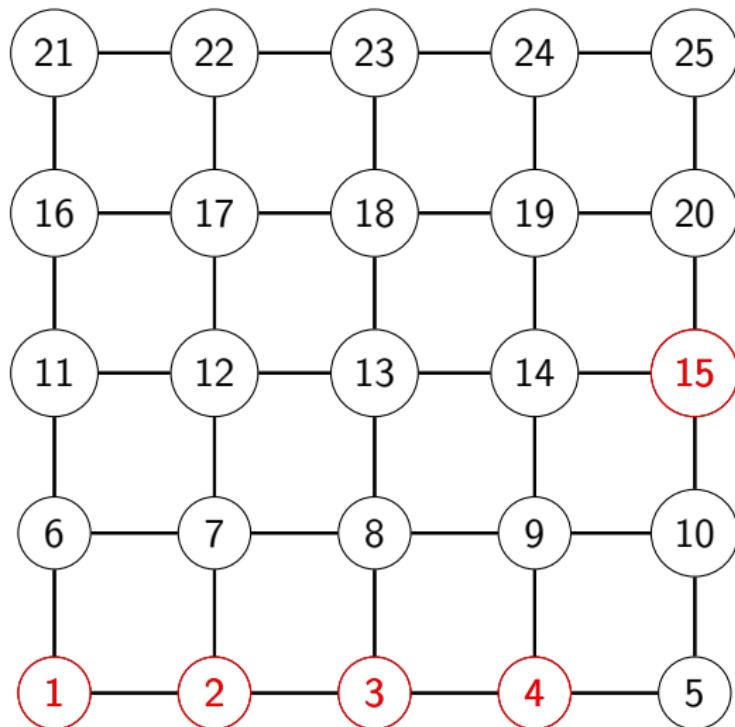


$$d(8, 15) = \sqrt{5}$$

$$d(4, 15) = \sqrt{5}$$

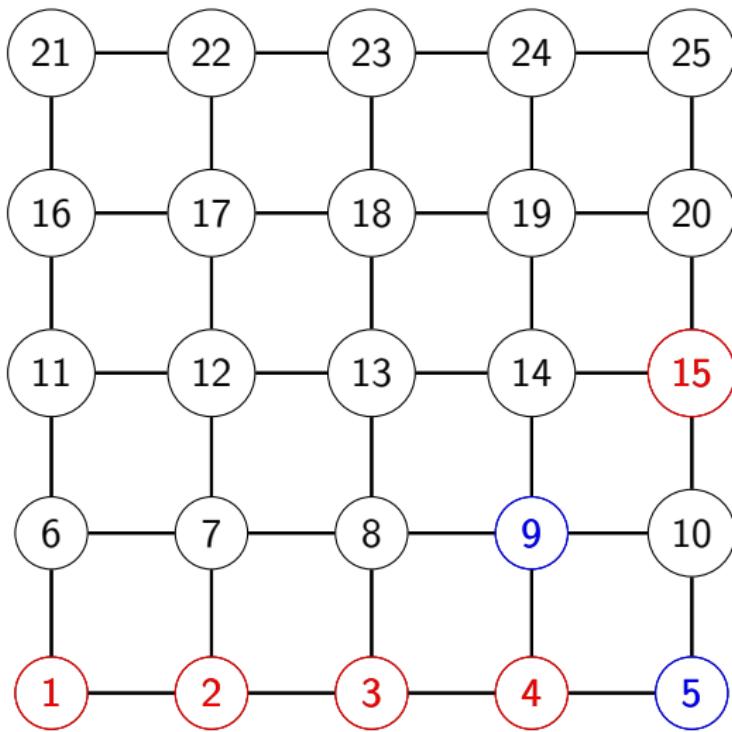
*operations+ = 2 = 7*

## Example of decentralized search



$$d(4, 15) = \sqrt{5}$$

## Example of decentralized search

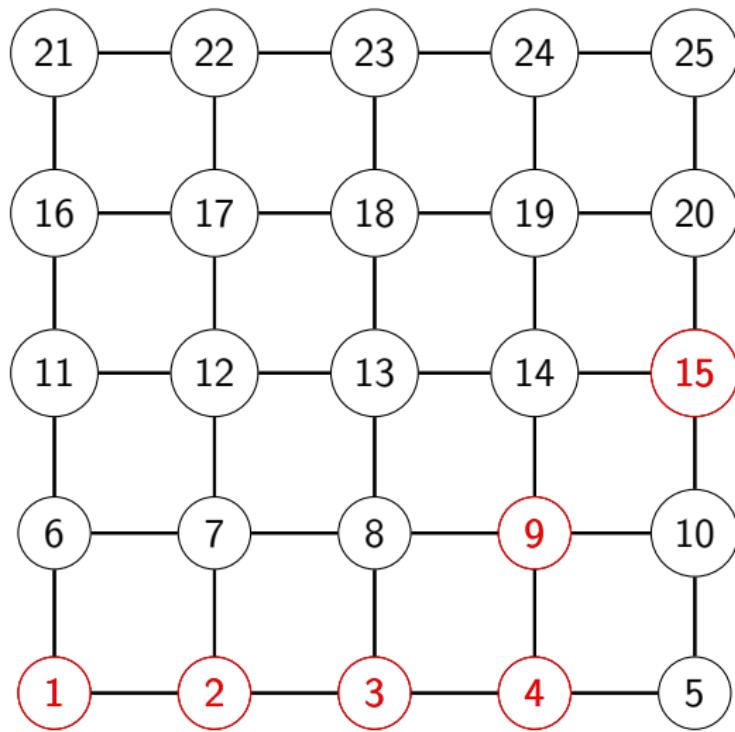


$$d(5, 15) = \sqrt{2}$$

$$d(9, 15) = \sqrt{4}$$

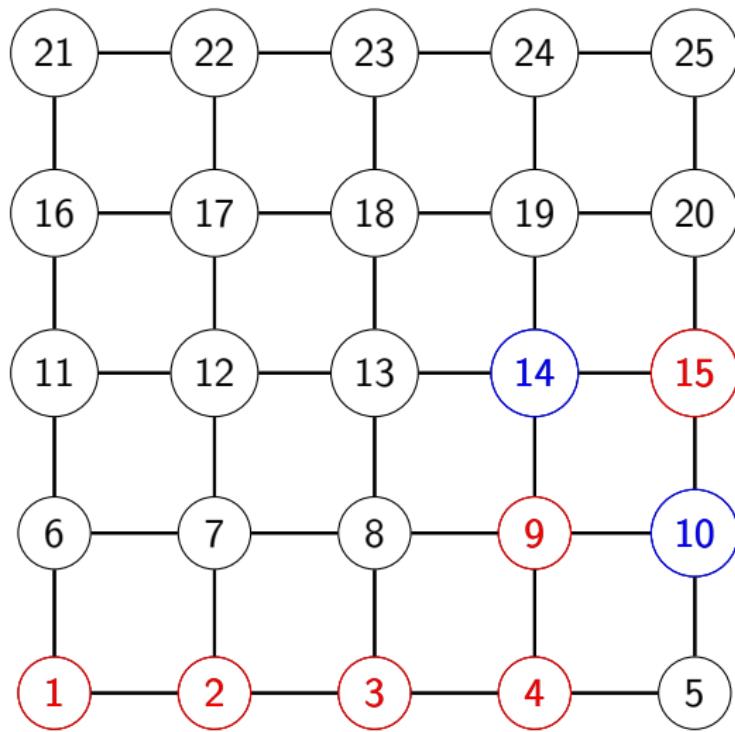
$$\text{operations}+ = 2 = 9$$

## Example of decentralized search



$$d(9, 15) = \sqrt{4}$$

## Example of decentralized search

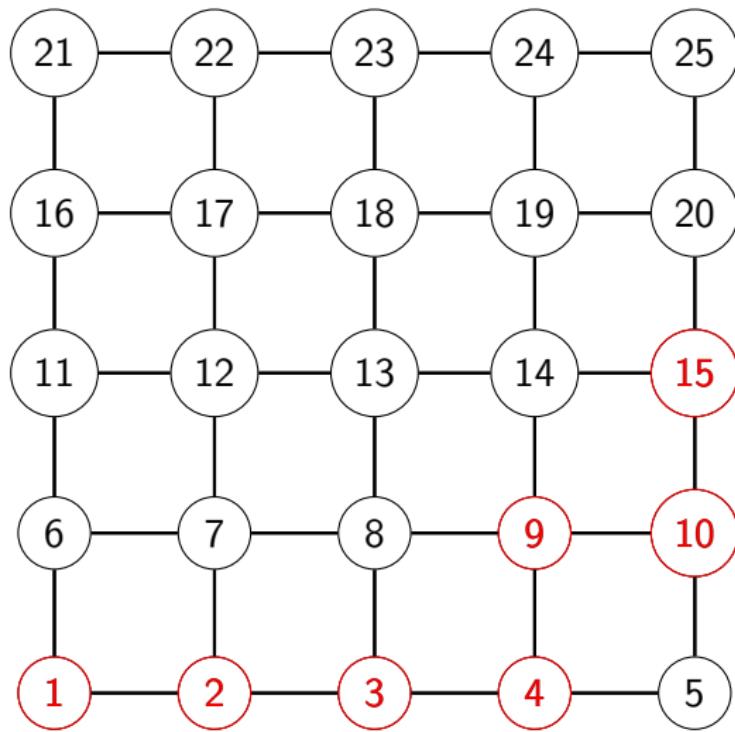


$$d(14, 15) = \sqrt{1}$$

$$d(10, 15) = \sqrt{1}$$

