Dynamic Shortest Paths

Camil Demetrescu

University of Rome "La Sapienza"

Giuseppe F. Italiano

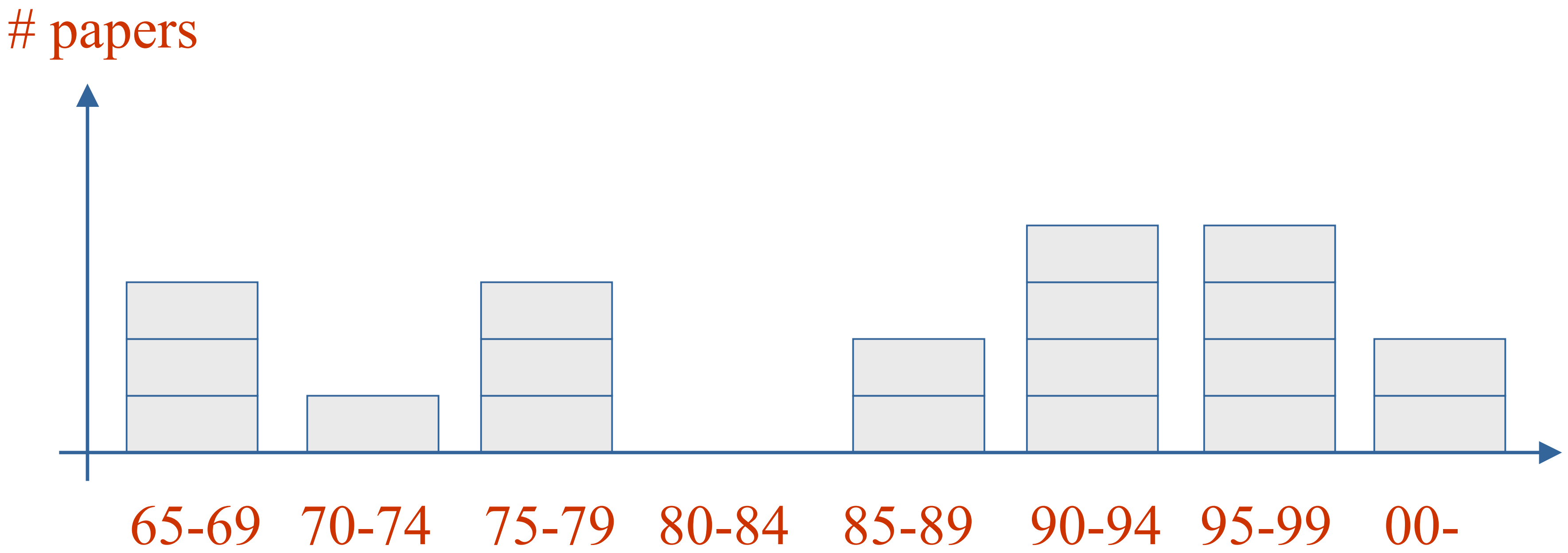
University of Rome "Tor Vergata"

Dynamic (All Pairs) Shortest Paths

Given a weighted directed graph G=(V,E,w), perform any intermixed sequence of the following operations:

`Update`(u,v,w):	update weight of edge (u,v) to w
`Query`(x,y):	return distance from x to y (or shortest path from x to y)

Previous work on dynamic APSP



Previous work on fully dynamic APSP

	Graph	Weight	Update	Query
Update recomputing from scratch	general	reals	$\tilde{O}(n^3)$	$O(1)$
		[0,C]	$\tilde{O}(n^{2.575} C^{0.681})$	
King 99	general	[0,C]	$O(n^{2.5} (C \log n)^{0.5})$	$O(1)$
Henzinger et al. 97	planar	[0,C]	$O(n^{9/7} \log(nC))$	
Fackcharoemphol, Rao 01	planar	reals	$O(n^{9/5} \log^{13/5} n)$	
D, Italiano 01	general	S reals	$O(n^{2.5} (S \log^3 n)^{0.5})$	$O(1)$
	general	reals	$? < o(n^3)$	$O(1)$

A new approach

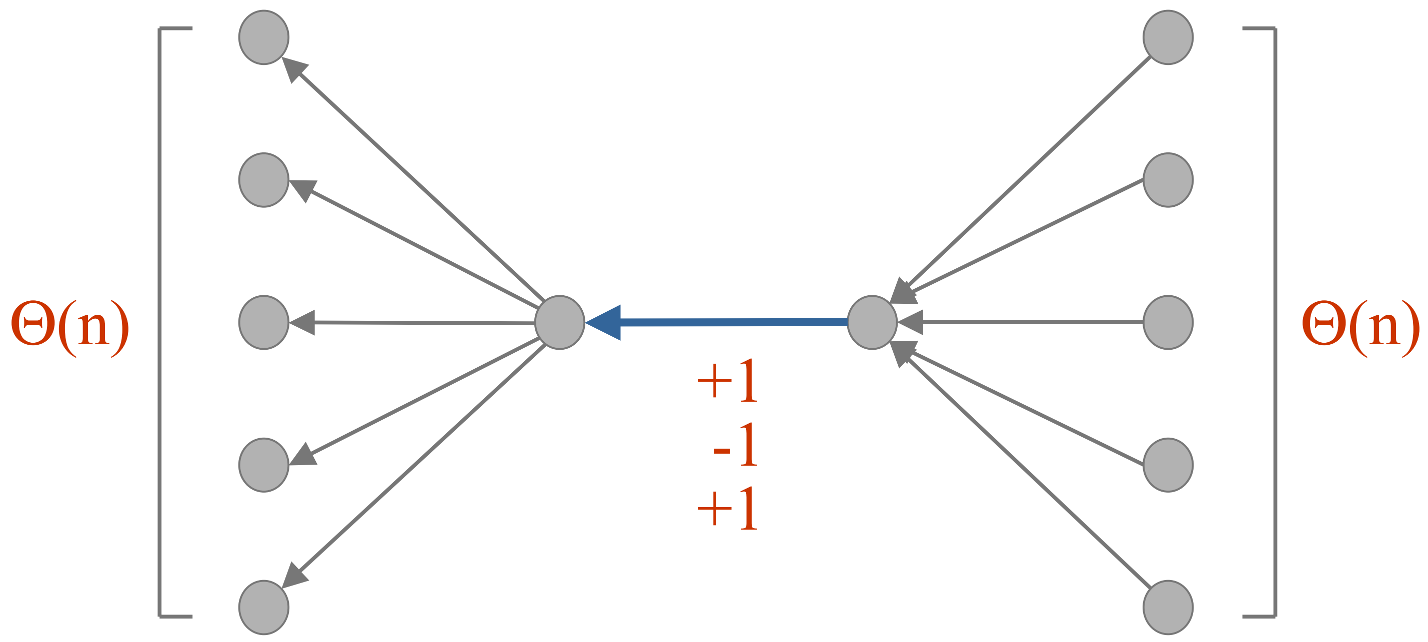
New combinatorial properties of graphs:

