

# ПАРАЛЛЕЛЬНЫЕ АЛГОРИТМЫ ПОИСКА КРАТЧАЙШИХ ПУТЕЙ НА ГРАФАХ

---

Выполнил: Ткаченко Г.С.

Руководитель: Корнеев Г.А.

13 мая 2015 г.

Университет ИТМО

# ПРОБЛЕМА И ЗАДАЧА

---

- Недостаточное разнообразие параллельных алгоритмов для поиска кратчайших путей
- Низкая производительность отдельных алгоритмов на специфичных графах

- Эффективное применение алгоритмов поиска кратчайшего пути на **многопроцессорных** архитектурах
- Разработка алгоритмов для поиска пути от одной вершины до всех (**one-to-many**)
- Разработка алгоритмов для поиска пути кратчайшего расстояния между каждой парой вершин (**many-to-many**)

## ЗАДАЧА ONE-TO-MANY

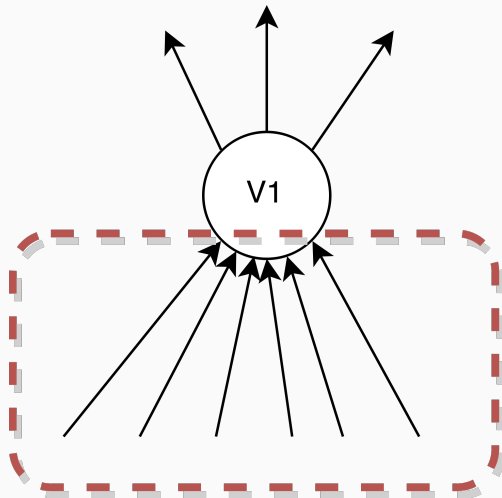
---

- Алгоритм Беллмана-Форда
  - Классический
  - На основе обхода в ширину
- Алгоритм Дейкстры
- Алгоритм Джонсона (Дейкстра с потенциалами)
- Алгоритмы  $A^*$  и  $D^*$

Три подхода

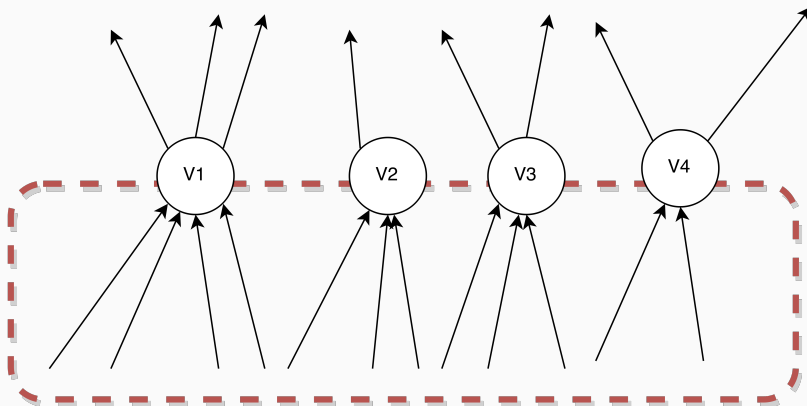
- Параллелизация по ребрам вершины
- Параллелизация по всем ребрам
- Использование параллельного обхода в ширину

