← $O(N^2)$

1. Хейкстрой в реальное время ничего не решить - долго!
2. Двусторонний дейкстра : ↔ . (все ребра обрабатываются, поэтому точно сойдутся пути)

Аморийм (любой):

1) препроцессирование
2) запрос

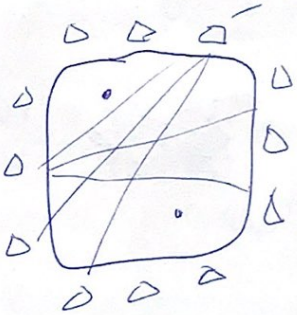
Ex для карт Европы препроц. 5 лет строит таблицу и занимает 1 гигабайт. Зато запрос очень быстро. - Дейкстра

3. Усовершенствование Дейкстры - A*

Усложнённый A* используется в реальной жизни.

4. ↔ двухсторонний A*

5. ALT ← неравенство △
↑ A* ↑ опорные точки
↑ отдельный алгоритм их выбора



Для опорных точек заранее знаем кратчайшие пути. Так для любых двух точек сможем находить кратчайшие пути быстро.

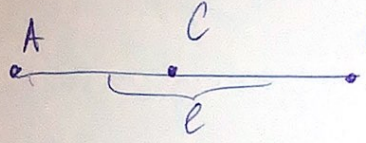
Ex. кучу запросов в яндекс картах, а уже заранее для опорных просчитано.

• Хорошая эвристика "до" 1-ой точки и "после" 2-ой.

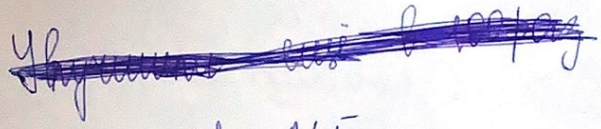
~~Получается в 100 раз быстрее Дейкстры~~

Между опорными (·) считается A*.

6. Когда мы точно знаем, что чтобы попасть из A в B нужно пойти по пути l. Например, единственный мост через воду.



C - reach - точка
и shortcuts - точки через которые часто ездят



7. REAL : Reach + ALT

8. Строим иерархию дорог.
Например, ищем путь из ~~A~~ одного города в другой, а потом внутри графа.

9. Оптимизация - флаги рёбер

чем больше рёбер, тем меньше площадь

какие рёбра относятся к кратчайшим для определённых вершин.

10. Две области например. И из одной нескольких въездов, ~~из~~ в другую несколько въездов. Например границы

11. 9 + 10,
Так ~~запрос~~ запрос будет работать не 4365 мс, а 0,002 мс
Препроцессирование от ∞ до 229 минут.

↗ Алгоритм поиска потока методом проталкивания предпотока

Форда-Фалкерсона: $O(F \cdot E)$ — Вставка

↑ вел-на макс. потока

↑ число рёбер

чтобы получить хоть какую оценку, не зная F, можно посчитать сумму выходящих потоков из истока – скорее всего он не будет максимальным.

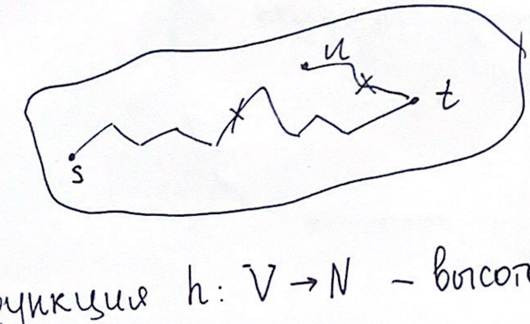
новый вид алгоритмов

- наиболее доказанный алгоритм с низкой сложностью поиска макс. потока в графе.
- предпоток: $f: V \times V \to \mathbb{R}$
  - → кососимметричная (поток в одн. сторону, равен в другую) (этим похожа на обычн. поток)
  - → предпоток на ребре $\leq$ пропускной способности этого ребра (как и у потока)
  - → втекать в вершину может $\geq$ чем вытекать
    - *избыток в вершине u: $e(u)$
    - если $e(u) \geq 0$ => вершина переполнена

Цель: найти максимальный предпоток

Поток максимален, если нельзя найти дополняющих путей в остаточной сети.

