Additive, Near-Additive, and Multiplicative Approximations for APSP in Weighted Undirected Graphs: Trade-offs and Algorithms

Liam Roditty, Bar-Ilan University, Israel

Ariel Sapir, Bar-Ilan University, Israel

Summer Seminar on AGT 2025, Ben-Gurion University, Israel

Plan of Talk

- APSP and APASP
- Additive APASP: Weighted and Unweighted
- Hitting Sets
- Additive $+2W_1$-APASP
- Additive $+2\sum_{i=1}^{k+1} W_i$-APASP
- Additional Results
- Further Directions

Distances in Graphs

$G = (V, E, w)$ weighted undirected graph

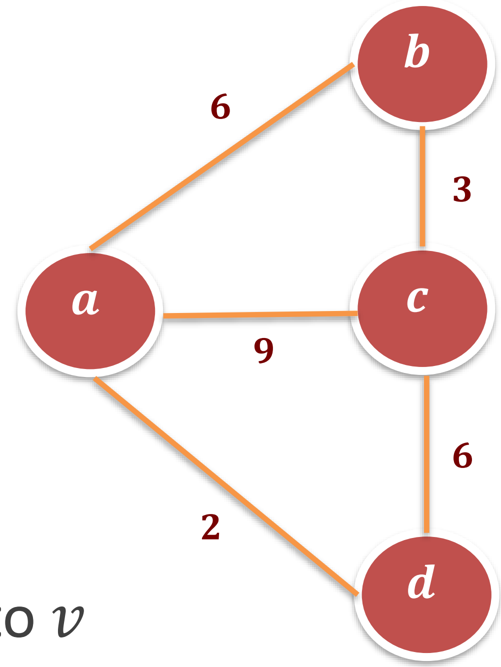
How do we define a distance?

For a path P: $w(P) = \sum_{e \in P} w(e)$

Let $u, v \in V$

Distance: $\delta(u, v) = \min_P w(P)$, over all P from u to v

<u>For unweighted graphs</u>: $w(P) =$ the number of edges in P (assume $w(e) = 1$)



$w(\langle a, b, c \rangle) = 9$

$\delta(a, c) = 8$

Problem(s) Definition

Common Input: $G = (V, E, w)$ weighted undirected graph.

Several problems:

	Single Source Shortest Paths (SSSP)	**Multi Source Shortest Paths (MSSP)**	**All Pairs Shortest Paths (APSP)**
Additional Input:	A single source $s \in V$	A subset of sources $S \subseteq V$	None ($S = V$)
Output:	Distances from s to all $v \in V$	Distances from any $s \in S$ to any $v \in V$	Distances from all $u \in V$ to all $v \in V$

Our focus: APSP, the others – utilized as a tool

APSP Conjecture

$|V| = n, |E| = m.$

How fast can we compute SSSP?

- Dijkstra (1956): $O(m + n \cdot \log n)$

How fast can we compute APSP?

- Floyd-Warshall (1962): $O(n^3)$
- Johnson (1977): $O(nm + n^2 \cdot \log n)$
- ...
- Williams (2014): $O\left(\frac{n^3}{2^{\sqrt{\Omega(\log n)}}}\right)$

None strictly better than n^3!

APSP Conjecture

Question 1: *Is there an $\varepsilon > 0$ for which APSP can be computed in $\tilde{O}(n^{3-\varepsilon})$?*

APSP Conjecture: There exists no such ε!

Question 2: *Can **A**ll **P**airs **A**pproximated **S**hortest **P**aths (APASP) be computed faster than n^3?*

Short Answer: Yes! Many approximations in $\tilde{O}(n^{3-\varepsilon})$

How do we define an approximation?

All-Pairs Approximate Shortest Paths

<ins>For example</ins>: $\delta(a,c) = 8,$

$\delta(b,d) = 8.$

Estimated distance: $d[u,v]$

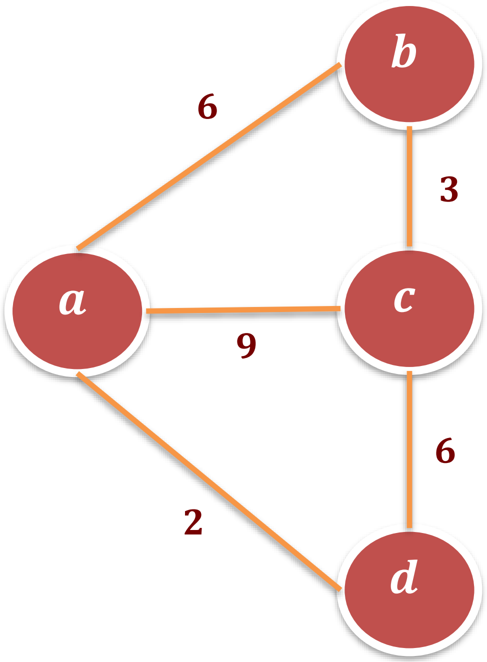
(α,β)-APASP: $d[u,v] \in [\delta(u,v), \alpha \cdot \delta(u,v) + \beta]$

$d[u,v] = w(P)$, for some P between u and v

<ins>For example</ins>: $\alpha = 1, \beta = 1 \Rightarrow$ (1,1)-APASP

$d[a,c] = 9,$

$d[b,d] = 8.$



Major Approximation Categories

(α, β)-APASP: $d[u, v] \in [\delta(u, v), \alpha \cdot \delta(u, v) + \beta]$

Multiplicative α-APASP: $\beta = 0$

Additive $+\beta$-APASP: $\alpha = 1$

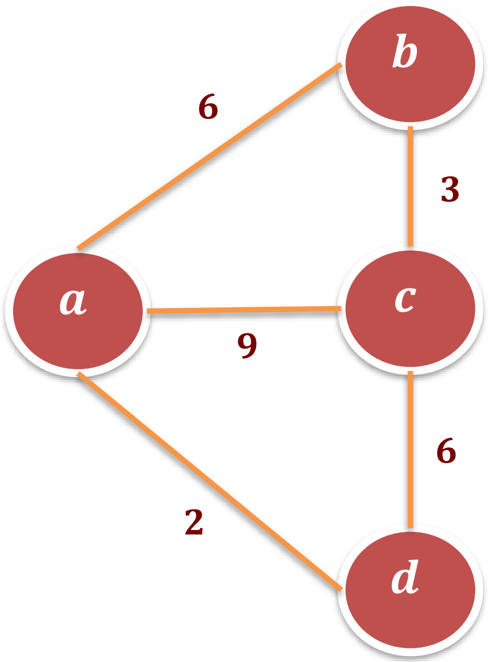
For small $\varepsilon > 0$: Nearly-Additive $(1 + \varepsilon, \beta)$-APASP

Which is better?

Our Setting

Directed?	*Undirected?*
Unweighted?	*Weighted?*
Negative Weights?	*Non-negative weights?*

Our focus: ⇑



Plan of Talk

- APSP and APASP
- Additive APASP: Weighted and Unweighted
- Hitting Sets
- Additive $+2W_1$-APASP
- Additive $+2\sum_{i=1}^{k+1} W_i$-APASP
- Additional Results
- Further Directions

Known Additive APASP for Unweighted

Dor, Halperin and Zwick (1996): +2-APASP

$$d[u,v] \in [\delta(u,v), \delta(u,v)+2]$$

Two algorithms: For dense graphs with $\tilde{O}(n^{\frac{7}{3}})$ runtime

For sparse graphs with $\tilde{O}(n^{\frac{3}{2}}m^{\frac{1}{2}})$ runtime

In total: $\tilde{O}(\min\{n^{\frac{7}{3}}, n^{\frac{3}{2}}m^{\frac{1}{2}}\})$ runtime

Strictly less than n^3

What Can We Do for Weighted Graphs?

Observation: Weighted graphs ⇒ Weights can be scaled

Multiply all weights by any $c \in \mathbb{R}^+$: $w'(u,v) = c \cdot w(u,v)$

Shortest paths will remain shortest path

The distance $\delta'(u,v) = c \cdot \delta(u,v)$

We may assume $\forall_{e \in E}: w(e) \geq 1$

Question 3: *Can a weighted $+\beta$-APASP have a constant β?*

For example: $+2$-APSP? $+4$-APASP?

Short Answer: Yes, but it is equivalent to exact APSP.

What Can We Do for Weighted Graphs?

Question 3: *Can a weighted $+\beta$-APASP have a constant β?*

Short Answer: Yes, but it is equivalent to exact APSP.

Scale the weights: *What if $c = \beta + \varepsilon$?*

$$d'[u,v] \in [\delta'(u,v), \delta'(u,v) + \beta]$$

$$\Downarrow$$

$$w(e) \geq \beta + \varepsilon$$

$$\Downarrow$$

$$d'[u,v] = \delta'(u,v)$$

Exact APSP: $d[u,v] = \frac{d'[u,v]}{c} = \frac{\delta'(u,v)}{c} = \delta(u,v)$

What Can We Do for Weighted Graphs?

β can depend somehow on $w: E \to \mathbb{R}$

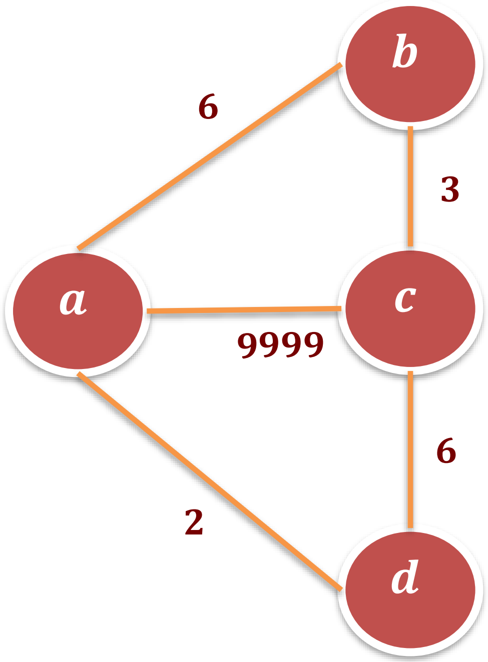
For example: $W_{\max} = \max w(e)$

Unweighted: $+2$-APASP

Weighted: $+2W_{\max}$-APASP

For example: $d[a,c] \in [8,26]$ / $[8,20006]$

Is it a "good" guarantee?



What Can We Do for Weighted Graphs?

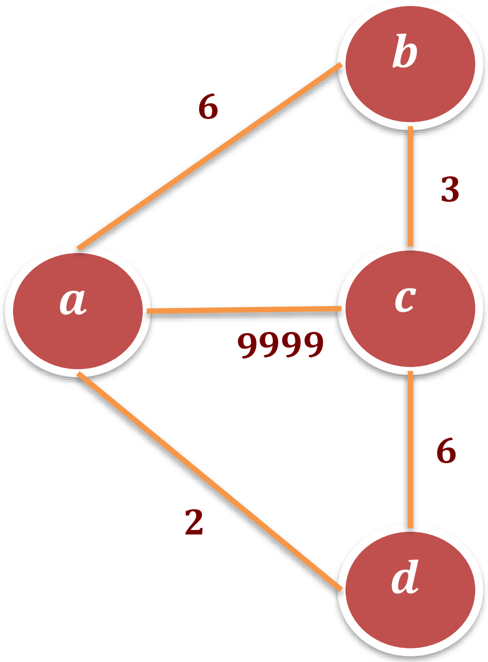
Better definition?

