Sparse Fault-Tolerant BFS Trees

Merav Parter and David Peleg

Weizmann Institute Of Science

BIU-CS Colloquium

16-01-2014

Breadth First Search (BFS) Trees

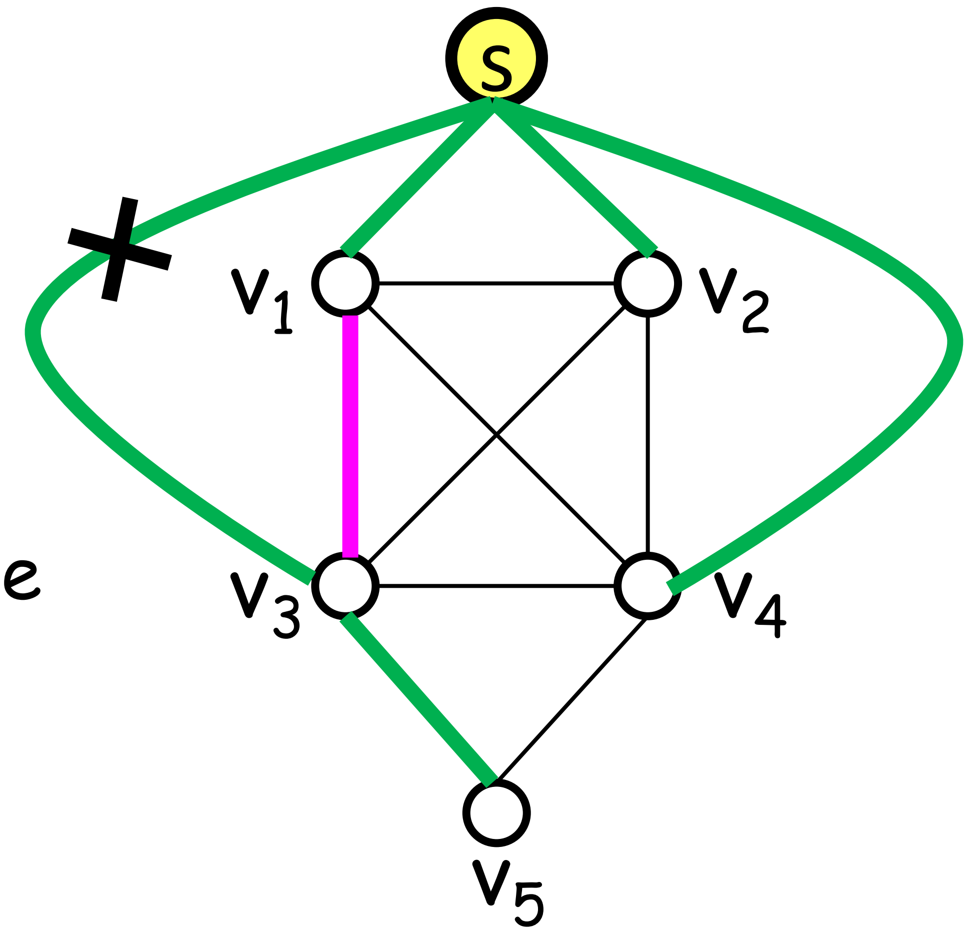
- Unweighted graph $G=(V,E)$, source vertex $s \in V$.
- Shortest-Path Tree (BFS) rooted at s.

Sparse solution:

$n-1$ edges.

Problem:

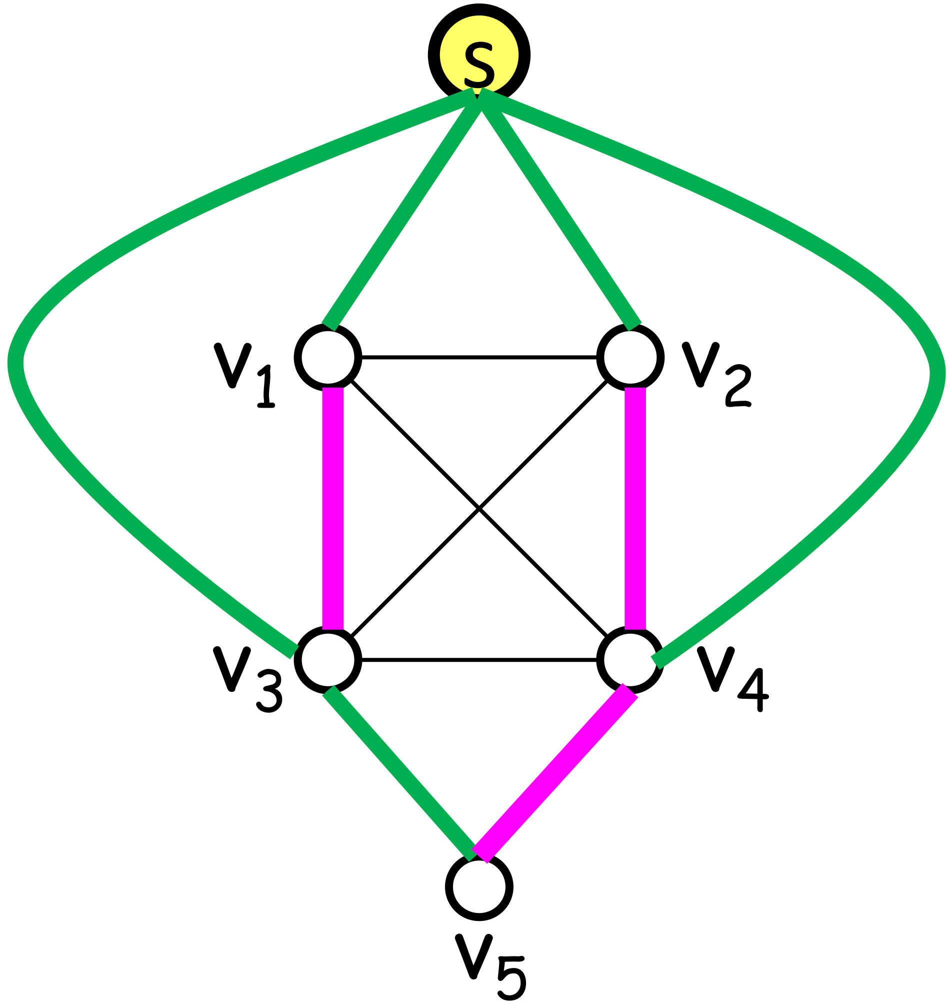
Not robust against edge and vertex faults.



Fault Tolerant BFS Trees

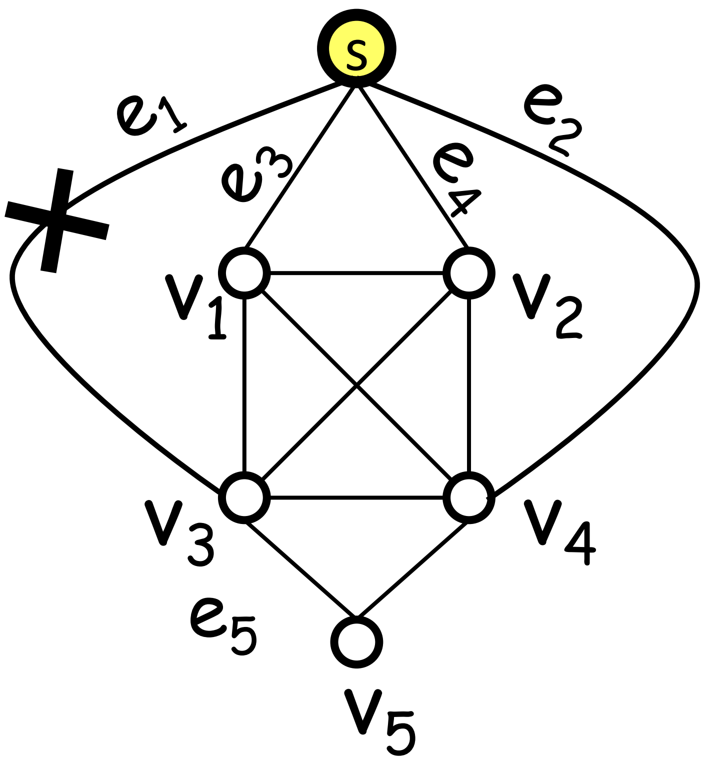
Objective:

Purchase a collection of edges (BFS + backup edges) that is robust against edge faults.

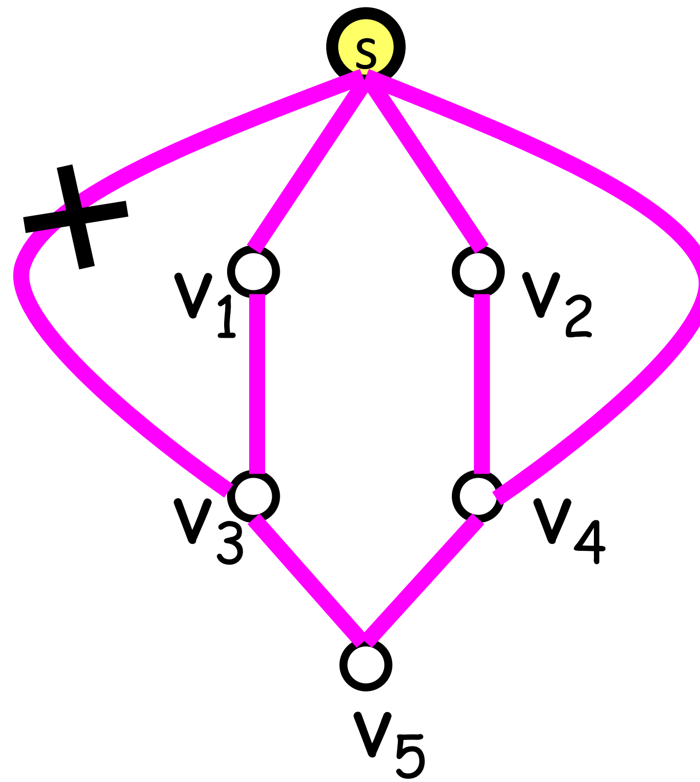


Fault-Tolerant BFS Trees

Subgraph H that contains a BFS tree in G\{e}
for every edge failure e in G.



