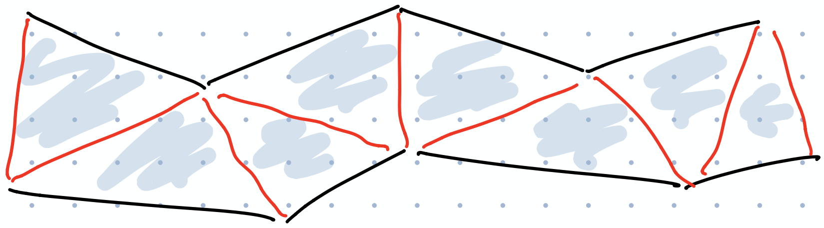
Old and new algorithms for cyclic dynamic programming

Sequence comparisons

- Edit distance
- Longest common subseq / Shortest common superseq
- Dynamic time warping
- Surface interpolation



$$OPT(i,j) = \min \begin{cases} OPT(i+1, j) + \alpha(i,j) \\ OPT(i, j+1) + \beta(i,j) \\ OPT(i+1, j+1) + \gamma(i,j) \end{cases} \Rightarrow \underline{O(n^2)}$$

Edit: $\alpha = 1$ $\quad \beta = 1$ $\quad \gamma(i,j) = [A[i] \neq B[j]]$

Surface: $\alpha = \text{area}(\triangle A[i], A[i+1], B[j])$ $\quad \beta = \ldots$