Let u~~v be shortest path between u and v

$W_i(u{\sim\sim}v)$ is the weight of the i^{th} heaviest edge

For example: $W_1(a{\sim\sim}c) = 6,$

$W_2(b{\sim\sim}d) = 2.$



What Can We Do for Weighted Graphs?

$+f(W_1, \ldots, W_k)$-APASP:

$$d[u,v] \le w(P) + f\big(W_1(P), \ldots, W_k(P)\big)$$

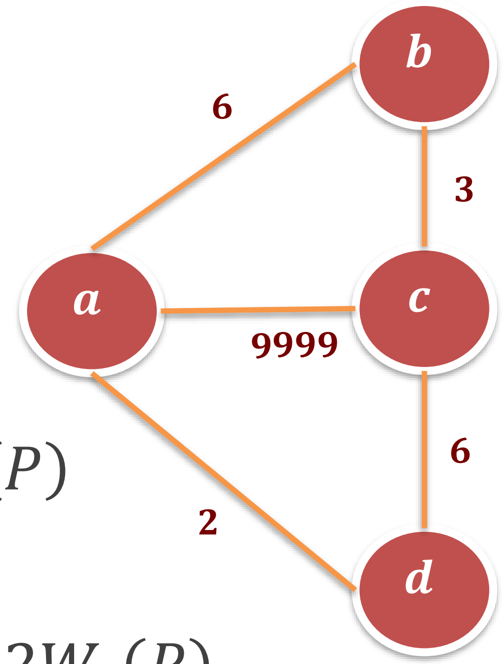
Over all shortest paths P between u and v

For example: $+2W_1$-APASP:

$$d[u,v] \le w(P) + 2W_1(P)$$

$+2W_1 + 2W_2$-APASP:

$$d[u,v] \le w(P) + 2W_1(P) + 2W_2(P)$$



The guarantee for $d[u,v]$ is "local" and not "global"

An Additive APASP With a "Local" Guarantee

Cohen and Zwick (1997): $+2W_1$-APASP

$\delta(u,v) \leq d[u,v] \leq w(P) + 2W_1(P)$

Two algorithms: For dense graphs with $\tilde{O}(n^{\frac{7}{3}})$ runtime

For sparse graphs with $\tilde{O}(n^{\frac{3}{2}}m^{\frac{1}{2}})$ runtime

In total: $\tilde{O}(\min\{n^{\frac{7}{3}}, n^{\frac{3}{2}}m^{\frac{1}{2}}\})$ runtime

The same runtime as the unweighted setting!

Discussion: Commensurate

(α, β)-APASP for unweighted

$$d[u, v] \leq \delta(u, v) + \beta$$

$$d[u, v] \leq w(P) + \beta$$

Over all shortest paths P between u and v

A weighted version of this?

Recall $G = (V, E, w)$ and let $f(\beta, G, P)$ be a function

Consider an $\left(\alpha, f(\beta, G, P)\right)$-APASP for weighted

$$d[u, v] \leq w(P) + f(\beta, G, P)$$

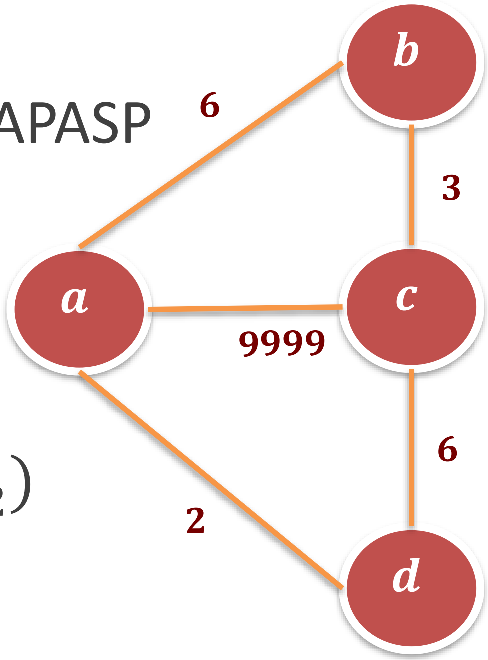
Discussion: Commensurate

Unweighted: $d[u, v] \leq w(P) + \beta$

Weighted: $d[u, v] \leq w(P) + f(\beta, G, P)$

If: when $\forall_{e \in E}: w(e) = 1 \Rightarrow f(\beta, G, P) = \beta$

Then: $(\alpha, f(\beta, G, P))$-APASP is a *Commensurate Version* of (α, β)-APASP



<u>Examples:</u>

$+2$ and $+2W_{\max}$	$f(\beta, G, P) = \beta W_{\max}$
$+2$ and $+2W_1$	$f(\beta, G, P) = \beta W_1$
$+2$ and $+W_1 + W_2$	$f(\beta, G, P) = \frac{\beta}{2}(W_1 + W_2)$

Discussion: Strongly Commensurate

Problems can be commensurate

What if their algorithms are not of the same "hardness"?

We need to consider the runtimes

$\mathcal{A}_1$ algorithm for unweighted (α, β)-APASP with a runtime $T(n)$

$\mathcal{A}_2$ algorithm for a commensurate $\big(\alpha, f(\beta, G, P)\big)$-APASP

Discussion: Strongly Commensurate

If: the runtime of $\mathcal{A}_2$ is $\tilde{O}(T(n) \cdot (\log W_{\max})^c)$ for some $c \in \mathbb{R}^+$

Then: $\mathcal{A}_2$ is a *Strongly Commensurate Version* of $\mathcal{A}_1$

Question 3: *What are the strongly commensurate versions of an* (α, β)*-APASP algorithm for some* α, β*?*

Partial Answer: $+2$-APASP algorithms of DHZ and $+2W_1$-APASP algorithms of CZ

Extended Additive APASP for Unweighted

Dor, Halperin and Zwick (1996): two $+2 \cdot (k+1)$-APASP

$$d[u,v] \leq \delta(u,v) + 2 \cdot (k+1)$$

<u>Two algorithms</u>: For dense graphs with $\tilde{O}(n^{2+\frac{1}{3k+2}})$ runtime

For sparse graphs with $\tilde{O}(n^{2-\frac{1}{k+2}} m^{\frac{1}{k+2}})$ runtime

In total: $\tilde{O}(\min\{n^{2+\frac{1}{3k+2}}, n^{2-\frac{1}{k+2}} m^{\frac{1}{k+2}}\})$ runtime

Naïve Strongly Commensurate Versions

For unweighted graphs: $+2 \cdot (k+1)$-APASP

The same algorithm: $+2 \cdot (k+1) \cdot W_{\max}$-APASP

No change in the algorithm, same runtime

Several changes:$+2 \cdot (k+1) \cdot W_1$-APASP

The runtime of both algorithms remains the same

Question 3: *What are the strongly commensurate versions of an (α, β)-APASP algorithm for some α, β?*

Naïve Strongly Commensurate Versions

Additional Answer: $+2 \cdot (k+1)$-APASP algorithms of DHZ and "similar" $+2 \cdot (k+1) \cdot W_{\max}$-APASP algorithms or $+2 \cdot (k+1) \cdot W_1$-APASP algorithms

Is it possible to do "better"?

Are there tighter weighted APASP algorithms which are strongly commensurate versions of the $+2 \cdot (k+1)$*-APASP algorithms of DHZ?*

An Additive APASP With a “Local” Guarantee

Cohen and Zwick (1997): $+2\sum_{i=1}^{k+1} W_i$-APASP

$$d[u,v] \leq w(P)+2\sum_{i=1}^{k+1} W_i(P)$$

Over all shortest paths P between u and v

When $\forall_{e\in E}: w(e) = 1$ then $+2\sum_{i=1}^{k+1} W_i = +2 \cdot (k+1)$

Observation: $+2\sum_{i=1}^{k+1} W_i$-APASP is a commensurate version of $+2 \cdot (k+1)$-APASP

An Additive APASP With a “Local” Guarantee

Are there strongly commensurate algorithms for these problems?

Only a single algorithm: For sparse graphs with $\tilde{O}(n^{2-\frac{1}{k+2}}m^{\frac{1}{k+2}})$ runtime

Nothing for dense graphs ☹

We present: $+2\sum_{i=1}^{k+1} W_i$-APASP algorithm for dense graphs with $\tilde{O}(n^{2+\frac{1}{3k+2}})$ runtime

Question 3: *What are the strongly commensurate versions of an (α, β)-APASP algorithm for some α, β?*

Additional answer to Q3: $+2\sum_{i=1}^{k+1} W_i$-APASP and $+2 \cdot (k+1)$-APASP

Additive: Unweighted vs Weighted

DHZ96: Dor, Halperin and Zwick (1996)

CZ97: Cohen and Zwick (1997)

RS25: Roditty and Sapir (2025)

Unweighted	Runtime	Ref	Weighted	Runtime	Ref
$+2$	$n^{\frac{3}{2}}m^{\frac{1}{2}}$	DHZ96	$+2W_1$	$n^{\frac{3}{2}}m^{\frac{1}{2}}$	CZ97
$+2$	$n^{\frac{7}{3}}$	DHZ96	$+2W_1$	$n^{\frac{7}{3}}$	CZ97
$+2\cdot(k+1)$	$n^{2-\frac{1}{k+2}}m^{\frac{1}{k+2}}$	DHZ96	$+2\sum_{i=1}^{k+1} W_i$	$n^{2-\frac{1}{k+2}}m^{\frac{1}{k+2}}$	CZ97
$+2\cdot(k+1)$	$n^{2+\frac{1}{3k+2}}$	DHZ96	$+2\sum_{i=1}^{k+1} W_i$	$n^{2+\frac{1}{3k+2}}$	RS25

Plan of Talk

- APSP and APASP
- Additive APASP: Weighted and Unweighted
- Hitting Sets
- Additive $+2W_1$-APASP
- Additive $+2\sum_{i=1}^{k+1} W_i$-APASP
- Additional Results
- Further Directions

Hitting Sets

A universe of elements $\mathcal{U} = \{u_1, \dots, u_n\}$

A collection of subsets: $T_1, T_2, \dots, T_\ell$

$T_i \subseteq \mathcal{U}$

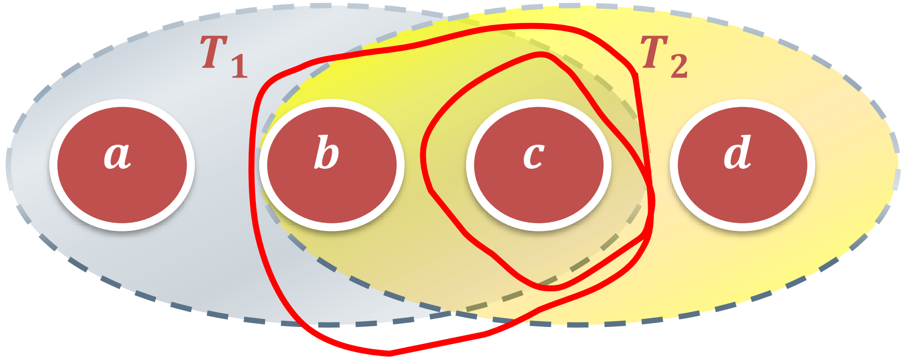
A *hitting set* is a set $S \subseteq \mathcal{U}$ s.t. $S \cap T_i \neq \emptyset$ for all $1 \leq i \leq \ell$

For example: $\mathcal{U} = \{a, b, c, d\}$

$T_1 = \{a, b, c\},\ T_2 = \{b, c, d\}$

$S = \{b, c\}$ is a hitting set

$S = \{c\}$ is a hitting set



Hitting Sets

How fast can we compute a hitting set?

