Challenges in Distributed Shortest Paths Algorithms

Danupon Nanongkai

KTH Royal Institute of Technology, Sweden

About this talk

Main focus

s-t Distance

Known

(1+ε)-approx. in $\Theta(n^{1/2}+D)$ time

Open problem	Technical challenge
1. Exact $O(n^{1-\varepsilon}+D)$ time	Avoid bounded-hop distances!
2. Directed $O(n^{1/2}+D)$ time	Avoid sparse spanner, etc.!

Note

polylog terms will be
hidden most of the time

Plan

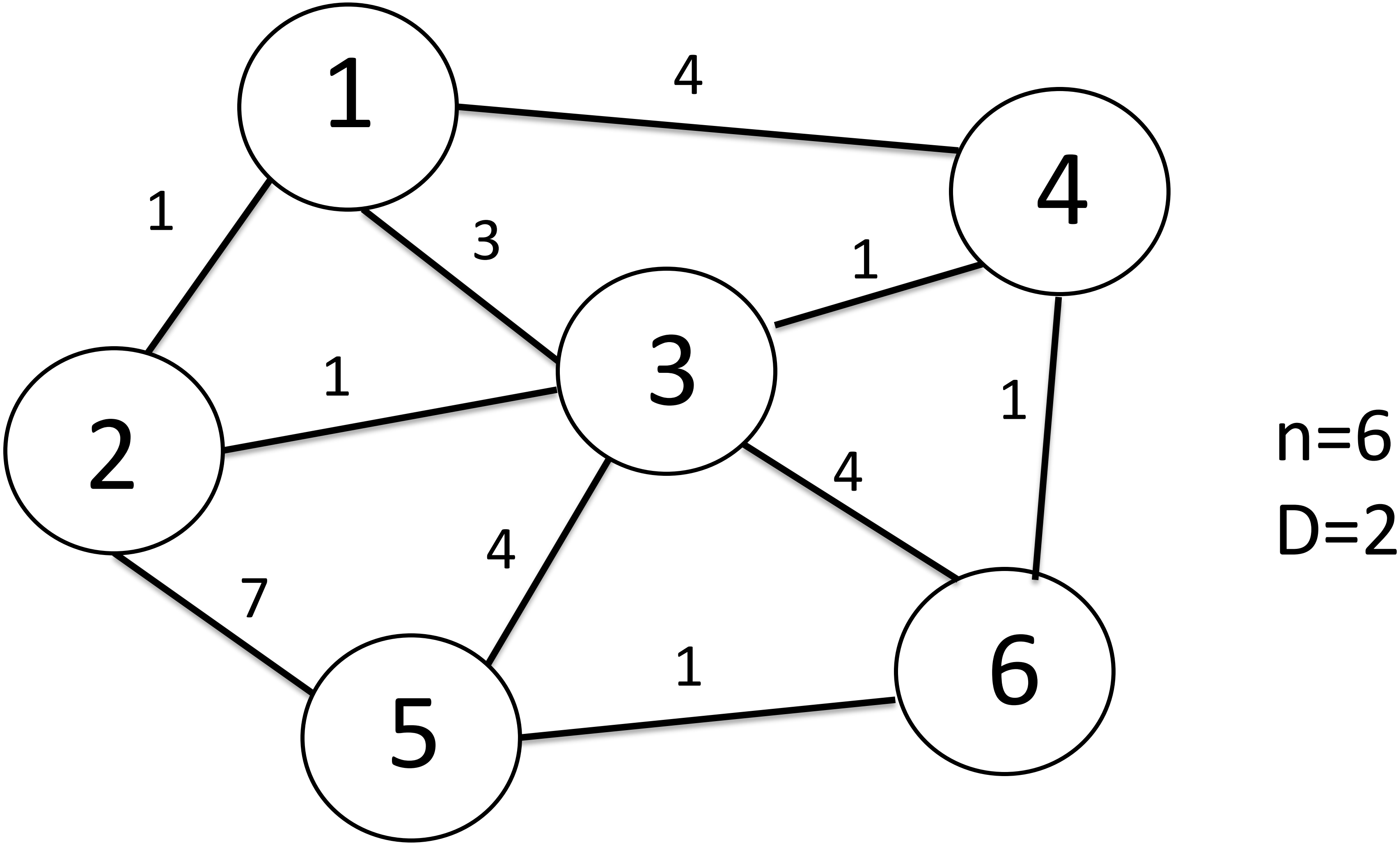
1. Problem & Known Results

2. Open Problems

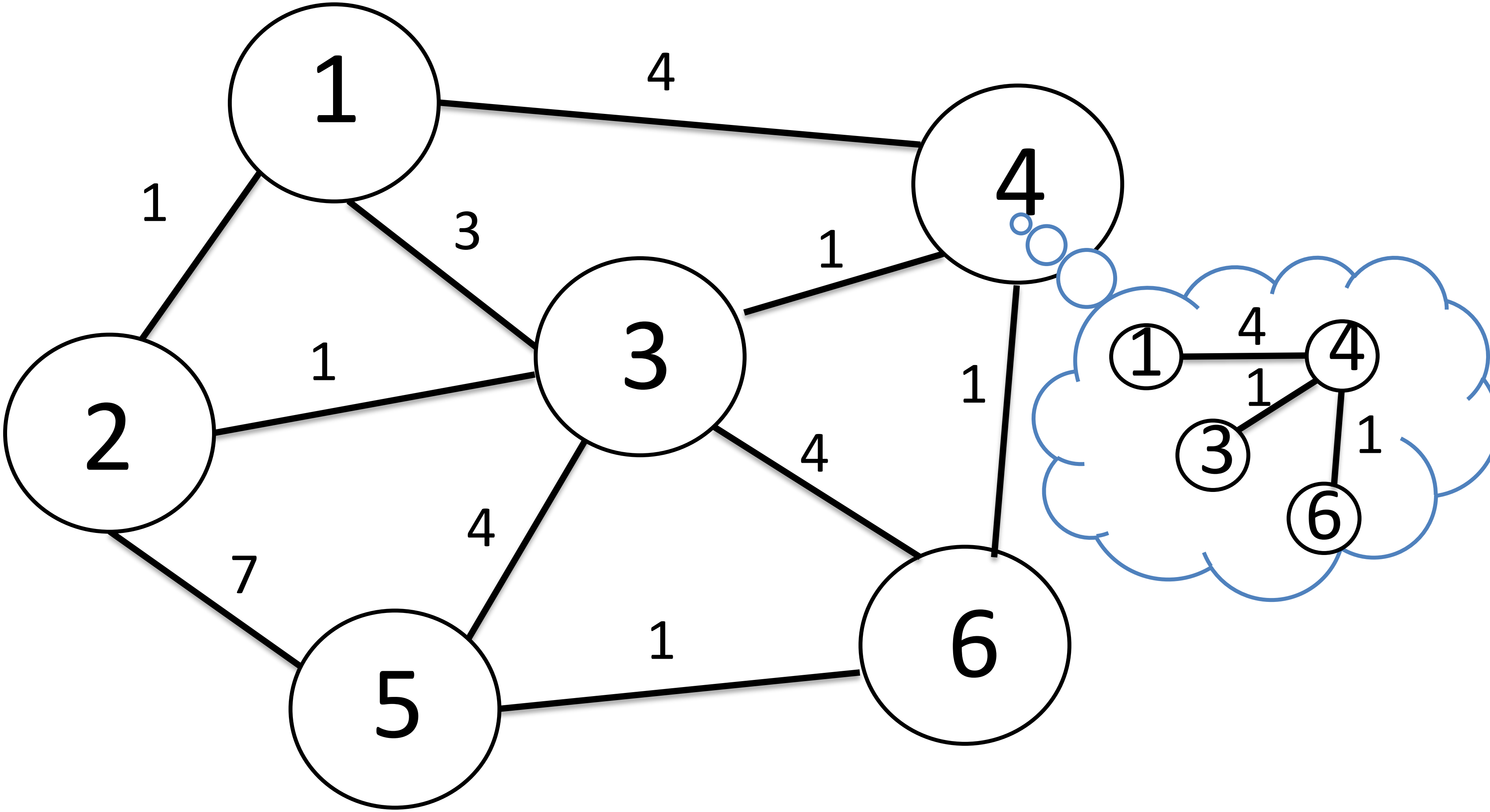
3. Technical Challenges

Part 1.1

CONGEST Model



Network represented by a weighted graph **G** with **n** nodes and hop-diameter **D**.