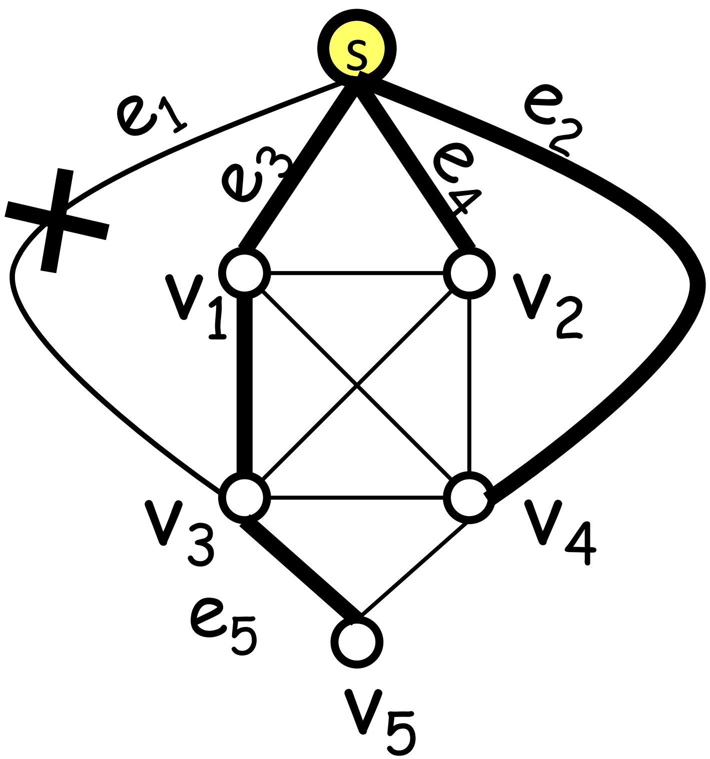
$G \setminus \{e_1\}$



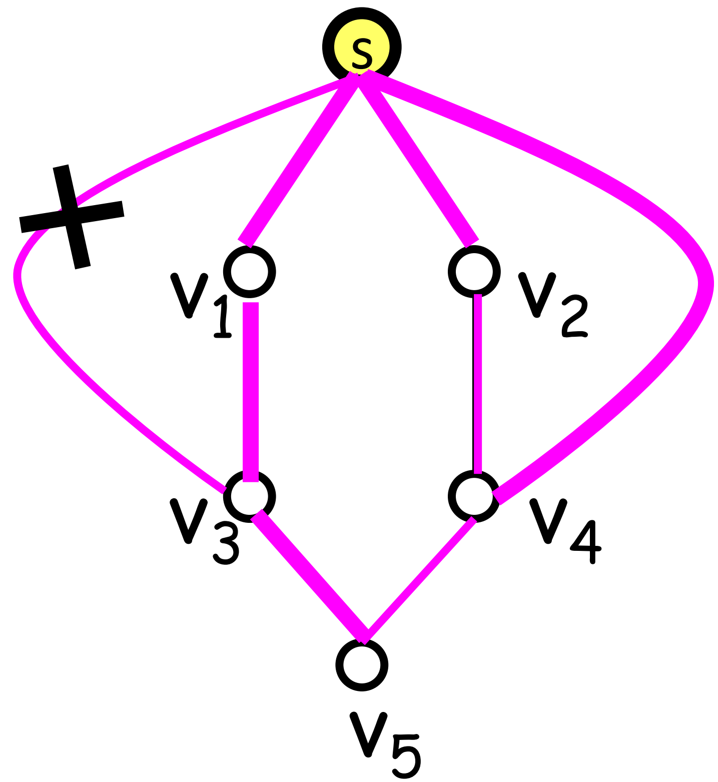
$H \setminus \{e_1\}$

Fault-Tolerant (FT) BFS Trees

Subgraph H that contains a BFS tree in G\{e} for every edge failure e in G.



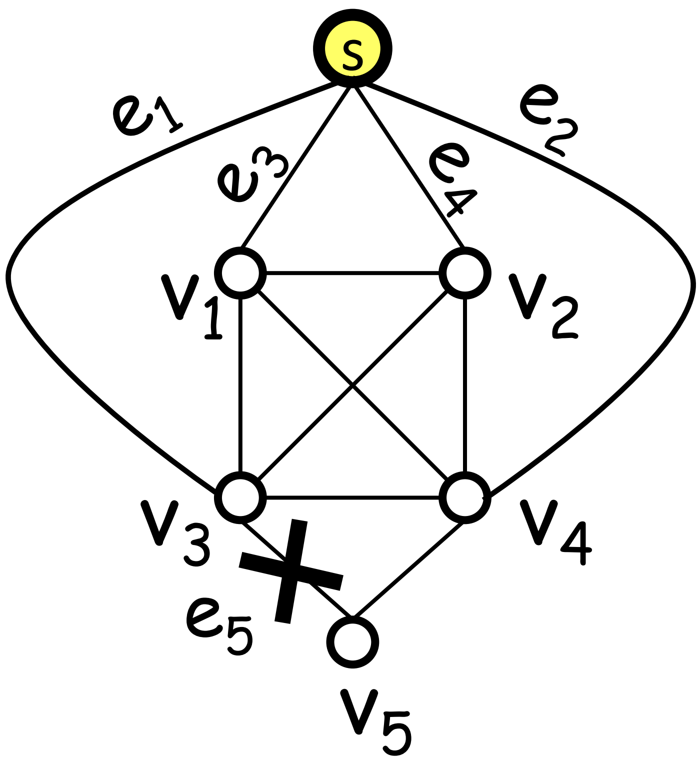
$G \setminus \{e_1\}$



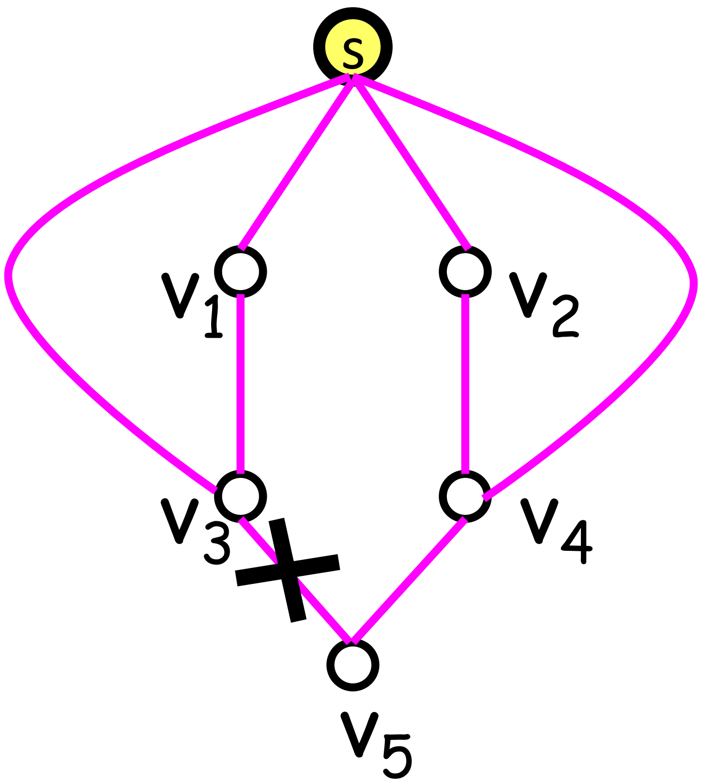
$H \setminus \{e_1\}$

Fault Tolerant (FT) BFS Trees

Subgraph H that contains a BFS tree in G\{e} for every edge failure e in G.



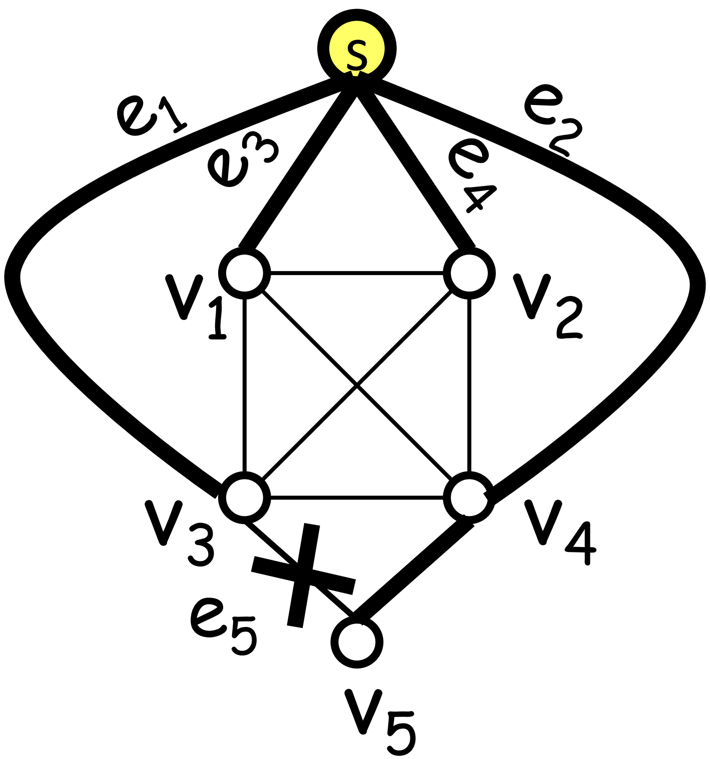
$G \setminus \{e_1\}$



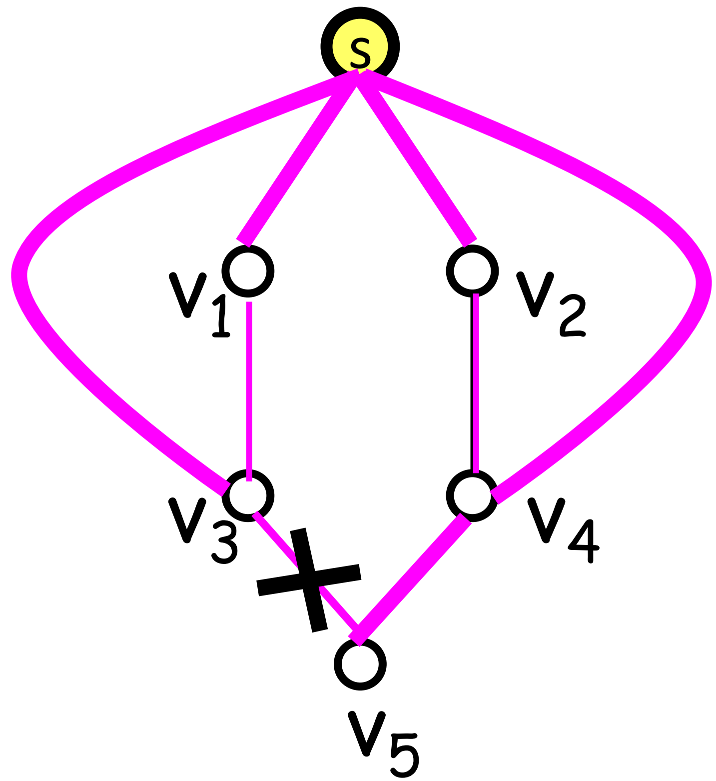
$H \setminus \{e_1\}$

Fault Tolerant (FT) BFS Trees

Subgraph **H** that contains a BFS tree in **G\{e}** for every edge failure **e** in **G**.



$G \setminus \{e_5\}$



$H \setminus \{e_5\}$

FT-BFS Tree - Formal Definition

- Consider an unweighted graph $G=(V,E)$ and a source vetrex s.

- A subgraph H is an FT-BFS of G and s if for every v in V **and** e **in** E:

$$d(s,v,\ H \setminus \{e\}) = d(s,v,\ G \setminus \{e\})$$

FT-BFS for Multiple Sources (FT-MBFS)

- Consider an unweighted graph $G=(V,E)$ and a source set S in V.

- A subgraph H is an FT-MBFS of G if for every s in S, v in V and e in E:

$$d(s,v, H \setminus \{e\}) = d(s,v, G \setminus \{e\})$$

The Minimum FT-BFS tree Problem

- Input: unweighted graph $G=(V,E)$
  source vertex s in V.

- Output:
  An FT-BFS subgraph $H \subseteq G$ with minimum number of edges.

Outline

- Related work
- Lower bound construction
- Upper bound
- Hardness and approximation algorithm.

Related Work

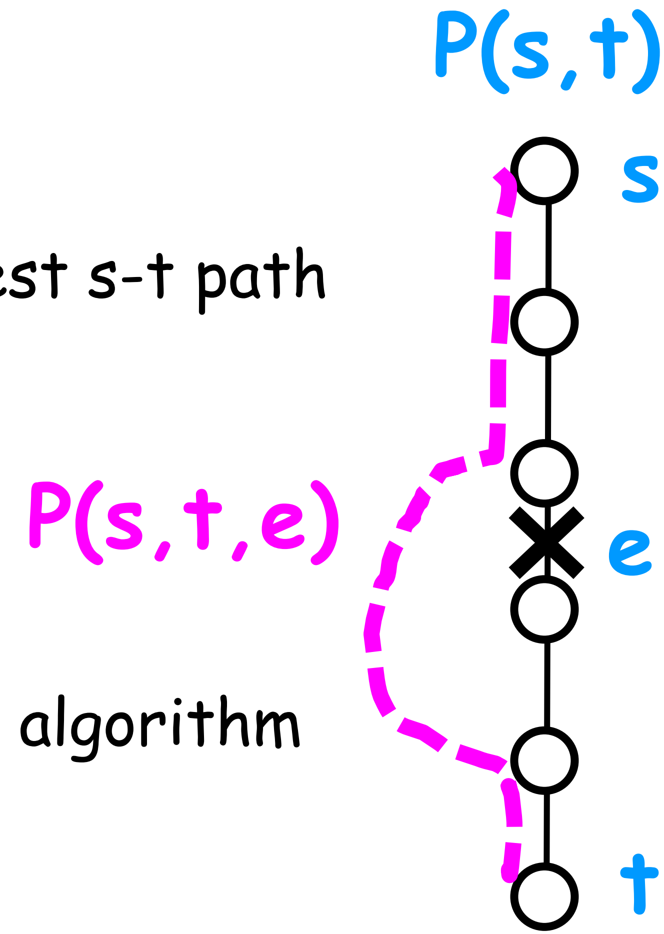
- Replacement Path
- Fault-Tolerant Spanners

A related problem: the replacement path problem

$P(s,t,e)$: s-t shortest path in $G \setminus \{e\}$

Problem definition:

Given a source s, destination t, for every $e \in P(s,t)$, compute $P(s,t,e)$ the shortest s-t path that avoids e.



