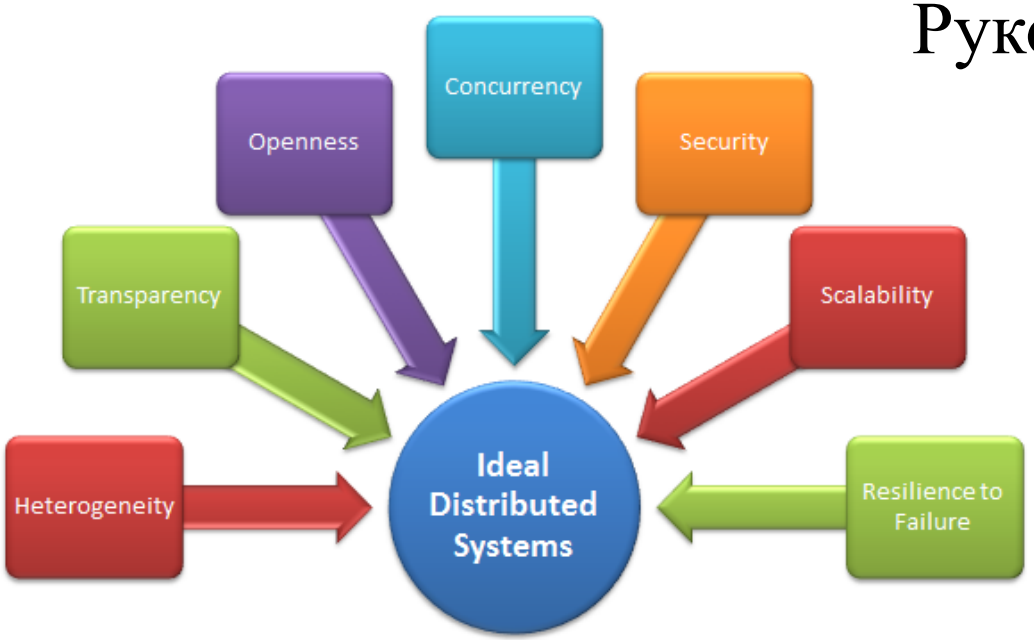
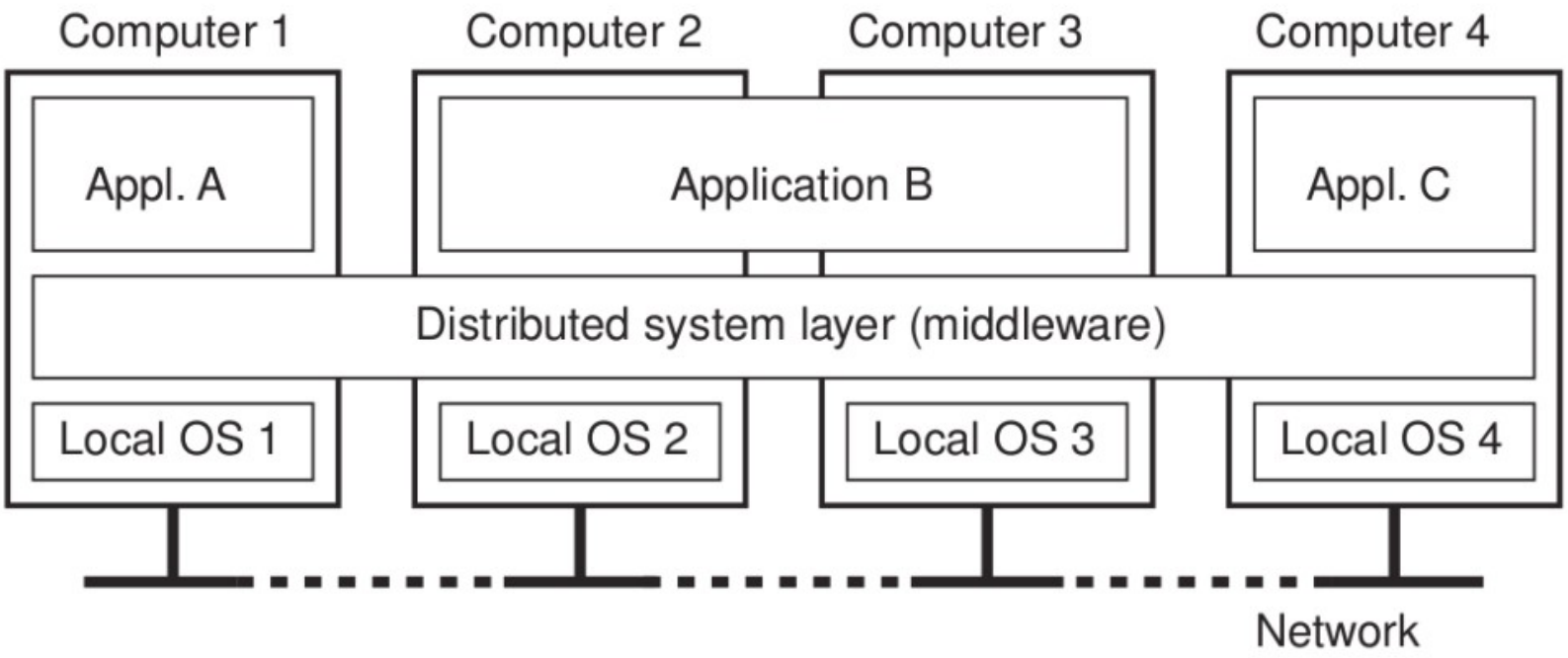
Распределенные системы. Алгоритмы балансирования весов орграфа