$\gamma = \infty$

Cyclic sequences

A B R A C A P O C U S

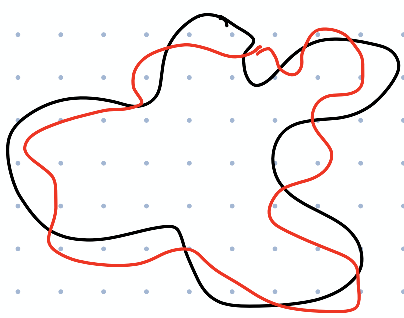
D A B R A H O C U S C A

$O(n^3)$ time immediate

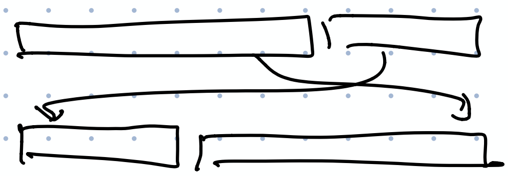
For $s \leftarrow 1$ to n

compare A with $\underline{B \circlearrowleft s}$

Shape alignment

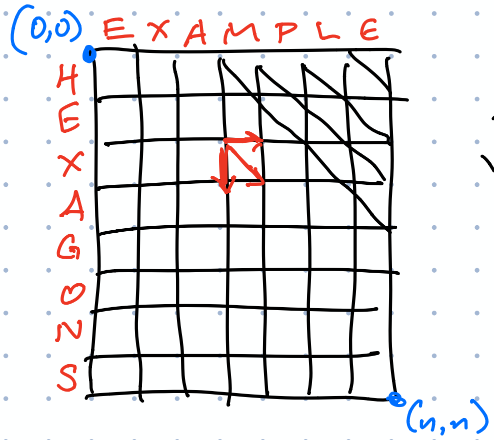


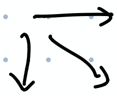
Circular RNA/DNA

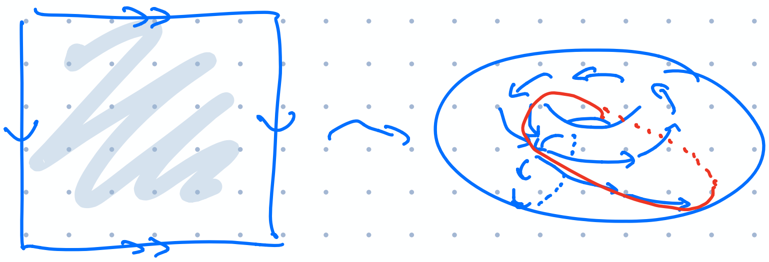


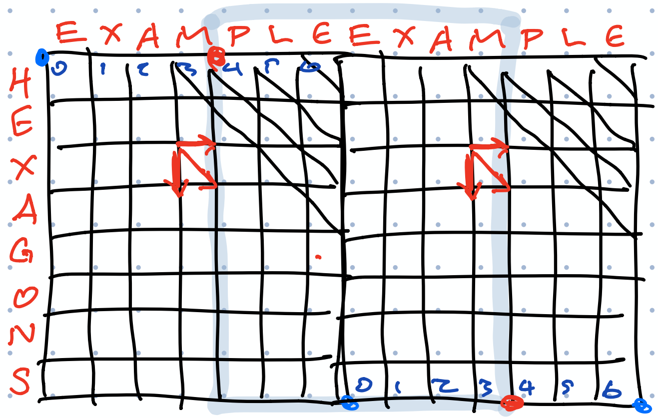
$\boxed{\tilde{O}(n^2) \text{ time}}$

Dynamic Programming = Shortest Path in DAG

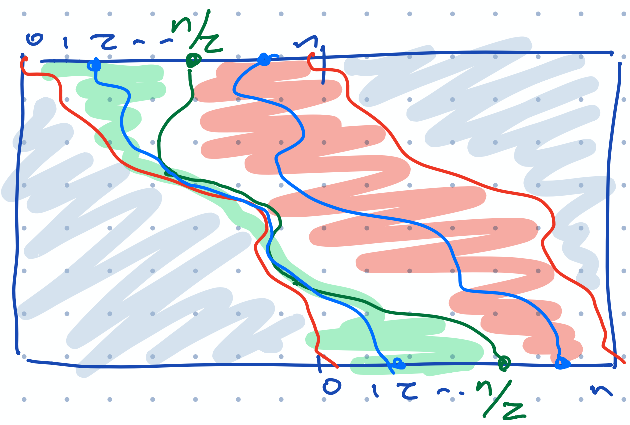








Shortest paths
from $(0,i)$ to $(n,n+i)$
for all i



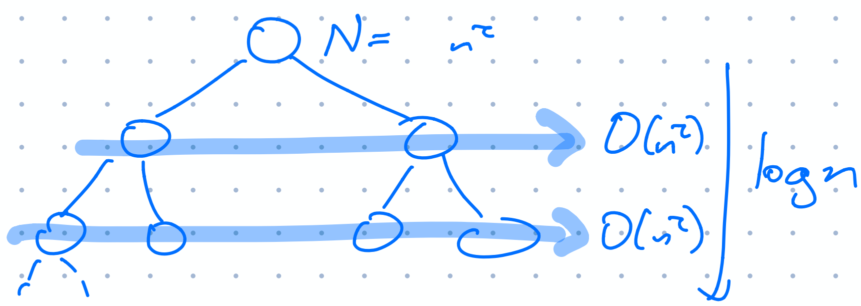
Divide + conquer

Fuchs Kedem Uselton 77
Maes 90
⋮

(WLOG) Shortest paths don't cross

Compute median path — $O(N)$
split graph into L + R
Recurse in L — $T(n/2)$
Recurse in R — $T(n/2)$ | depth $O(\log n)$

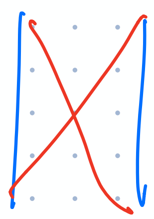
$\boxed{O(n^2 \log n)}$

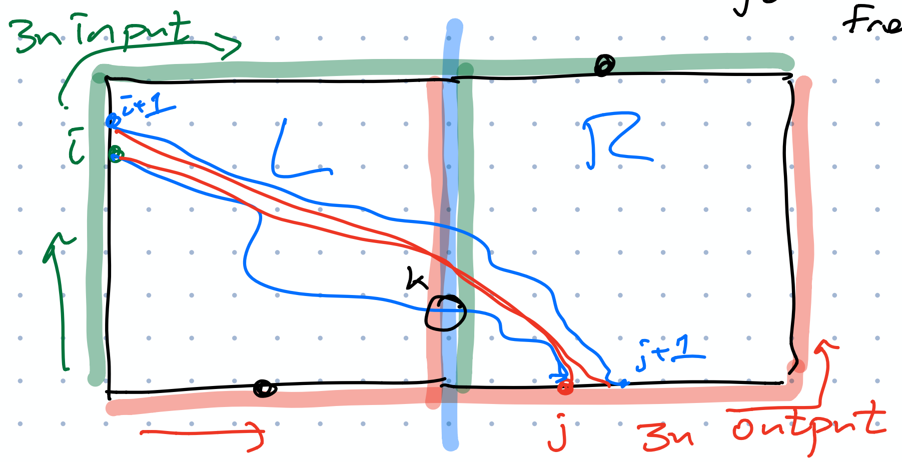


Divide + Conquer

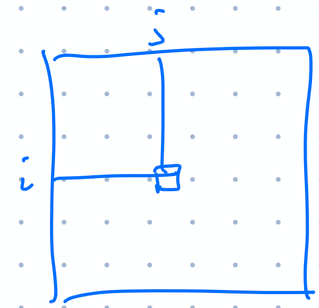
[Apostolico
Attalah
Larmore
McFadden '87]