Finding the smallest hitting set is NP-Hard!

Our usage: $|T_i| \geq r, \quad \ell = n$

Aingworth, Chekuri, Indyk and Motwani (1996):

Lemma 1: A hitting set S of size $|S| \in \tilde{O}\left(\frac{n}{r}\right)$ can be computed in $\tilde{O}(nr)$ runtime.

Hitting Sets for Graphs?

How do hitting sets relate to a graph $G = (V, E)$*?*

Let $\mathcal{U} = V$

$T_v = \Gamma(v) =$ neighbours of v

Focus on high-degree vertices

$\deg v \geq n^{\alpha}$ for some $\alpha \in (0,1)$

$|S| \in \tilde{O}\left(\frac{n}{n^{\alpha}}\right) = \tilde{O}(n^{1-\alpha})$

Pivots

For each $v \in V: \Gamma(v) \cap S \neq \emptyset$

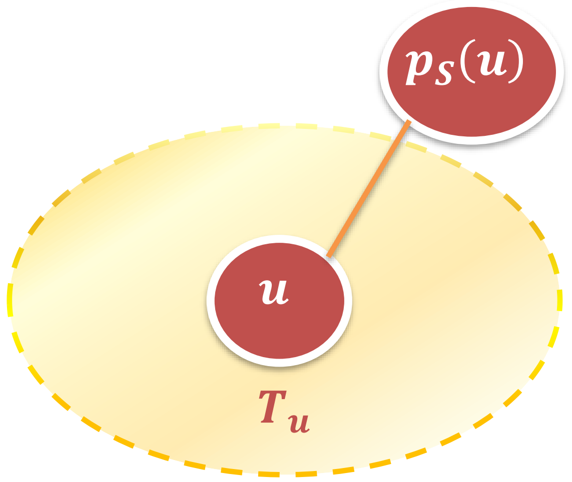
There exists a vertex in $\Gamma(v) \cap S$

Let $p_S(u)$ be the nearest to u (by distance)

$p_S(u)$ is the *pivot* of u, relatively to S

$H = \{(u, p_S(u)) \mid u \in V\}$

$|H| \in O(n)$



Hitting Sets for APASP?

How do hitting sets relate to APASP?

<u>One way to compute APSP</u>: Invoke SSSP from all $u \in V$

Yields precise distances (=APSP)

What is the issue?

$|V|$ iterations of SSSP $\Rightarrow \tilde{O}(nm)$ runtime

What if invoke SSSP only from a subset $S \subseteq V$ *of vertices?*

The runtime: $\tilde{O}(|S| \cdot m)$

Hitting Sets for APASP?

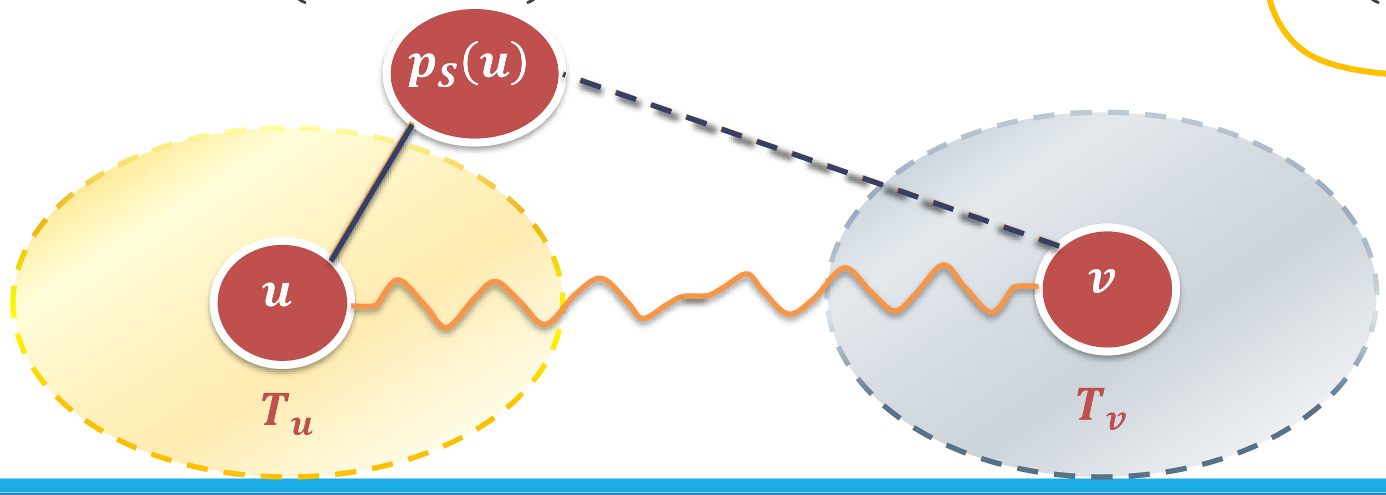
On a high scale, the approach for APASP: Each vertex considers its neighbours $\Gamma(u)$

Invoke SSSP from a hitting set $S \subseteq V$

For $u, v \in V$: Estimate the distance through pivots

By the triangle inequality: $\delta(p_S(u), v) \leq \delta\big(u, p_S(u)\big) + \delta(u, v)$

$$d[u,v] \leq \delta\big(u, p_S(u)\big) + \delta(p_S(u), v) \leq \delta(u,v) + 2\delta\big(u, p_S(u)\big)$$



Plan of Talk

- APSP and APASP
- Additive APASP: Weighted and Unweighted
- Hitting Sets
- Additive $+2W_1$-APASP
- Additive $+2\sum_{i=1}^{k+1} W_i$-APASP
- Additional Results
- Further Directions

Base for Our Approach

Cohen and Zwick (1997): $+2W_1$-APASP

Two algorithms: For dense graphs with $\tilde{O}(n^{\frac{7}{3}})$ runtime

For sparse graphs with $\tilde{O}(n^{\frac{3}{2}}m^{\frac{1}{2}})$ runtime

Cohen and Zwick (1997): $+2\sum_{i=1}^{k+1} W_i$-APASP

Only one: For sparse graphs with $\tilde{O}(n^{2-\frac{1}{k+2}}m^{\frac{1}{k+2}})$ runtime

Base for Our Approach

Our goal: Extend the $+2W_1$-APASP algorithm with $\tilde{O}(n^{\frac{7}{3}})$ runtime

$+2\sum_{i=1}^{k+1} W_i$-APASP algorithm with $\tilde{O}(n^{2+\frac{1}{3k+2}})$ runtime

Present a simplified version: $+2W_1$-APASP algorithm of Cohen and Zwick

Toolkit: hitting-sets, pivots, SSSP invocations over smaller sets of edges

Warmup: $+2W_1$-APASP

An undirected weighted graph $G = (V, E, w)$

$\Gamma(u, n^\beta)$ = n^β nearest neighbours of u, $\beta \in (0,1)$

Each vertex $u \in V$ considers $T_u = \Gamma(u, n^\beta)$

Find a hitting set S_1 for $\{\Gamma(u, n^\beta) \mid u \in V\}$

Lemma 1: A hitting set S of size $|S| \in \tilde{O}\left(\frac{n}{r}\right)$ can be computed in $\tilde{O}(nr)$ runtime.

In our case: $r = n^\beta$

$|S_1| \in \tilde{O}(n^{1-\beta})$

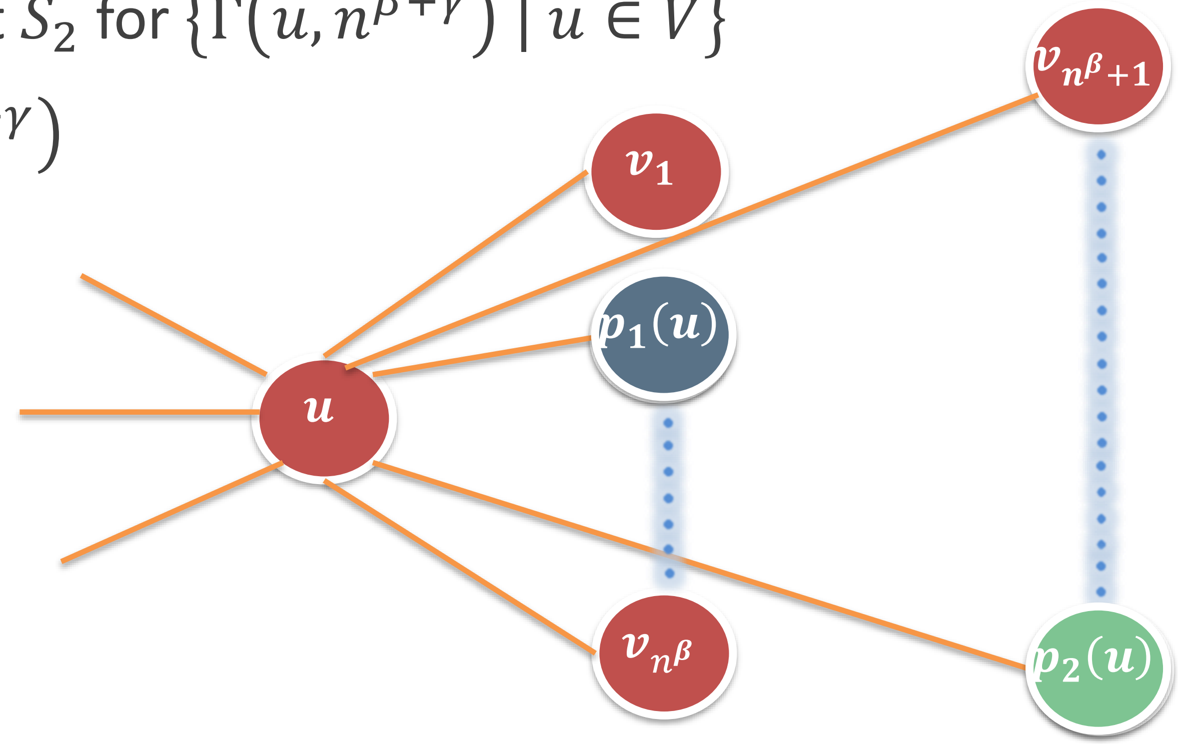
Warmup: $+2W_1$-APASP

$\Gamma(u, n^{\beta+\gamma})$ = $n^{\beta+\gamma}$ nearest neighbours of u, $\gamma \in (0,1)$