Выполнил: Ёров Собир
Руководитель: Мальковский Н. В.

Актуальность темы

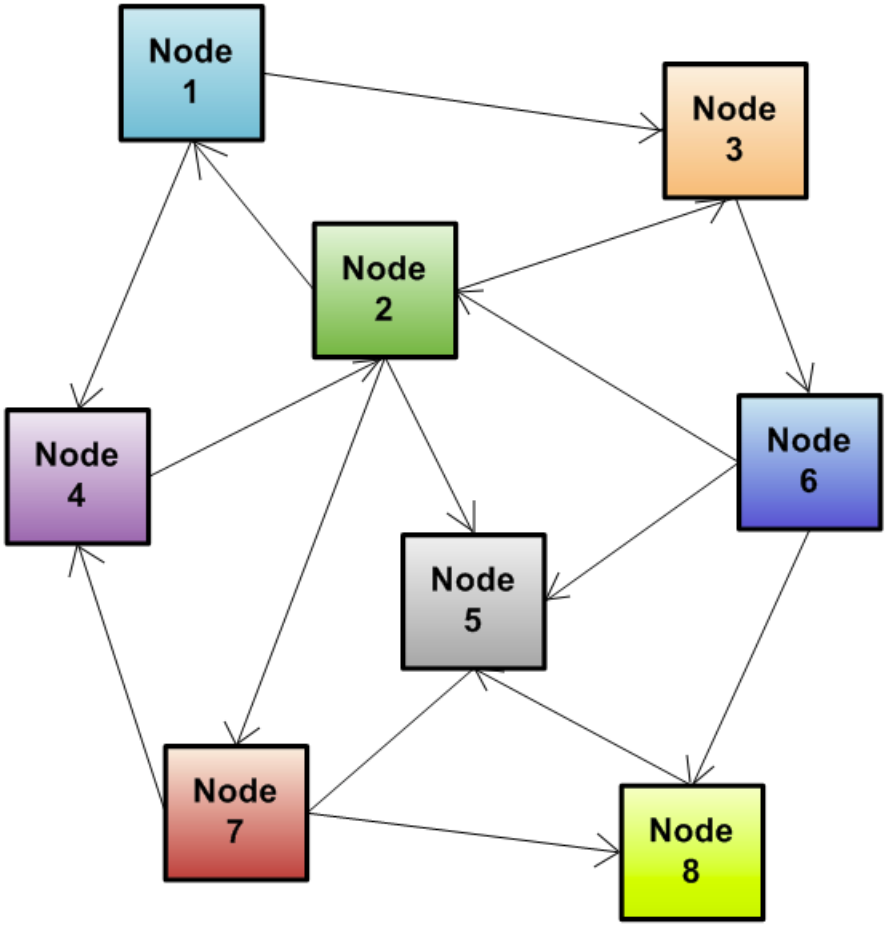
- **Распределенные системы**
  - это …
  - плюсы …



Неформальное описание задачи

- **Есть распределенная система, где все связи односторонние.**
- **Можем представить это как ориентированный граф.**
- **Хотим определить можем ли мы получить информацию от узла, которого мы видим напрямую, а он нас напрямую не видит.**
- **Это же определение сильной связности орграфа!**

Постановка задачи



1. Определить сильную связность в распределенной сети.