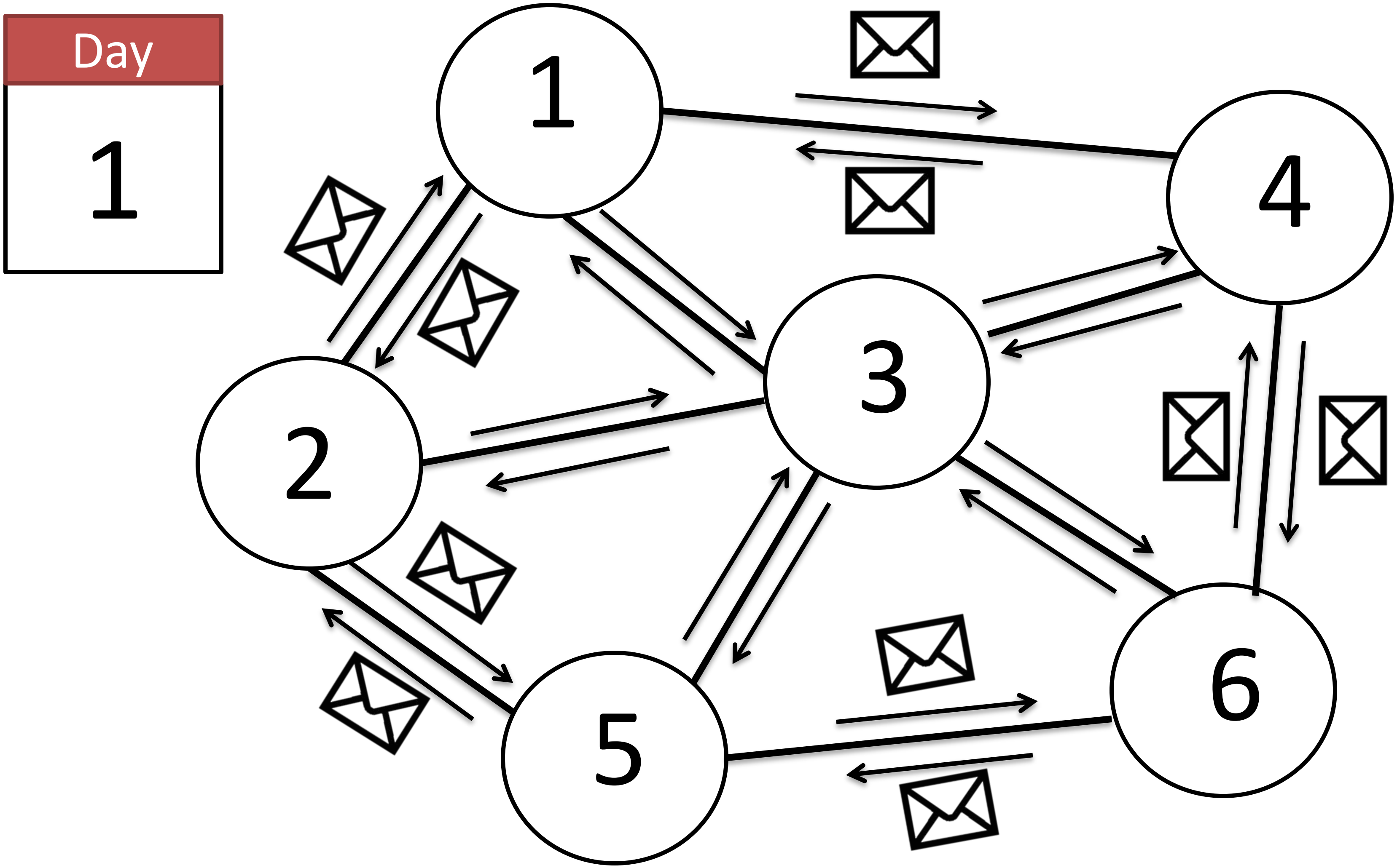


Nodes know only local information

Time complexity

“number of days”