Consider again $T_u = \Gamma(u, n^{\beta+\gamma})$

Find a hitting set S_2 for $\{\Gamma(u, n^{\beta+\gamma}) \mid u \in V\}$

$|S_2| \in \tilde{O}(n^{1-\beta-\gamma})$



Warmup: $+2W_1$-APASP

Each vertex considers edges to nearest neighbours

$$E_1(u) = \{(u,v) \mid v \in \Gamma(u, n^{\beta})\}$$

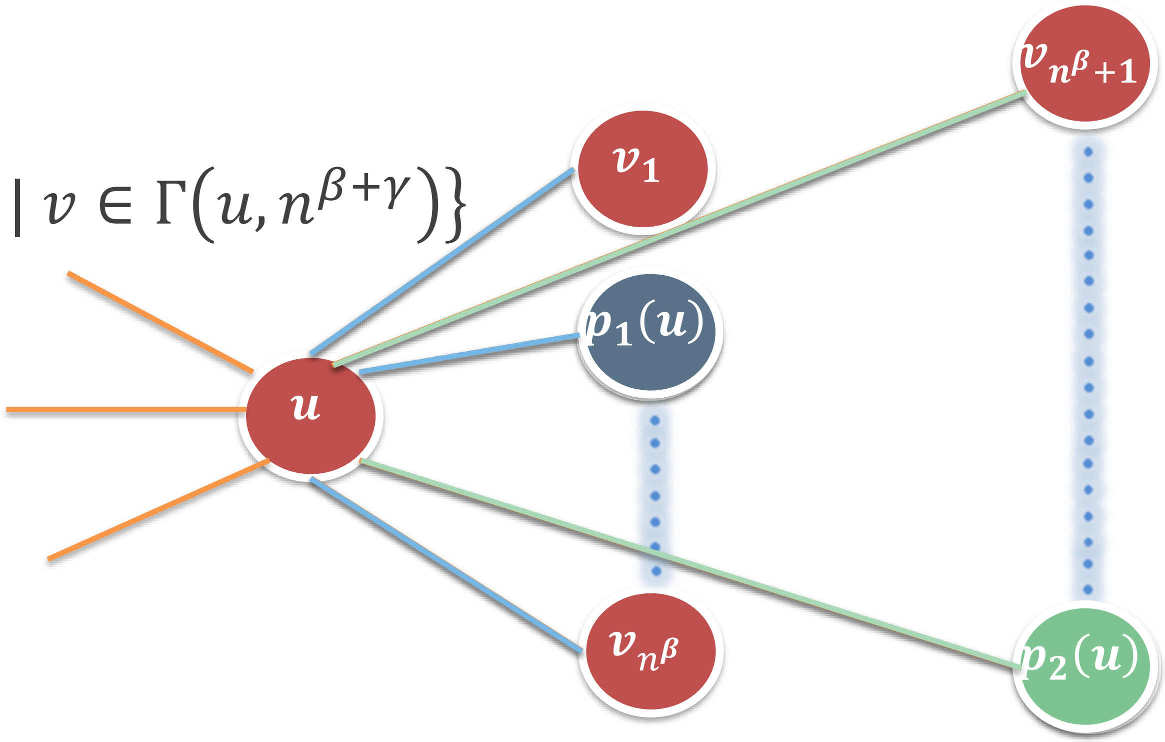
$$E_1 = \bigcup_{u \in V} E_1(u)$$

$$E_2(u) = \{(u,v) \mid v \in \Gamma(u, n^{\beta+\gamma})\}$$

$$E_2 = \bigcup_{u \in V} E_2(u)$$

$$|E_1| = n^{1+\beta}$$

$$|E_2| = n^{1+\beta+\gamma}$$



$+2W_1$-APASP Algorithm Overview

1. Find $p_1(u)$ (resp. $p_2(u)$) for every $u \in V$

$\tilde{O}(m)$

2. Set $d[u, p_1(u)] = \delta(u, p_1(u))$ (resp. $d[u, p_2(u)] = \delta(u, p_2(u))$)

3. For $s \in S_2$: Invoke SSSP over E and update d $\tilde{O}(|S_2| \cdot |E|) = \tilde{O}(mn^{1-\beta-\gamma})$

4. For $s \in S_1$: Invoke SSSP over E_2 and update d $\tilde{O}(|S_1| \cdot |E_2|) = \tilde{O}(n^{1-\beta} \cdot n^{1+\beta+\gamma}) = \tilde{O}(n^{2+\gamma})$

$\tilde{O}(|V| \cdot (|E_1| + |V| \cdot |S_2|)) = \tilde{O}\left(n \cdot (n^{1+\beta} + n \cdot n^{1-\beta-\gamma})\right) = \tilde{O}(n^{2+\beta} + n^{3-\beta-\gamma})$

5. For $u \in V$: Invoke SSSP over $E_1 \cup \{(u,v) | v \in V\} \cup (S_2 \times V) \cup H$ and update d

Total: $\tilde{O}(n^{2+\beta} + n^{2+\gamma} + n^{3-\beta-\gamma}) \Rightarrow \beta = \gamma = \frac{1}{3} \Rightarrow \tilde{O}(n^{\frac{7}{3}})$

$+2W_1$-APASP Algorithm Correctness

Let $u, v \in V$

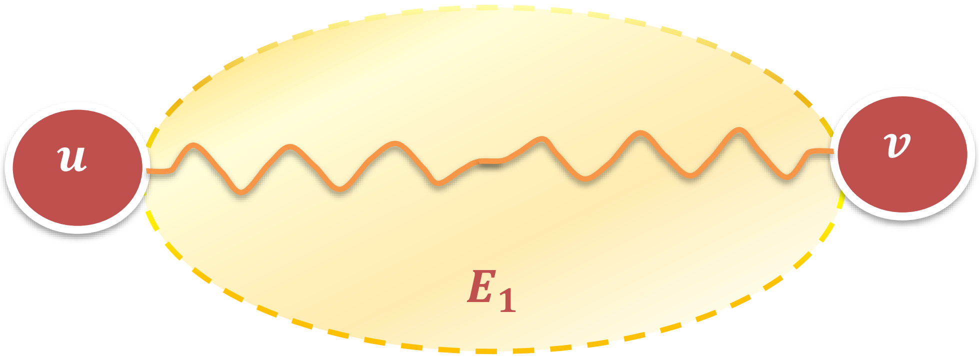
Our aim: $d[u, v] \in [\delta(u, v), \delta(u, v) + 2W_1]$

Distinguish between three possible cases:

1. $u \rightsquigarrow v \subseteq E_1$
2. $u \rightsquigarrow v \subseteq E_2$ yet $u \rightsquigarrow v \nsubseteq E_1$
3. $u \rightsquigarrow v \nsubseteq E_2$

Case 1

$$u \rightsquigarrow v \subseteq E_1$$



$+2W_1$-APASP Algorithm Overview

1. Find $p_1(u)$ (resp. $p_2(u)$) for every $u \in V$

2. Set $d[u, p_1(u)] = \delta\big(u, p_1(u)\big)$ (resp. $d[u, p_2(u)] = \delta\big(u, p_2(u)\big)$)

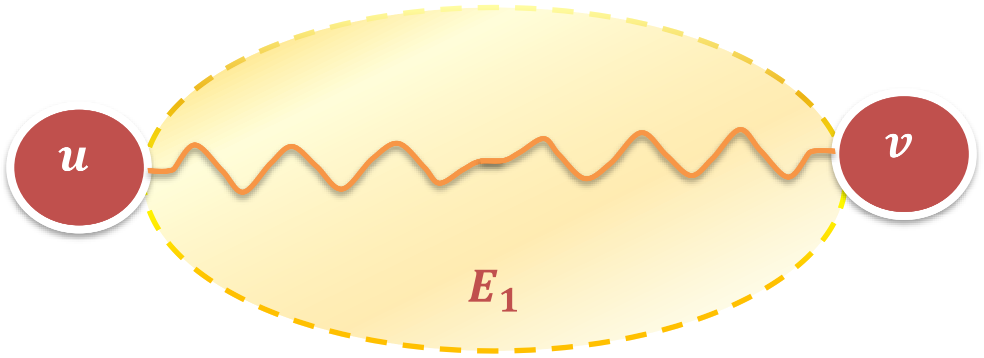
3. For $s \in S_2$: Invoke SSSP over E and update d

4. For $s \in S_1$: Invoke SSSP over E_2 and update d

5. For $u \in V$: Invoke SSSP over $E_1 \cup \{(u, v) | v \in V\} \cup (S_2 \times V) \cup H$ and update d

Case 1

$$u \rightsquigarrow v \subseteq E_1$$



$$d[u, v] = \delta(u, v)$$

$+2W_1$-APASP Algorithm Correctness

Let $u, v \in V$

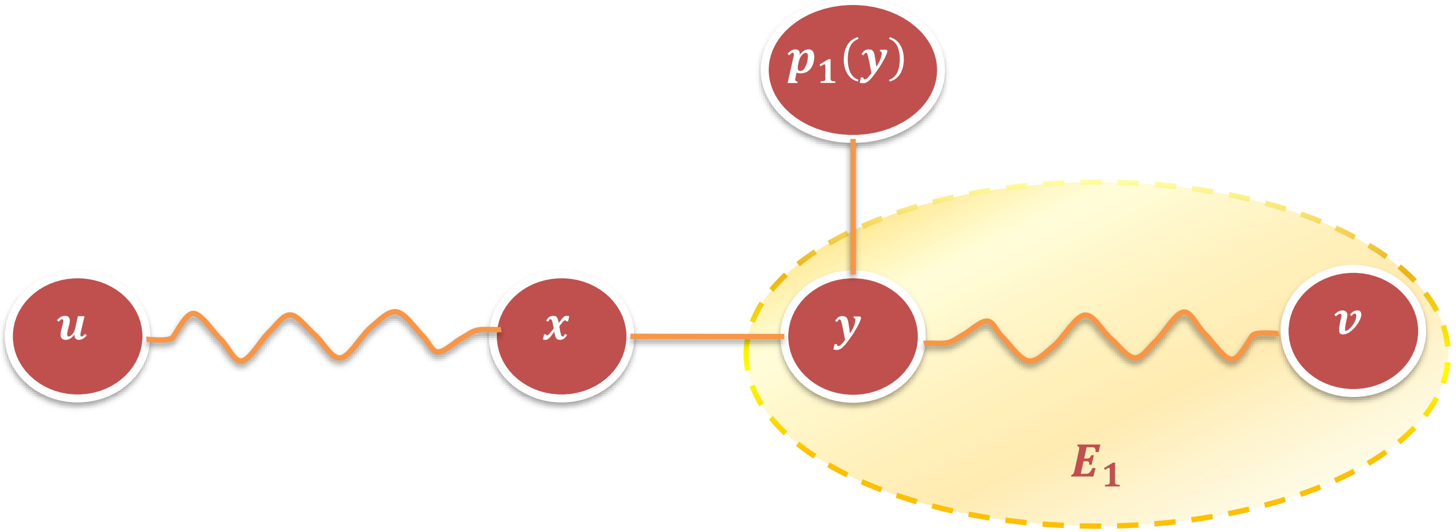
Our aim: $d[u,v] \in [\delta(u,v), \delta(u,v) + 2W_1]$