2. Написать приложение моделирующее работу сети и позволяющее для двух произвольных “узлов” сети сказать связаны они или нет.

3. Написать сетевое приложение, с той же функциональностью.

Альтернативные подходы

- **Алгоритмы нахождения компонент сильной связности**
  - Алгоритм Тарьяна
  - Алгоритм Косарайю
  - Алгоритм DCSC (Компоненты сильной связности по принципу “Разделяй и властвуй”)

- **Чем лучше наш алгоритм??**

Как хотим проверить сильную связность?

- **Воспользуемся алгоритмом балансирования весов орграфа!**
- **Теорема. Орграф сильно связен тогда и только тогда, когда его ребра можно сбалансировать (см. лит.).**

- **Используем итеративный алгоритм со следующей матрицей перехода:**

$$P = I - B + BD^{-1}A,\ B = diag(\beta_j),\ D = diag(outdeg(j)),\ A - \text{матрица смежности}$$

Этапы

Знакомство с алгоритмами балансирования весов орграфа распределенно:

- A. I. Rikos and C. N. Hadjicostis, “Distributed balancing of a digraph with integer weights”
- Apostolos I. Rikos, Themistoklis Charalambous, “Distributed weight balancing over digraph”

Приложение, моделирующее работу сети на одной машине, для определения сильной связности графа взаимодействий в сети:

- Многопоточность в Java

Сетевое приложение для определения связности вершин:

- TCP
- UDP

Основные проблемы в ходе решения задачи

new Thread(client).start();

