

Кратчайшие пути в графе дорог

Андроник Ордиян

Научный руководитель: Алексей Гуревич

Академический университет

14 декабря 2013 г.

Обзор

1 Введение

2 Мотивация

3 Алгоритмы

4 Результаты

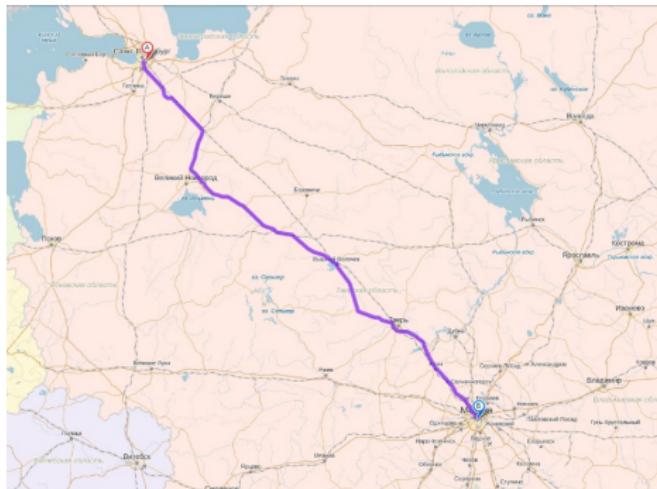
Задача кратчайшего пути

Вход

- Орграф $G = (V, E)$
- Веса рёбер $w(u, v) \geq 0$
- Начальная вершина s
- Конечная вершина t

Цель: найти кратчайший путь от s до t

- V – перекрёстки $\sim 10^7$
- E – отрезки дорог, их соединяющие $\sim 10^7$
- $w(u, v)$ – расстояние или время в пути



Мотивация

① Используется в:

- Яндекс.Карты, GoogleMaps, Yahoo! Maps
- Mapquest, Microsoft MapPoint
- GPS устройства
- Приближение для задачи TSP
- Некоторые задачи AI
- и т.д.

② Не всё так просто:

- Алгоритм Дейкстры не слишком быстрый
- Предпосчитать всё заранее не получится:
 - время (Европа): $n \times Dijkstra \sim 5$ лет
 - память(Европа): $n \times n$ таблица ~ 1 петабайт
- Нужно нечто среднее

Двунаправленный алгоритм

Двунаправленный Дейкстра

- запускаем **forward** Dijkstra от s , находим $d_f(v)$
- запускаем **reverse** Dijkstra от t , находим $d_r(v)$
- переключаемся после каждой итерации

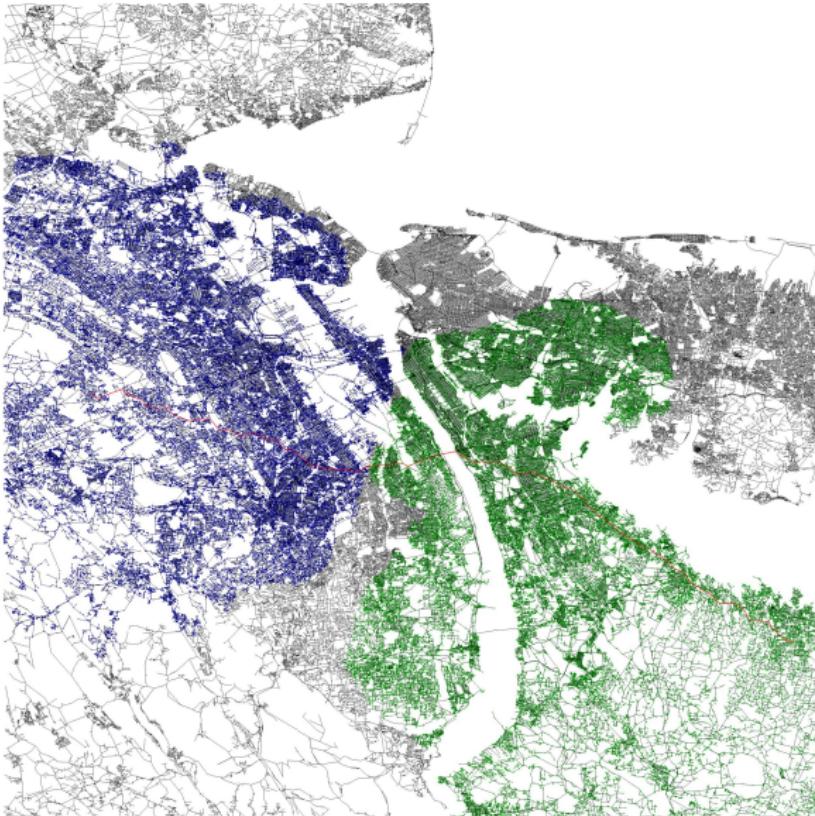
Когда найдётся вершина просканированная дважды:

- просматриваем все рёбра с концами, просмотренными разными Dijkstra
- находим минимум $d_f(u) + w(u, v) + d_r(v)$

Алгоритм Дейкстры



Двунаправленный Дейкстра



Сравнение

Метод	Предподсчёт (сек./mb)	scanned	ms
Dijkstra	—/30	506708	416.569
biDijkstra	—/30	380603	361.195

Таблица : США, Флорида, 1.2 млн

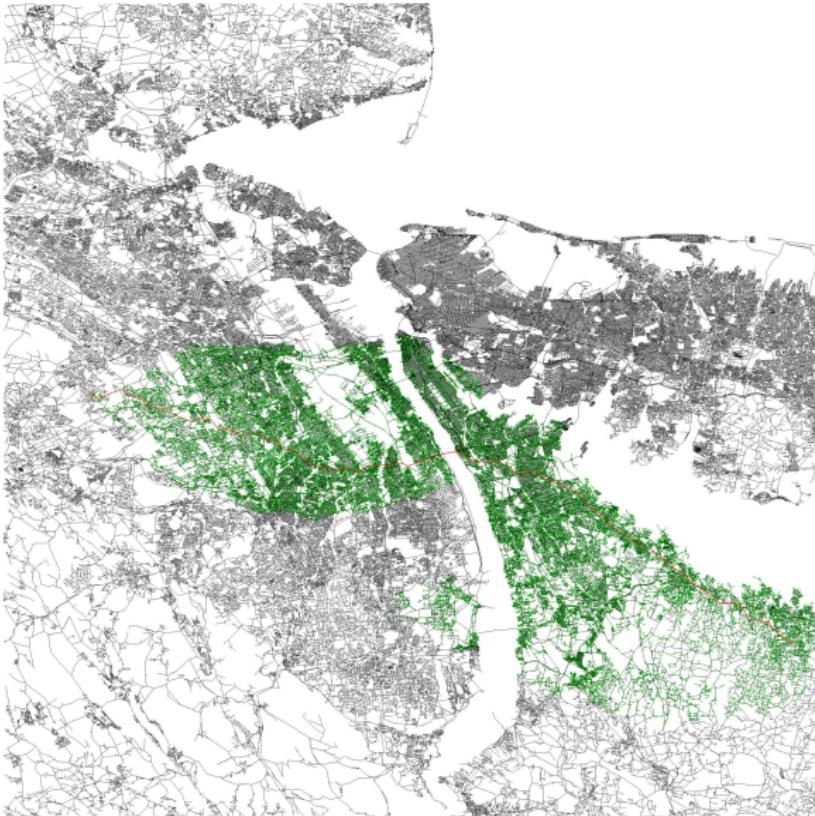
A* поиск

- Возьмём любую функцию (потенциала) $\pi: V \rightarrow \mathbb{R}$
- A* поиск \equiv Dijkstra с расстоянием w_π
- $w_\pi(u, v) = w(u, v) - \pi(u) + \pi(v)$
- Для корректности нужно: $w_\pi \geq 0$
- π даёт нижнюю оценку на расстояние
- Чем точнее оценка, тем быстрее сходимость
- Эффект: целенаправленный поиск