Distinguish between three possible cases:

✅ 1. $u \rightsquigarrow v \subseteq E_1$
2. $u \rightsquigarrow v \subseteq E_2$ yet $u \rightsquigarrow v \nsubseteq E_1$
3. $u \rightsquigarrow v \nsubseteq E_2$

Case 2

$u \rightsquigarrow v \subseteq E_2$ yet $u \rightsquigarrow v \not\subseteq E_1$



Let y be such $(x, y) \notin E_1$, assume y is nearest to v

$+2W_1$-APASP Algorithm Overview

1. Find $p_1(u)$ (resp. $p_2(u)$) for every $u \in V$

2. Set $d[u, p_1(u)] = \delta\big(u, p_1(u)\big)$ (resp. $d[u, p_2(u)] = \delta\big(u, p_2(u)\big)$)

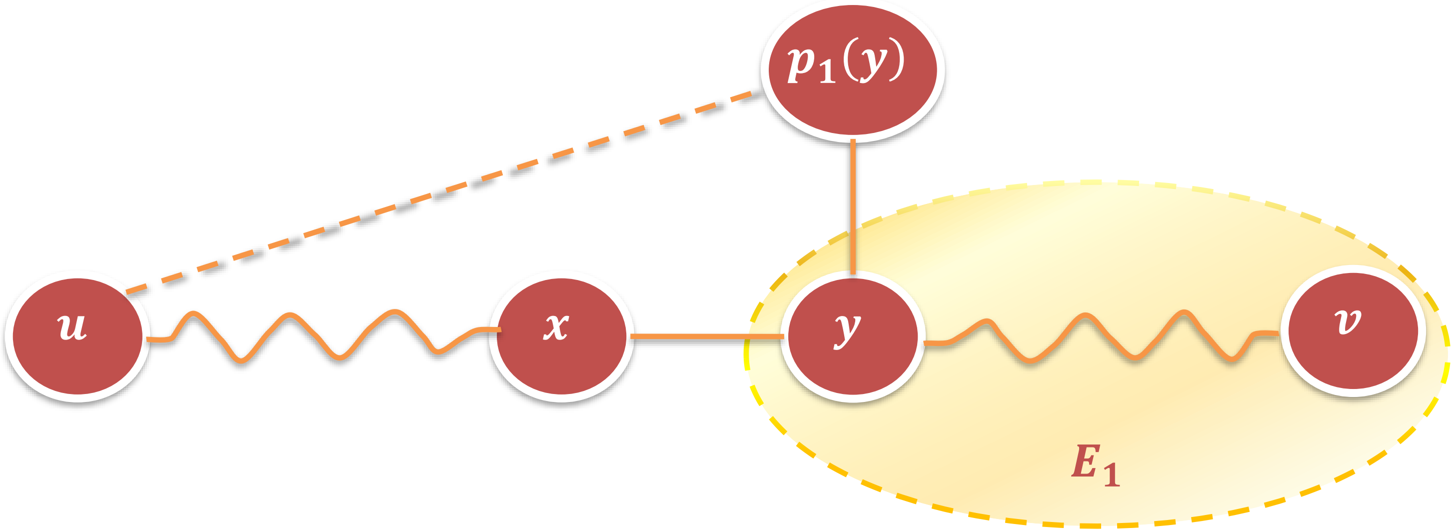
3. For $s \in S_2$: Invoke SSSP over E and update d

4. For $s \in S_1$: Invoke SSSP over E_2 and update d

5. For $u \in V$: Invoke SSSP over $E_1 \cup \{(u, v) | v \in V\} \cup (S_2 \times V) \cup H$ and update d

Case 2

$u \rightsquigarrow v \subseteq E_2$ yet $u \rightsquigarrow v \nsubseteq E_1$



Let y be such $(x, y) \notin E_1$, assume y is nearest to v

$$d[p_1(y), u] = \delta(p_1(y), u) \leq \delta(u, y) + \delta(y, p_1(y))$$

$+2W_1$-APASP Algorithm Overview

1. Find $p_1(u)$ (resp. $p_2(u)$) for every $u \in V$

2. Set $d[u, p_1(u)] = \delta\big(u, p_1(u)\big)$ (resp. $d[u, p_2(u)] = \delta\big(u, p_2(u)\big)$)

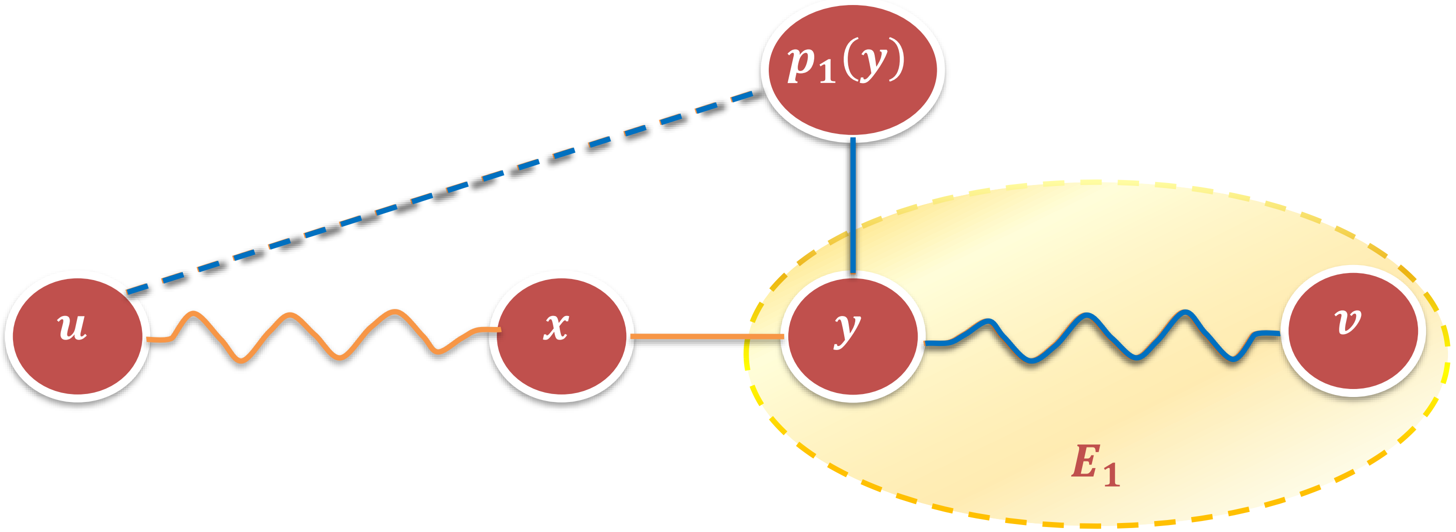
3. For $s \in S_2$: Invoke SSSP over E and update d

4. For $s \in S_1$: Invoke SSSP over E_2 and update d

5. For $u \in V$: Invoke SSSP over $E_1 \cup \{(u, v) | v \in V\} \cup (S_2 \times V) \cup H$ and update d

Case 2

$u \rightsquigarrow v \subseteq E_2$ yet $u \rightsquigarrow v \nsubseteq E_1$



Let y be such $(x, y) \notin E_1$, assume y is nearest to v

$$d[p_1(y), u] = \delta(p_1(y), u) \leq \delta(u, y) + \delta\big(y, p_1(y)\big)$$

$$d[u, v] \leq d[u, p_1(y)] + d[p_1(y), y] + \delta(y, v) \leq \delta(u, y) + 2\delta\big(y, p_1(y)\big) + \delta(y, v)$$
$$\leq \delta(u, v) + 2w(x, y) \leq \delta(u, v) + 2W_1(u, v)$$

$+2W_1$-APASP Algorithm Correctness

Let $u, v \in V$

Our aim: $d[u, v] \in [\delta(u, v), \delta(u, v) + 2W_1]$

Distinguish between three possible cases:

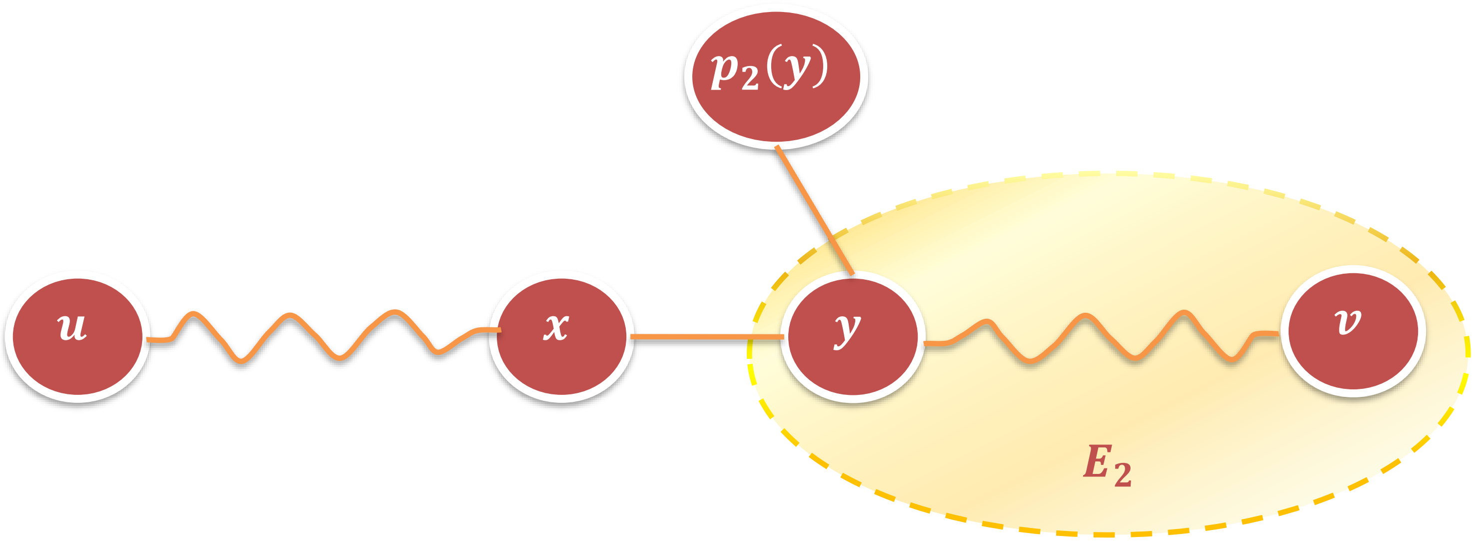
✅ 1. $u \rightsquigarrow v \subseteq E_1$

✅ 2. $u \rightsquigarrow v \subseteq E_2$ yet $u \rightsquigarrow v \nsubseteq E_1$