DIST matrix

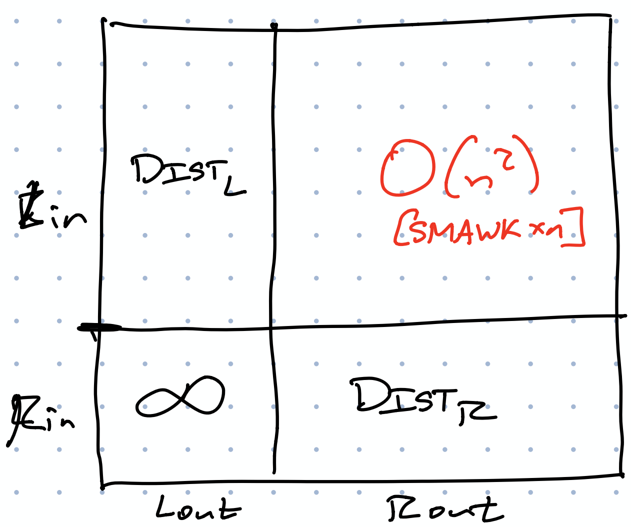




DIST[i,j] = shortest path distance from i^{th} input to j^{th} output



Compute $DIST_L$ and $DIST_R$ recursively



$$DIST[i,j] = \min_k \left(DIST_L[i,k] + DIST_R[k,j] \right)$$

Distance matrix multiplication

DIST is MONGE

$$M[i,j] + M[i+1,j+1] \le M[i,j+1] + M[i+1,j]$$

SMAWK
$\Rightarrow$ Monge dist mult in $O(n^2)$ time

$$T(n,m) = 2T\left(\tfrac{n}{2}, m\right) + O(n\,m)$$

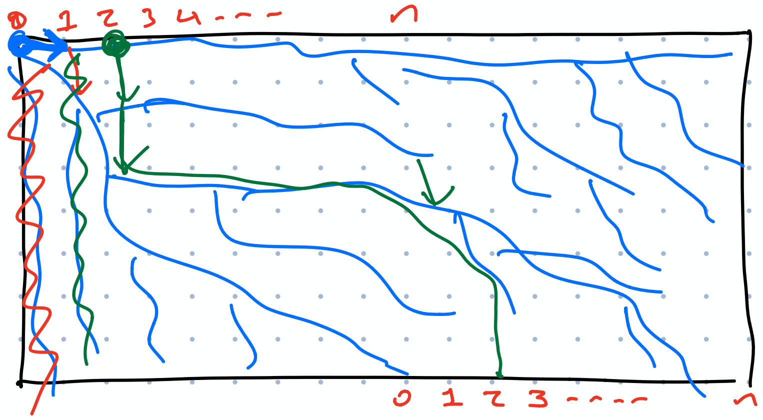
$$\Rightarrow \boxed{O(n^2 \log n)}$$

$$T(n) = \cancel{2}\,4T\left(\tfrac{n}{2}\right) + O(n^2)$$

Multiple-source shortest paths

[Landau Meyers Schmidt 98]

[Klein 05]
[Cabello Chambers E 08]



① compute shortest path tree rooted at 0

② drag root across ceiling
keep the tree correct

$\boxed{\text{Klein}}$ Total # changes $= O(n^2)$

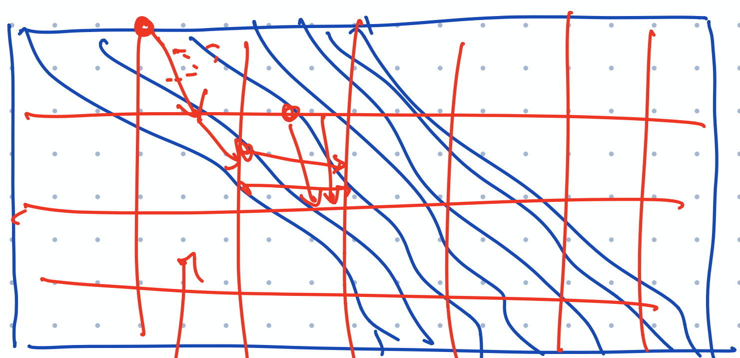
Every edge enters T ≤ 1 time
leaves T ≤ 1 time
Proof: Jordan curve thm

amortized analysis
↓
splay tree
↓
dynamic forest ⟶ [implicit representation]
O(log n) query

$\boxed{\text{Klein}}$ O(log n) per change

$\boxed{O(n^2 \log n) \text{ time}}$ ←

Lots of redundant computation



"DIST" using MSSP

[Italiano Nussbaum
Sankowski Wulff-Nilsen 10]

[Fakcharoenphol Rao 01]