Uniform paths

A new fully dynamic algorithm

	Graph	Weight	Update	Query
King 99	general	[0,C]	$O(n^{2.5} (C \log n)^{0.5})$	$O(1)$
Henzinger et al. 97	planar	[0,C]	$O(n^{9/7} \log(nC))$	
Fackcharoemphol, Rao 01	planar	reals	$O(n^{9/5} \log^{13/5} n)$	
D, Italiano 01	general	S reals	$O(n^{2.5} (S \log^3 n)^{0.5})$	$O(1)$
D, Italiano 02	general	reals	$O(n^2 \log n)$	$O(1)$

$\Theta(n^2)$ changes per update



Main ingredients of previous results

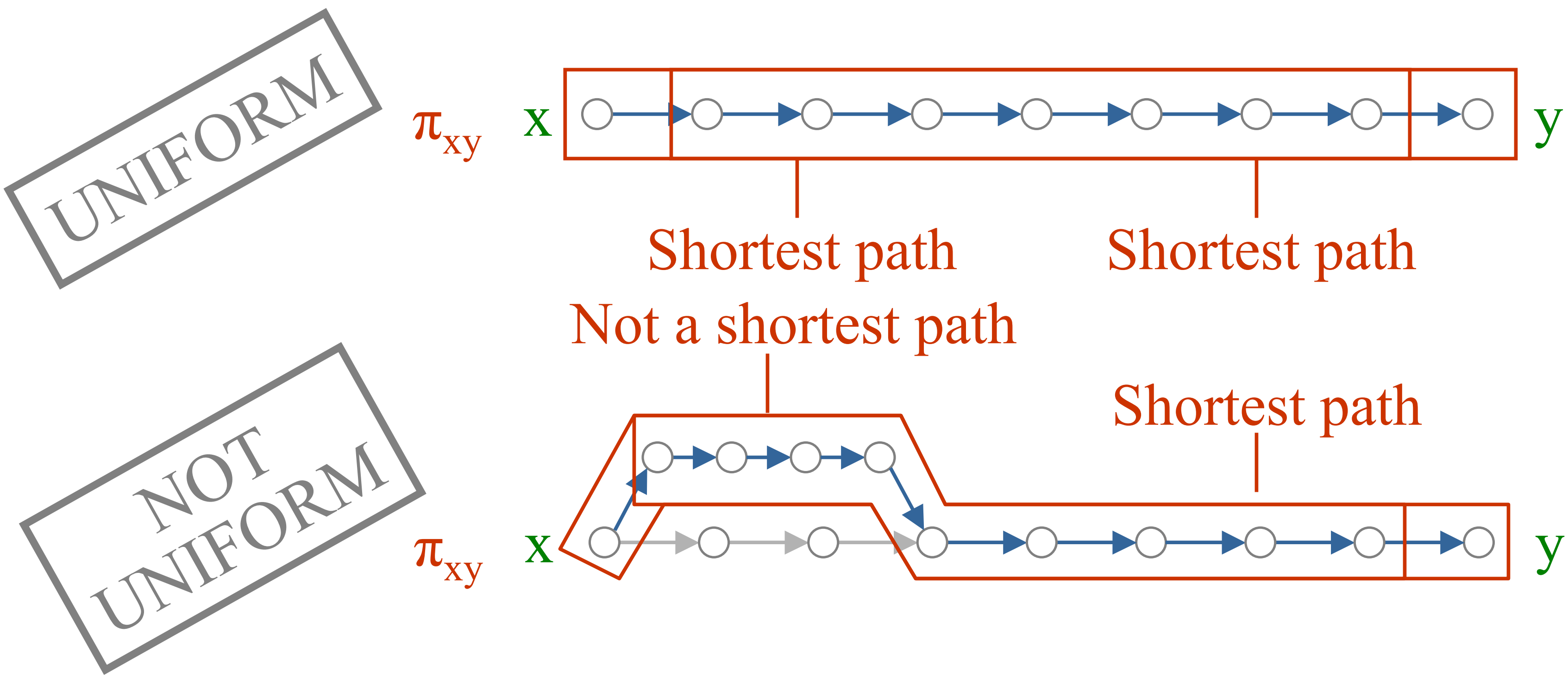
Maintaining breadth-first trees under edge deletions
[Even, Shiloach 85]
(results for general graphs with integer weights in [0,C])

Dynamic reevaluation of products of matrices
[D, Italiano 00]
(results for general graphs with S real weights per edge)

Long paths property
[Greene, Knuth 82]
(all fully dynamic algorithms on general graphs)

Uniform paths

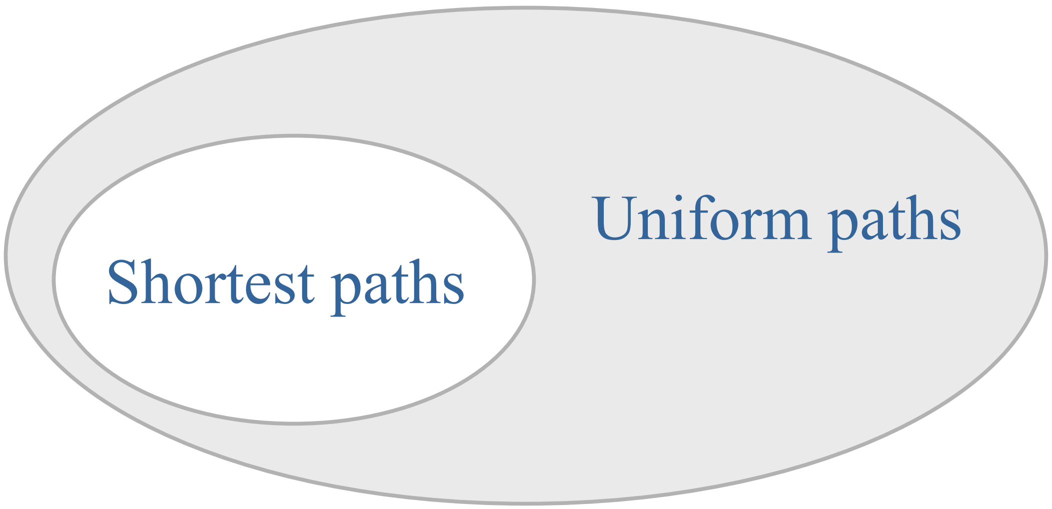
A path is *uniform* if all of its subpaths are shortest paths