3. $u \rightsquigarrow v \nsubseteq E_2$

Case 3

$u \rightsquigarrow v \not\subseteq E_2$



An arbitrary edge $(x, y) \notin E_2$

$+2W_1$-APASP Algorithm Overview

1. Find $p_1(u)$ (resp. $p_2(u)$) for every $u \in V$

2. Set $d[u, p_1(u)] = \delta\big(u, p_1(u)\big)$ (resp. $d[u, p_2(u)] = \delta\big(u, p_2(u)\big)$)

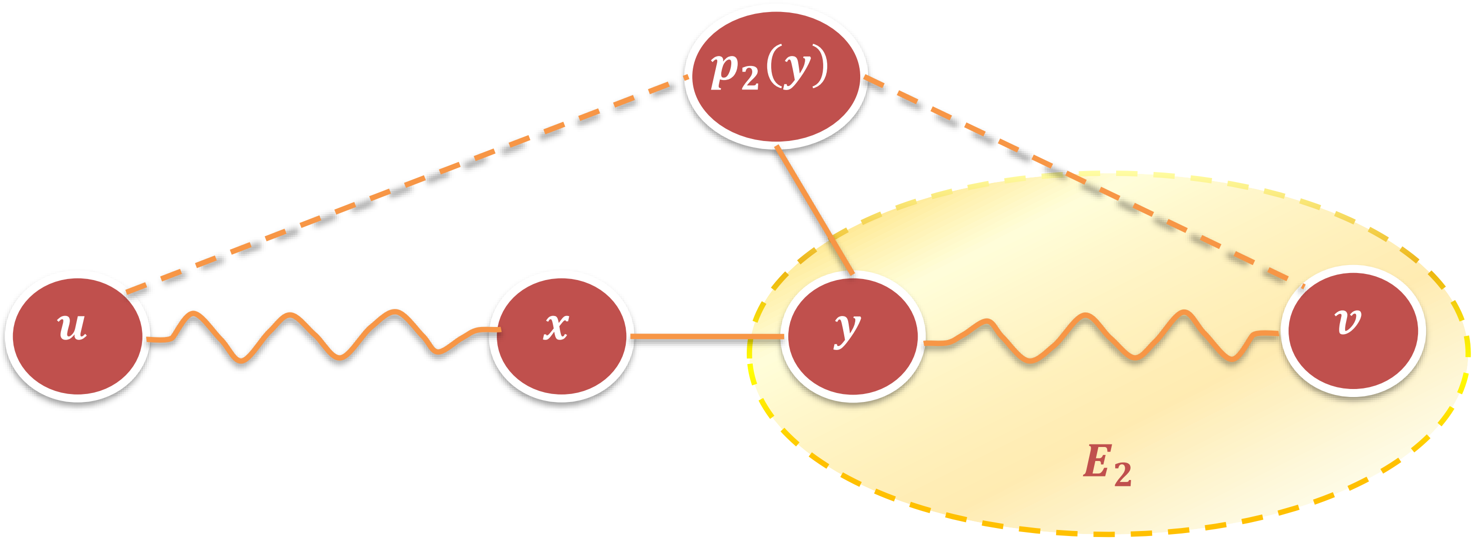
3. For $s \in S_2$: Invoke SSSP over E and update d

4. For $s \in S_1$: Invoke SSSP over E_2 and update d

5. For $u \in V$: Invoke SSSP over $E_1 \cup \{(u, v) | v \in V\} \cup (S_2 \times V) \cup H$ and update d

Case 3

$u \rightsquigarrow v \not\subseteq E_2$



An arbitrary edge $(x, y) \notin E_2$

$d[p_2(y), u] = \delta(p_2(y), u)$ and $d[p_2(y), v] = \delta(p_2(y), v)$

$+2W_1$-APASP Algorithm Overview

1. Find $p_1(u)$ (resp. $p_2(u)$) for every $u \in V$

2. Set $d[u, p_1(u)] = \delta\big(u, p_1(u)\big)$ (resp. $d[u, p_2(u)] = \delta\big(u, p_2(u)\big)$)

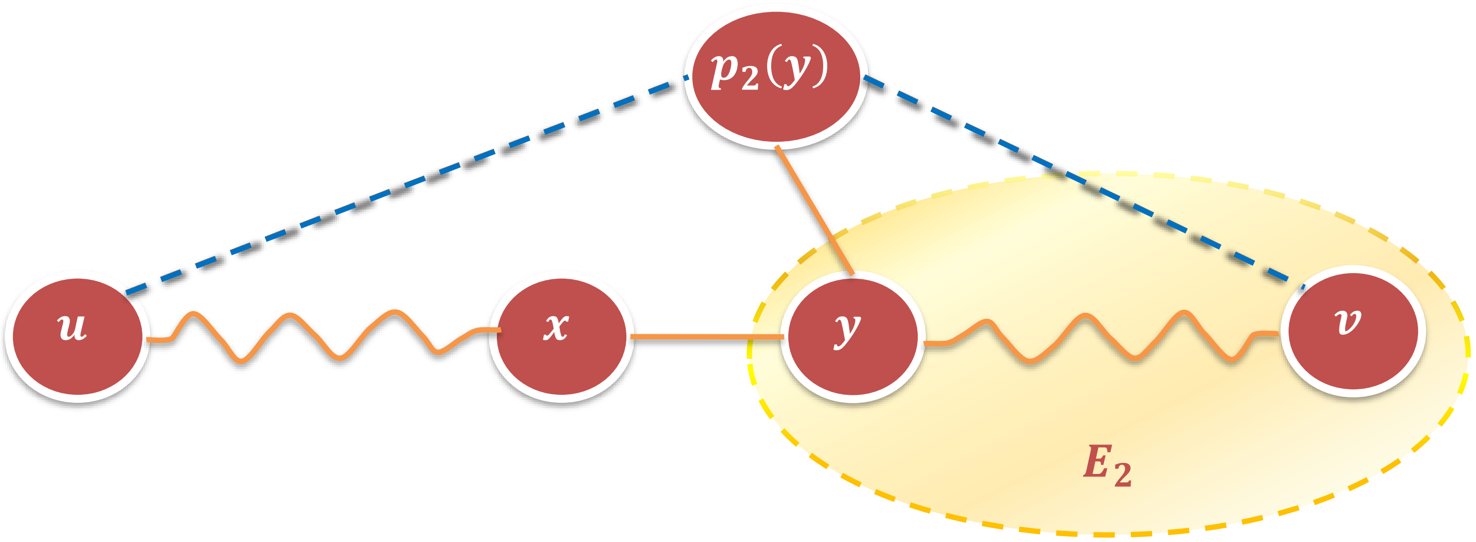
3. For $s \in S_2$: Invoke SSSP over E and update d

4. For $s \in S_1$: Invoke SSSP over E_2 and update d

5. For $u \in V$: Invoke SSSP over $E_1 \cup \{(u, v) | v \in V\} \cup (S_2 \times V) \cup H$ and update d

Case 3

$u \rightsquigarrow v \not\subseteq E_2$



An arbitrary edge $(x, y) \notin E_2$

$d[p_2(y), u] = \delta(p_2(y), u)$ and $d[p_2(y), v] = \delta(p_2(y), v)$

$d[u, v] \leq d[u, p_2(y)] + d[p_2(y), v] = \delta\big(u, p_2(y)\big) + \delta\big(v, p_2(y)\big) \leq \delta(u, y) + 2\delta\big(y, p_2(y)\big) + \delta(y, v) \leq \delta(u, v) + 2w(x, y) \leq \delta(u, v) + 2W_1(u, v)$

$+2W_1$-APASP Algorithm Correctness

Let $u, v \in V$

Our aim: $d[u,v] \in [\delta(u,v), \delta(u,v) + 2W_1]$

Distinguish between three possible cases:

✅ 1. $u \rightsquigarrow v \subseteq E_1$

✅ 2. $u \rightsquigarrow v \subseteq E_2$ yet $u \rightsquigarrow v \nsubseteq E_1$

✅ 3. $u \rightsquigarrow v \nsubseteq E_2$

Conclusion: This algorithm computes a $+2W_1$-APASP and requires $\tilde{O}(n^{\frac{7}{3}})$ runtime.

Plan of Talk

- APSP and APASP
- Additive APASP: Weighted and Unweighted
- Hitting Sets
- Additive $+2W_1$-APASP
- Additive $+2\sum_{i=1}^{k+1} W_i$-APASP
- Additional Results
- Further Directions

Only Two Levels?

Cohen and Zwick's $+2W_1$-APASP algorithm: $\beta, \gamma \in (0,1)$

They considered: $\Gamma(u, n^{\beta})$ and $\Gamma(u, n^{\beta+\gamma})$

Hitting sets: S_1 and S_2

$|S_1| \in \tilde{O}(n^{1-\beta}),\ |S_2| \in \tilde{O}(n^{1-\beta-\gamma})$

Edges to nearest neighbours: E_1 and E_2

$|E_1| \in O(n^{1+\beta}),\ |E_2| \in O(n^{1+\beta+\gamma})$

What if we add more levels?

Adding More Levels

Simply $k \in \mathbb{N}$ levels?

We skipped a single level ($k = 1$)?

For $k = 1$ we still get a $+2W_1$-APASP

The runtime will be $\tilde{O}\left(n^{2+\beta} + n^{3-\beta}\right)$

Select $\beta = \frac{1}{2}$

The runtime becomes $\tilde{O}(n^{\frac{5}{2}})$

Worse than $\tilde{O}(n^{\frac{7}{3}})$

Adding More Levels

What about $k = 3$?