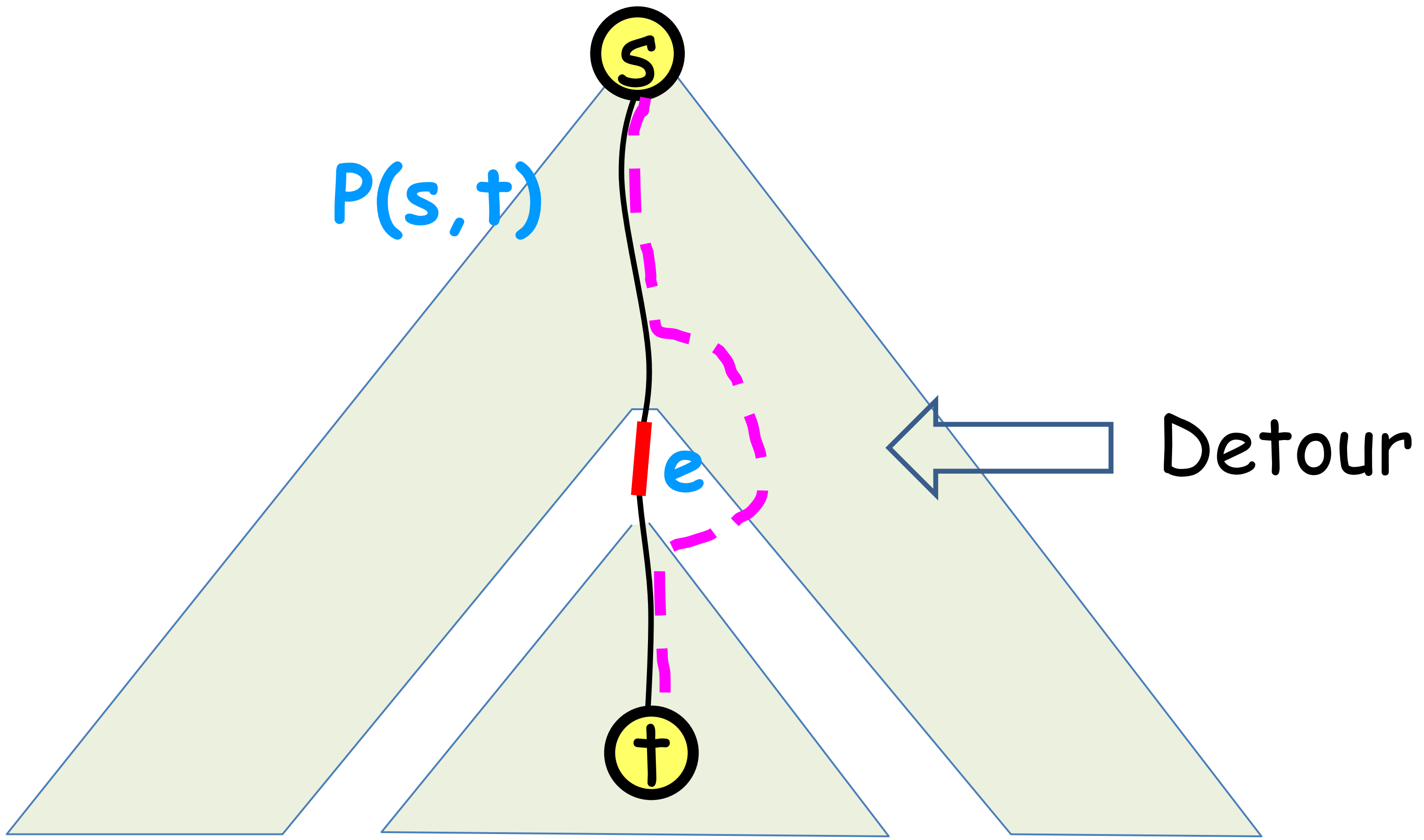
Trivial algorithm:

For every edge $e \in P(s,t)$, run Dijkstra's algorithm from s in $G \setminus \{e\}$.

Time complexity: $O(mn)$

The structure of a replacement path

$P(s,t,e)$: s-t shortest path in $G \setminus \{e\}$



The replacement paths problem

Better bounds available for replacement paths problem for

Undirected graphs:

Time complexity: $O(m+n \log n)$

[Gupta et al. 1989]

[Hershberger and Suri, 2001]

Unweighted directed graphs:

Time complexity: $O(m\sqrt{n})$ **(Randomized MonteCarlo algorithm)**

[Roditty and Zwick 2005]

Single-source replacement paths

Problem definition:

Given a source s, compute $P(s,t,e)$ efficiently for each t in V and every $e \in P(s,t)$.

Time complexity: $O(n^{\omega})$

[Grandoni and Williams, FOCS'12]

FT-BFS tree revisited:

An FT-BFS tree H contains the collection of all single source replacement paths.

New!

Complexity measure: size of H (#edges).

Spanners

- Graph $G=(V,E)$

- A subgraph H is an k-spanner if

  for every u,v in V:

$$d(u,v,H) \leq k \cdot d(u,v,G).$$

Fault-Tolerant Spanners

A subgraph H is an

f-edge fault tolerant k-spanner

if for every u,v in V and every set of

f edges $F=\{e_1,e_2,\ldots,e_f\}$:

$$d(u,v,H \setminus F) \leq k \cdot d(u,v,G \setminus F).$$

Fault-Tolerant Spanners

d(u,v,H \F) $\leq (2k-1) \cdot$ d(u,v,G\F) for all u,v in V

Robust to f-vertex faults:

Stretch: 2k-1

#edges:

$\tilde{O}\left(f^2 k^{f+1} \cdot n^{1+\frac{1}{k}}\right)$ [Chechik et al., 2009]

$\tilde{O}\left(f^{2-\frac{1}{k}} \cdot n^{1+\frac{1}{k}}\right)$ [Dinitz and Krauthgamer, 2011]

Fault-Tolerant Spanners

d(u,v,H \F) $\leq (2k-1) \cdot$ d(u,v,G\F) for all u,v in V

Robust to f-edge faults:

Stretch: 2k-1

#edges: $O\left(f\, n^{1+\frac{1}{k}}\right)$ [Chechik et al., 2009]

FT-Spanners vs. FT-BFS trees

FT-Spanners

All-pairs
$V \times V$

approximate

FT-BFS tree

Single source
$s \times V$

⇨ FT-BFS's easier

exact

⇨ FT-BFS's harder

Outline

- ❑ Related work
- ❑ Lower bound construction
- ❑ Upper bound
- ❑ Hardness and approximation algorithm.

Lower Bound

Theorem [Single source]:

For every integer $n \geq 1$, there exists an n-vertex graph $G=(V,E)$ and a source vertex $s \in V$ such that every **FT-BFS** tree H has $\Omega(n\sqrt{n})$ edges.

Generalization to multiple sources (FT-MBFS)

Theorem [Multiple sources]:

For every integer $n \geq 1$, there exists an n-vertex graph $G=(V,E)$ and a source set $S \subseteq V$ such that every FT-BFS tree H has $\Omega(n\sqrt{|S|\,n})$ edges.

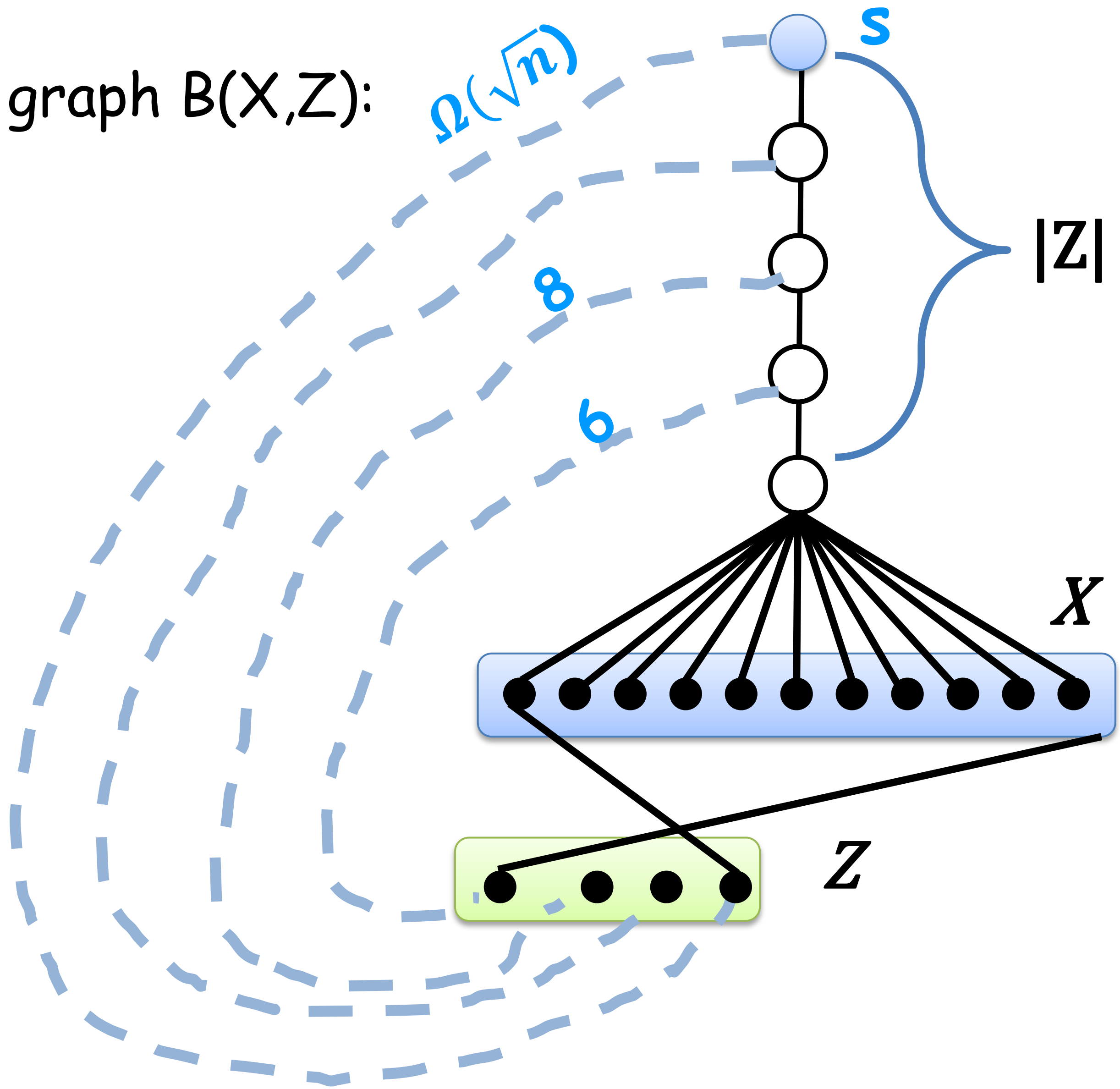
The Lower Bound Construction

- Complete bipartite graph B(X,Z):

$|X|=\Omega(n)$, $|Z|=\Omega(\sqrt{n})$

- Path of length $|Z|$

- Collection of $|Z|$ paths which are
- Vertex disjoint
- of monotone increasing lengths.