Properties of Uniform paths

Theorem I

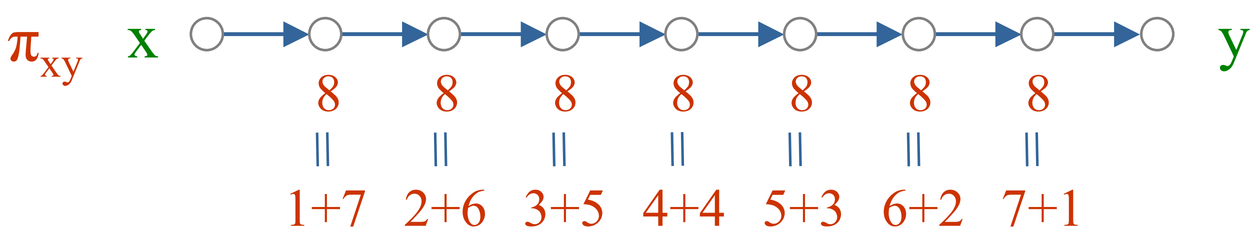
Shortest paths $\subseteq$ Uniform paths



Properties of Uniform paths

Theorem II

A path is uniform ⇔ for any internal vertex, the sum of the distances to the path endpoints is equal to the weight of the path



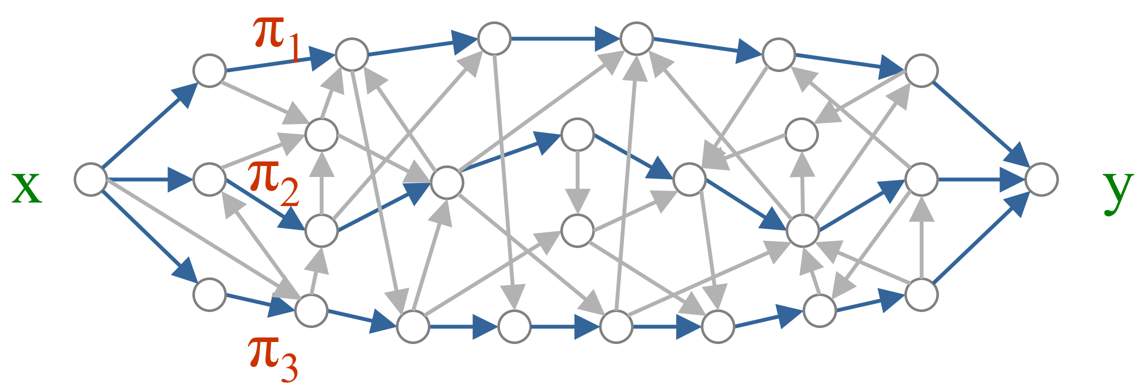
Properties of Uniform paths

For the sake of simplicity, we assume that no two paths in the graph have the same weight.

(Ties can be broken by adding a tiny fraction to the weight of each edge)

Theorem II

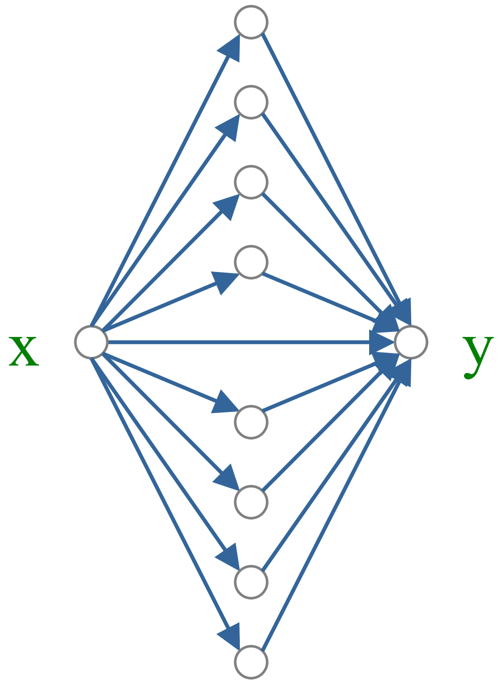
Uniform paths π_{xy} are internally vertex-disjoint



Properties of Uniform paths

Theorem III

There are at most n-1 uniform paths connecting x,y



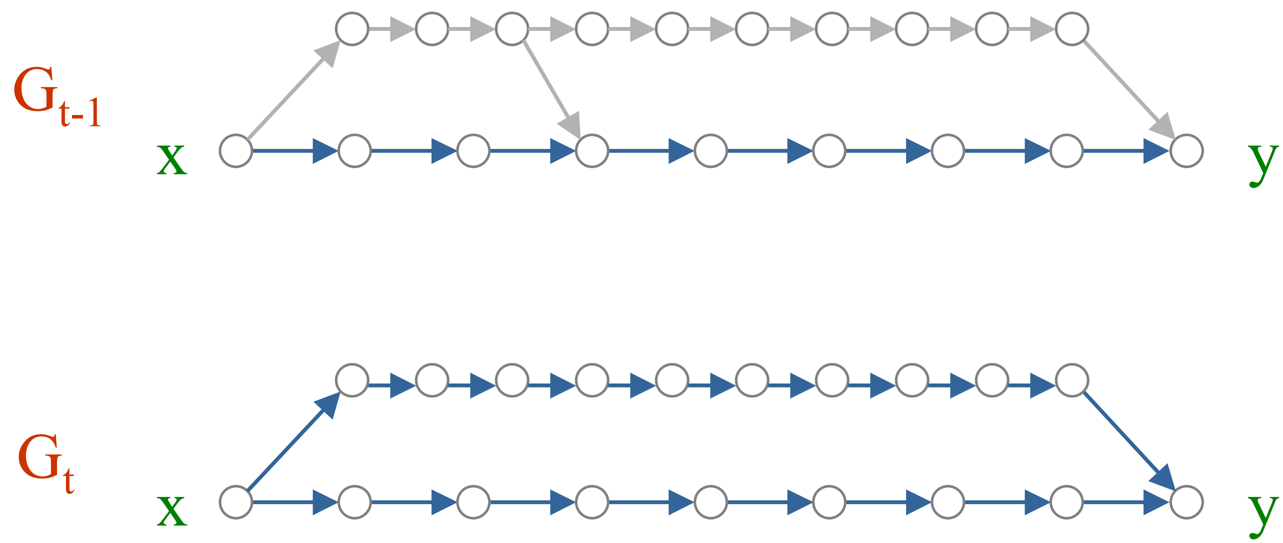
This is a consequence of vertex-disjointess…

Dynamic graphs

We call dynamic graph a sequence of graphs $<G_0,\ldots,G_k>$ such that, for any t, G_{t-1} and G_t differ in the weight of exactly one edge