The runtime will be $\tilde{O}\left(n^{2+\beta} + n^{2+\gamma} + n^{2+\delta} + n^{3-\beta-\gamma-\delta}\right)$

Select $\beta = \frac{1}{4}$

The runtime becomes $\tilde{O}(n^{\frac{9}{4}})$

But we compute a $+2W_1 + 2W_2$-APASP

Weaker guarantee than $+2W_1$-APASP

Adding More Levels

For $k = 4$: $+2W_1 + 2W_2$-APASP in $\tilde{O}(n^{\frac{11}{5}})$ runtime

Better than $k = 3$: $+2W_1 + 2W_2$-APASP in $\tilde{O}(n^{\frac{9}{4}})$ runtime

Not every $k \in \mathbb{N}$ is “useful”

$3k + 2$ levels

Parameters: $\beta_1, \beta_2, \ldots, \beta_{3k+2} \in (0,1)$

$3k + 2$ Levels

$$\alpha_j = \sum_{i=1}^{j} \beta_i$$

Consider $\Gamma(u, n^{\alpha_j})$ for $1 \leq j \leq 3k + 2$

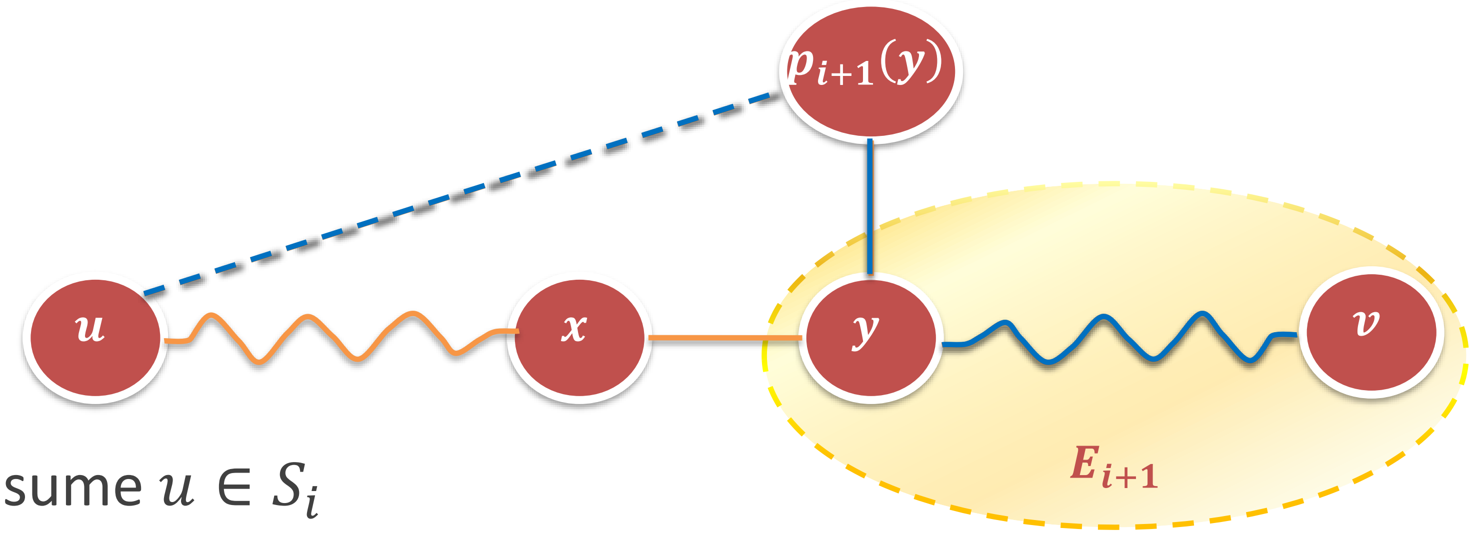
Hitting sets: S_j, $|S_j| \in \tilde{O}(n^{1-\alpha_j})$

Edges to nearest neighbours: E_j, $|E_j| \in O(n^{1+\alpha_j})$

Similar SSSP invocations

SSSP Invocations

Which edges should we consider in each SSSP invocation?



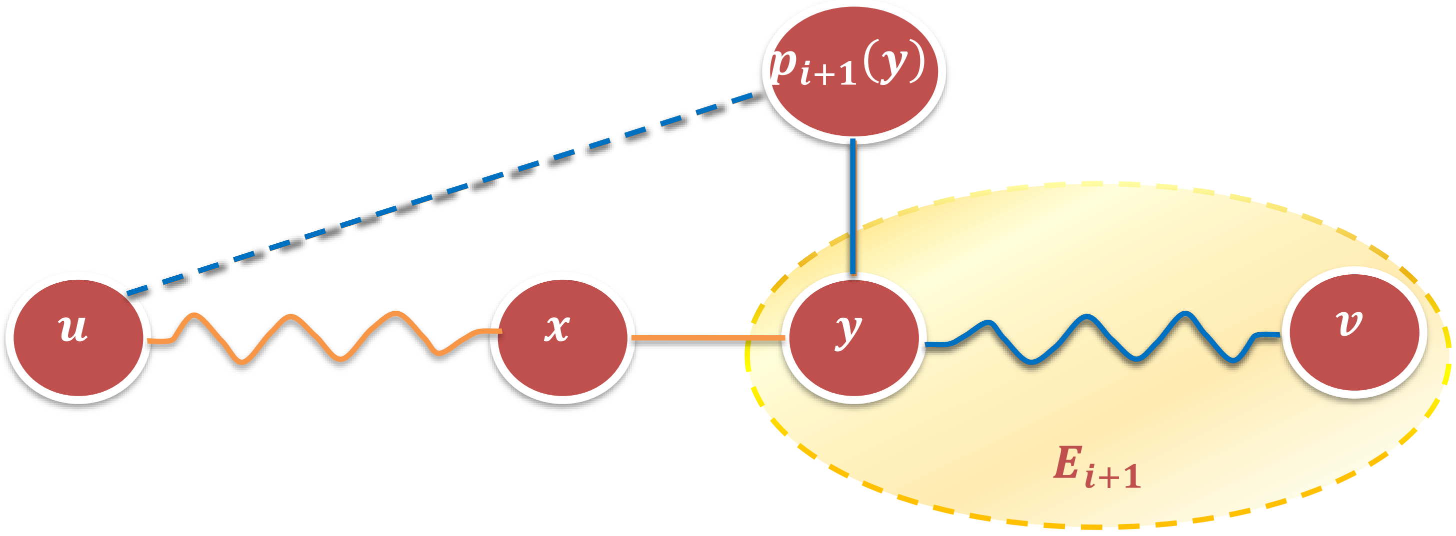
Assume $u \in S_i$

u will "see" E_{i+1}

We need to have $d[p_{i+1}(y), u]$

SSSP Invocations

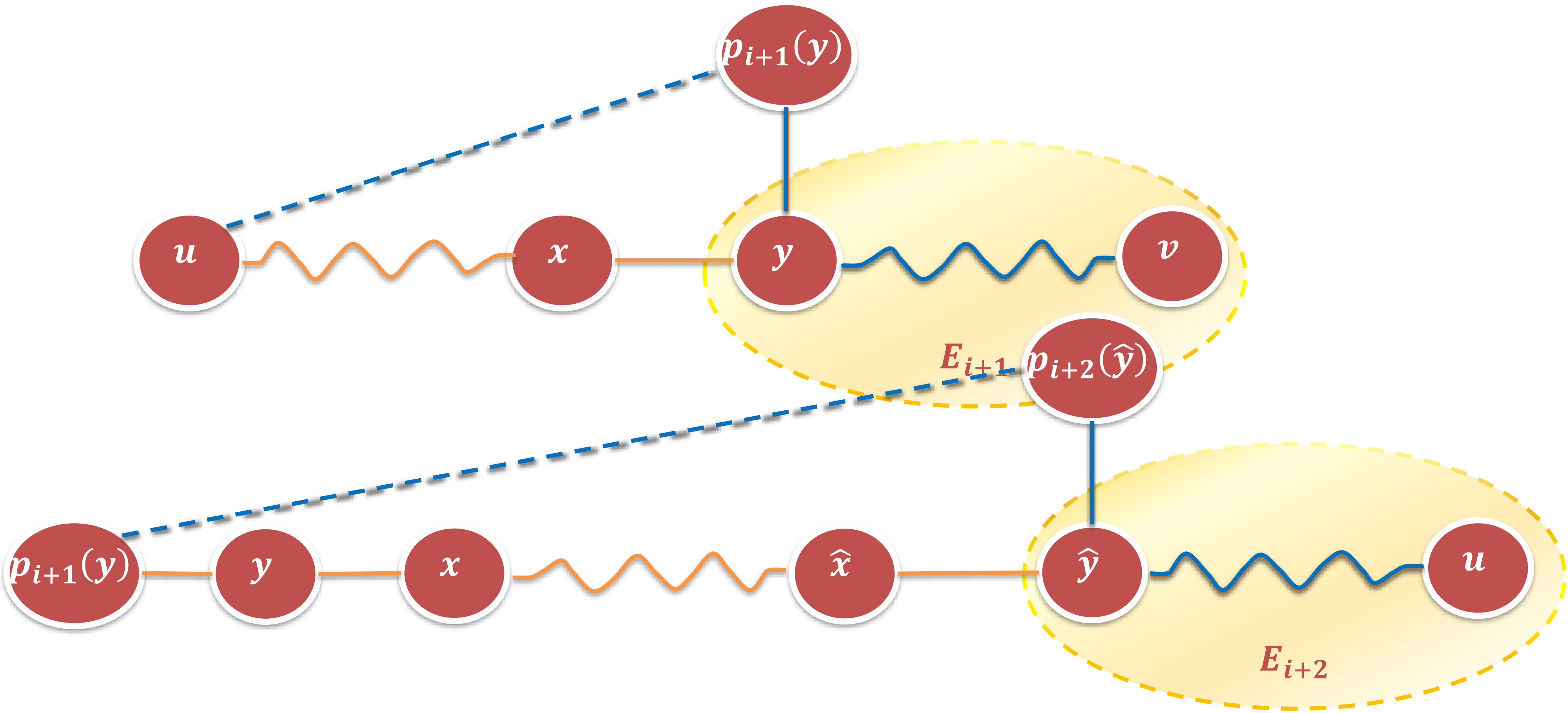
Let $\Delta(u \rightsquigarrow v)$ be an upper bound for $d[u,v] - \delta(u,v)$



Here: $\Delta(u \rightsquigarrow v) = \Delta\big(u \rightsquigarrow x - y \rightsquigarrow p_{i+1}(y)\big) + 2w(x,y)$

Recursively: $p_{i+1}(y) \in S_{i+1} \ldots$

Recursive Upper Bound for the Estimation



Recursive Upper Bound for the Estimation

Trivia: *Does this guarantee an upper-bound that depends on $W_1, W_2, \dots, W_{k+1}$?*