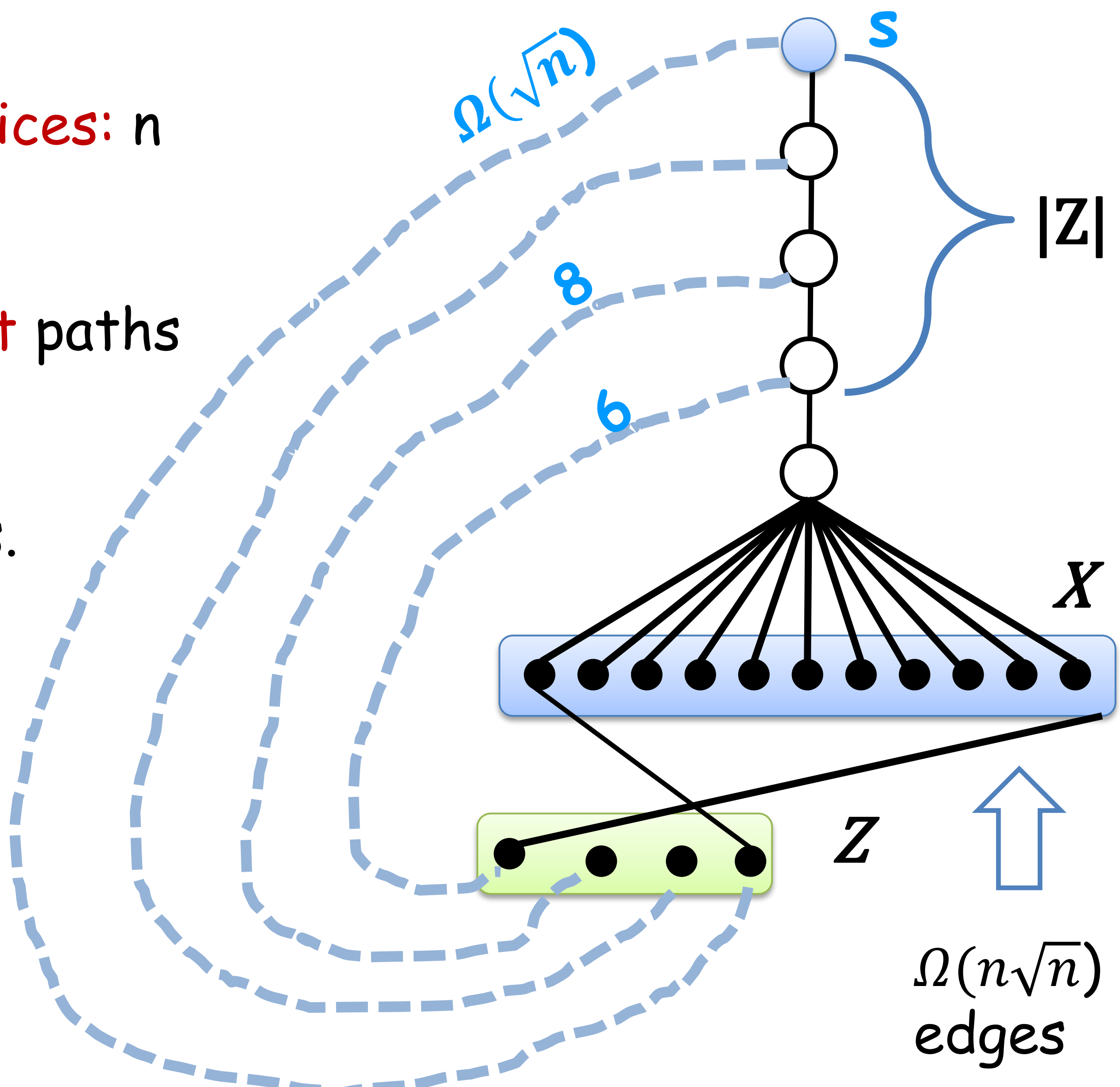


The Construction

Total number of vertices: n

$\Omega(\sqrt{n})$ vertex disjoint paths of increasing length contain $\Omega(n)$ vertices.

Total number of edges: $\Omega(n\sqrt{n})$



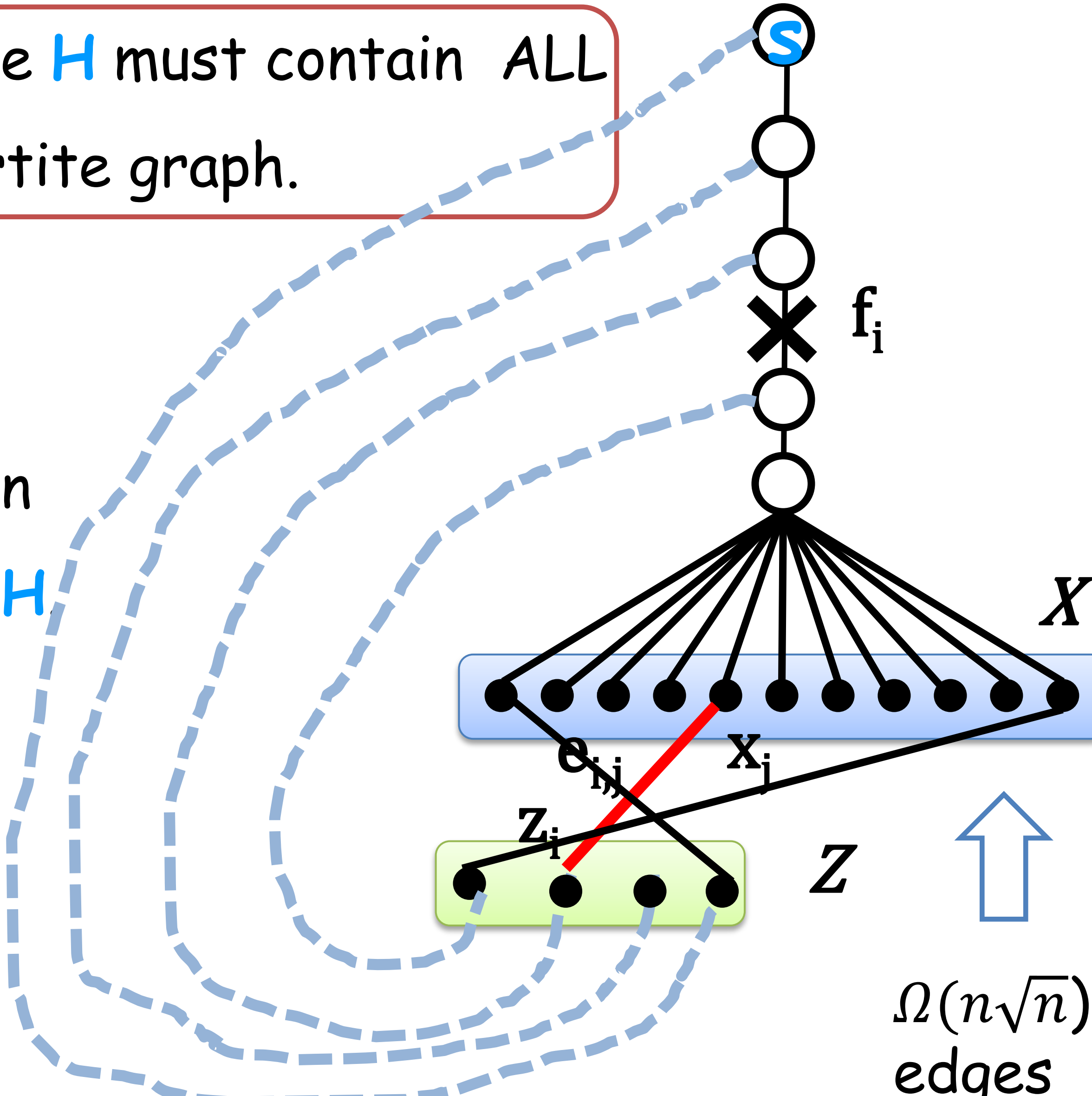
The Construction

Cl. : Every FT-BFS tree H must contain ALL the edges of the bipartite graph.

- By contradiction:

Assume there exists an edge $e_{i,j}$ that is not in H.

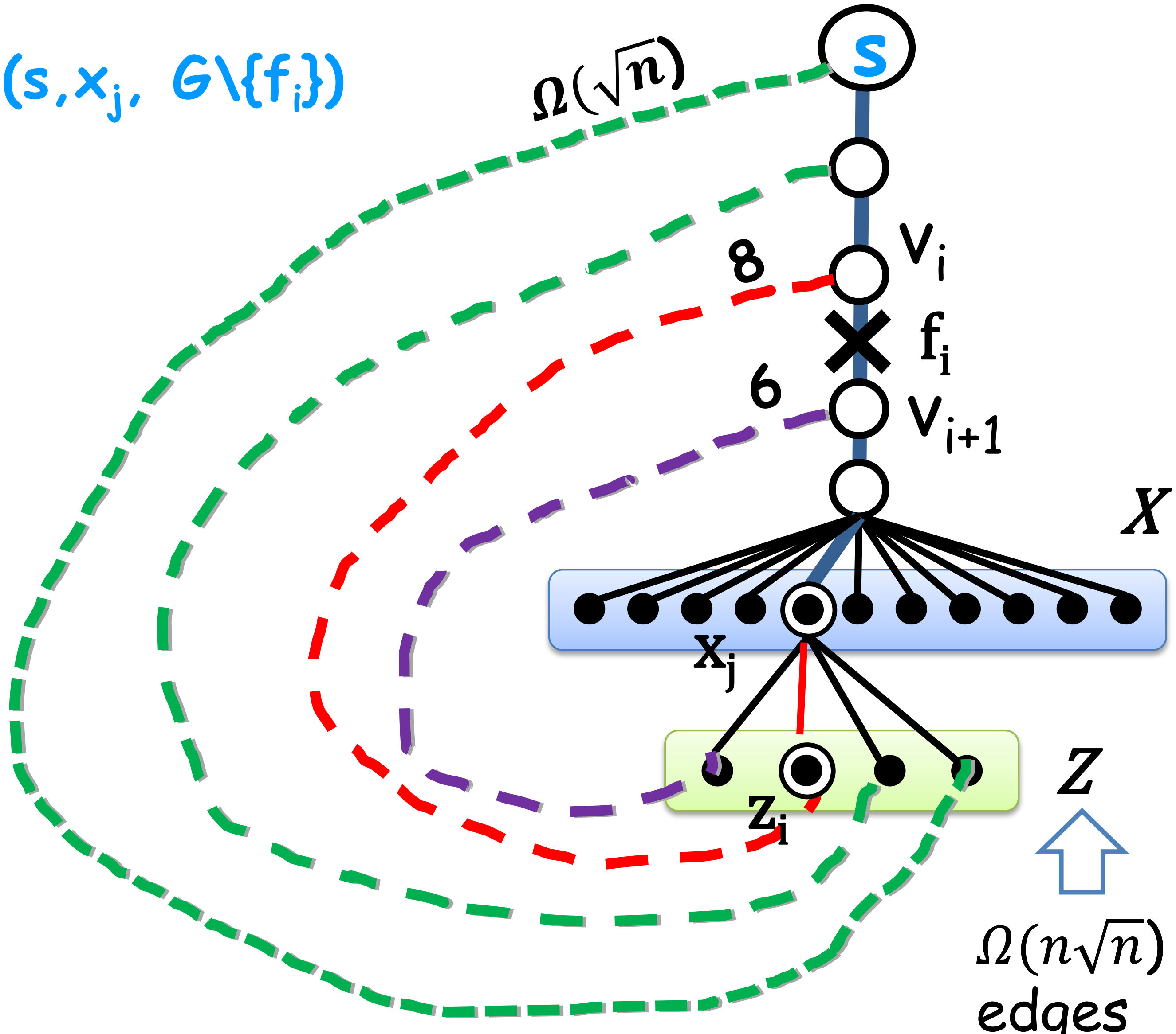
- Consider the case where f_i fails.



The Construction

$d(s,x_j, H \setminus \{f_i\}) > d(s,x_j, G\setminus \{f_i\})$

Contradiction since H is an FT-BFS tree.



Outline

- ❑ Related work
- ❑ Lower bound construction
- ❑ Upper bound
- ❑ Hardness and approximation algorithm.

Matching Upper Bound

Theorem:

For every graph $G=(V,E)$ and every source $s \in V$ there exists a (polynomially constructible) FT-BFS tree H with $O(n\sqrt{n})$ edges.