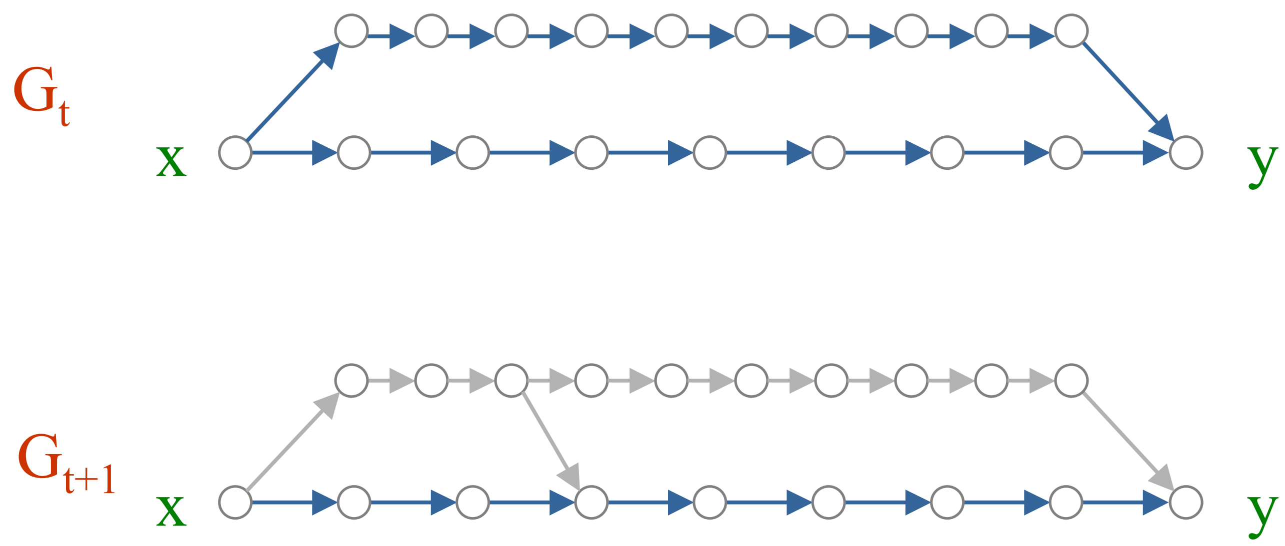
Uniform paths in dynamic graphs

A uniform path in G_t is appearing if it is not uniform in G_{t-1}



Uniform paths in dynamic graphs

A uniform path in G_t is disappearing if it is not uniform in G_{t+1}



Uniform paths in dynamic graphs

Theorem IV

At most n uniform paths can appear after an increase

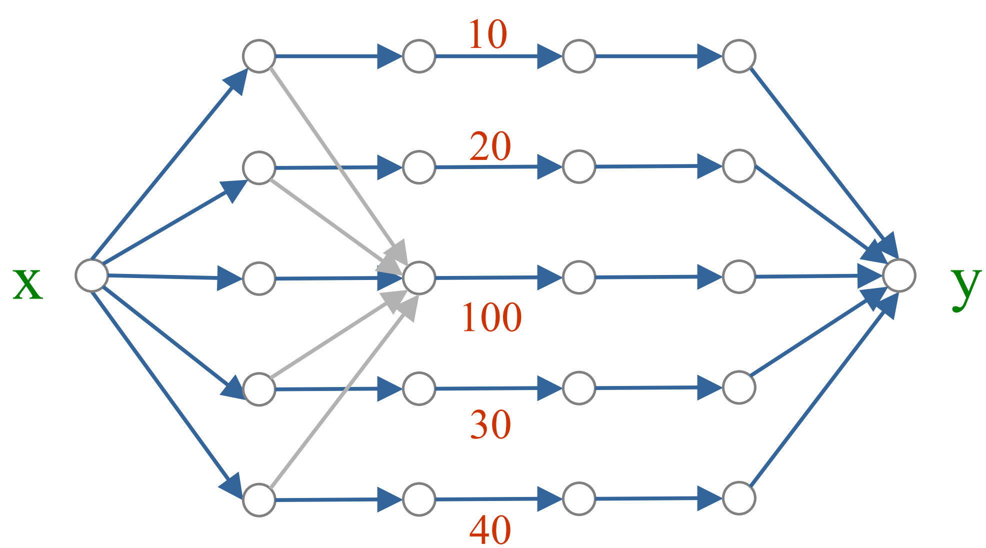
At most 1 uniform path can disappear after an increase

Corollary

The amortized number of appearing uniform paths per update in an increase-only sequence is O(1)

Uniform paths in dynamic graphs

What about fully dynamic sequences?



The weights of uniform paths that disappear and then reappear do not change…

Uniform paths in dynamic graphs

Theorem V

For any pair (x,y), the amortized number of uniform paths π_{xy} appearing with a new weight per update in a fully dynamic sequence is O(log n)

We conjecture that the bound is O(1)…

Sketch of proof of Theorem VI (1/3)

Construct the *graph of uniform paths* H defined as:

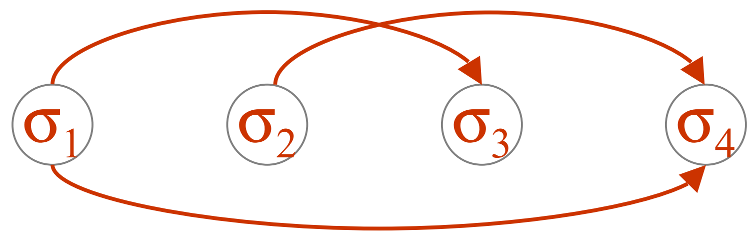
Vertices of H are operations σ_t in the dynamic graph

Each edge (σ_1, σ_2) in H corresponds to a path π_{xy} in the dynamic graph such that:

1	π_{xy} contains both edges updated by σ_1 and σ_2
2	π_{xy} is uniform at least once between σ_1 and σ_2
3	π_{xy} is not touched between σ_1 and σ_2

Sketch of proof of Theorem VI (2/3)

H cannot contain the following forbidden subgraph:

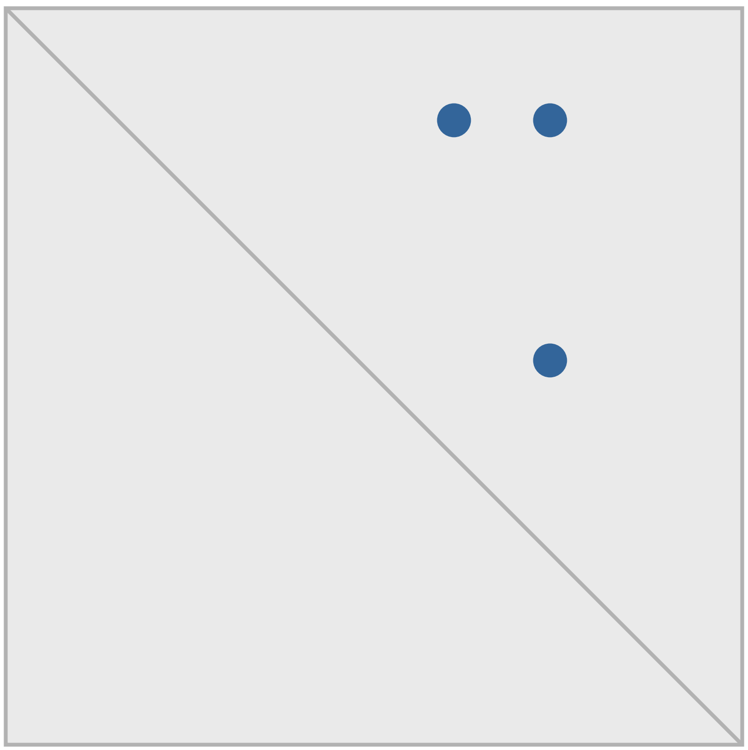


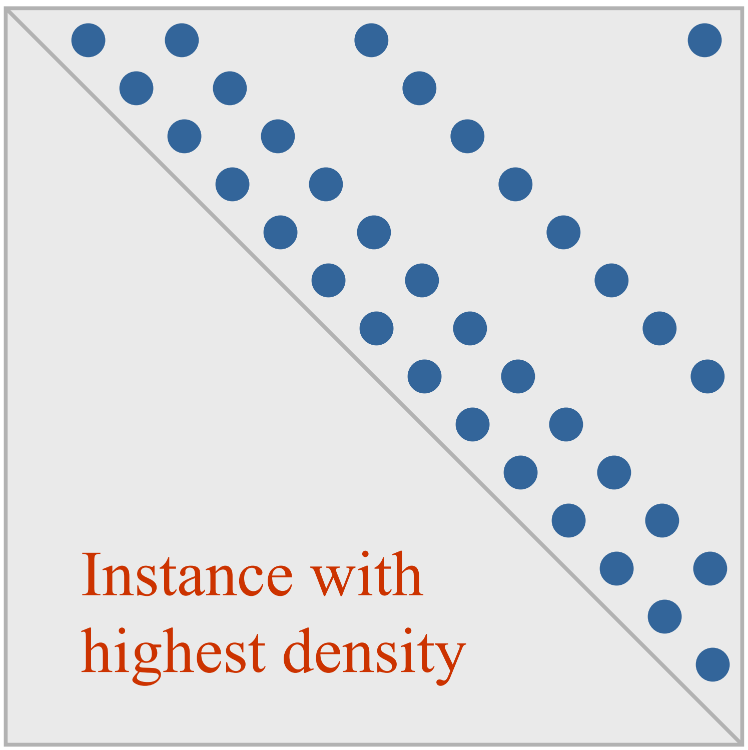
Where: $\sigma_1 < \sigma_2 < \sigma_3 < \sigma_4$

This implies that the *graph of uniform paths* H has average degree log(k), where k is the number of vertices of H

Sketch of proof of Theorem VI (3/3)

Map each edge (σ_1, σ_2) of H into a point $p=(\sigma_1, \sigma_2)$ in the square $k \times k$

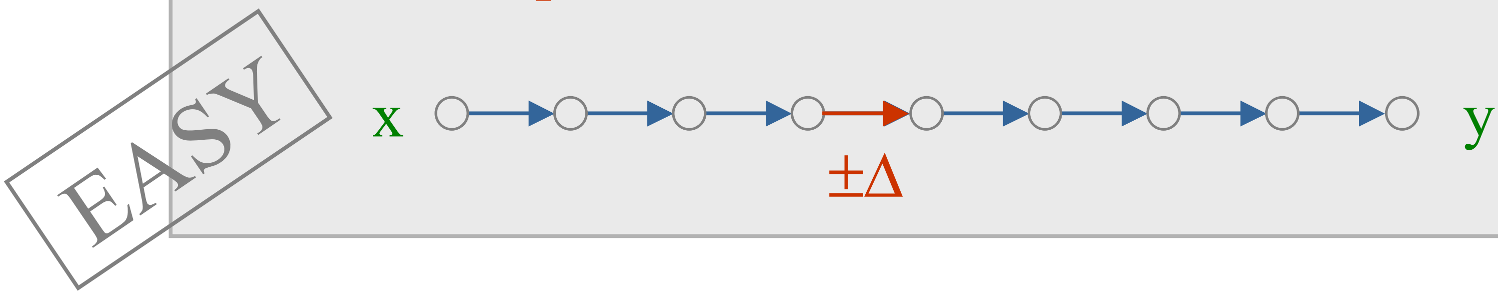




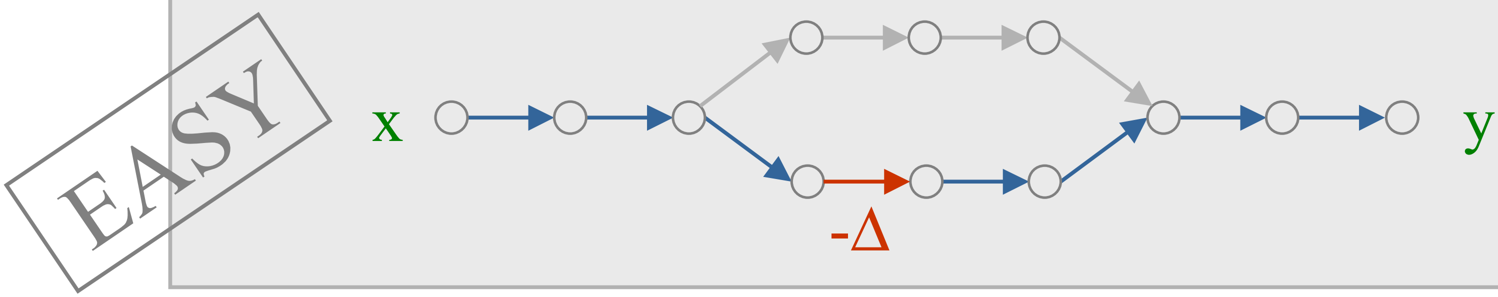
Shortest paths and edge weight updates

How does a shortest path change after an update?