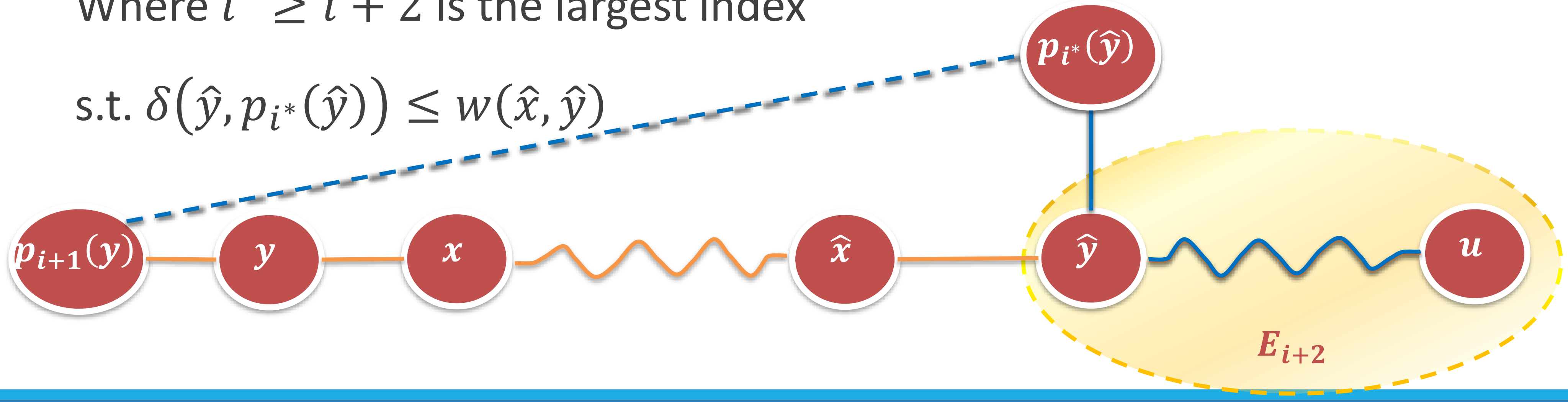
Answer: Almost…

How can we guarantee that the same W_i is not used more than once? 🤷

Instead of $p_{i+2}(\hat{y})$ we need to consider $p_{i^*}(\hat{y})$

Where $i^* \geq i+2$ is the largest index

s.t. $\delta(\hat{y}, p_{i^*}(\hat{y})) \leq w(\hat{x}, \hat{y})$

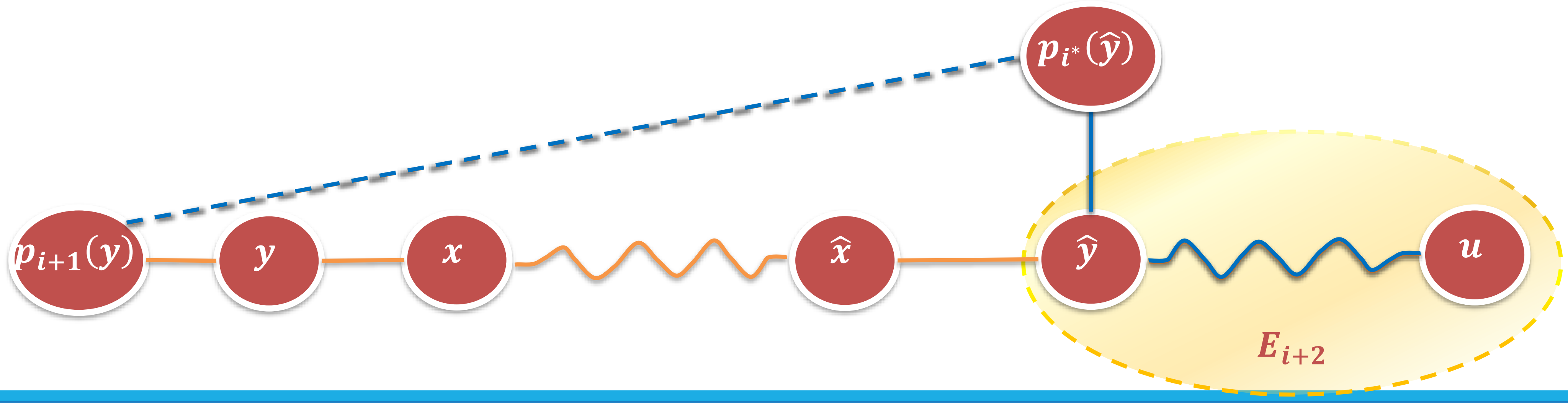


A Word About Runtime

To compute the runtime:

1. List exactly the edges being used
2. Enumerate the number of recursive calls

Total runtime: $\tilde{O}(n^{2+\frac{1}{3k+2}})$



Additive $+2\sum_{i=1}^{k+1} W_i$-APASP

Cohen and Zwick's $+2W_1$-APASP algorithm

Runtime: $\tilde{O}(n^{\frac{7}{3}})$

Our result: $+2\sum_{i=1}^{k+1} W_i$-APASP algorithm

Runtime: $\tilde{O}(n^{2+\frac{1}{3k+2}})$

(Only the runtime for the base case differs…)

Plan of Talk

- APSP and APASP
- Additive APASP: Weighted and Unweighted
- Hitting Sets
- Additive $+2W_1$-APASP
- Additive $+2\sum_{i=1}^{k+1} W_i$-APASP
- Additional Results
- Further Directions

Nearly Additive APASP

Purely additive (α, β)-APASP $\Rightarrow \alpha = 1$

Nearly additive: $\alpha = 1 + \varepsilon$, for some small $\varepsilon > 0$

Cohen and Zwick's algorithm actually computed a $+2\min\{2W_1, 4W_2\}$-APASP

Saha and Ye (2024) computed a $(1 + \varepsilon, 2W_1)$-APASP

Their runtime: $\tilde{O}\left(\left(\frac{1}{\varepsilon}\right)^{O(1)} \cdot n^{2.15135313} \cdot \log W\right)$

In the same runtime, we compute a $(1 + \varepsilon, 2\min\{2W_1, 4W_2\})$-APASP

Multiplicative APASP

Cohen and Zwick (1997), Baswana and Kavitha (2010), Kavitha (2012):

2-APASP, $\frac{7}{3}$-APASP, $\frac{5}{2}$-APASP, 3-APASP

Roditty and Akav (2021) extended these specific approximations:

$$\frac{3\ell+4}{\ell+2}\text{-APASP}$$

We consider a similar family:

$$\left(\frac{3\ell+4}{\ell+2}+\varepsilon\right)\text{-APASP}$$

Tradeoffs

In general:

(α_1, β_1)-APASP algorithm $\mathcal{A}_1$ and an(α_2, β_2)-APASP algorithm $\mathcal{A}_2$

Running both (assuming they have the same runtime…):

$$\begin{cases} d[u,v] \le \alpha_1 \cdot \delta(u,v) + \beta_1 \\ d[u,v] \le \alpha_2 \cdot \delta(u,v) + \beta_2 \end{cases}$$

$$\Downarrow$$

$$d[u,v] \le \frac{\alpha_1 + \alpha_2}{2} \cdot \delta(u,v) + \frac{\beta_1 + \beta_2}{2}$$

Tradeoffs

Running both yields a $\left(\frac{\alpha_1+\alpha_2}{2}, \frac{\beta_1+\beta_2}{2}\right)$-APASP algorithm $\mathcal{A}_3$

Same runtime as $\mathcal{A}_1,\ \mathcal{A}_2$

Any type of weighted average:

$$d[u,v] \leq \frac{\alpha_1 \cdot \gamma + \alpha_2 \cdot \tau}{\gamma + \tau} \cdot \delta(u,v) + \frac{\beta_1 \cdot \gamma + \beta_2 \cdot \tau}{\gamma + \tau}$$

A $\left(\frac{\alpha_1 \cdot \gamma + \alpha_2 \cdot \tau}{\gamma + \tau}, \frac{\beta_1 \cdot \gamma + \beta_2 \cdot \tau}{\gamma + \tau}\right)$-APASP algorithm

Tradeoffs: Concrete Examples

Our algorithm: $+2\sum_{i=1}^{k+1} W_i$-APASP algorithm with $\tilde{O}(n^{2+\frac{1}{3k+2}})$ runtime

Akav and Roditty (2021): $\frac{3\ell+4}{\ell+2}$-APASP algorithm with $\tilde{O}(n^{2-\frac{3}{\ell+2}} m^{\frac{2}{\ell+2}} + n^2)$ runtime

For $m = n^2$ and $\ell = 3k$ it is a $\frac{9k+4}{3k+2}$-APASP algorithm with $\tilde{O}(n^{2+\frac{1}{3k+2}})$ runtime

Running both: $\left(\frac{(9k+4)\cdot\gamma+(3k+2)\cdot\tau}{\gamma+\tau}, \frac{2\tau}{\gamma+\tau} \cdot \sum_{i=1}^{k+1} W_i\right)$-APASP

<u>For example</u>: $\left(\frac{6k+3}{3k+2}, \sum_{i=1}^{k+1} W_i\right)$-APASP with $\tilde{O}(n^{2+\frac{1}{3k+2}})$ runtime

Plan of Talk

- APSP and APASP
- Additive APASP: Weighted and Unweighted
- Hitting Sets
- Additive $+2W_1$-APASP
- Additive $+2\sum_{i=1}^{k+1} W_i$-APASP
- Additional Results
- Further Directions

Further Directions

Runtime gap between the runtimes: base case ($k = 0$) and general case

$$\tilde{O}(n^{\frac{7}{3}}) \text{ and } \tilde{O}(n^{2+\frac{1}{3k+2}})$$

The above holds as well for the unweighted setting

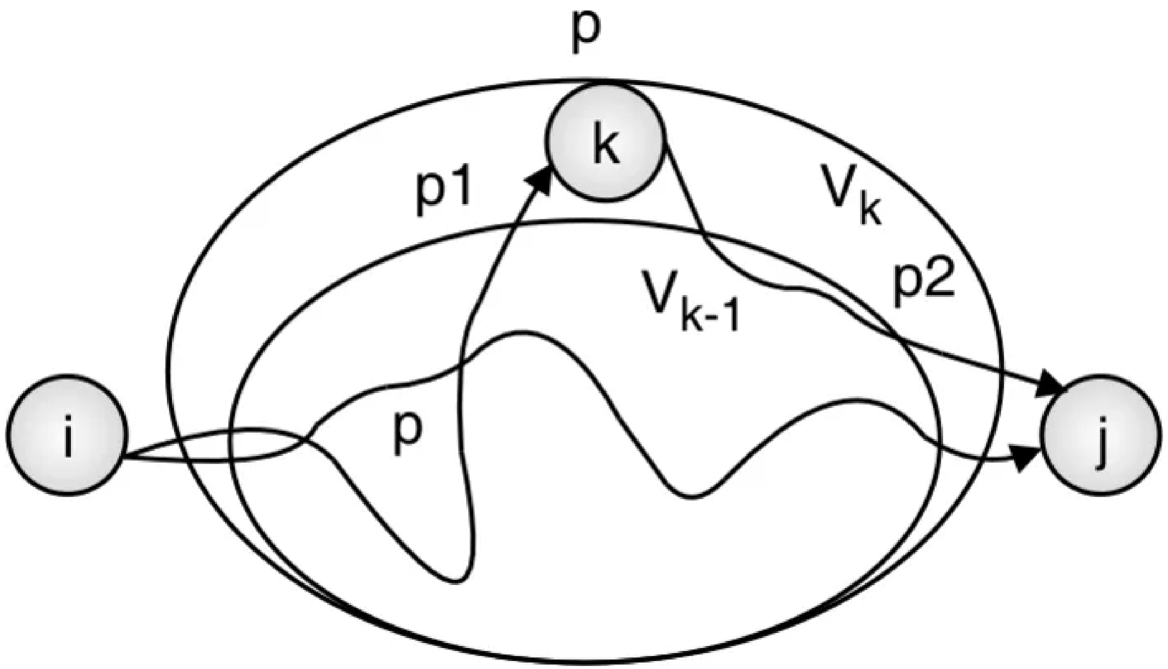
Strongly Commensurate: Other approaches except W_i?

Additive to Multiplicative? $+2W_2$-APASP $\Rightarrow$ 2-APASP

Plan of Talk

- APSP and APASP
- Additive APASP: Weighted and Unweighted
- Hitting Sets
- Additive $+2W_1$-APASP
- Additive $+2\sum_{i=1}^{k+1} W_i$-APASP
- Additional Results
- Further Directions

The End (for today)



To Be Continued