Algorithm for constructing FT-BFS

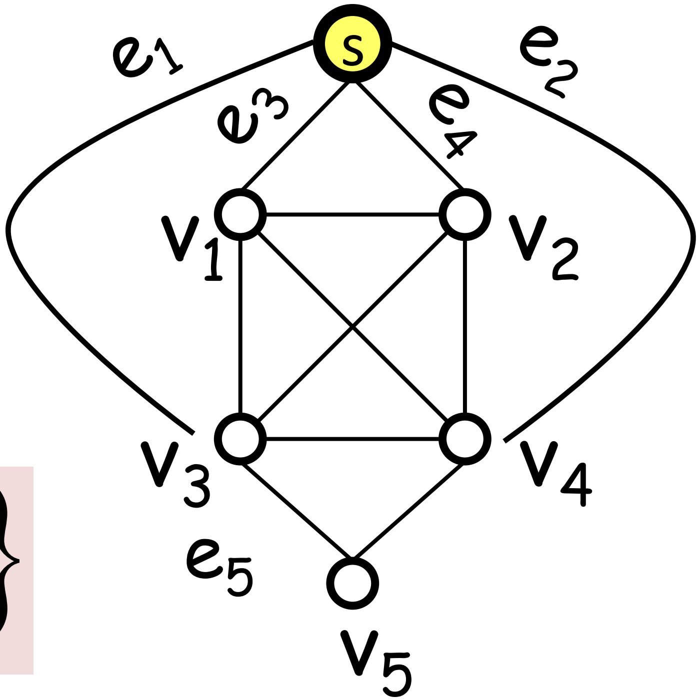
Input: unweighted graph $G=(V,E)$, source vertex s.

Output: FT-BFS tree $H \subseteq G$.

** Assume that all shortest paths in G are unique.*

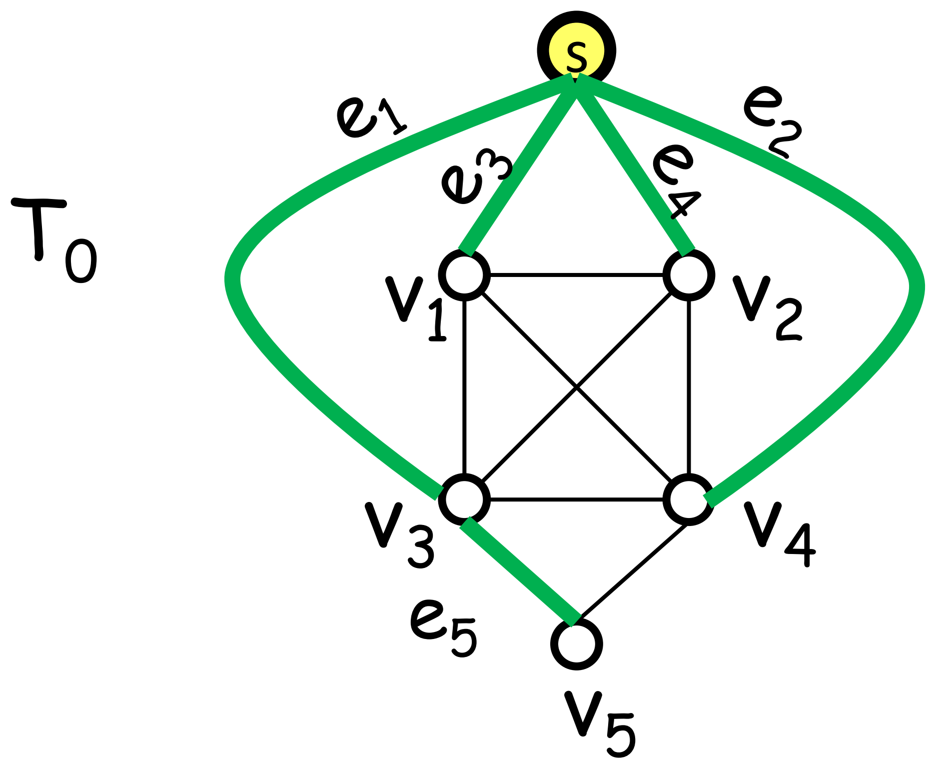
- ❑ $T_0 := \mathrm{BFS}(s, G)$
- ❑ $T_e := \mathrm{BFS}(s, G \setminus \{e\})$

$$H = T_0 \bigcup \{T_e \mid e \in T_0\}$$



Algorithm for constructing FT-BFS

- $T_0 := BFS(s, G)$
- $T_e := BFS(s, G \setminus \{e\})$

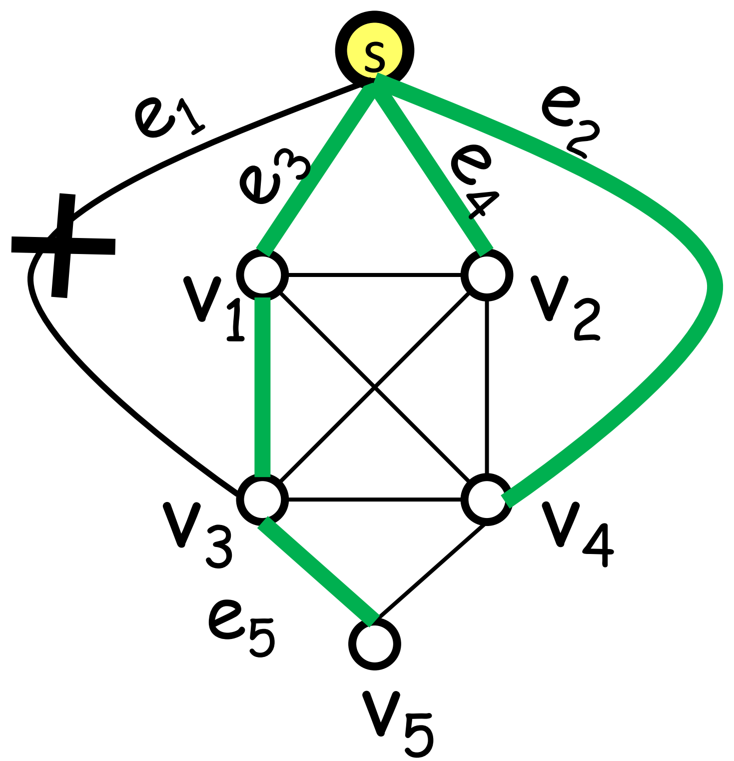


Algorithm for constructing FT-BFS

- $T_0 := BFS(s, G)$

- $T_e := BFS(s, G \setminus \{e\})$

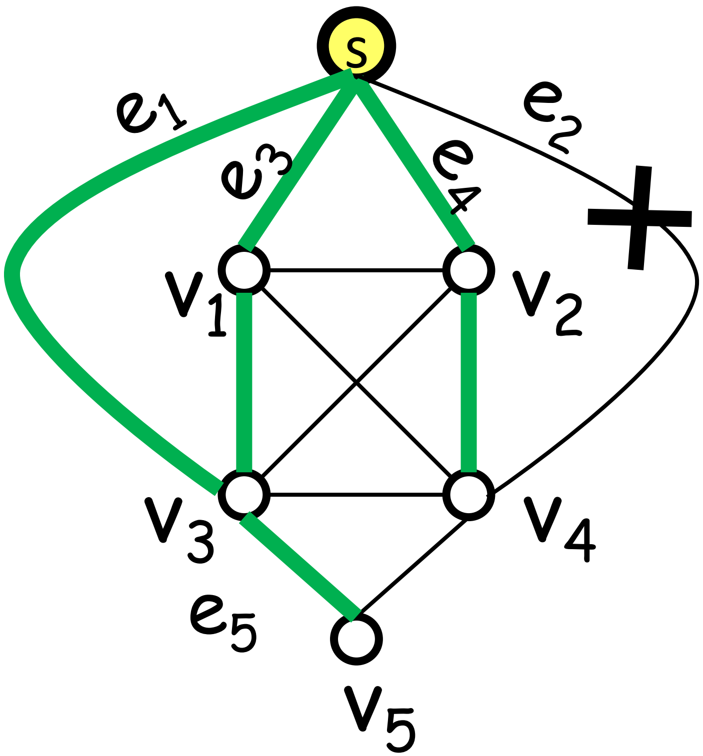
T_{e_1}



Algorithm for constructing FT-BFS

- ❑ $T_0 := BFS(s, G)$
- ❑ $T_e := BFS(s, G \setminus \{e\})$

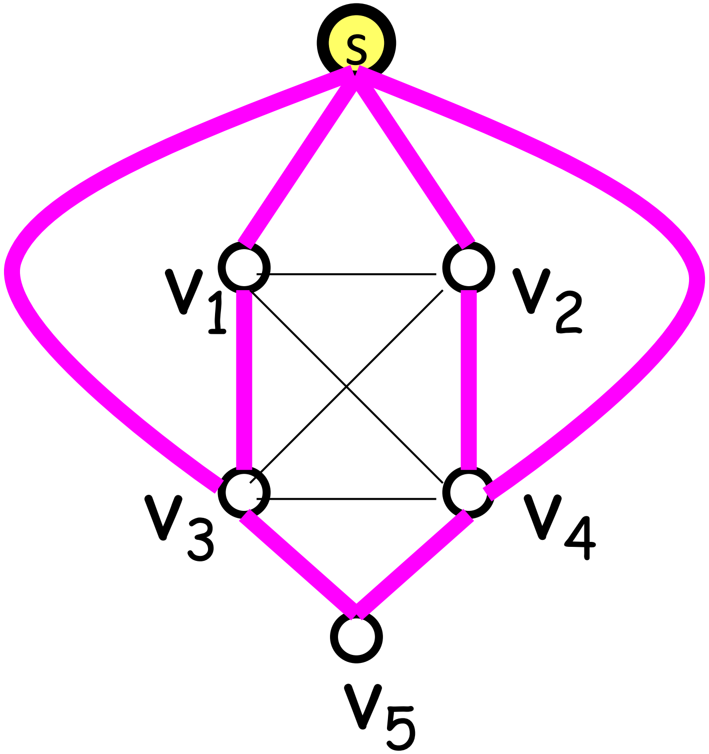
T_{e_2}



Algorithm for constructing FT-BFS

- T_0 := BFS(s, G)
- T_e := BFS(s, G \{e})

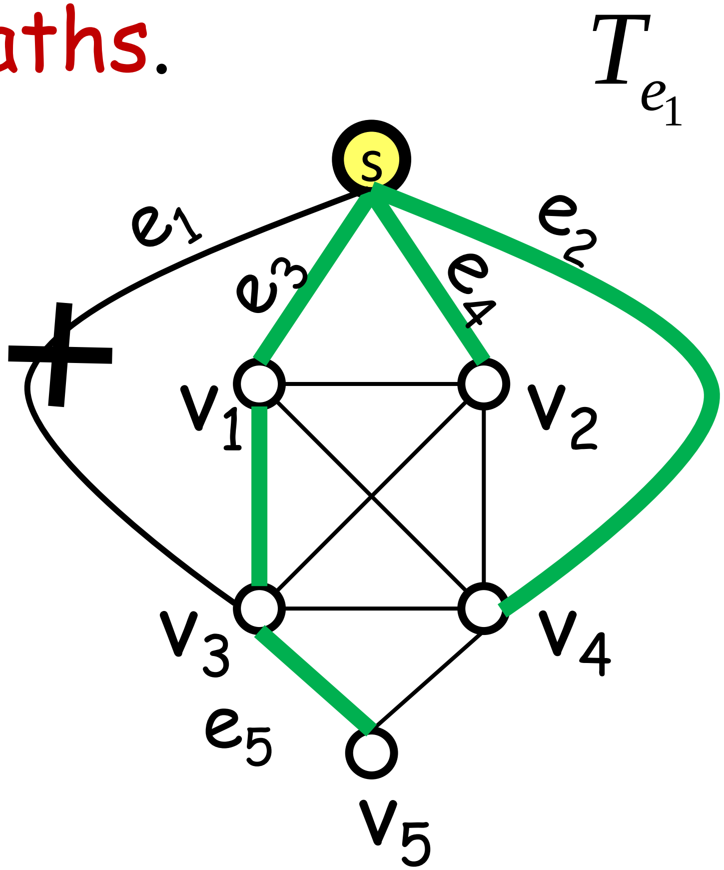
H



Correctness

Recall: $P(s,t,e)$ is the s-t shortest path in $G\backslash\{e\}$.