The shortest path is the same, but has different weight:

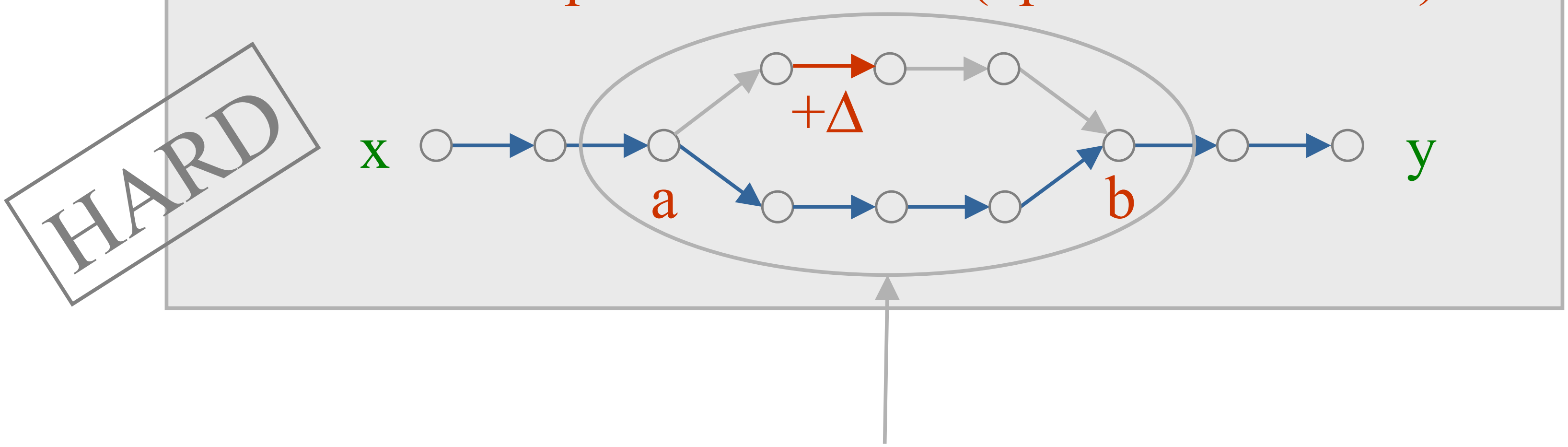


The shortest path is different (update = decrease):



Shortest paths and edge weight updates

The shortest path is different (update = increase):



If we look closer, we realize that the new shortest path from a to b was already uniform before the update!

A new approach to dynamic APSP

Main idea:

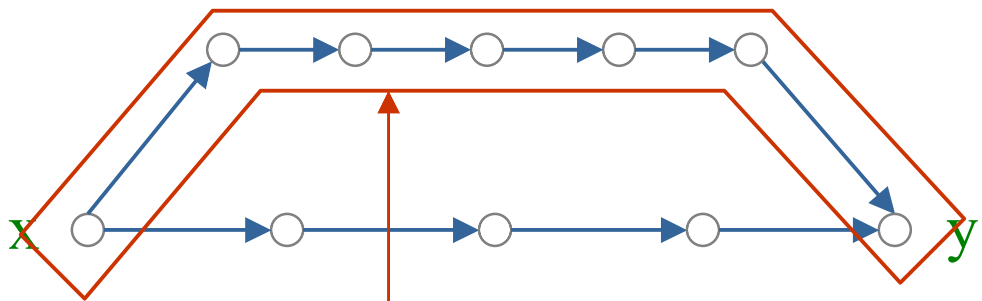
For each pair x,y, maintain in a data structure uniform paths connecting x to y

The combinatorial properties of uniform paths imply that only a small piece of information needs to be updated at each time…

Are we done?

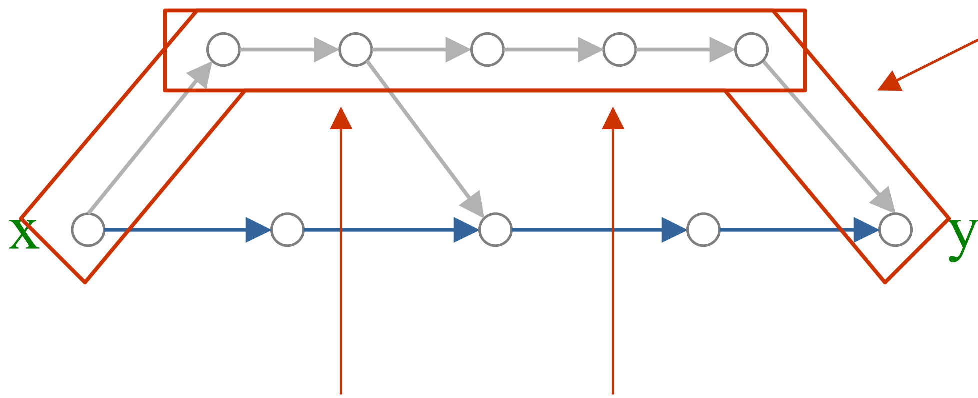
NO

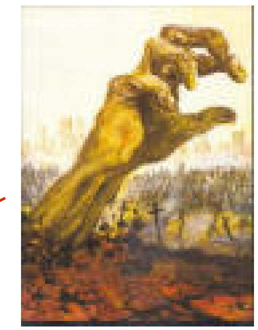
How to pay only once?



This path stays the same while flipping
between uniform and non-uniform:
We would like to have an update algorithm
that pays only once for it
over the whole sequence...

Looking at the substructure





may
resurrect!

This path is no longer a shortest path
after the insertion…

This path remains a shortest path
after the insertion…

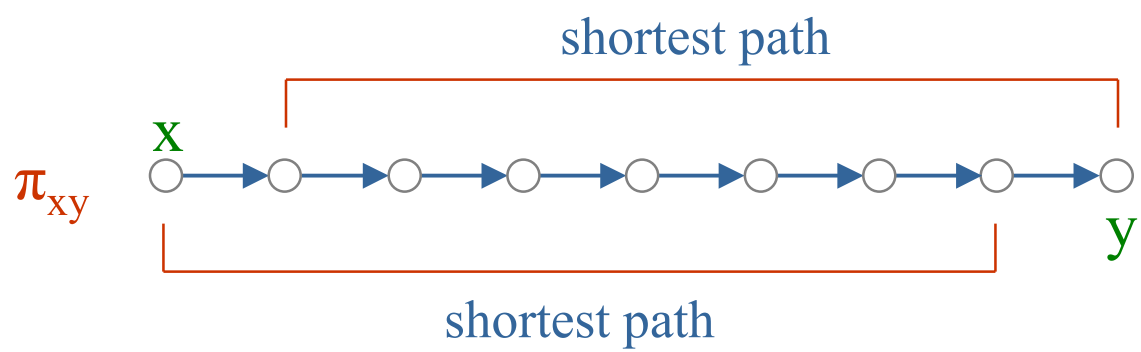
…but if we removed the edge
it would get a shortest path again!