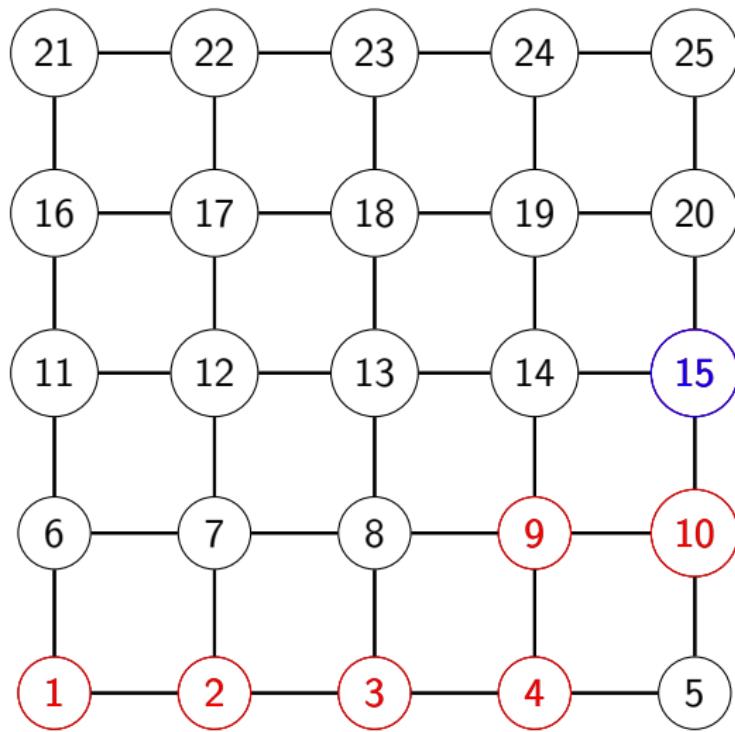
*operations+ = 2 = 11*

## Example of decentralized search



$$d(10, 15) = \sqrt{1}$$

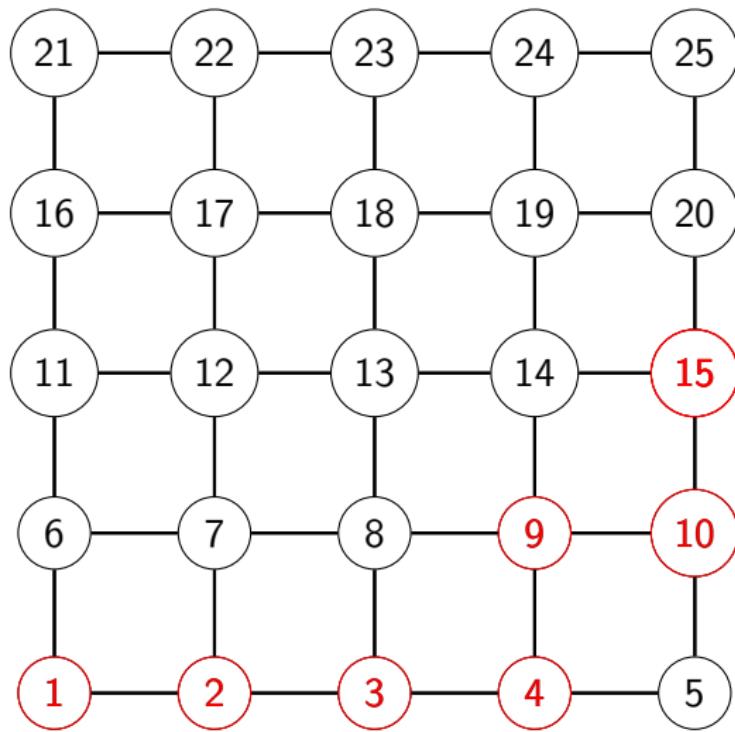
## Example of decentralized search



$$d(15, 15) = 0$$

$$\text{operations}+ = 1 = 12$$

## Example of decentralized search



$$d(15, 15) = 0$$

$$\text{operations}+ = 1 = 12$$

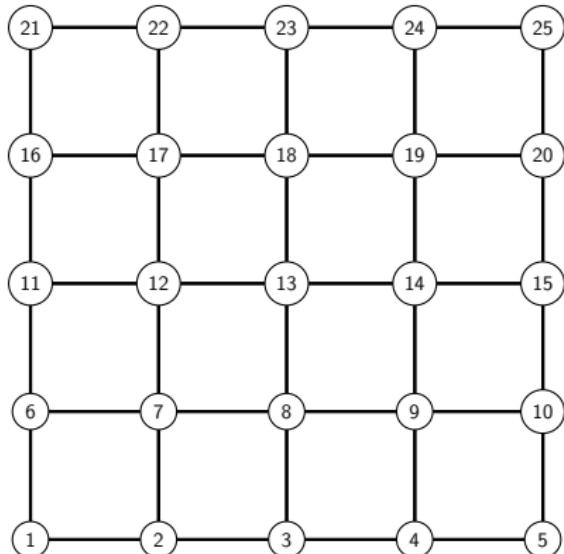
## Goal

Goal is to find optimal structure of graph  $G(V,E)$ , which gives us minimum of function  $f$ .

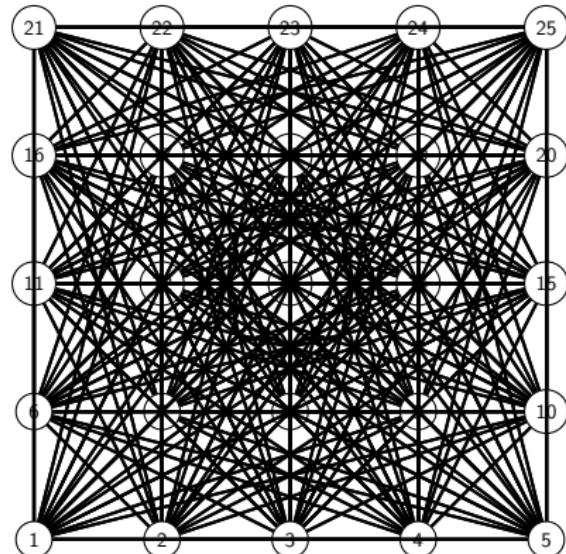
$$f = \frac{\sum_{i=1}^n \sum_{j=1}^n \text{operations}(i,j)}{n^2},$$