Days: Exchange one bit



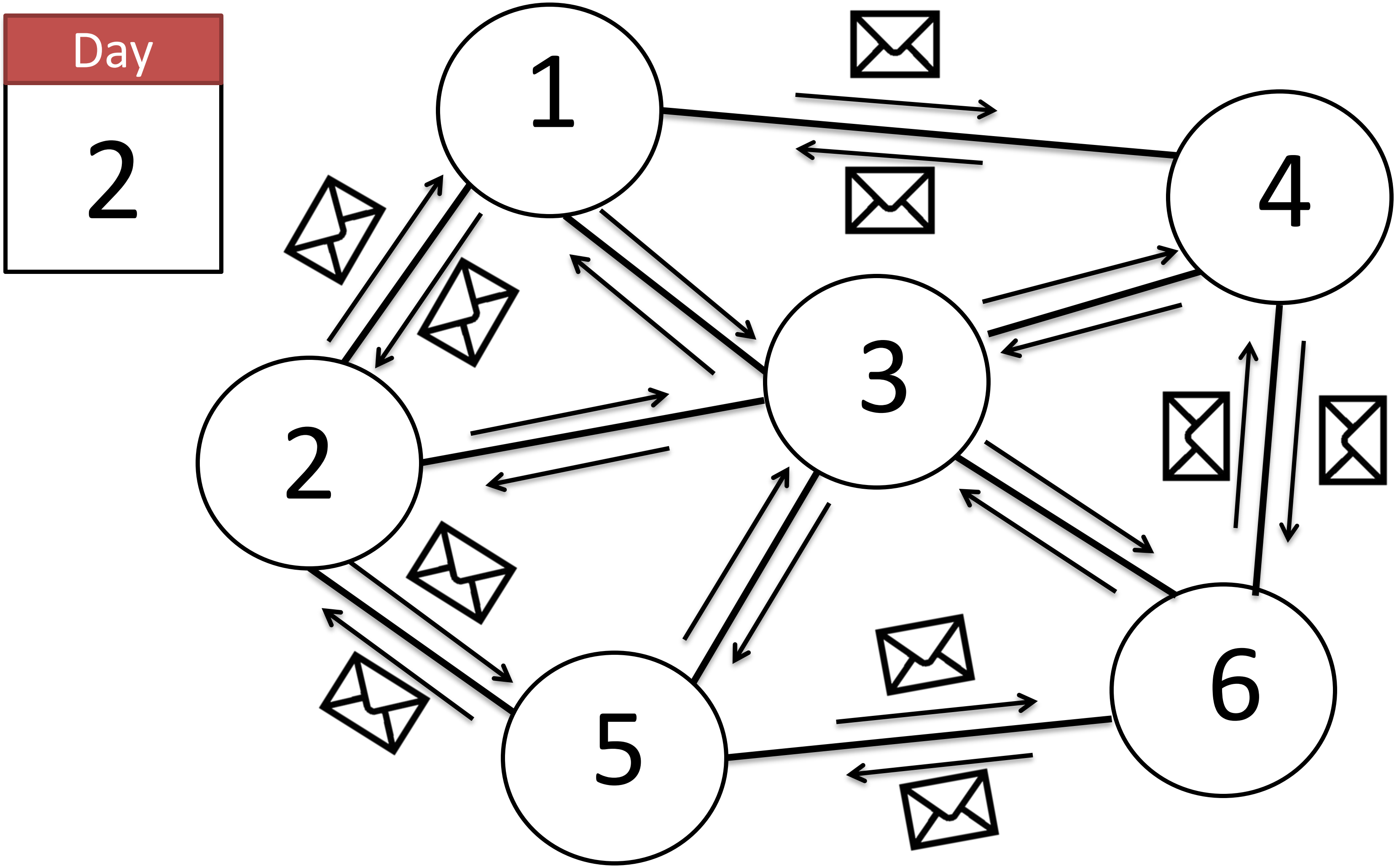
Nights: Perform local computation

Night

1

<u>Assume:</u> Any calculation finishes in one night

Days: Exchange one bit



Nights: Perform local computation

Night

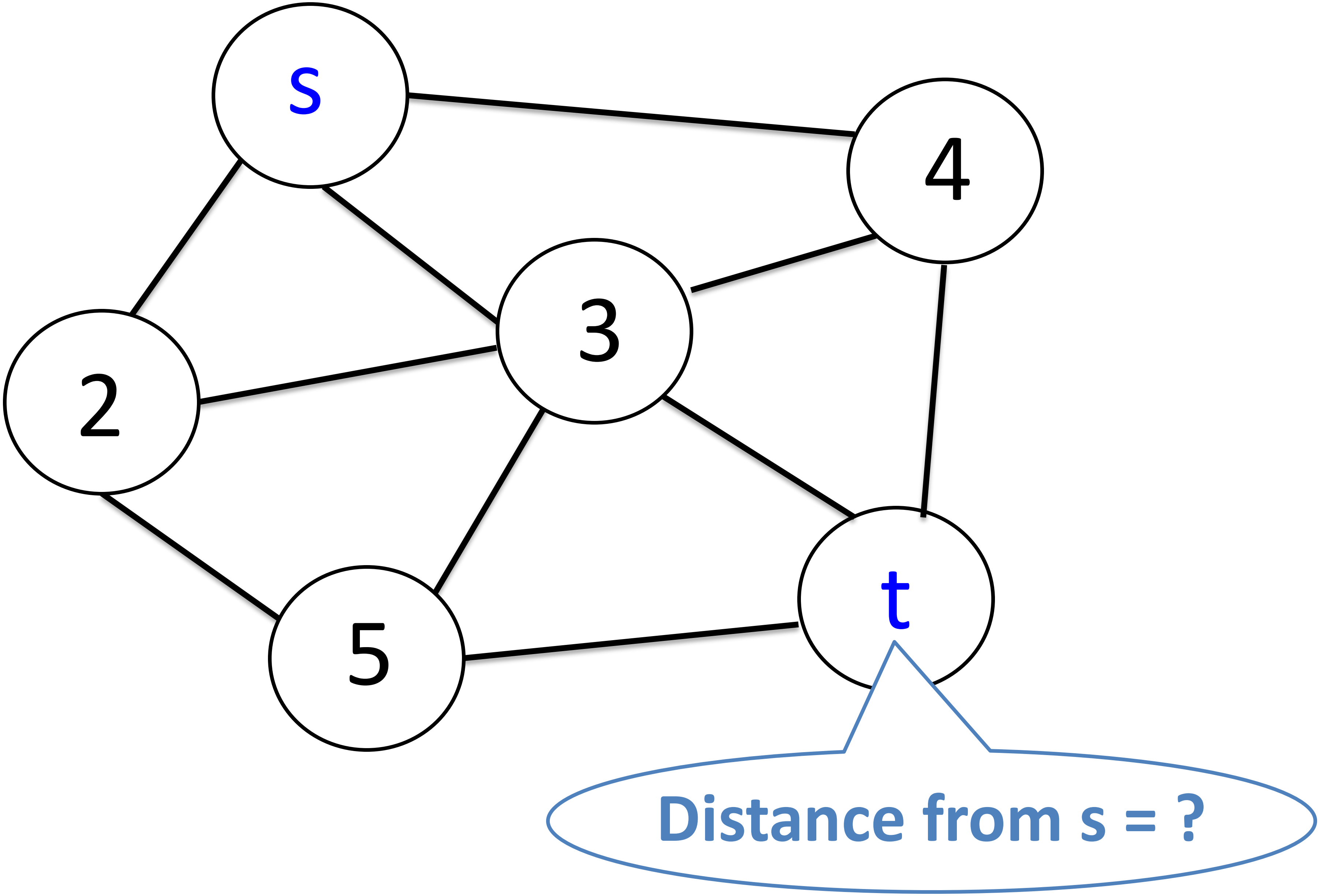
2

Finish in t days

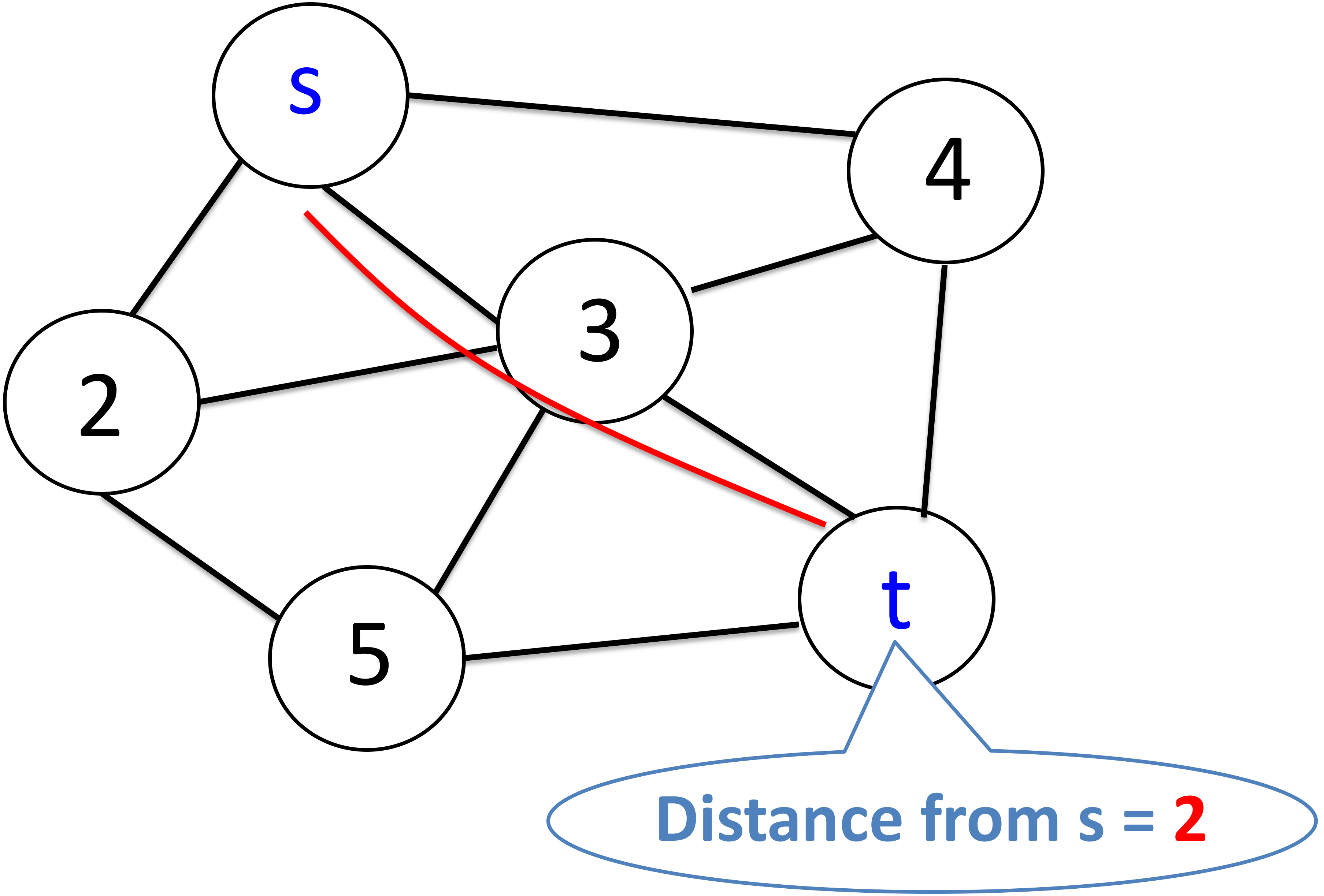
→ Time complexity = t

Part 1.2

Unweighted s-t distance



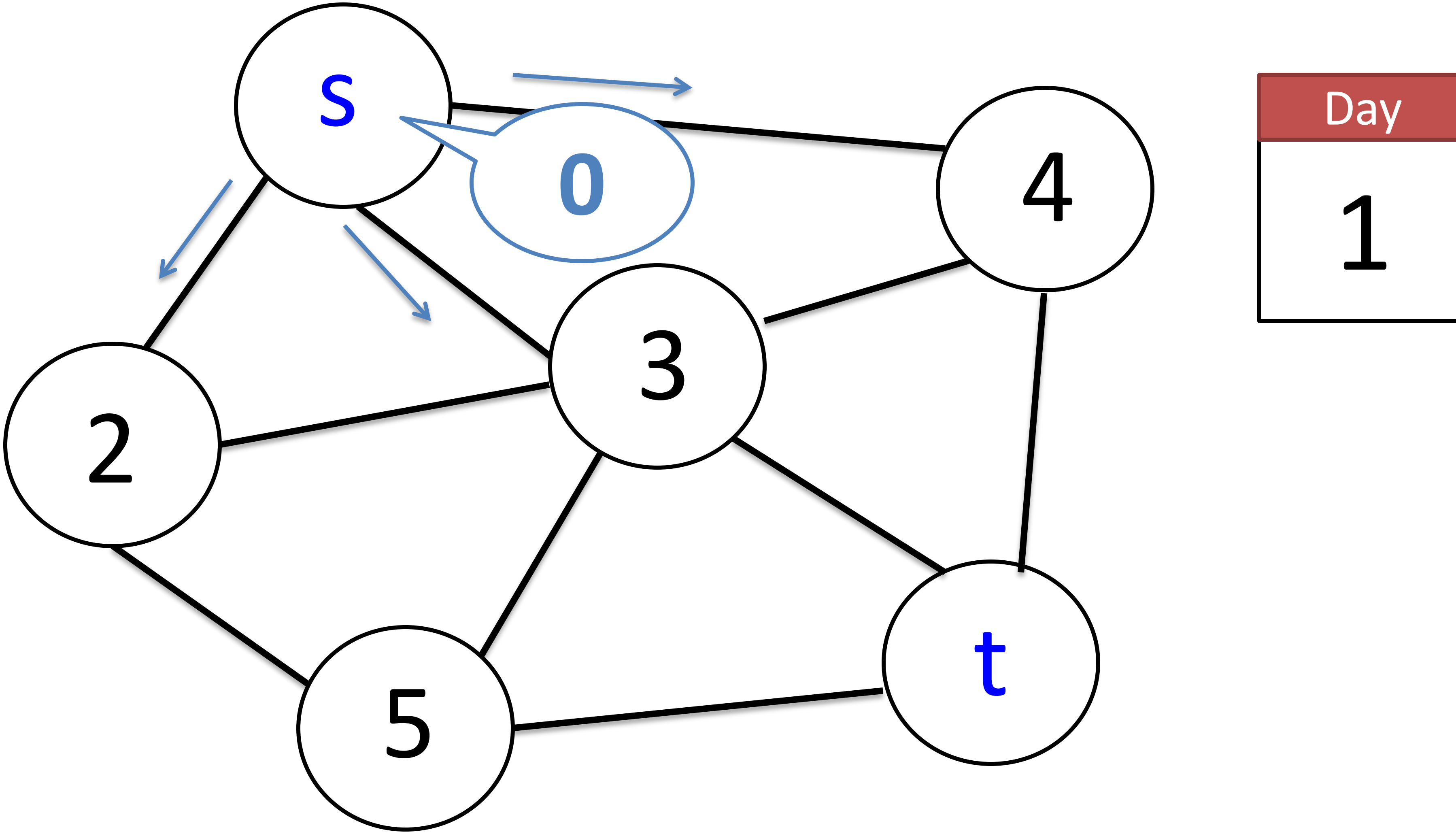
<u>Goal:</u> Node t knows distance from s