## ПАРАЛЛЕЛИЗАЦИЯ ПО РЕБРАМ ВЕРШИНЫ

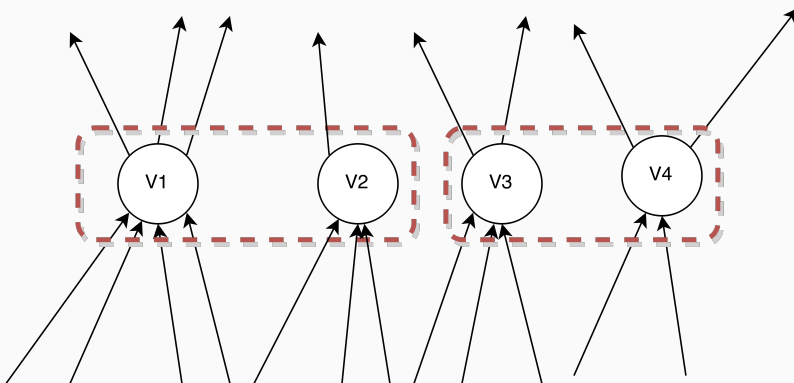




## ПАРАЛЛЕЛИЗАЦИЯ ПО ВСЕМ РЕБРАМ

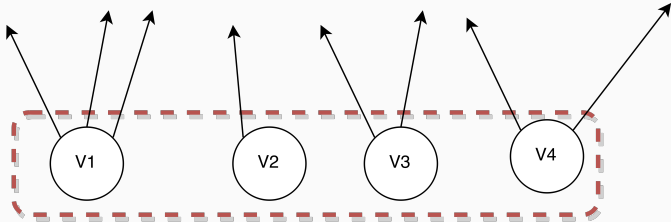


# ПАРАЛЛЕЛИЗАЦИЯ ПО ВСЕМ РЕБРАМ

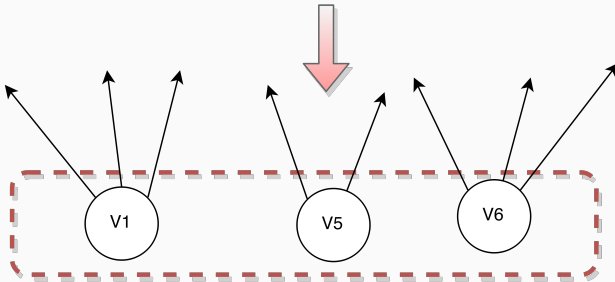


# ИСПОЛЬЗОВАНИЕ ПАРАЛЛЕЛЬНОГО ОБХОДА В ШИРИНУ

J



J + 1



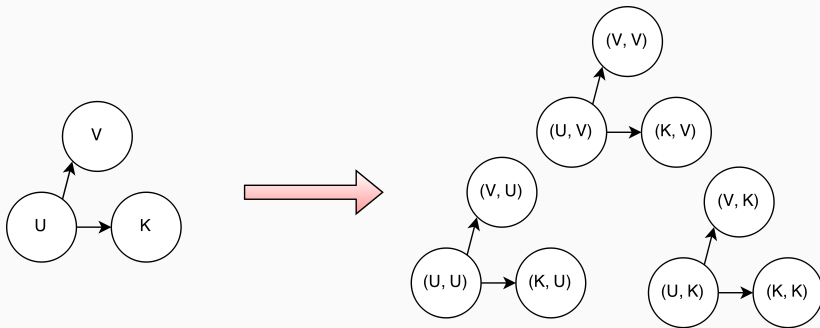
# ЗАДАЧА MANY-TO-MANY

---

- В некоторых случаях классический алгоритм оказывается медленнее наивных алгоритмов
- Для каждой вершины можно использовать любой алгоритм поиска кратчайшего пути

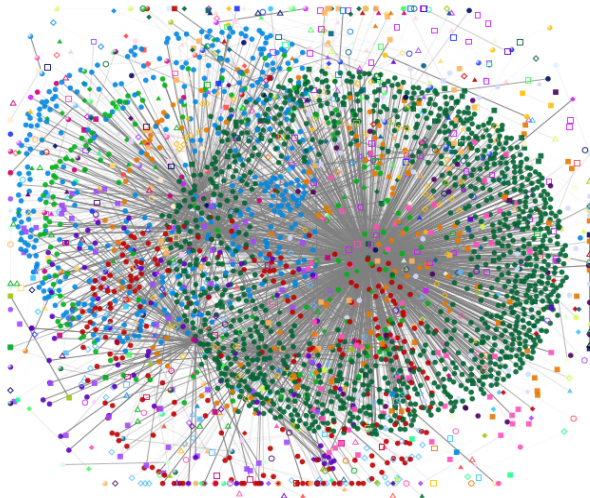
```
1: procedure ALLPAIRSPAR1(G)
2:   return HANDLEVERTICES(G, 0, |G.vertices|)
3:
4: procedure HANDLEVERTICES(G, startVertex, endVertex)
5:   if endVertex – startVertex < threshold then
6:     distances  $\leftarrow$  run Bellman-Ford for [startVertex, endVertex)
7:     return distances
8:   else
9:     midV  $\leftarrow$  (startVertex + endVertex)/2
10:    fork2(
      HANDLEVERTICES(G, startV, midV),
      HANDLEVERTICES(G, midV, endV));
```