where  $|V| = n$ ,  $\text{operations}(i,j)$  returns the number of operations in search from node  $i$  to node  $j$ .

# Example



$$f = 9.26$$



$$f = 24.04$$

# Definitions

- Number of fails - how many times we can not improve our solution before we stop algorithm.
- Number of tabu - how many edges we can strict
- Flag - boolean value, either we choose first good improvements or choose the best.

# Algorithm

```
function LOCALMIN
    while (nonSopCondition) do
        if (flag == true) then
            choose first good improvement among all we can do
        else
            choose the best improvement among all we can do
        if (min == operations) then local min is found
            if (min < bestKnown) then
                fails = 0
            else
                fails ++
            if (fails == numberOffails) then
                saveResult
                end of algorithm
            else
                clear strict edges
                Tabu(numberOfTabu)
        else
            Do chosen improvement
```

# Algorithm

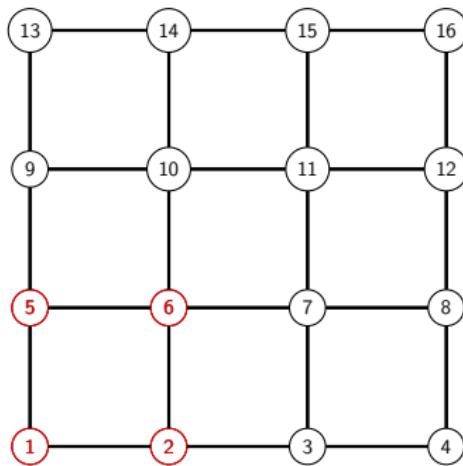
```
function TABU(numberOfTabu)
  while (numberOfTabu! = 0) do
    Do the least bad improvement
    Strict this improvement to be undone
    numberOfTabu --
```