s
4
3
2
t
5
Distance from s = 2

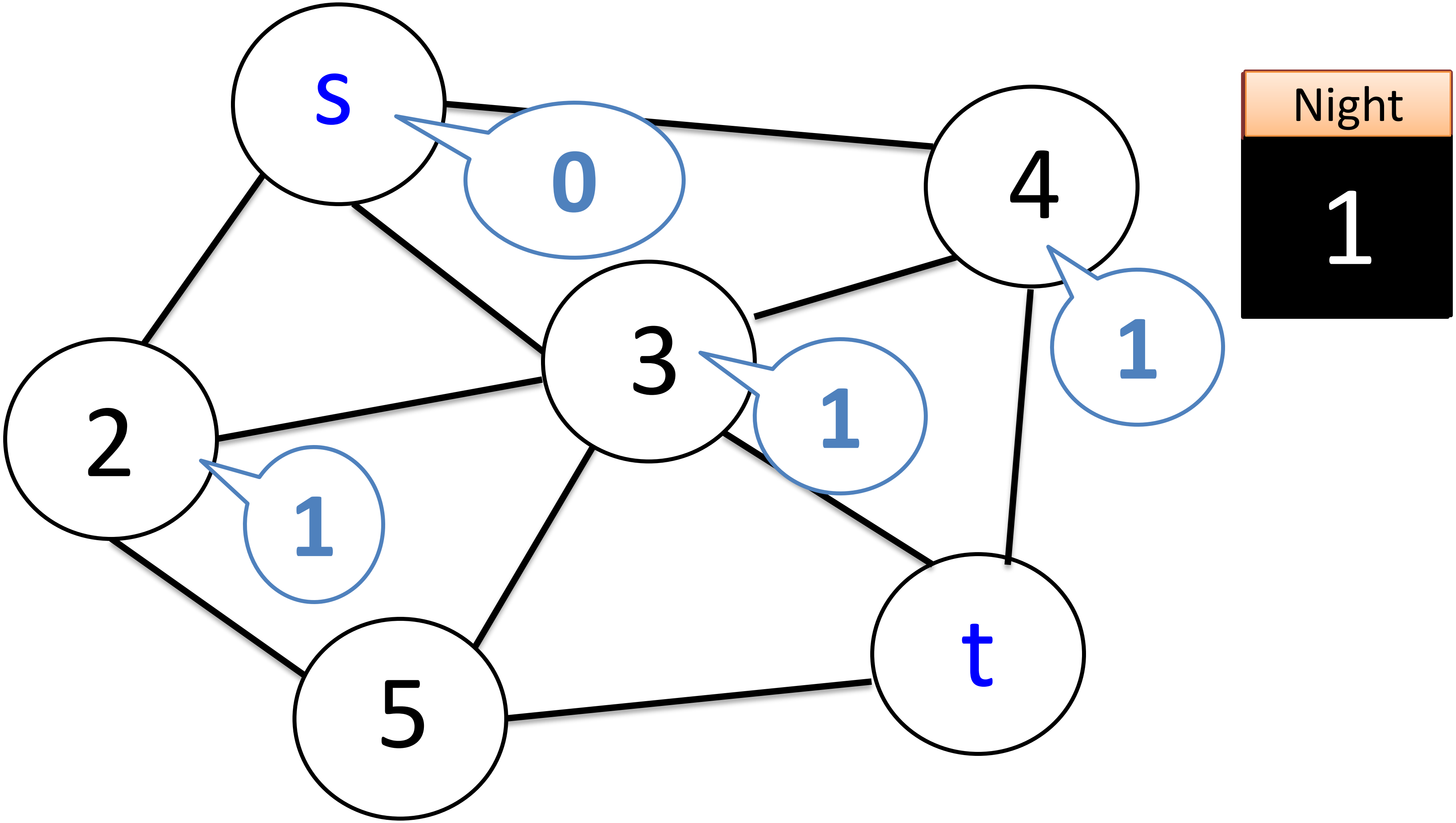
Unweighted Case

$O(D)$ time using
Breadth-First Search (BFS) algorithm.

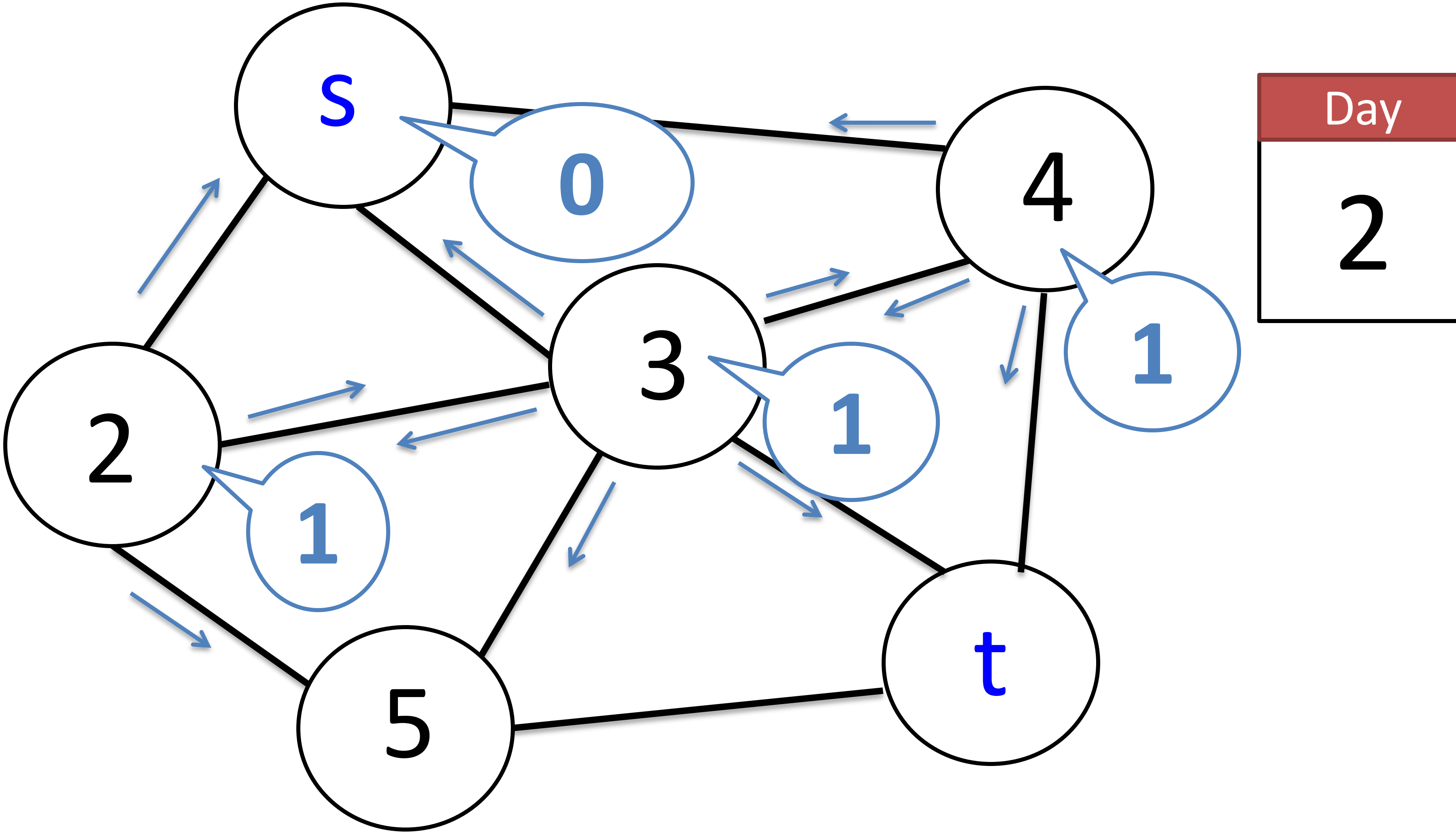
There is an **$\Omega(D)$** lower bound.