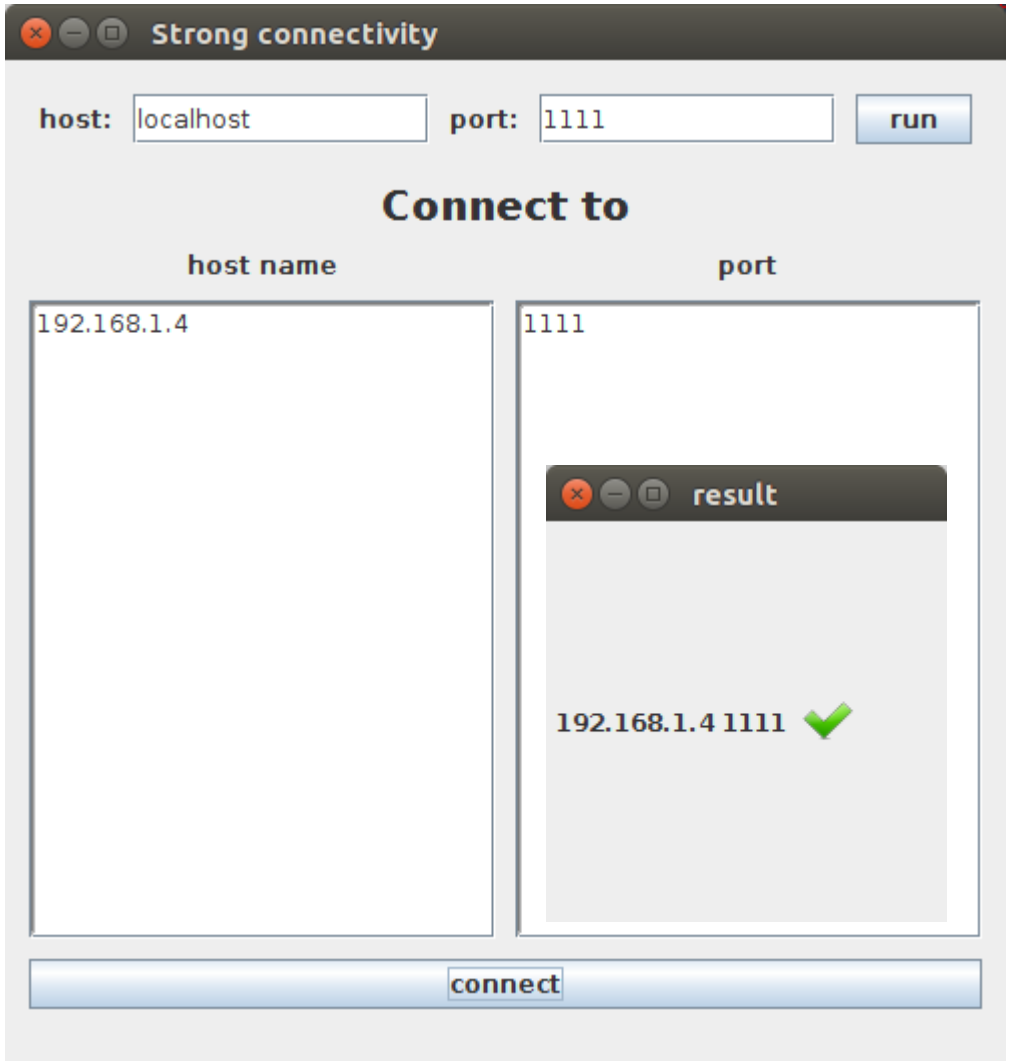
- **Неиспользование пула трэдов**

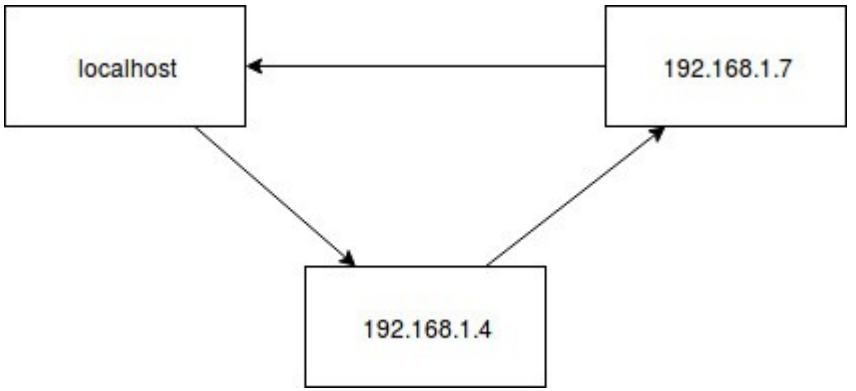
- **Проблемы с синхронизацией.**
  - Один узел находится в одной стадии итерации алгоритма а другой узел в другом

Результаты

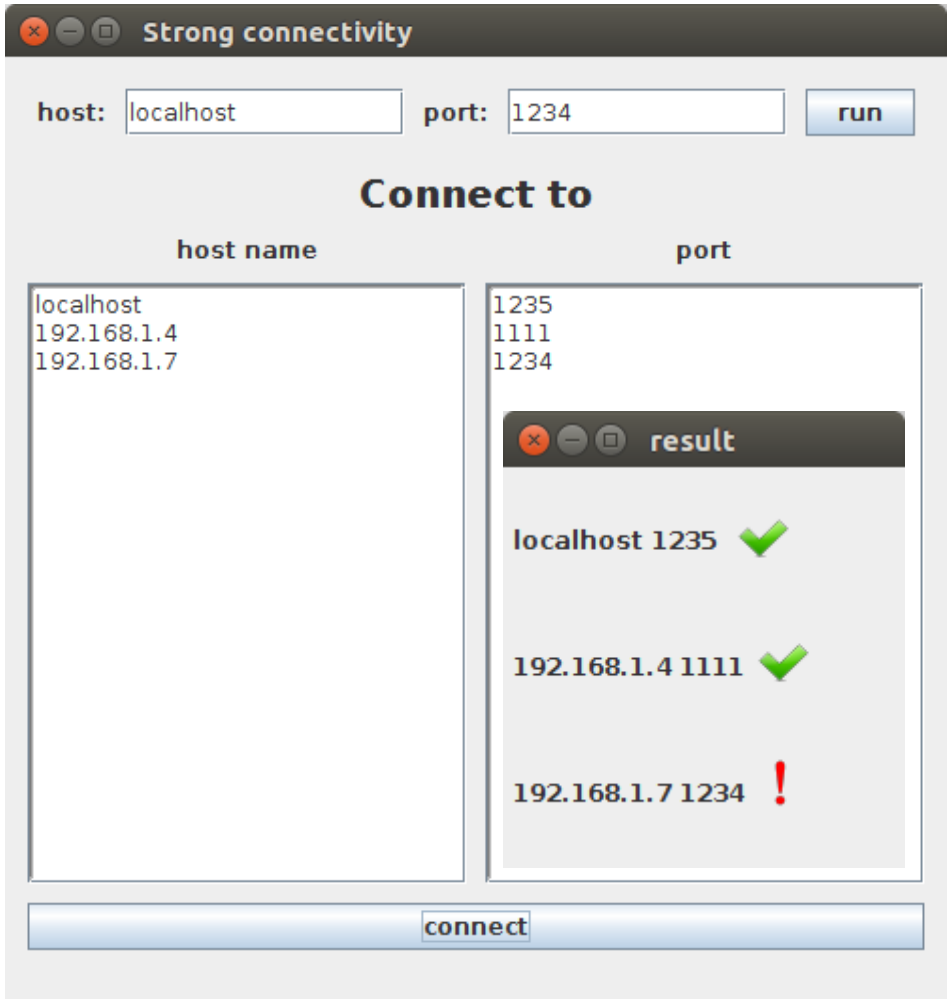
- **Многопоточное приложение**
  - Моделирующее работу сети – граф взаимодействий в сети
  - Возможность проверки сильной связности
- **Сетевое приложение с той же функциональностью**

