ПАРАЛЛЕЛЬНЫЕ АЛГОРИТМЫ ПОИСКА КРАТЧАЙШИХ ПУТЕЙ НА ГРАФАХ

Выполнил: Ткаченко Г.С.
Руководитель: Корнеев Г.А.

14 июня 2015 г.

Университет ИТМО

ПРОБЛЕМА И ЗАДАЧА

РЕШАЕМАЯ ПРОБЛЕМА

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

ПОСТАНОВКА ЗАДАЧИ

- 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

ОБЗОР РЕШЕНИЙ

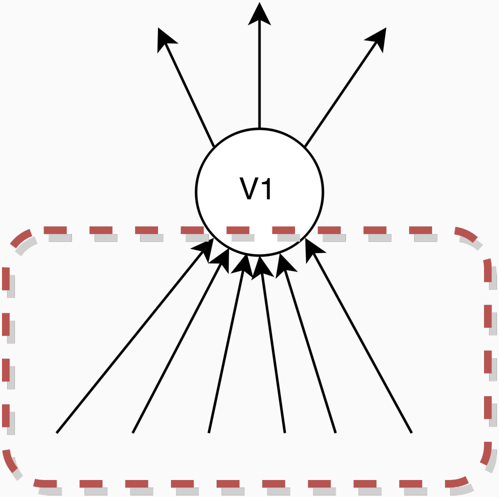
- Алгоритм Беллмана-Форда
    - Классический
    - На основе обхода в ширину
- Алгоритм Дейкстры
- Алгоритмы A*
- Алгоритмы D*