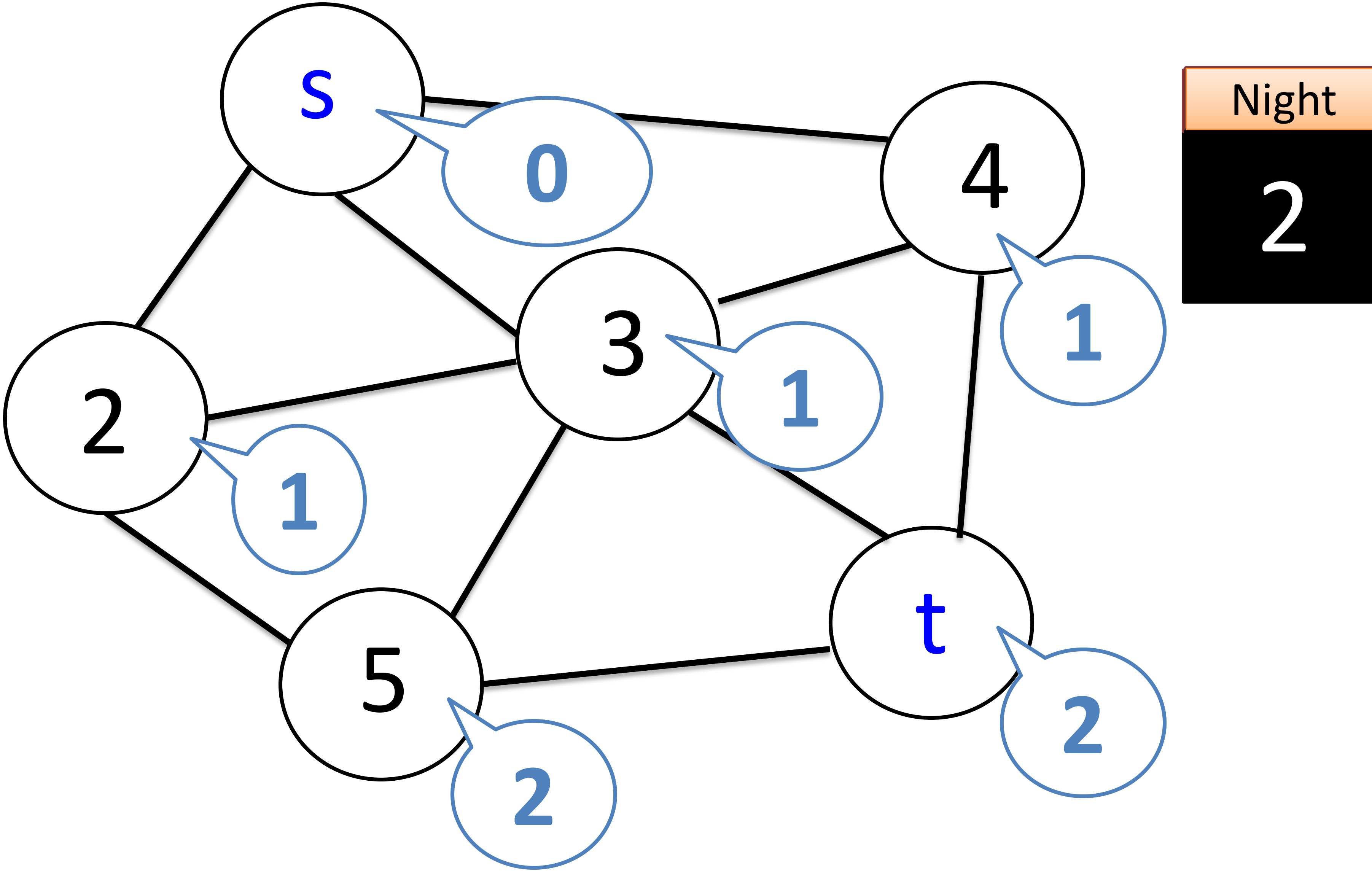
Source node sends its distance to neighbors



Each node updates its distance



Nodes tell new knowledge to neighbors

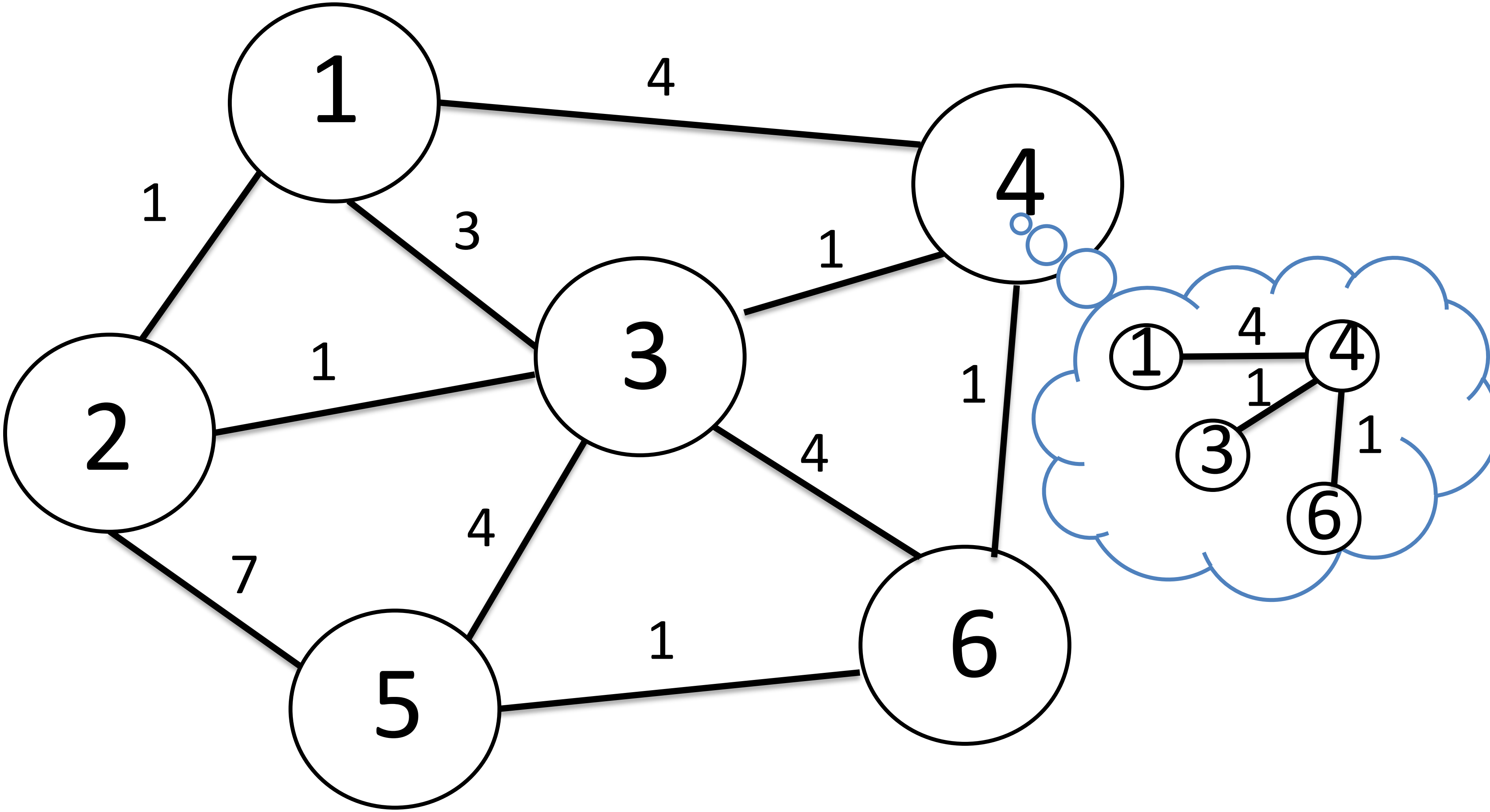


Each node updates its distance

This algorithm takes $\Theta(D)$ time

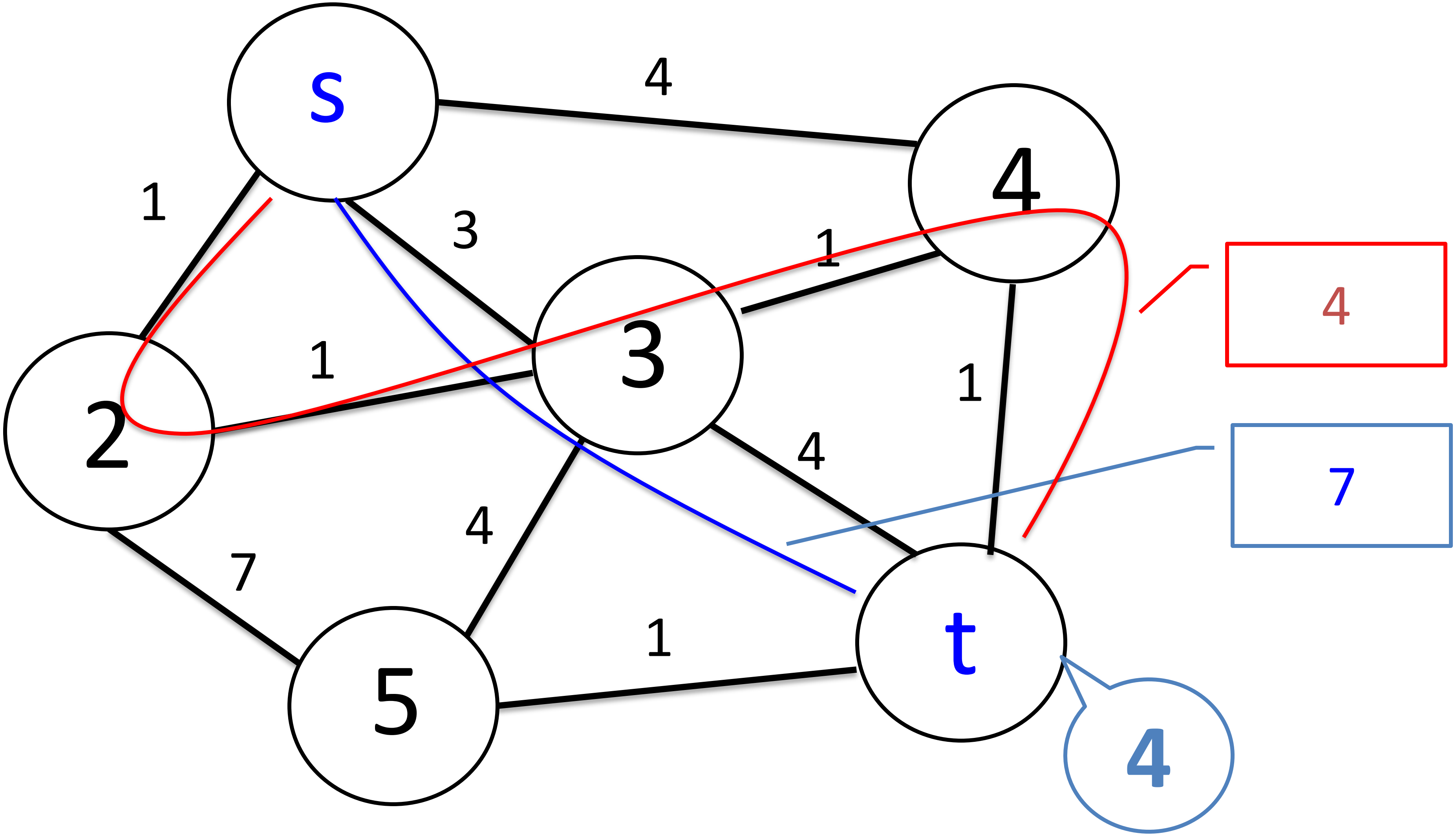
Part 1.3

How about **weighted** graphs?