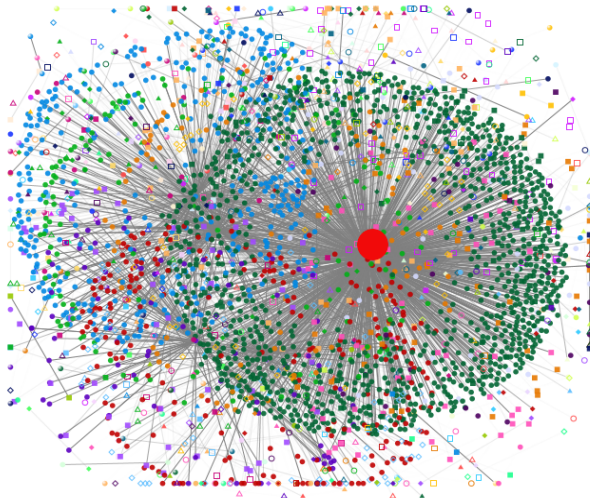
# АЛГОРИТМ ДЛЯ ОБЪЕДИНЕННОГО ГРАФА

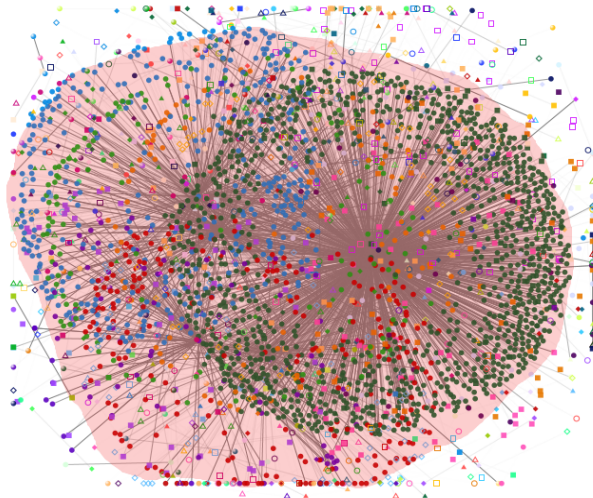


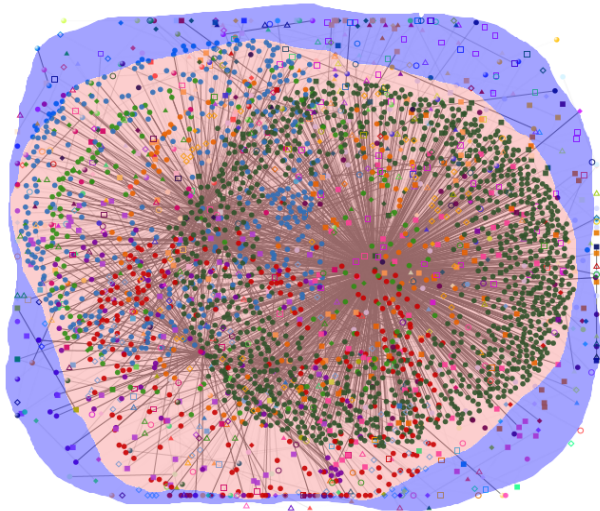
- Основан на теории "Шести рукопожатий"
- Работает не неориентированных невзвешенных социальных графах
- Использует идею динамического программирования











- $\text{mask}[u][i]$  — множество вершин, расстояние от которых до  $u$  равно  $i$
- $\text{calc}[u][i]$  — не более  $i$
- Для вершины  $u$  имеют смысл расстояния  $[d - K, d + K]$ , где  $d = \text{dist}[\text{base}][u]$

$$\text{mask}[v][i] = \neg \text{calc}[v][i - 1] \wedge \bigvee_{\exists (u,v) \in E} \text{mask}[u][i - 1] \quad (1)$$

$$\text{calc}[v][i] = \text{calc}[v][i - 1] \vee \text{mask}[v][i] \quad (2)$$

# РЕЗУЛЬТАТЫ

---

# СРАВНЕНИЕ ПАРАЛЛЕЛЬНЫХ ВЕРСИЙ БЕЛЛМАНА-ФОРДА

Идея алгоритма	Полный			Дерево		Решетка	
	TS	+	+-	0.5	1	+	+-
Ребра вершины	2.43	4.65	$\infty$	116.31	9.04	5.49	13.40
Все ребра	5.17	0.18	10.84	3.59	3.08	5.92	7.10
Обход в ширину	44.63	0.37	23.55	0.44	0.31	4.42	0.58

**Таблица:** Классические графы

Идея алгоритма	Разреженный			Плотный		
	0.5+	0.5-	0.96+	0.5+	0.5-	0.96+
Ребра вершины	$\infty$	$\infty$	24.35	$\infty$	$\infty$	5.01
Все ребра	2.77	14.68	2.42	0.48	6.38	0.46
Обход в ширину	0.98	22.59	0.76	0.60	10.25	0.71

**Таблица:** Случайные графы

Алгоритм	"Twitter graph"
Стандартная параллельная версия	427.217
Алгоритм для социальных графов	210.322

Таблица: Сравнение алгоритмов



Вопросы?