ПАРАЛЛЕЛЬНЫЙ БЕЛЛМАН-ФОРД

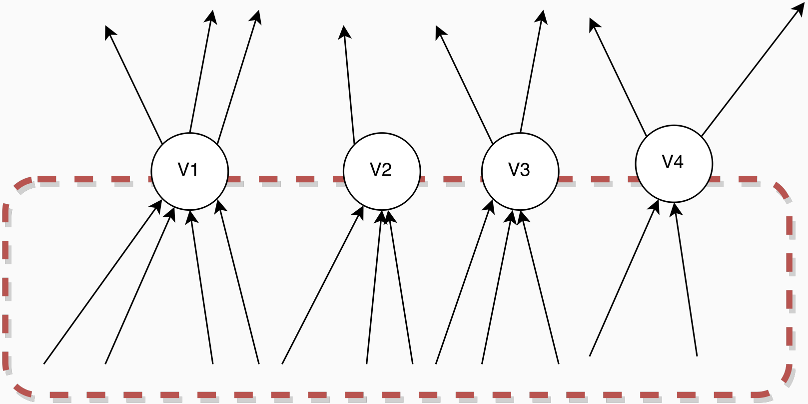
Три подхода

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

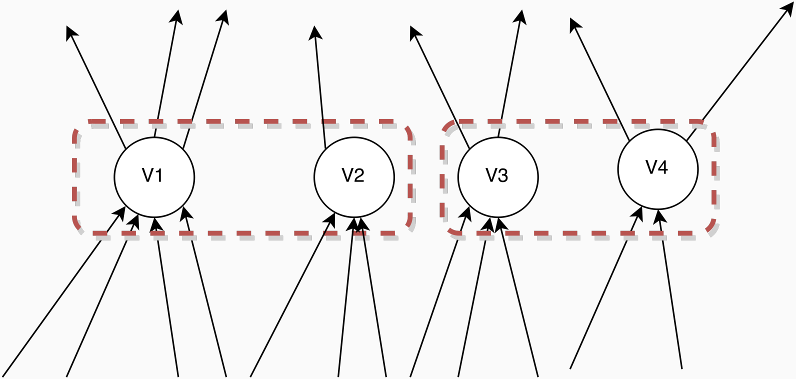
ПАРАЛЛЕЛИЗАЦИЯ ПО РЕБРАМ ВЕРШИНЫ



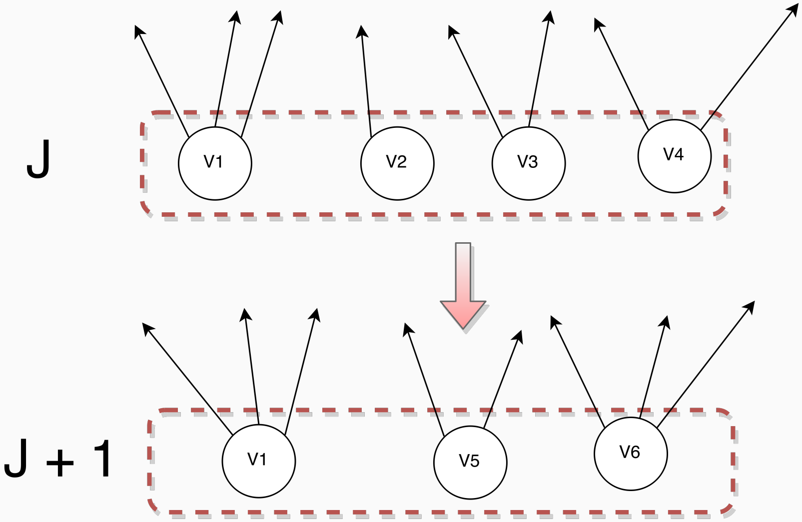
ПАРАЛЛЕЛИЗАЦИЯ ПО ВСЕМ РЕБРАМ



ПАРАЛЛЕЛИЗАЦИЯ ПО ВСЕМ РЕБРАМ



ИСПОЛЬЗОВАНИЕ ПАРАЛЛЕЛЬНОГО ОБХОДА В ШИРИНУ



ЗАДАЧА MANY-TO-MANY

АЛГОРИТМ ФЛОЙДА

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

НАИВНАЯ ПАРАЛЛЕЛЬНАЯ ВЕРСИЯ

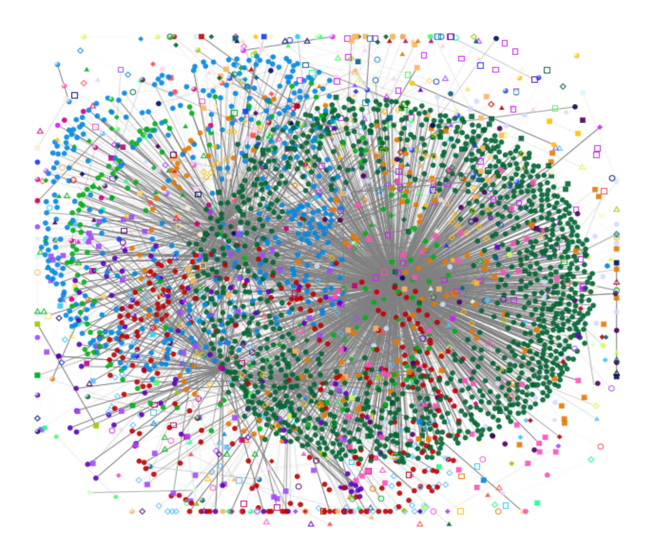
```
procedure AllPairsPar1(G)
    return HandleVertices(G, 0, |G.vertices|)

procedure HandleVertices(G, startV, endV)
    if endV − startV < threshold then
        run Bellman-Ford for [startV, endV)
    else
        midV ← (startV + endV)/2
        fork2(
            HandleVertices(G, startV, midV),
            HandleVertices(G, midV, endV));
```