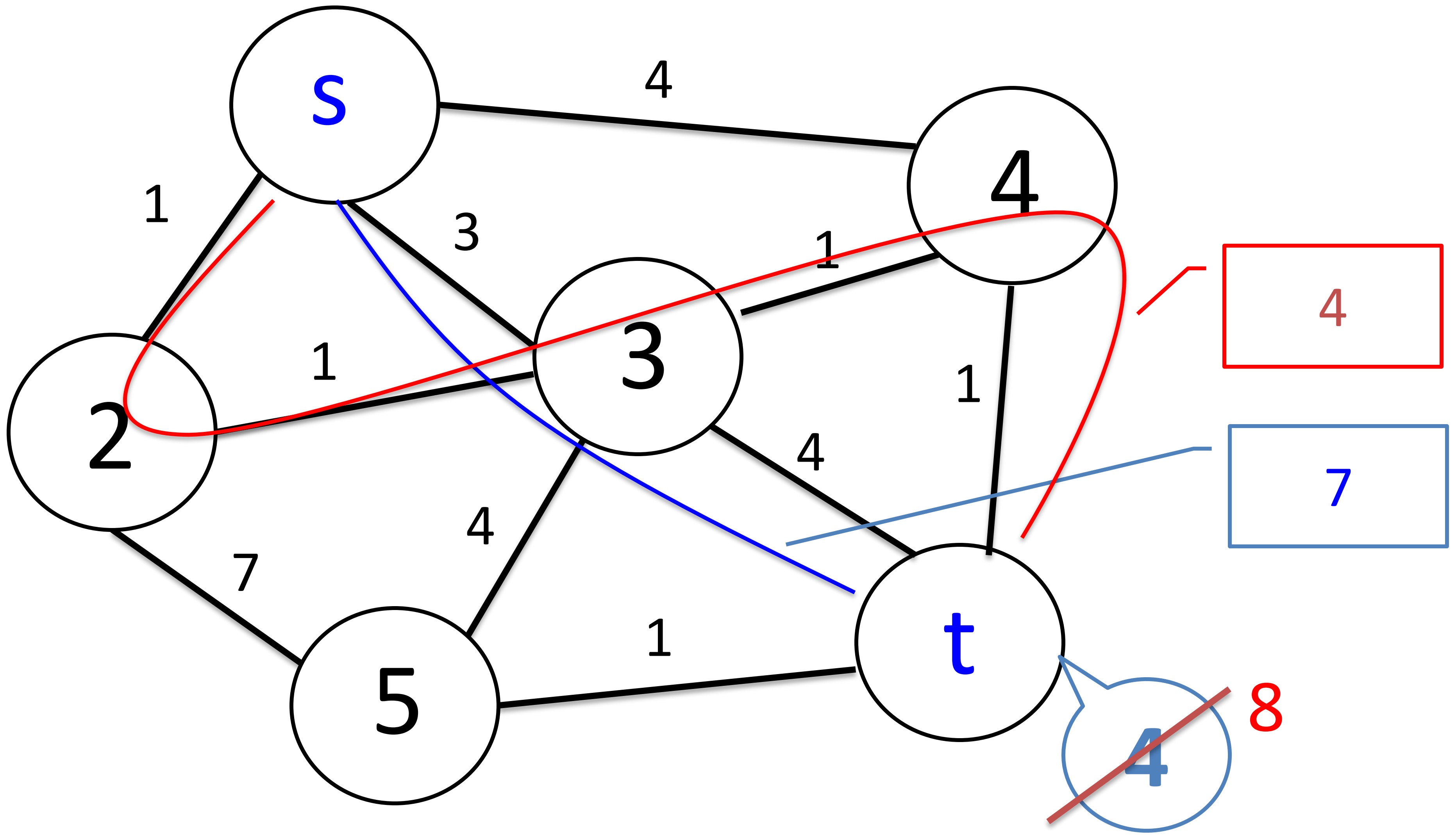


Input: weighted network

Remark: Weights do not affect the communication, edge weights ≤ O(polylog n)



s-t distance



2-approximation

A naïve solution

Aggregate everything into one node.
Then solve the problem on that node.

Time = O(# of edges)

(using “pipelining” technique)

Can we do better?

Why distributed s-t distance?

- Among active research on **distributed algorithms for basic graph problems**
  - MST, Connectivity, Matching, etc.
- **Connection** to other distributed algorithmic problems
  - Routing, APSP, Diameter, Eccentricity, Radius, etc.
- Provide **distributed algorithmic viewpoint**
  - Complement with data streams, dynamic algorithms, parallel algorithms, etc.

Part 1.4

Known Results for s-t distance

(All results also hold for **sing-source** distance.)

Reference	Time	Approximation
Folklore	$\Omega(D)$	any

Reference	Time	Approximation
Folklore	$\Omega(D)$	any
Bellman&Ford [1950s]	$O(n)$	exact

Reference	Time	Approximation
Folklore	$\Omega(D)$	any
Bellman&Ford [1950s]	$O(n)$	exact
Elkin [STOC 2006]	$\Omega((n/\alpha)^{1/2} + D)$	any α

- polylog(n/ε) factors are hidden

Reference	Time	Approximation
Folklore	$\Omega(D)$	any
Bellman&Ford [1950s]	$O(n)$	exact
Elkin [STOC 2006]	$\Omega((n/\alpha)^{1/2} + D)$	any α
Das Sarma et al [STOC 2011] Elkin et al. [PODC 2014]	$\Omega(n^{1/2} + D)$	any α also quantum

- polylog(n/ε) factors are hidden

Reference	Time	Approximation
Folklore	$\Omega(D)$	any
Bellman&Ford [1950s]	$O(n)$	exact
Elkin [STOC 2006]	$\Omega((n/\alpha)^{1/2} + D)$	any α
Das Sarma et al [STOC 2011] Elkin et al. [PODC 2014]	$\Omega(n^{1/2} + D)$	any α also quantum
Lenzen,Patt-Shamir [STOC 2013]	$O(n^{1/2+\varepsilon} + D)$	$O(1/\varepsilon)$

- polylog(n/ε) factors are hidden
- Lenzen&Patt-Shamir actually achieve more than computing distances

Reference	Time	Approximation
Folklore	$\Omega(D)$	any
Bellman&Ford [1950s]	$O(n)$	exact
Elkin [STOC 2006]	$\Omega((n/\alpha)^{1/2} + D)$	any α
Das Sarma et al [STOC 2011] Elkin et al. [PODC 2014]	$\Omega(n^{1/2} + D)$	any α also quantum
Lenzen,Patt-Shamir [STOC 2013]	$O(n^{1/2+\varepsilon} + D)$	$O(1/\varepsilon)$
N [STOC 2014]	$O(n^{1/2}D^{1/4} + D)$	$1+\varepsilon$

- polylog(n/ε) factors are hidden
- Lenzen&Patt-Shamir actually achieve more than computing distances

Reference	Time	Approximation
Folklore	$\Omega(D)$	any
Bellman&Ford [1950s]	$O(n)$	exact
Elkin [STOC 2006]	$\Omega((n/\alpha)^{1/2} + D)$	any α
Das Sarma et al [STOC 2011] Elkin et al. [PODC 2014]	$\Omega(n^{1/2} + D)$	any α also quantum
Lenzen,Patt-Shamir [STOC 2013]	$O(n^{1/2+\varepsilon} + D)$	$O(1/\varepsilon)$
N [STOC 2014]	$O(n^{1/2}D^{1/4} + D)$	$1+\varepsilon$
Henzinger,Krinninger,N [STOC 2016]	$O(n^{1/2+o(1)} + D^{1+o(1)})$ **(Deterministic)**	$1+\varepsilon$

- polylog(n/ε) factors are hidden
- Lenzen&Patt-Shamir actually achieve more than computing distances

Reference	Time	Approximation	
Folklore	$\Omega(D)$	any	
Bellman&Ford [1950s]	$O(n)$	exact	
Elkin [STOC 2006]	$\Omega((n/\alpha)^{1/2} + D)$	any α	
Das Sarma et al [STOC 2011] Elkin et al. [PODC 2014]	$\Omega(n^{1/2} + D)$	any α also quantum	*
Lenzen,Patt-Shamir [STOC 2013]	$O(n^{1/2+\varepsilon} + D)$	$O(1/\varepsilon)$	
N [STOC 2014]	$O(n^{1/2}D^{1/4} + D)$	$1+\varepsilon$	
Henzinger,Krinninger,N [STOC 2016]	$O(n^{1/2+o(1)} + D^{1+o(1)})$ (Deterministic)	$1+\varepsilon$	
⇨ Becker, Karrenbauer, Krinninger, Lenzen [2016]	$O(n^{1/2} + D)$ (Deterministic)	$1+\varepsilon$	*