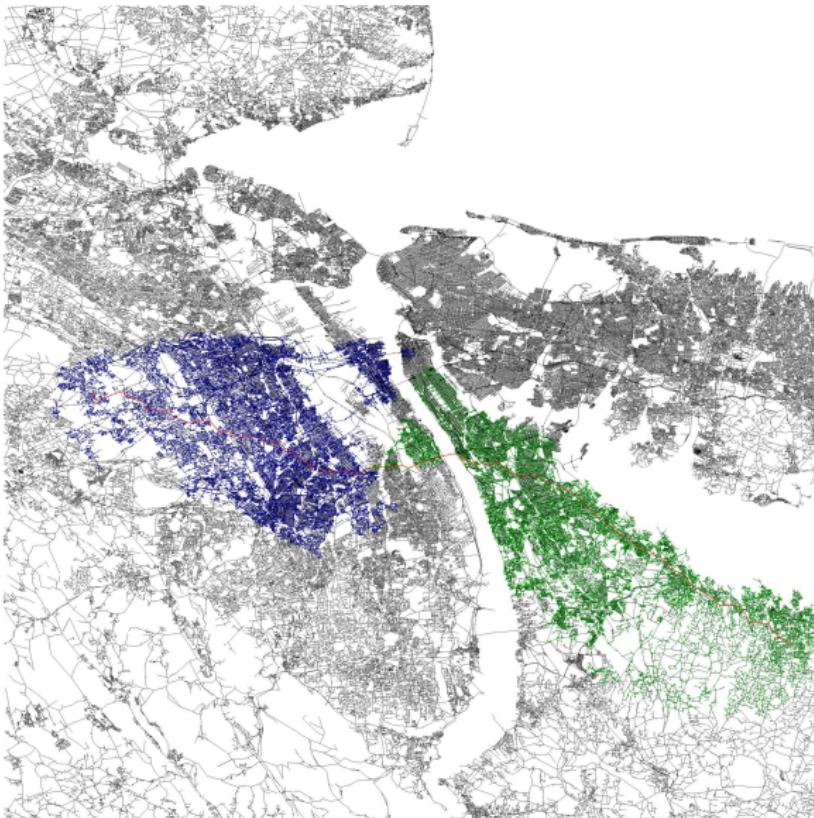
Двунаправленный A* поиск

- ① Нужно две функции потенциала: π_f и π_r
- ② Проблема: получатся разные расстояния!
- ③ Решение: усредняем потенциалы
- ④ $\pi_f = \frac{\pi_f - \pi_r}{2}$
- ⑤ $\pi_r = \frac{\pi_r - \pi_f}{2}$

A* поиск



Двунаправленный A*



Сравнение

Метод	Предподсчёт (сек./mb)	scanned	ms
Dijkstra	—/30	506708	416.569
biDijkstra	—/30	380603	361.195
A*	—/30	161969	206.392
biA*	—/30	149847	222.381

Таблица : США, Флорида, 1.2 млн

- A* + Landmarks + Triangle inequality: ALT [werneck]
- Предподсчёт: выбираем несколько (например, 16) вершин как **ориентиры**
- Считаем расстояние от них до всех остальных
- Сохраняем эти расстояния
- Запрос: A* поиск с ориентирами, используя неравенство треугольника