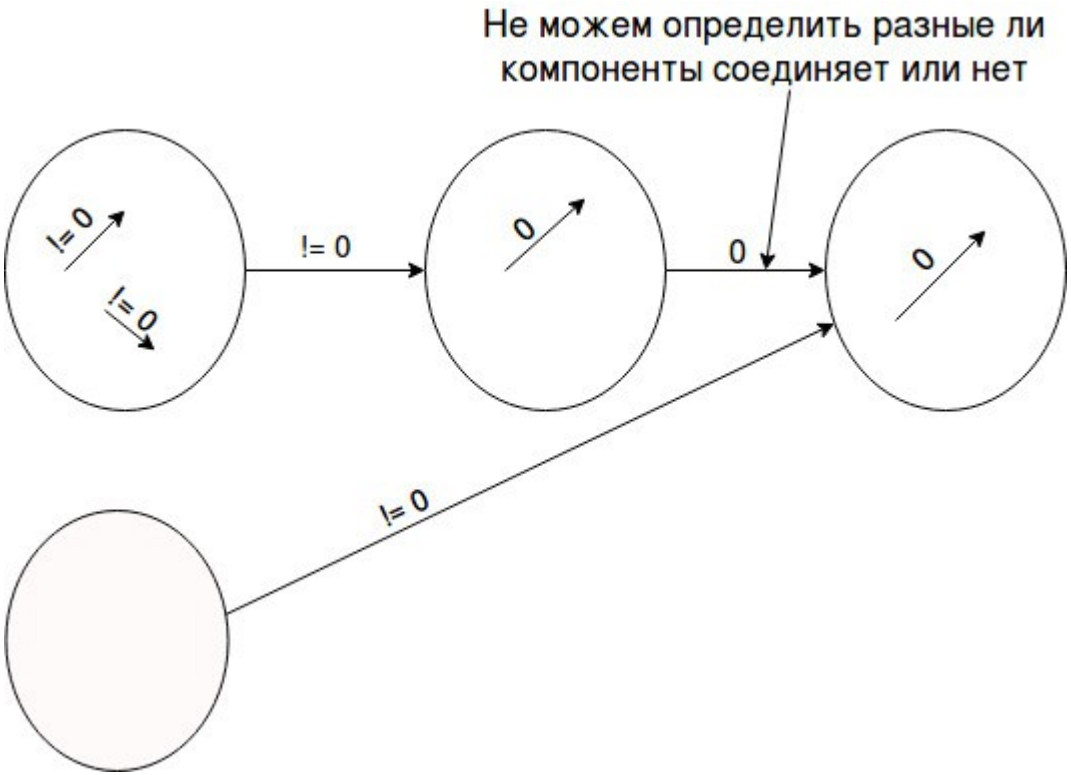
Пример работы программы





Когда алгоритм не работает





Технологии

- **Java**
  - Многопоточность
  - TCP/IP

Что сделали и что хотим сделать

- **Выполнили основную задачу:**
  - Сетевое приложение для проверки сильной связности графа взаимодействий в сети.

- **Хотим сделать:**
  - Доработать алгоритм, чтобы мы могли точно сказать про каждое ребро, то что она соединяет вершины одной компоненты или нет.
  - Провести сравнение с другими алгоритмами проверки сильной связности.
  - Попробовать ускорить.

Спасибо за внимание!

- Код доступен по ссылке:
  - https://github.com/YorovSobir/second-semester-research