Предпоток максимален, если 1) мы не можем найти путей, дополняющих из истока в сток, но и 2) если мы не можем найти дополнительных путей из переполненной вершины в наш сток.



т.к. если будет переполнена вершина u, мы можем дотолкать в сток

- функция $h: V \to N$ – высота/пометка

$h(s) = |V|$

$h(t) = 0$

$\forall (u,v): h(u) \leq h(v) + 1$

скакать снизу вверх через слой нельзя

высота для разбиения на слои

C - остаточная сеть, пропуск. способ

У нас никогда не будет дополняющего пути из истока в сток слоев от 0 до $|V|$

Алгоритм Болберга относится к классу push/relable

f - предпоток

две процедуры

push (u,v): условия
1) $e(u) > 0$
2) $c_f(u,v) > 0$
3) $h(u) = h(v) + 1$

протолкнуть можем $\underbrace{\min(e(u), c(u,v))}_{d(u,v)}$

$f(u,v) = f(u,v) + d(u,v)$

? предполный предпоток

$f(u,v) = -f(u,v)$ строим обратные ребра

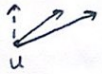
$e(u) = e(u) - d(u,v)$

$e(v) = e(v) + d(u,v)$

relable (u): 1) $e(u) > 0$

2) $\forall (u,v) \in E_f$ ↑ ребра из этой вершины

можем поднять вершину



$h(u) \le h(v)$

$h(u) = 1 + \min(h(v) \mid (u,v) \in E)$

Алгоритм:

$h(s) = |V|$

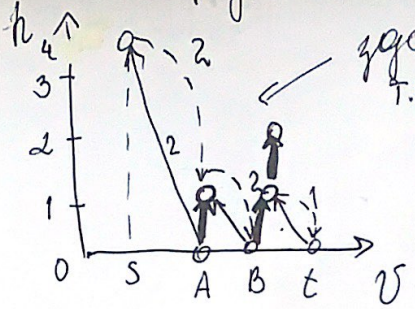
$h(t) = 0$

В цикле применяем пока можем push/relable

push (u,v) или relabe (u) – в произв. порядке

⇓

как. max предпоток = max поток



здесь нарушение инварианта, т.к. больше чем через один слой $S \to A$ сделали AS, теперь ок

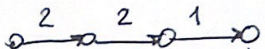
$h(s) = |V| = 4$

$h(t) = 0$

для первого шага:

$$f(u,v) = \begin{cases} C(u,v), & u = s \\ -f(u,v), & v = s \\ 0 \end{cases}$$

- высота истока и стока не изменяются
- заканчиваем, когда избытка нет нигде, кроме стартовой и конечной (= когда нельзя сделать push/relable)

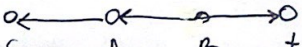
1) 

	S	A	B	t
h	4	0	0	0
e	-2	2	0	0

2) 

push применить не могу делаем relable

	S	A	B	t
h	4	1	0	0
e	-2	2	0	0

3) 

сделаем push 2 из A в B

	S	A	B	t
h	4	1	0	0
e	-2	0	2	0

4) relable

	S	A	B	t
h	4	1	1	0

5) push 1 из B в t

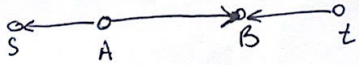
	S	A	B	t
h	4	1	1	0
e	-2	0	1	1

6) relable B

	S	A	B	t
h	4	1	2	0

в сток запомним что слили, теперь идём в исток

7) push из B в A



	S	A	B	t
h	4	1	2	0
e	-2	1	0	1

Рисуем на картинке и в конце по ней посчитаем значение потока

• каждая вершина может быть поднята не более 2|V| + 1 раз. (28)

$\parallel$

$O(V)$

в общем

relable $O(V \cdot E)$

при каждом поднятии анализируем ребро

• push

насыщающий
• в предпотоке полностью используется пропускная способность ребра
$O(V \cdot E)$

ненасыщающий
для оценки используется метод потенциалов
$O(V^2 E)$

Выгода алгоритма: 1) не зависит от max потока
2) квадратично зависит от V, а V всегда в таких задачах меньше, чем число E

Сложность: $O(V^2 E)$