- ① Используем неравенство \triangle

$$\text{dist}(u, v) \geq \text{dist}(A, v) - \text{dist}(A, u)$$

$$\text{dist}(u, v) \geq \text{dist}(u, A) - \text{dist}(v, A)$$

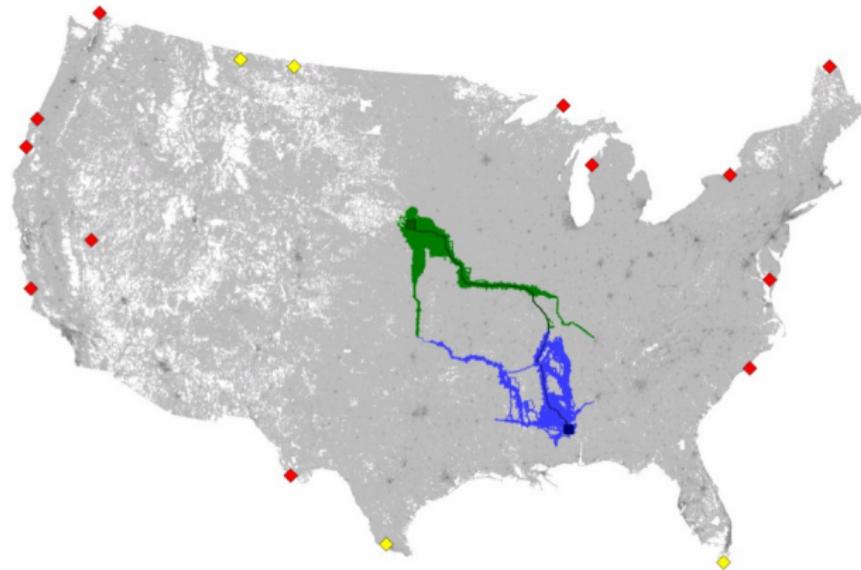
- ② Выбираем максимум нижних границ по всем ориентирам
③ Больше ориентиров \Rightarrow точнее оценка, больше памяти

ALT: выбор ориентиров

- ① Нужно выбрать ориентиры, которые хороши для всех запросов
- ② Несколько видов было опробовано:
 - random
 - planar
 - avoid

ALT: planar ориентиры

- 1 Делим карту на p равных секторов
- 2 Выбираем самую дальнюю от центра вершину в каждом секторе как ориентир



ALT: avoid ориентиры

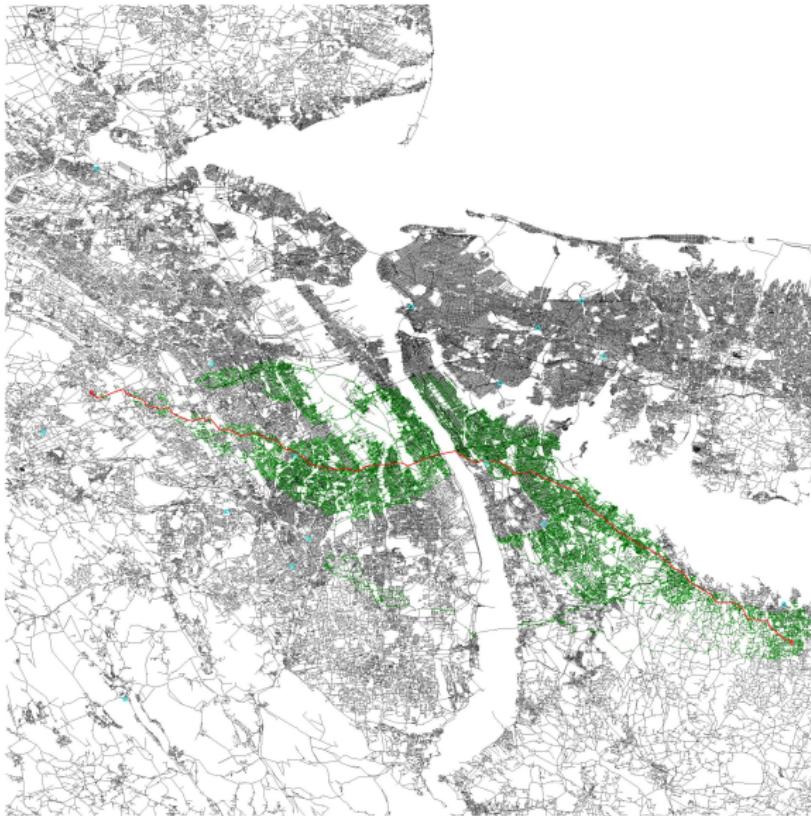
① Идея:

- Добавляем ориентиры по очереди
- Предпочтение областям, плохо покрытым уже выбранными ориентирами

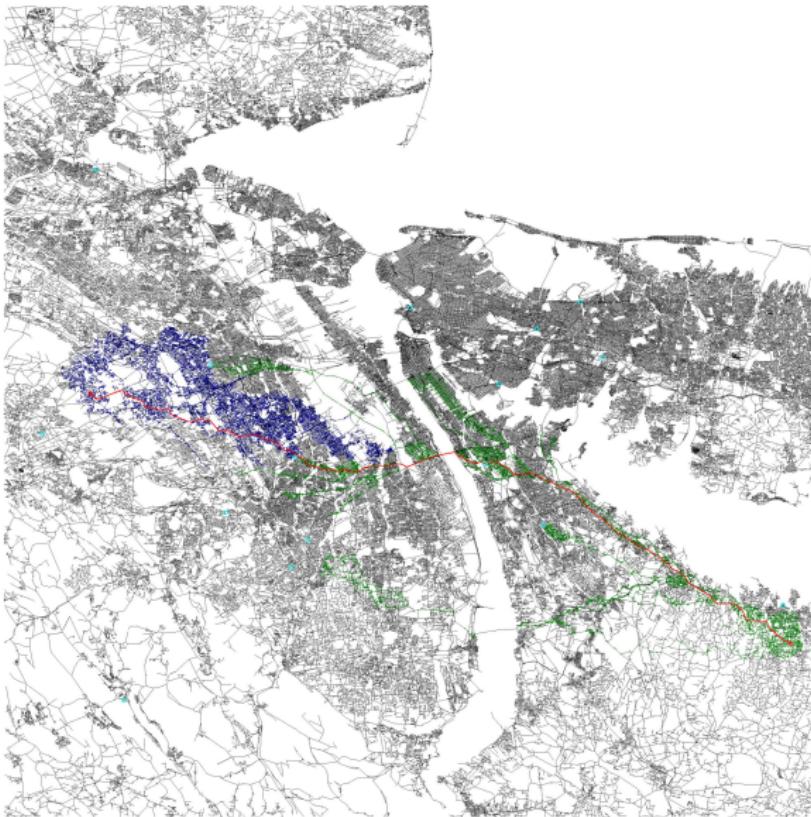
② Реализация:

- выбираем случайно r
- строим дерево кратчайших путей T от r
- для каждой вершины определим
 - $LB(v)$ — нижняя оценка на $dist(r, v)$
 - $weight(v) = dist(r, v) - LB(v)$
 - $size(v)$: сумма весов всех потомков v
(или 0, если среди потомков есть ориентиры)
- пусть w максимизирует $size(w)$
- начиная от w , спустимся по дереву, выбирая ребёнка с максимальным $size$
- выберем лист как новый ориентир

avoidALT



Двунаправленный avoidALT



Сравнение

Метод	Предподсчёт (мин./mb)	scanned	ms
Dijkstra	—/30	506708	416.569
biDijkstra	—/30	380603	361.195
A*	—/30	161969	206.392
biA*	—/30	149847	222.381
ALT	0.5/120	44360	102.071
biALT	0.5/120	31384	123.361

Таблица : США, Флорида, 1.2 млн

TNR

TODO

Результаты

Познакомился с

- ① Алгоритмами A*, ALT, TNR
- ② boost::optional, boost::test
- ③ форматом BMP

References

-  Renato F. Werneck (2010)
Shortest Paths and Experimental Evaluation of Algorithms
Microsoft Research Silicon Valley. MIDAS
-  A. V. Goldberg and C. Harrelson. (2005)
Computing the shortest path: A * search meets graph theory.
In Proc. 16th SODA, 156 – 165.
-  H. Bast, S. Funke, D. Matijevic, P. Sanders, and D. Schultes. (2007)
In Transit to Constant Shortest-Path Queries in Road Networks.
In Proc. 9th ALENEX, SIAM, 49 – 59

Вопросы?