Zombies

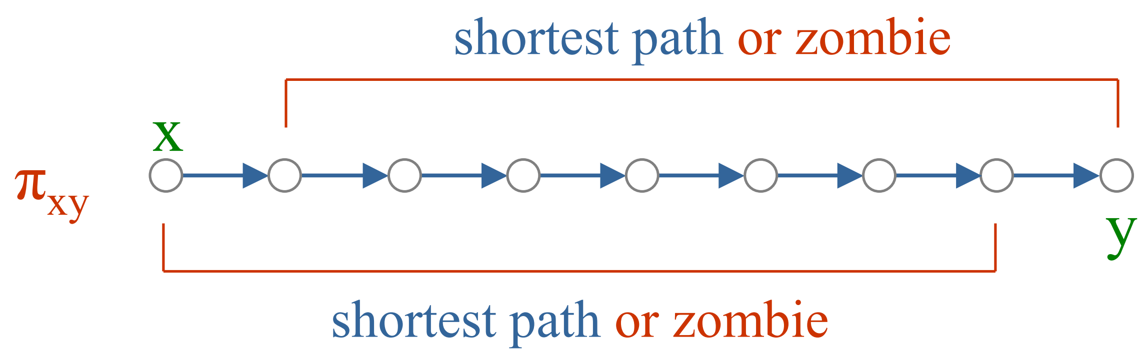
A path is a **zombie** if it used to be a shortest path, and its edges have not been updated since then

Potentially uniform paths

Uniform path



Potentially uniform path

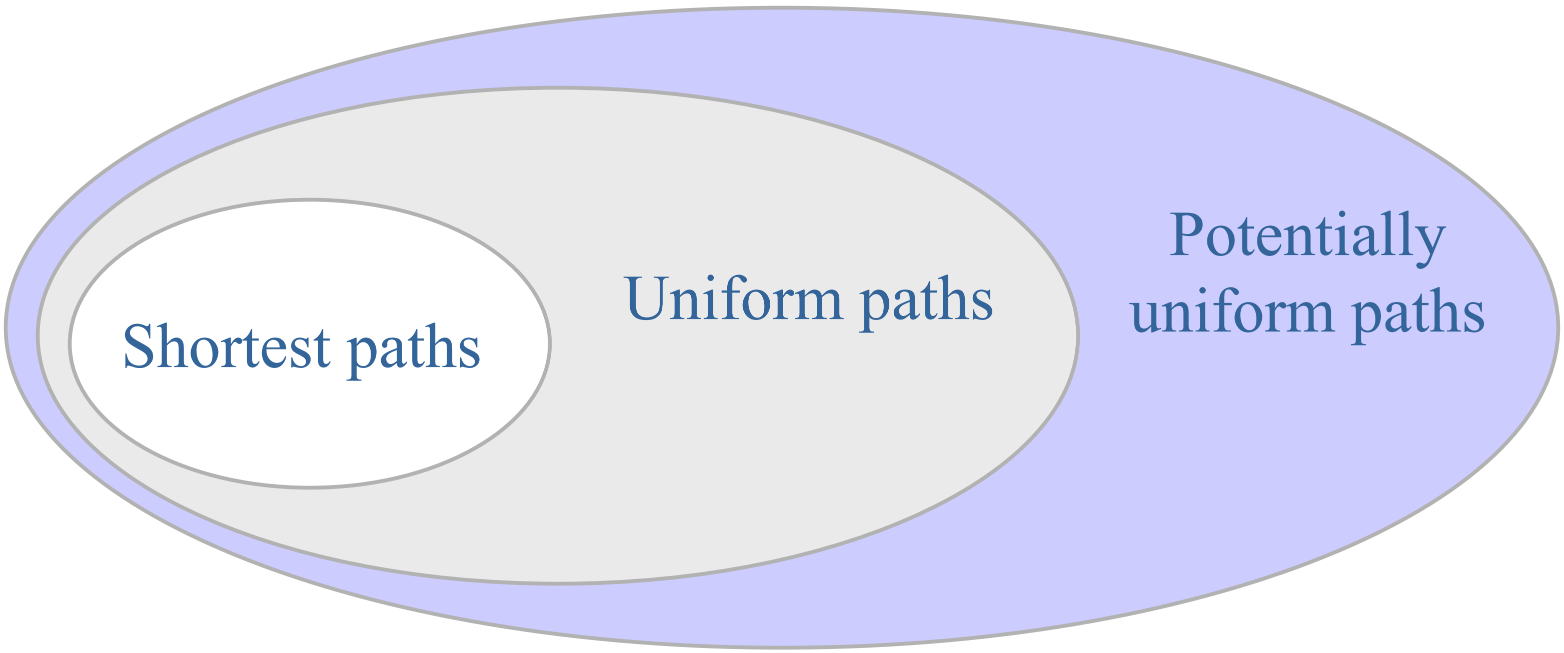


Relaxed notion of uniformity:
Subpaths do not need to be shortest at the same time

Properties of potentially uniform paths

Theorem I

Uniform paths $\subseteq$ Potentially uniform paths



Properties of potentially uniform paths

Theorem II

O(zn^2) zombies at any time

⇓

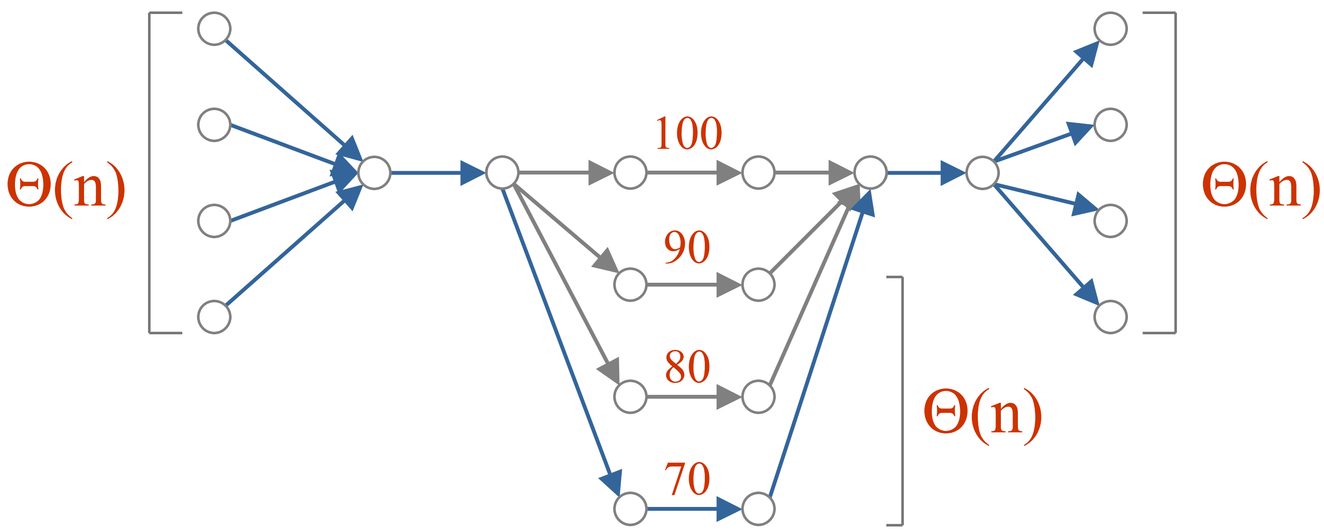
O(zn^2) new potentially uniform paths per update

How many zombies can we have?