H contains the collection of all single source replacement paths.



The replacement path

$P(s,v_5,e_1)$ is the s-t path in

T_{e1}=BFS(s, $G\backslash\{e_1\}$).

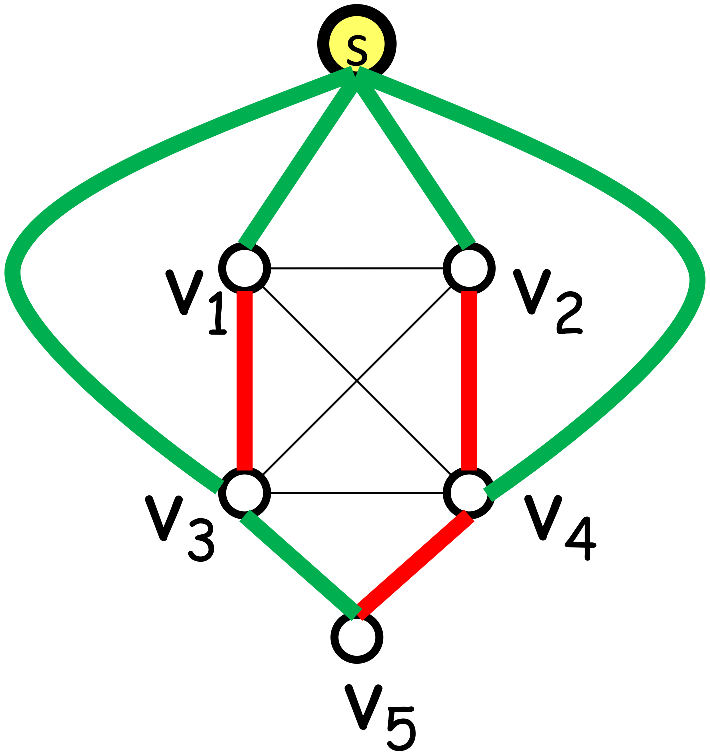
Size Analysis - Basic Intuition

An edge e in H is new if it is not in T_0.

Lemma:

Every vertex t has at most

$O(\sqrt{n})$ new edges in H.



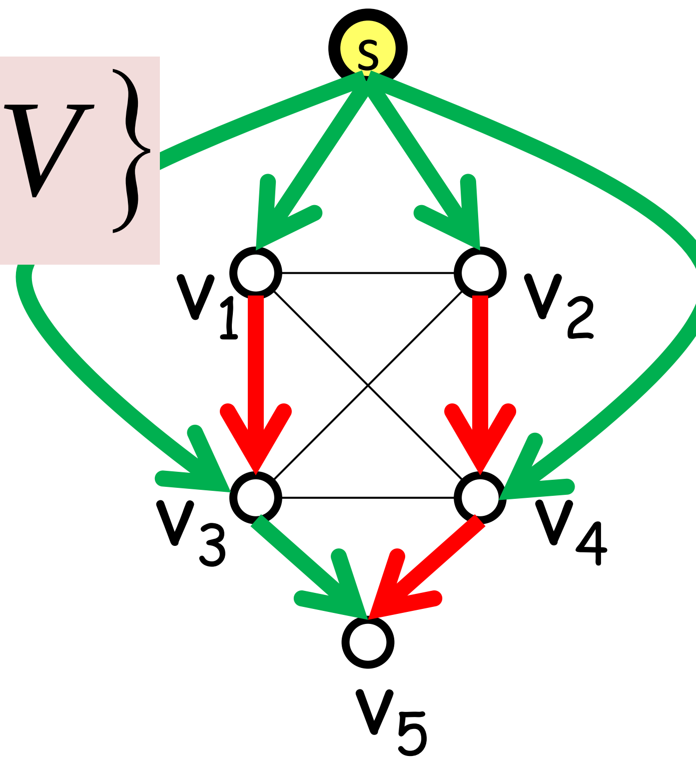
Size Analysis - Basic Intuition

$\pi(s,t,T)$: s-t path in tree T

New(t) ={ Last edge of $\pi(s,t,T_e)$, $e \in T_0$} \ T_0

$$H = T_0 \bigcup \{New(t), t \in V\}$$

$$H = T_0 \bigcup \{T_e \mid e \in T_0\}$$



Size Analysis - First Bound

$\pi(s,t,T)$: s-t path in tree T

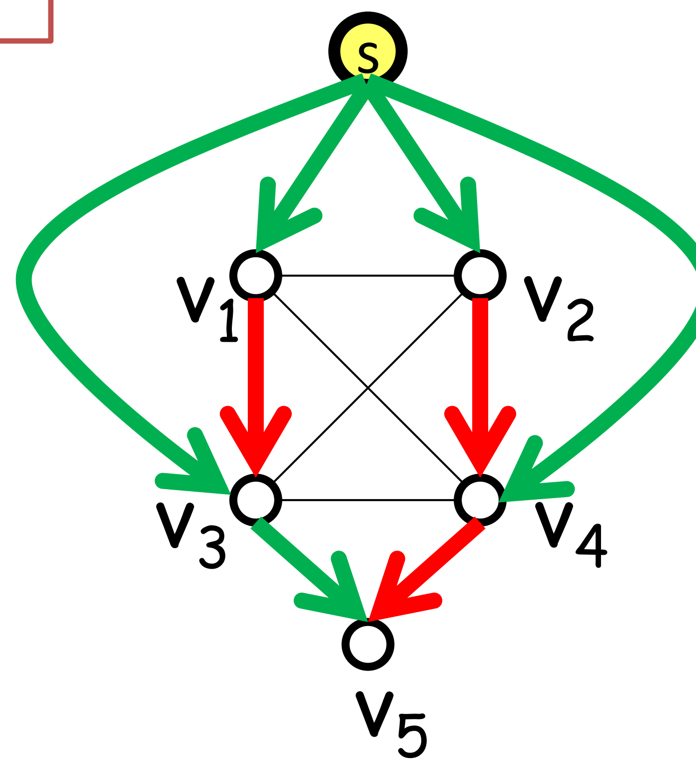
New(t) ={ Last edge of $\pi(s,t,Te)$, $e \in T_0$} \ T_0

Cl. 1: | New(t)| $\leq$ dist(s,t,G)

Proof:

If last edge of $\pi(s,t,T_e)$

is new then $e \in \pi(s,t,T_0)$

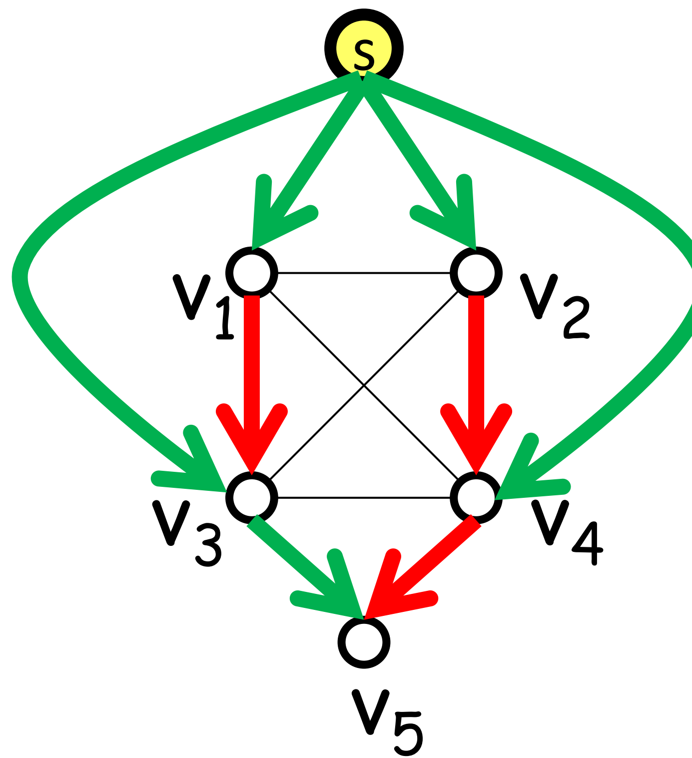


Size Analysis - Second Bound

$\pi(s,t,T)$: s-t path in tree T

New(t) $=\{$ Last edge of $\pi(s,t,T_e)$, $e \in T_0\} \setminus T_0$

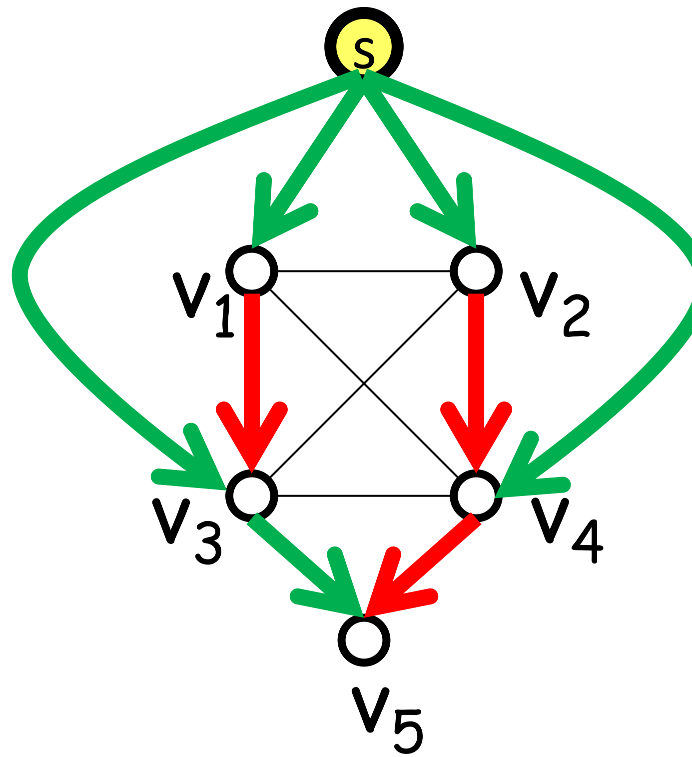
Cl. 2: $|\,\mathrm{New}(t)| \le \sqrt{2n}$