# Algorithm

```
function LOCALMIN
    while (nonSopCondition) do
        if (flag == true) then
            choose first good improvement among all we can do
        else
            choose the best improvement among all we can do
        if (min == operations) then local min is found
            if (min < bestKnown) then
                fails = 0
            else
                fails ++
            if (fails == numberOffails) then
                saveResult
                end of algorithm
            else
                clear strict edges
                Tabu(numberOfTabu)
        else
            Do chosen improvement
```

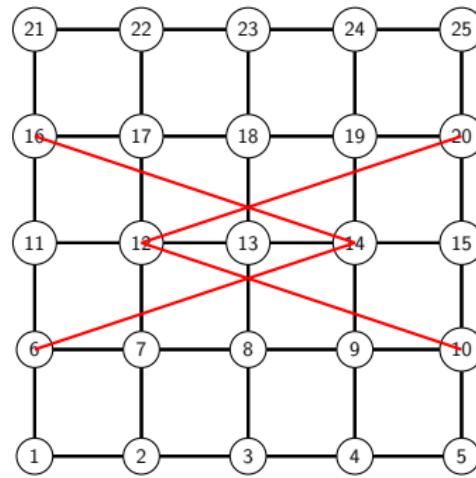
# Symmetrical Edges

In our algorithm we can add, delete or strict any symmetrical edges.  
That's why we consider edges which start only from red area and finish in any node.

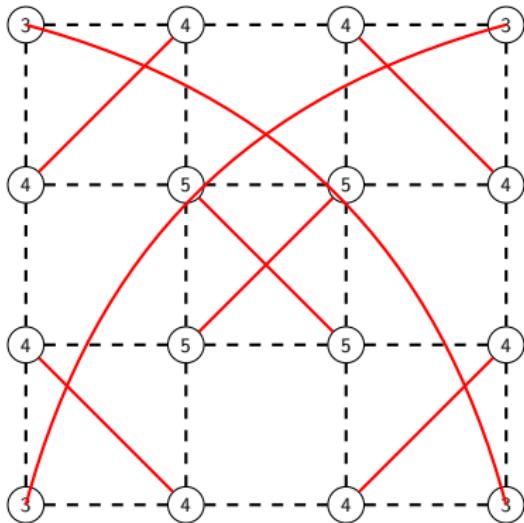


# Symmetrical Edges

When we add, delete or strict edge  $(a, b)$ , we also add, delete or strict all symmetrical, such as  $(a_1, b_1) = ((a/n+1)*n - a\%n, (b/n+1)*n - b\%n)$ ,  $(a_2, b_2) = (n * m - a, n * m - b)$  and  $(a_3, b_3) = ((a_2/n + 1)*n - a_2\%n, (b_2/n + 1)*n - b_2\%n)$ .