- All previous results except Becker et al. can compute shortest-paths tree

Summary of Part 1

Main focus

s-t Distance

Known

$(1+\varepsilon)$-approx. in $\Theta(n^{1/2}+D)$ time

Distributed ***approximate***
s-t distance are essentially ***solved****.*

Plan

1. Problem & Known Results

2. Open Problems

3. Technical Challenges

<u>Part 2.1</u>

Exact algorithms

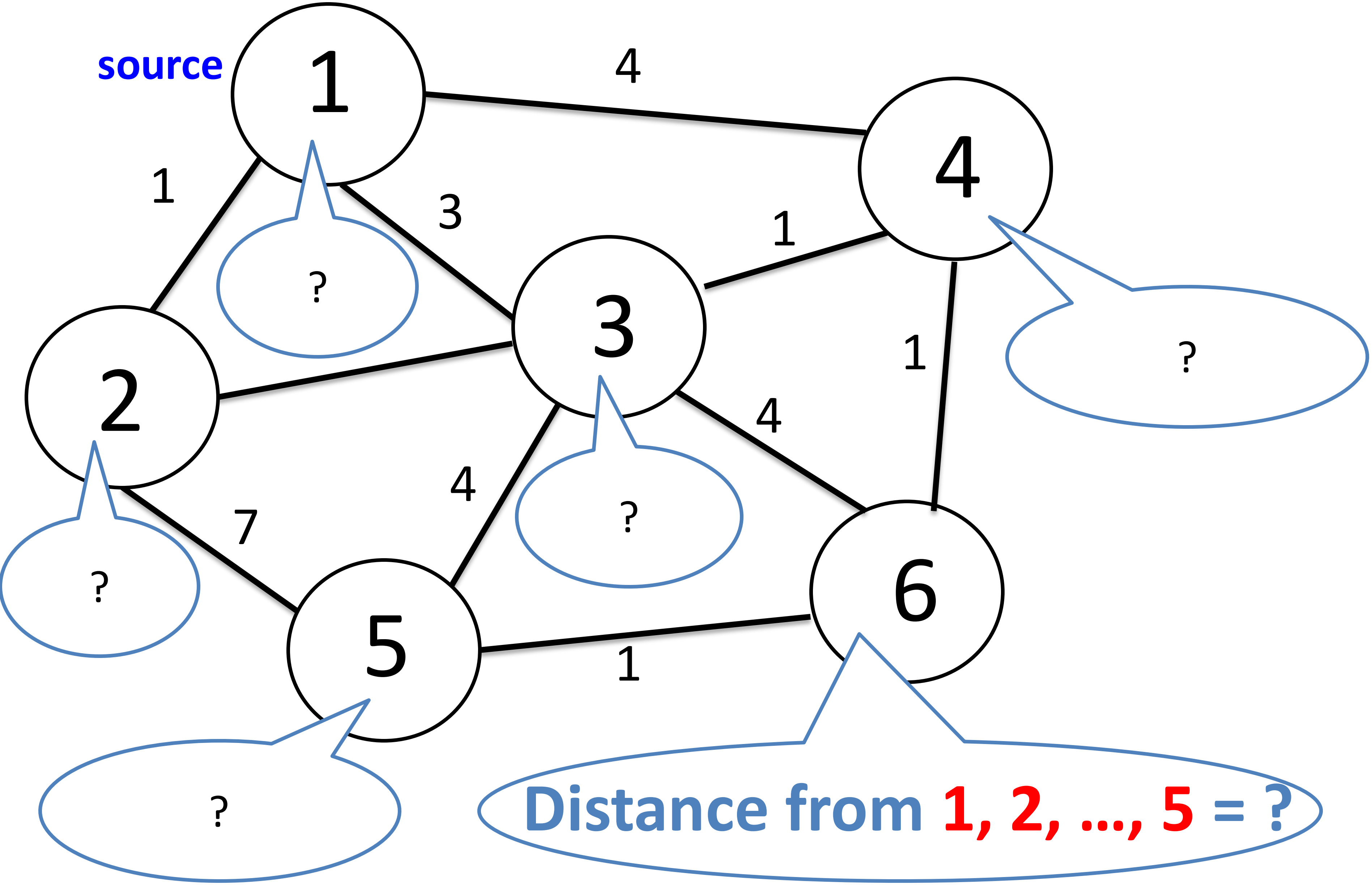
Is there a sublinear-time **exact** algorithm for s-t distance?

- Current lower bound: $\Omega(n^{1/2} + D)$
- $(1+\varepsilon)$-approx. algorithms need $O(n^{1/2} + D)$ time
- Exact algorithm: no $O(n^{1-\varepsilon} + D)$ known

Exact case also open for many other graph problems.

Is there a linear-time exact algorithm for **all-pairs distances** ?

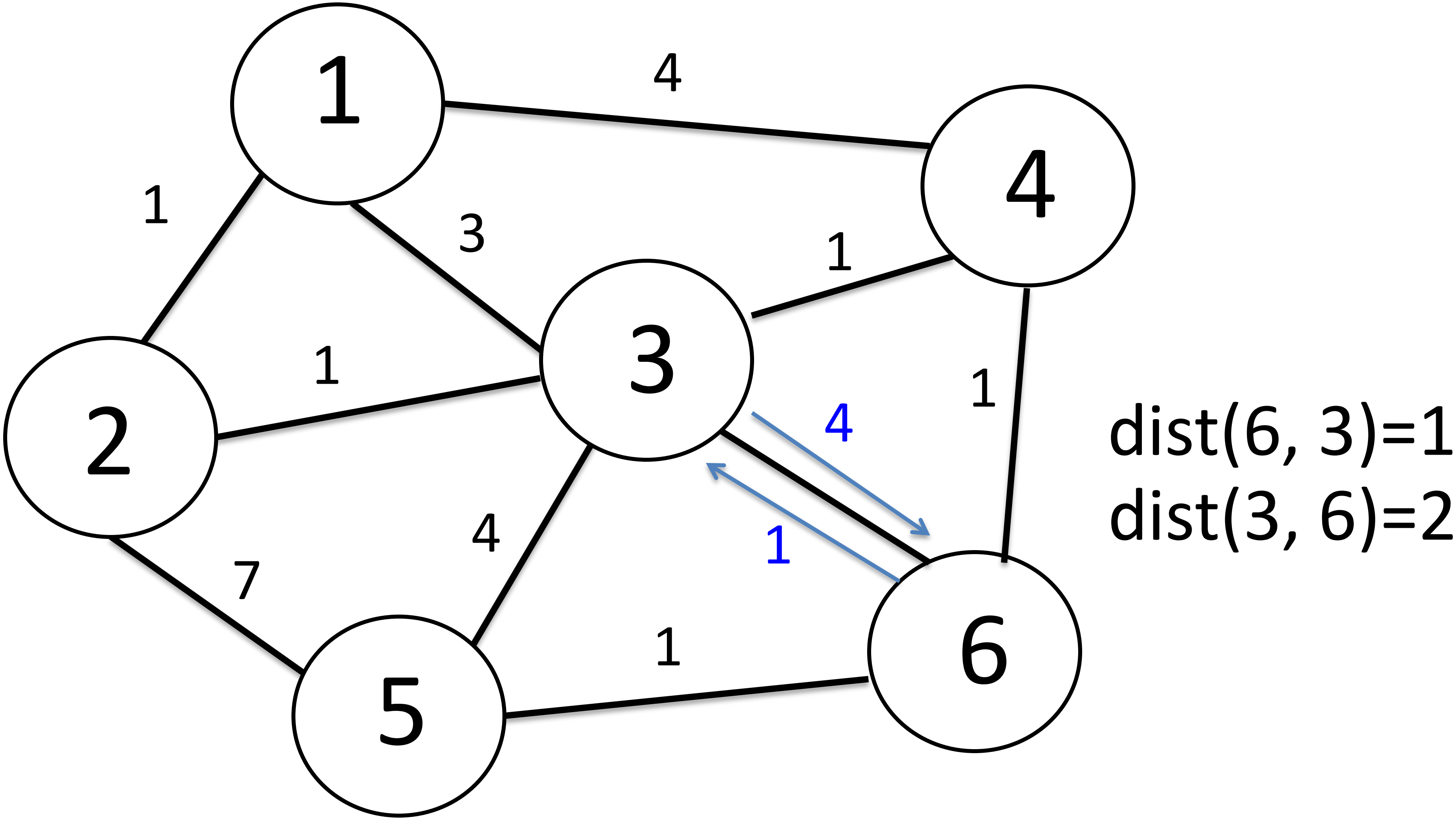
- Current lower bound: $\Omega(n)$.
- We have linear-time $(1+\varepsilon)$-approx. algorithm.



All-Pairs Shortest Paths

Part 2.2

Directed Case



Directed case

Note: **Two-way** communication, not affected by weights.