Size Analysis - Second Bound

$\pi(s,t,T)$: s-t path in tree T

New(t) ={ Last edge of $\pi(s,t,T_e)$, $e \in T_0$} \ T_0

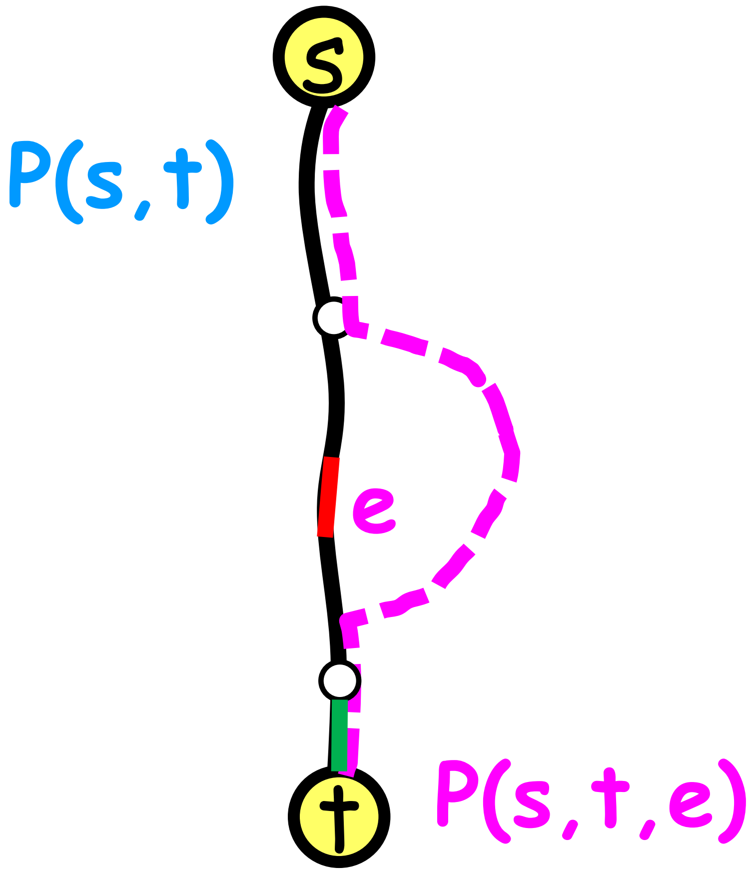


A replacement path P(s,t,e) whose last edge is new

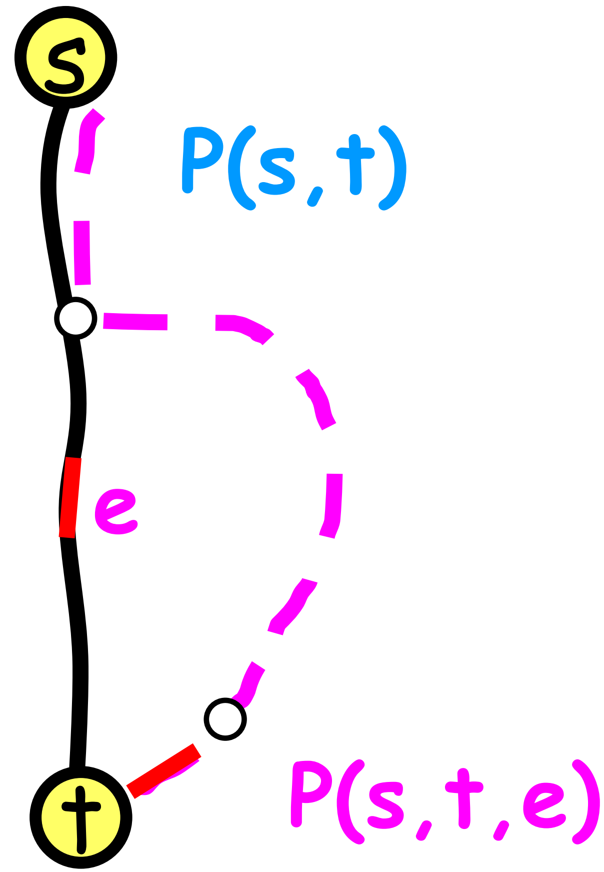
Count the number of new ending paths.

New Ending Replacement Paths

$P(s,t,e)$ is the s-t path in T_e=BFS(s, G\{e}).



Non-New Ending Path



New Ending Path

Analysis - Second Bound

Strategy: Count the number of new ending paths.

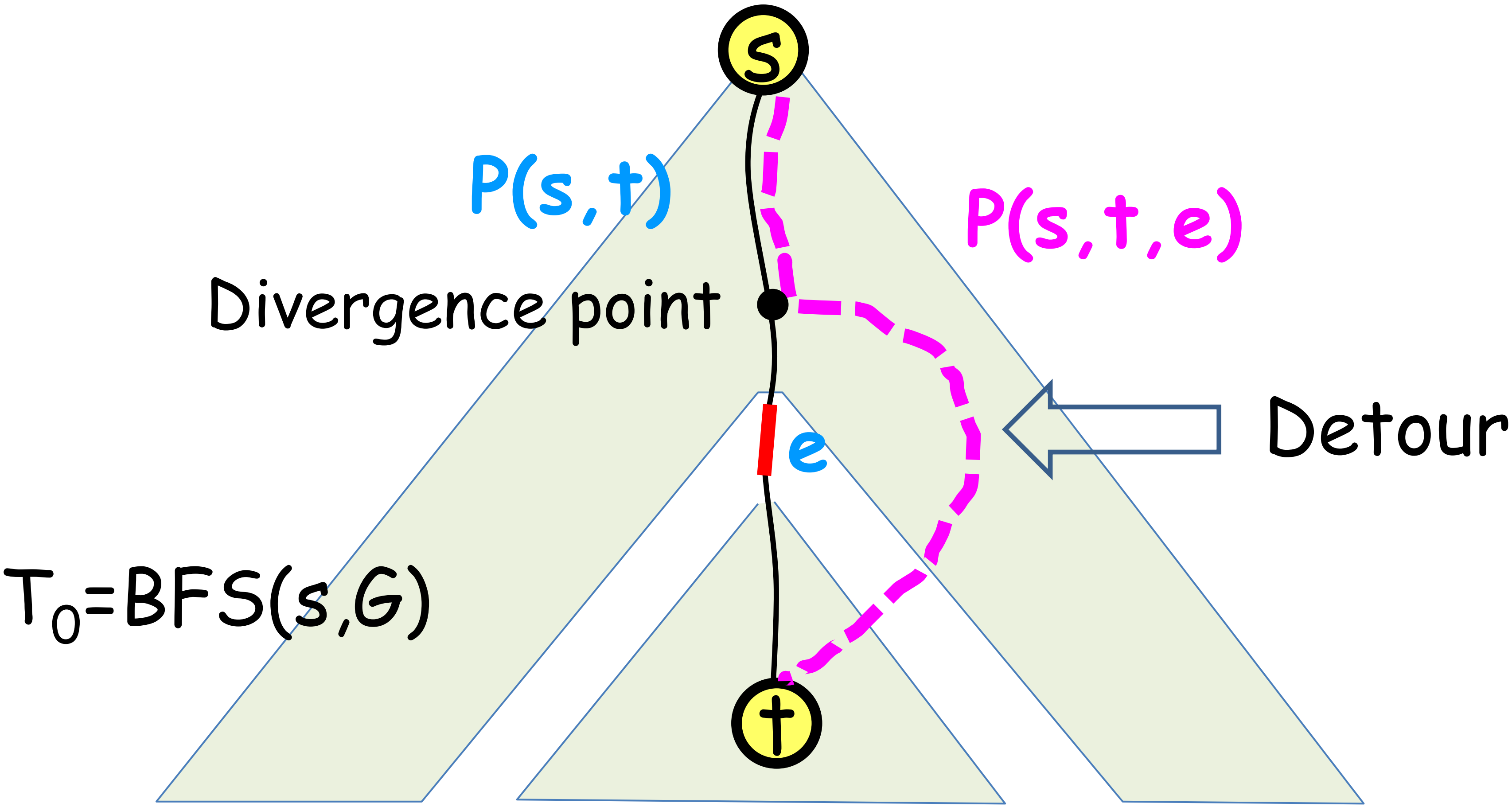
Consider the set of L new ending replacement paths

$P_1=P(s,t,e_1)$, $P_2=P(s,t,e_2)$, …., $P_L=P(s,t,e_L)$

where each P_i ends with a *distinct* new edge of t.

Show that $L \leq \sqrt{2n}$

The structure of a new ending replacement path



Lemma:

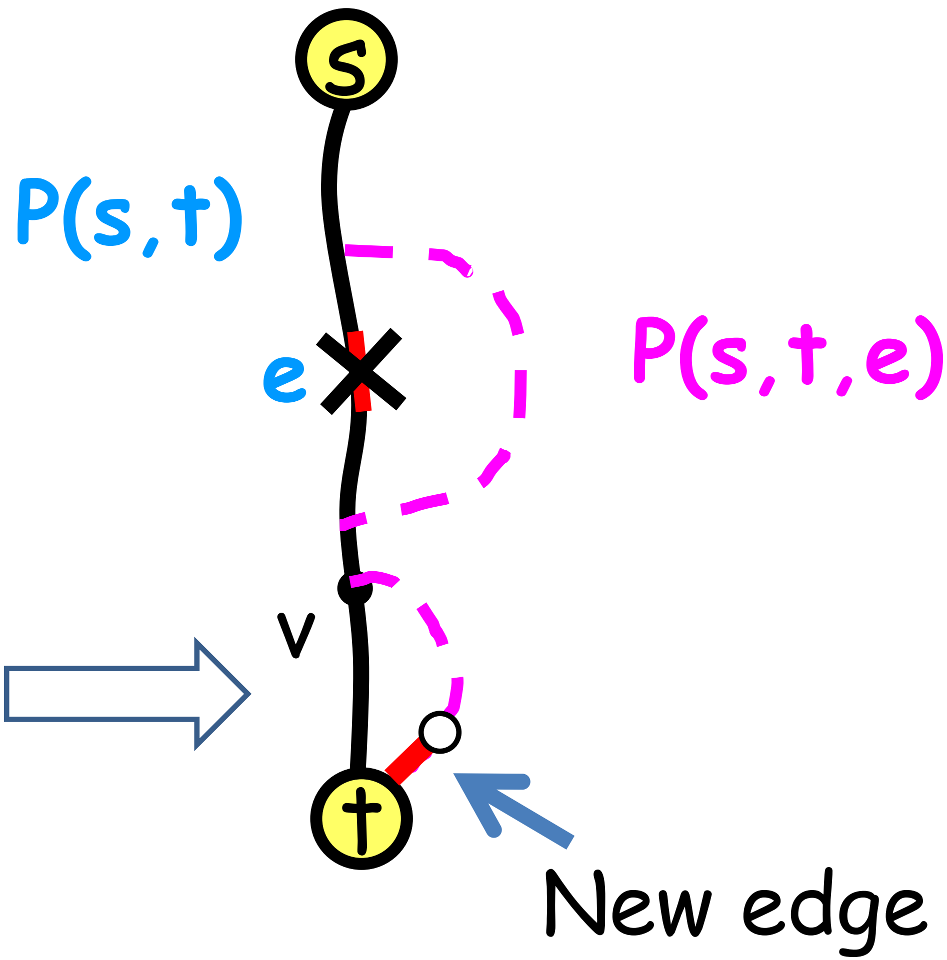
The detour segment is edge disjoint from P(s,t)

Analysis - Basic Intuition

Cl. 1: The detour segment is edge disjoint from P(s,t)

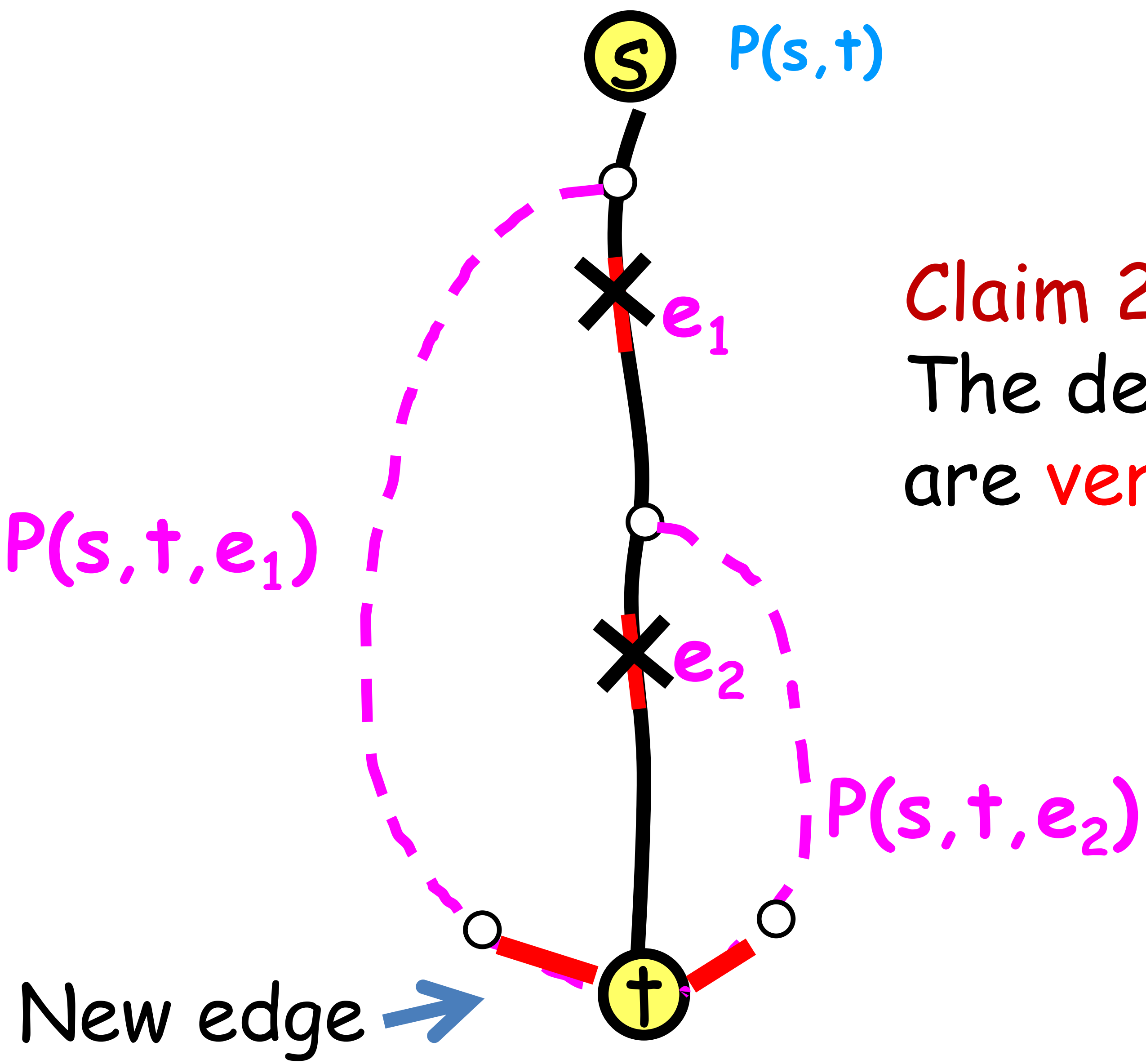
By Contradiction:

There are *two* v-t shortest paths in G\{e}.