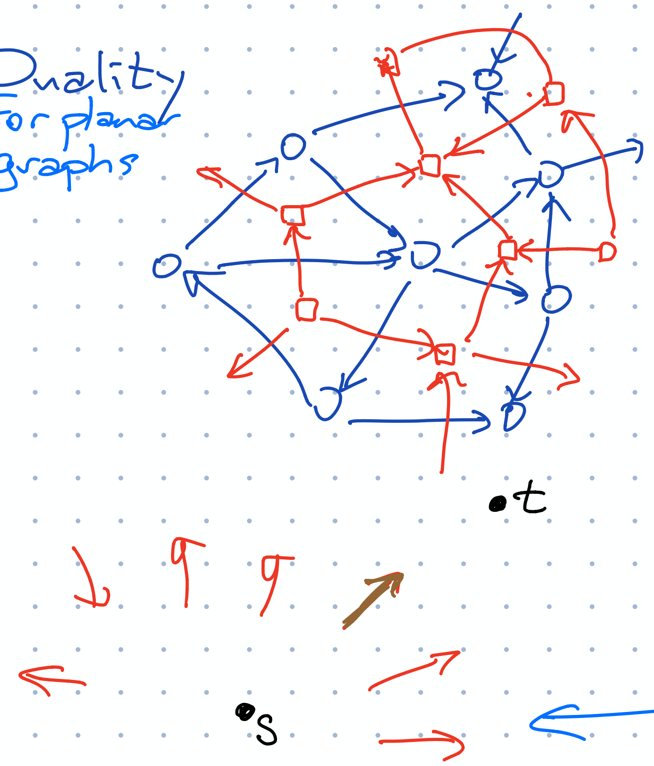
$\boxed{O(n^2 \log\log n) \text{ time}}$

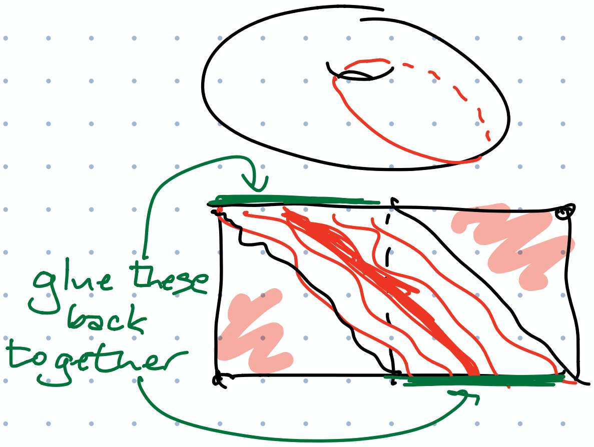
?

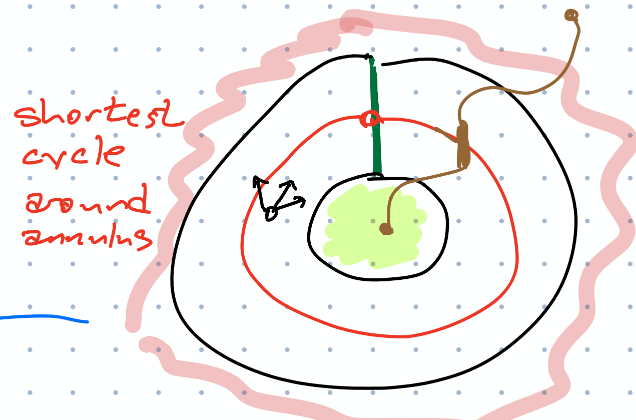
$\boxed{O(n^2)}$

But it's really a flow problem

Duality for planar graphs







Set of edges partition vertices
min total cost

$\boxed{\text{min } (s,t)\text{-cut}}$

$\downarrow$

$\boxed{\text{max } (s,t)\text{-flow}}$ $\boxed{O(n^2 \log n)}$

Set of edges partition Faces
min total cost

Borradaile Klein '08
Erickson '10 — MSSP

Edit distance — Edge wts all either 1 or 0

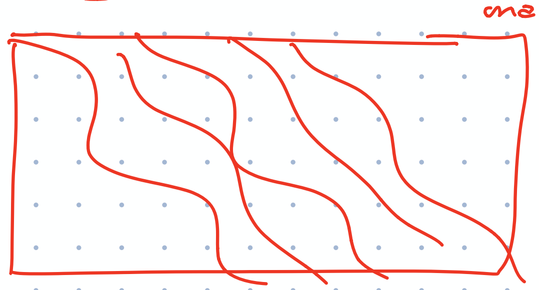
SCS — Edge wts all 1 ~~or 00~~

$\hookrightarrow$ Flow w/all cap = 1
= edge disjoint paths

[Brandes Wagner 90]
[Eisenstat Klein 15]

$\boxed{O(n^2)}$

Unweighted → directions don't matter!



Edge Disjoint path in undirected planar graph → [Weihe 87] → $\boxed{O(n^2)}$ simple