A lot!

We can construct a dynamic graph with

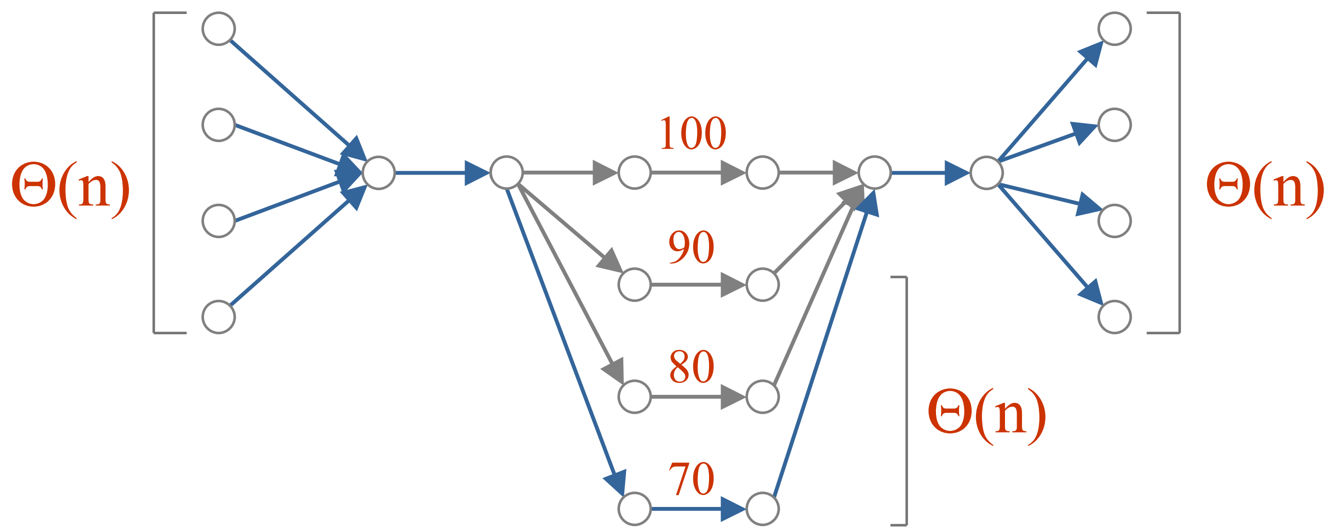
$\Theta(n^3)$ zombies at any time, amortized.



$$\# \text{ zombies} = \underbrace{\Theta(n^2) + \Theta(n^2) + \Theta(n^2)}_{\Theta(n)} = \Theta(n^3)$$

Reducing # of zombies: Smoothing

At each update we pick an edge with the maximum number of zombies passing through it, and we remove and reinsert it



zombies = 0 (in general, $O(n^2)$)

A new approach to dynamic APSP (II)

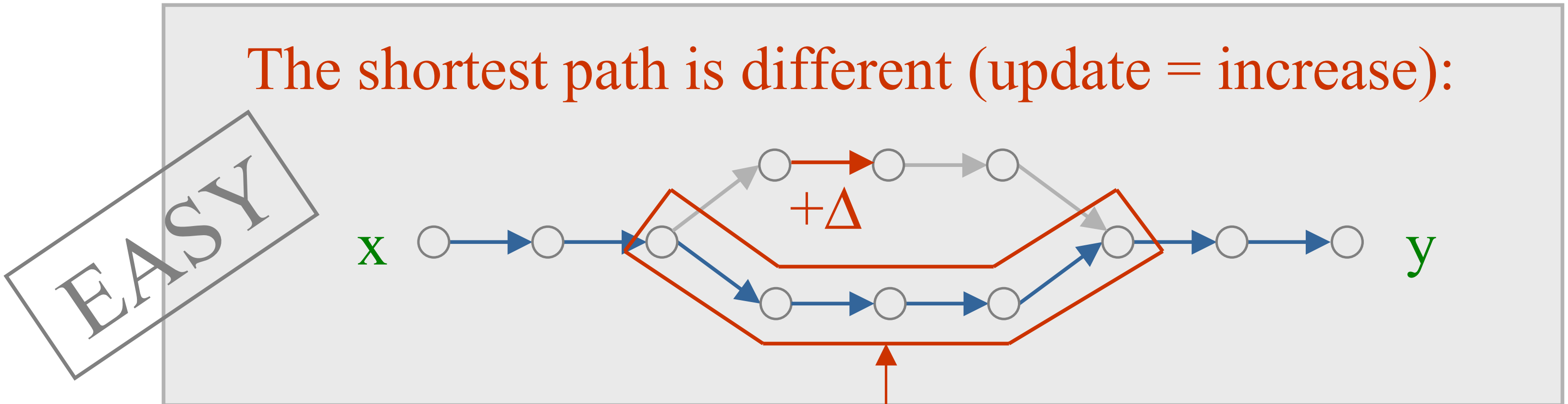
Main idea:

For each pair x,y, maintain in a data structure the **potentially** uniform paths connecting x to y

The combinatorial properties of potentially uniform paths imply that, if we do **smoothing**, we have only $O(n^2)$ new potentially uniform paths per update, amortized…

There exists and update algorithm that spends $O(\log n)$ time per potentially uniform path

Handling the hard case



If we maintain potentially uniform paths
using priority queues,
we can find this path in O(1) time!

The update algorithm

1 Remove from the data structure all potentially uniform paths containing the updated edge

2 Use remaining potentially uniform paths to find an upper bound to the distances after the update

3 Propagate changes in waves from the updated edge, finding the new potentially uniform paths and the new distances

Conclusions

Uniform paths are the heart of dynamic shortest paths

Combinatorially well-behaved in dynamic graphs

New approach based on maintaining
potentially uniform paths

Solves the problem in its full generality

Applications to Railway Optimization Problems?