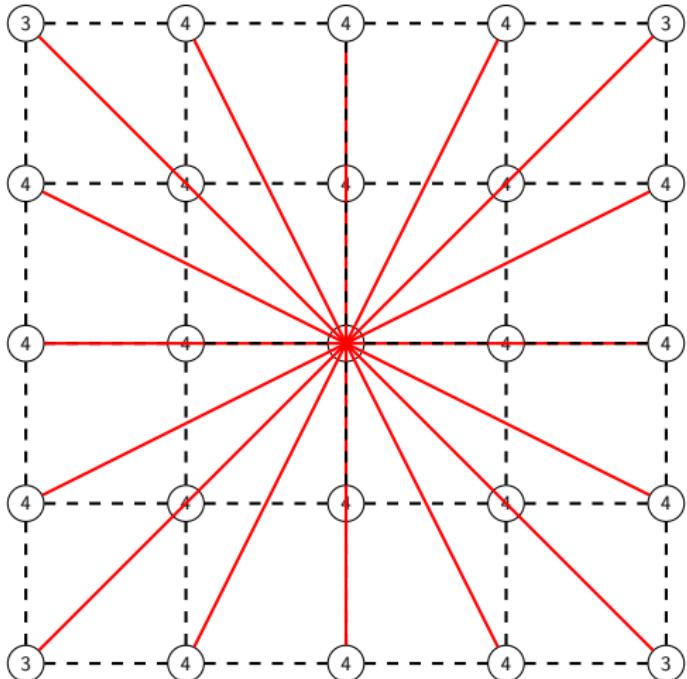


# Results



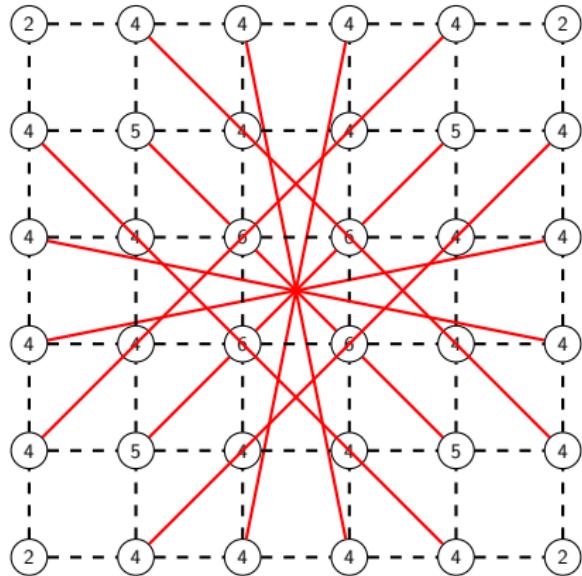
$$f = 7.094(7.266)$$

# Results



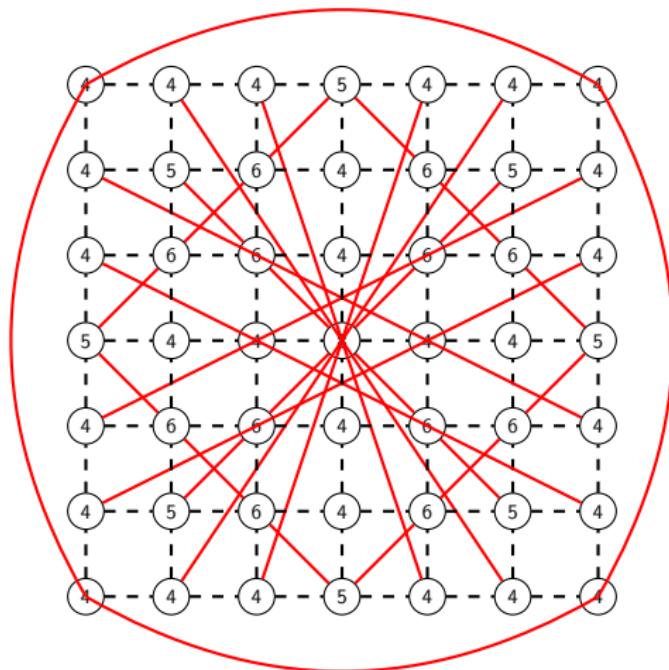
$$f = 8.974(9.259)$$

# Results



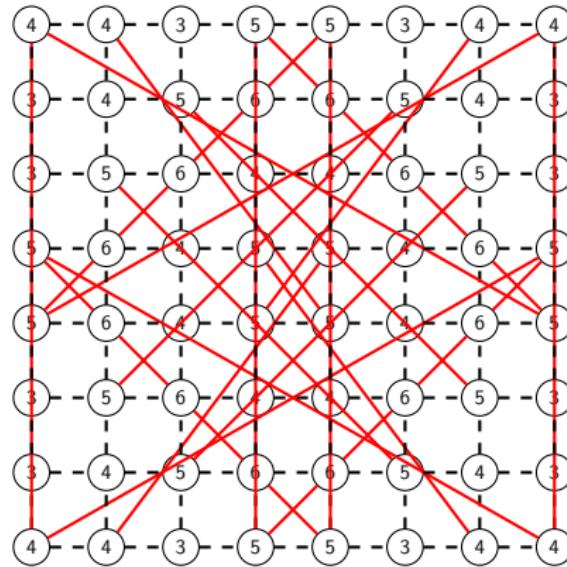
$$f = 10.531(11.139)$$

# Results



$$f = 12.139(12.948)$$

# Results



$$f = 13.56(14.709)$$

Thank you for your attention