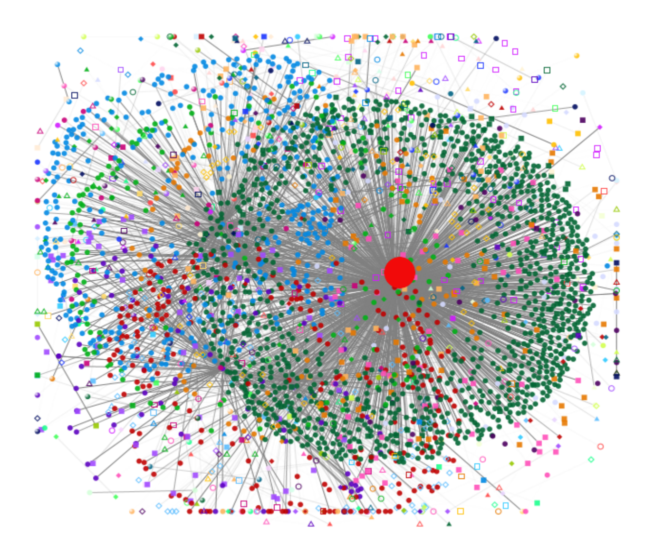
АЛГОРИТМ ДЛЯ СОЦИАЛЬНЫХ ГРАФОВ

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

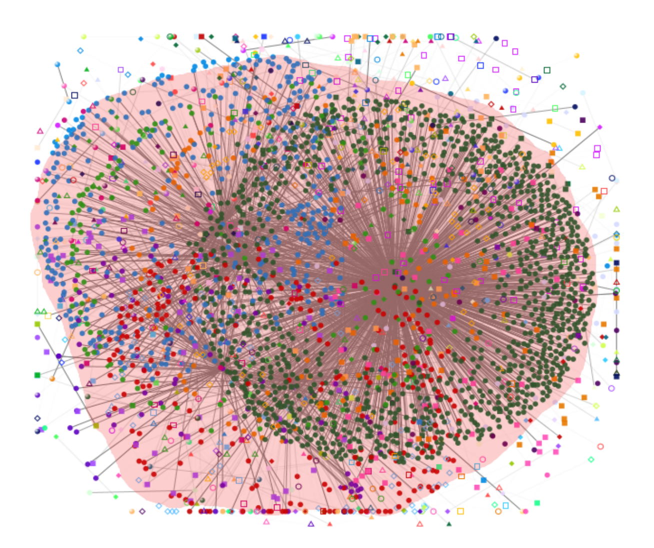
ИДЕЯ



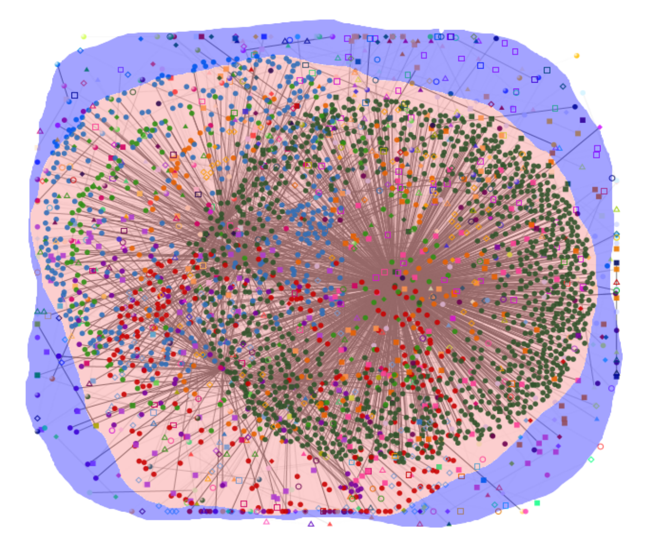
ИДЕЯ



ИДЕЯ



ИДЕЯ



- Основана на **битовых векторах**
- mask[v][i] — множество вершин U, что $\forall u \in U,\ p(u, v) = i$
- calc[v][i] — множество вершин U, что $\forall u \in U,\ p(u, v) < i$
- O(V) битовых векторов

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

$$calc[v][i] = calc[v][i-1] \vee mask[v][i] \tag{2}$$

РЕЗУЛЬТАТЫ

ТЕХНОЛОГИИ

- C++11
- PASL (Parallel Algorithm Scheduling Library)
- 40-core Intel machine (with hyper-threading)

СРАВНЕНИЕ ПАРАЛЛЕЛЬНЫХ ВЕРСИЙ БЕЛЛМАНА-ФОРДА

Идея алгоритма	Полный			Дерево		Решетка	
	TS	+	+-	0.5	1	+	+-
Ребра вершины	2.43	4.65	∞	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
Ligra	49.13	0.30	26.11	0.55	0.50	8.15	1.21

Идея алгоритма	Разреженный			Плотный		
	0.5+	0.5+-	0.96+	0.5+	0.5+-	0.96+
Ребра вершины	∞	∞	24.35	∞	∞	5.01
Все ребра	2.77	14.68	2.42	0.48	6.38	0.46
Обход в ширину	0.59	22.59	0.48	0.60	10.25	0.71
Ligra	0.58	25.19	0.54	1.12	14.15	1.20

РАССТОЯНИЕ МЕЖДУ КАЖДОЙ ПАРОЙ ВЕРШИН СОЦИАЛЬНОГО ГРАФА

Алгоритм	Twitter	Slashdot
Стандартная параллельная версия	427.217	254.567
Алгоритм для социальных графов	191.232	169.393

Таблица: Сравнение алгоритмов

итого

- Разработаны параллельные версии Беллмана-Форда
- Разработан алгоритм для социальных графов
- Алгоритмы показали высокую эффективность на практике
- Исследованы области применимости алгоритмов