Contradiction!

Analysis - Basic Intuition



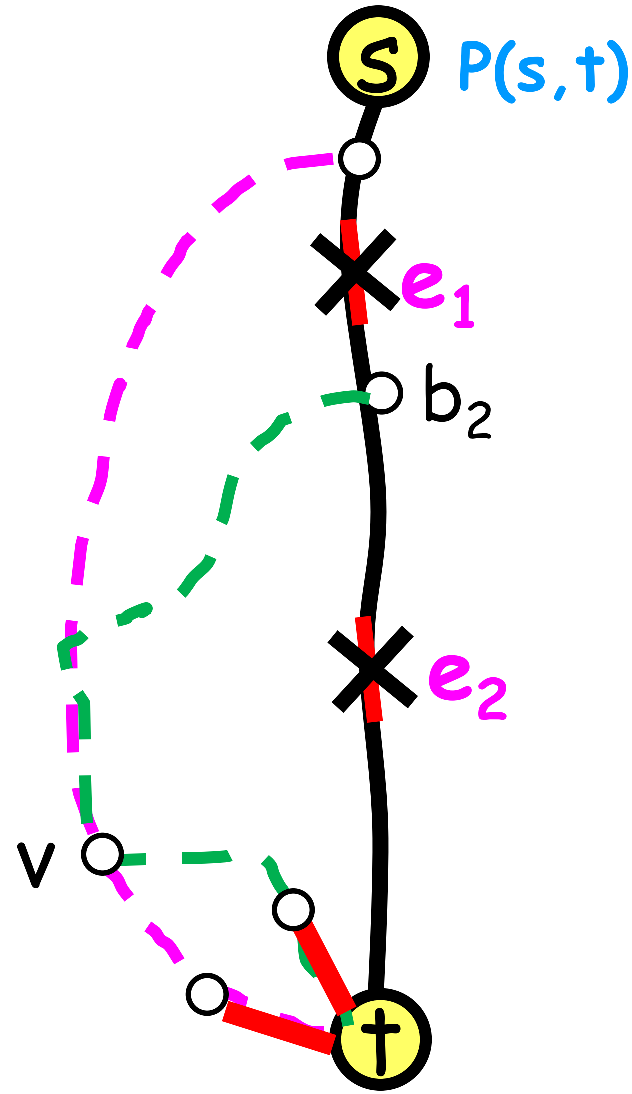
Claim 2:
The detour segments are vertex disjoint!

Analysis - Basic Intuition

Claim 2: the detours are vertex disjoint!

$P(s,t,e_1)$

$P(s,t,e_2)$



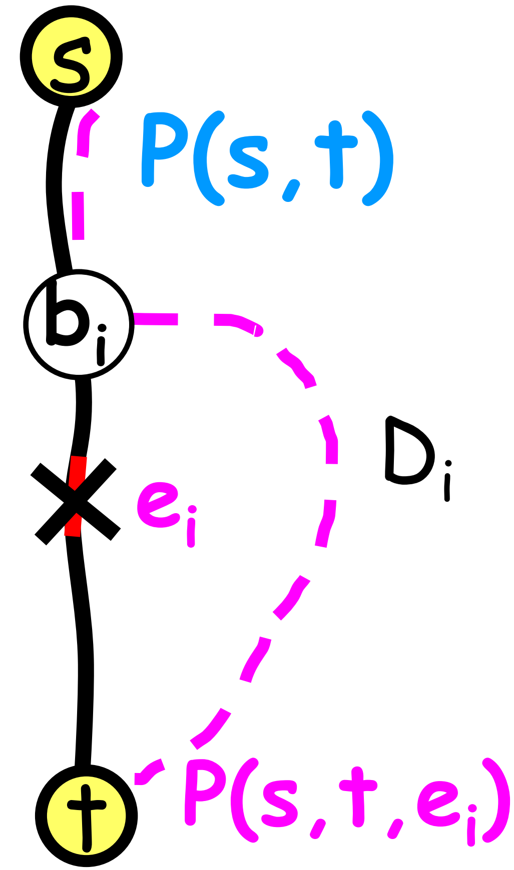
! there are two v-t shortest paths in $G \setminus \{e_1, e_2\}$.

New Ending Replacement Path

Notation:

b_i:= unique divergence point of $P(s,t,e_i)$ and $P(s,t)$.

D_i:=detour segment of $P(s,t,e_i)$.



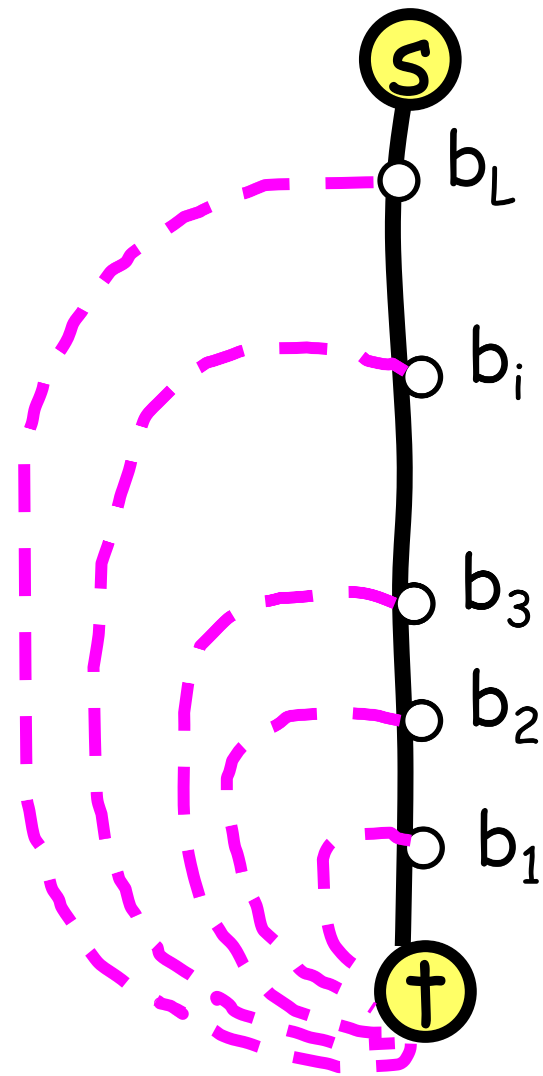
Analysis - Basic Intuition

Set of new ending replacement paths $P_1, P_2, \ldots, P_L$.

$d(s, b_1) \geq d(s, b_2) \geq \ldots d(s, b_L)$

The divergence points b_i are distinct!

$d(s, b_1) > d(s, b_2) > \ldots > d(s, b_L)$



Analysis - Basic Intuition

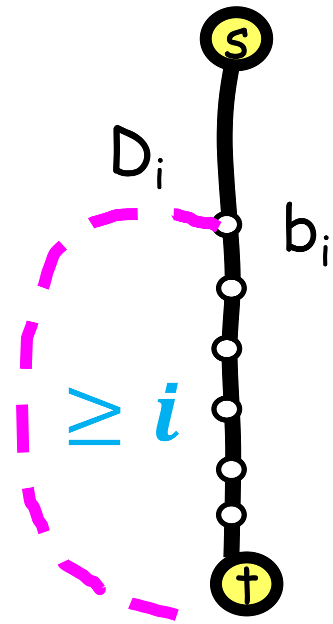
Set of new ending replacement paths $P_1, P_2, \ldots, P_L$.

- Towards contradiction assume $L > \sqrt{2n}$
- The total #vertices in the detours is:

$$|\cup_{i=1}^{L} D_i| = \sum_{i=1}^{L} |D_i| \geq \sum_{i=1}^{L} i > L^2 > n$$

Detours are vertex disjoint

Divergence points are are distinct



Contradiction!

Generalization to multiple sources (FT-MBFS)

Theorem [upper bound]

For every graph $G=(V,E)$ and every source set $S \subseteq V$

there exists a (polynomially constructible)

FT-MBFS tree H with $O(n\sqrt{|S|\,n})$ edges.

Outline

- ❑ Related work
- ❑ Lower bound construction
- ❑ Upper bound
- ❑ Hardness and approximation algorithm.

The Minimum FT-BFS tree Problem

Theorem [Hardness]

The **Minimum FT-BFS problem** is NP-hard and cannot be approximated to within a factor of **Ω(log n)** unless $\mathrm{NP} \subseteq \mathrm{TIME}(n^{\mathrm{ploylog}(n)})$.

(By a gap preserving reduction from Set-Cover)

The Minimum FT-BFS tree Problem

Theorem [Approximation]

The **Minimum FT-BFS problem** can be approximated within a factor of **O(log n)** .

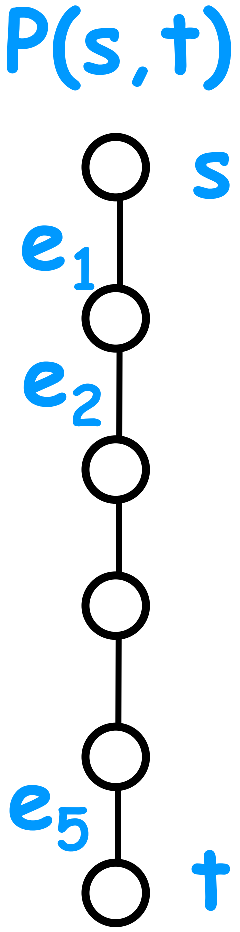
O(log n) Approximation algorithm for the Min-FT BFS problem

- Solve n-1 instances of Set-Cover.
- A Set-Cover instance of vertex t:
- Universe of vertex t: $U_t = E(P(s,t))$

Every neighbor v of t is a set S_{vt}:

$e \in P(s,t)$ is in the set S_{vt} if

$dist(s, t, G \setminus \{e\}) = dist(s, v, G \setminus \{e\}) + 1$



Summary

- FT-BFS with $O(n\sqrt{n})$ edges (tight!).
- FT-MBFS (S sources) with $O(n\sqrt{|S|\,n})$ edges (tight!).
- The **Minimum FT-MBFS problem** is NP-hard.
  - **O(log n)-approximation** (tight!).

What about **approximate** FT-BFS structure?

- **Multiplicative stretch =3:**

Upper bound: **4n** edges.

- **Additive stretch β:**

Lower bound: $\Omega(n^{1+\varepsilon_\beta})$ edges.

P, Peleg, SODA'14

Thanks !

Happy Tu-bishvat!

WEIZMANN INSTITUTE OF SCIENCE

מכון ויצמן למדע