Directed s-t & single-source distances

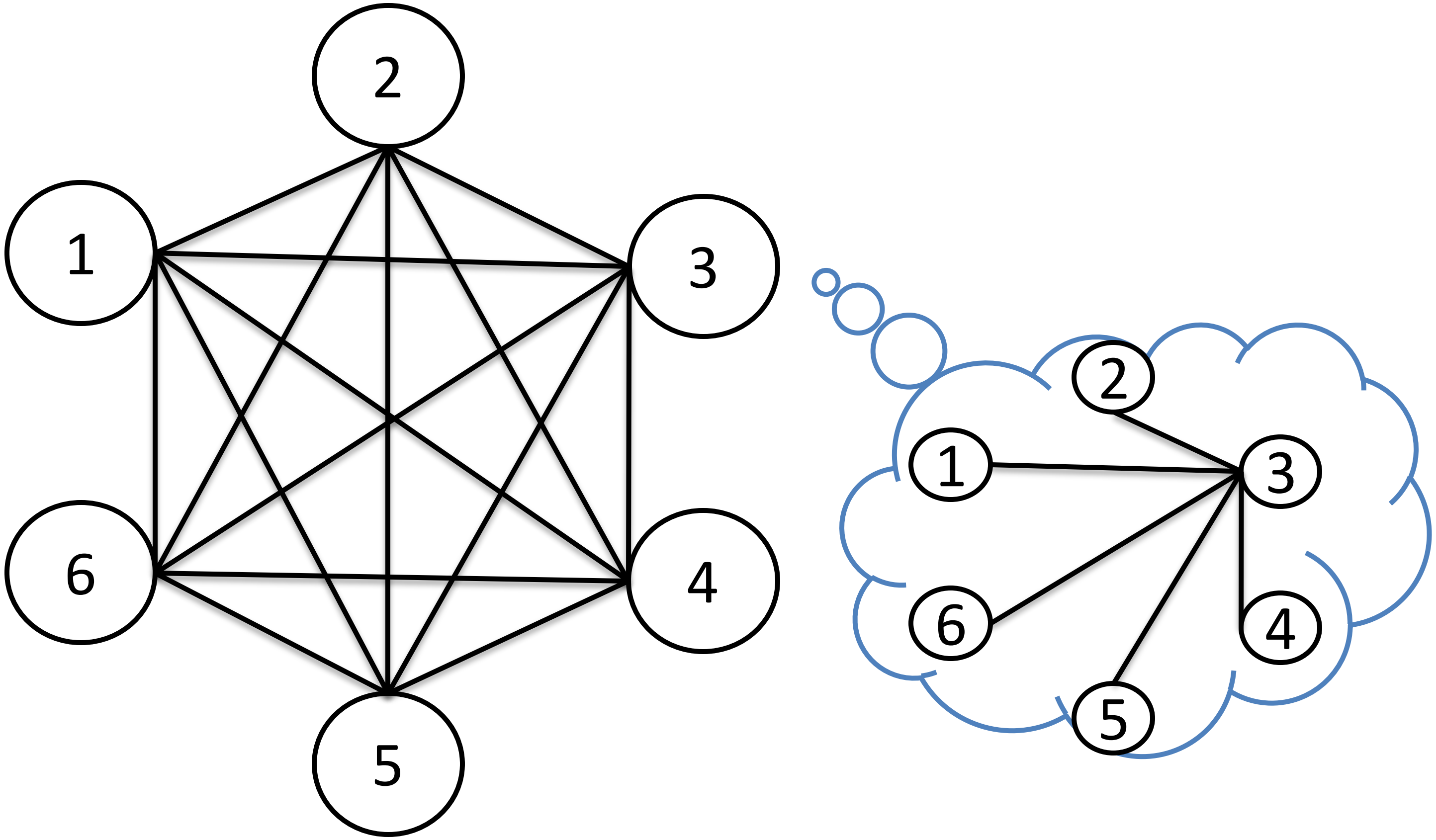
Reference	Time	Approximation
N [STOC'14]	$O(n^{1/2}D^{1/2}+D)$	$1+\varepsilon$
Ghaffari, Udwani [PODC'15]	$O(n^{1/2}D^{1/4}+D)$	Reachability

<u>Open</u>

$O(n^{1/2}+D)$-time (any)-approximation algorithm.

<u>Part 2.3</u>

Congested Cliques



<u>Congested Clique:</u>
The underlying network is *fully connected*

s-t distance, congested clique

Reference	Time	Approximation
N [STOC'14]	$O(n^{1/2})$	exact
Censor-Hillel et al. [PODC'15]*	$O(n^{1/3})$ $O(n^{0.15715})$	exact $1+\varepsilon$
Henzinger,Krinninger,N [STOC'16]	$O(n^{o(1)})$	$1+\varepsilon$
Becker, Karrenbauer, Krinninger, Lenzen [2016]	polylog(n)	$1+\varepsilon$

Open: Better exact algorithm? Lower bound?

* Censor-Hillel et al.'s result works for APSP

All-pairs distances

Reference	Time	Approximation
N [STOC'14]	$O(n^{1/2})$	$2+\varepsilon$
Censor-Hillel, Kaski, Korhonen, Lenzen, Paz, Suomela [PODC'15]	$O(n^{1/3})$ $O(n^{0.15715})$ **Connection to matrix multiplication**	exact $1+\varepsilon$
Le Gall [DISC'16]	• Additional algebraic tools (e.g. determinant) • Applications to, e.g., maximum matching	

<u>Open</u>:

1. Better exact and approximation algorithm.
2. Explore the power of algebraic techniques on congested cliques.
3. Lower bounds?

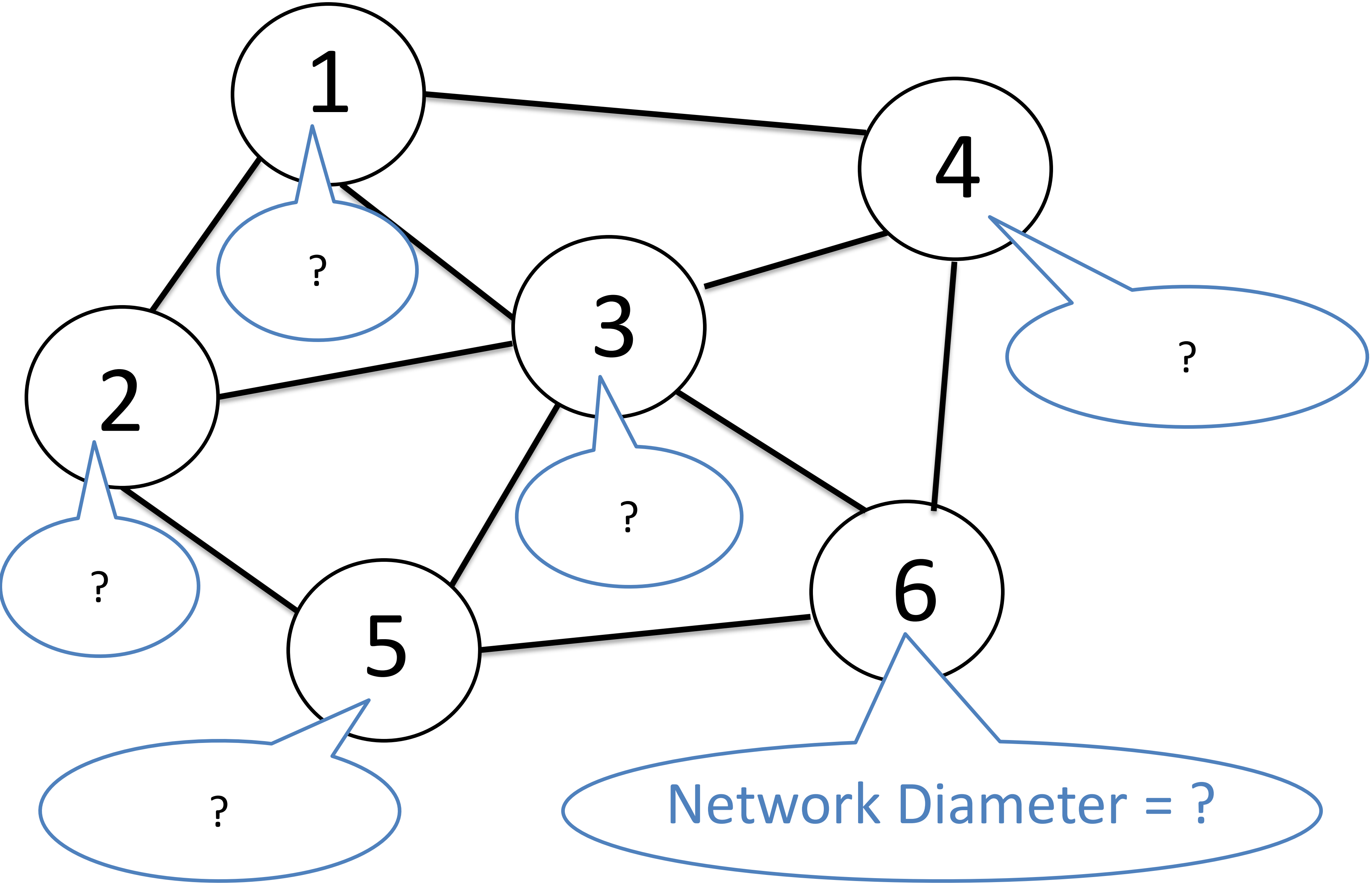
Lower Bounds on Congested Clique?

Drucker et al [PODC’14]:

- Not so easy.
- Will imply something big in *circuit complexity*.

Part 2.4

Other Related Problems



Diameter

Diameter (unweighted)

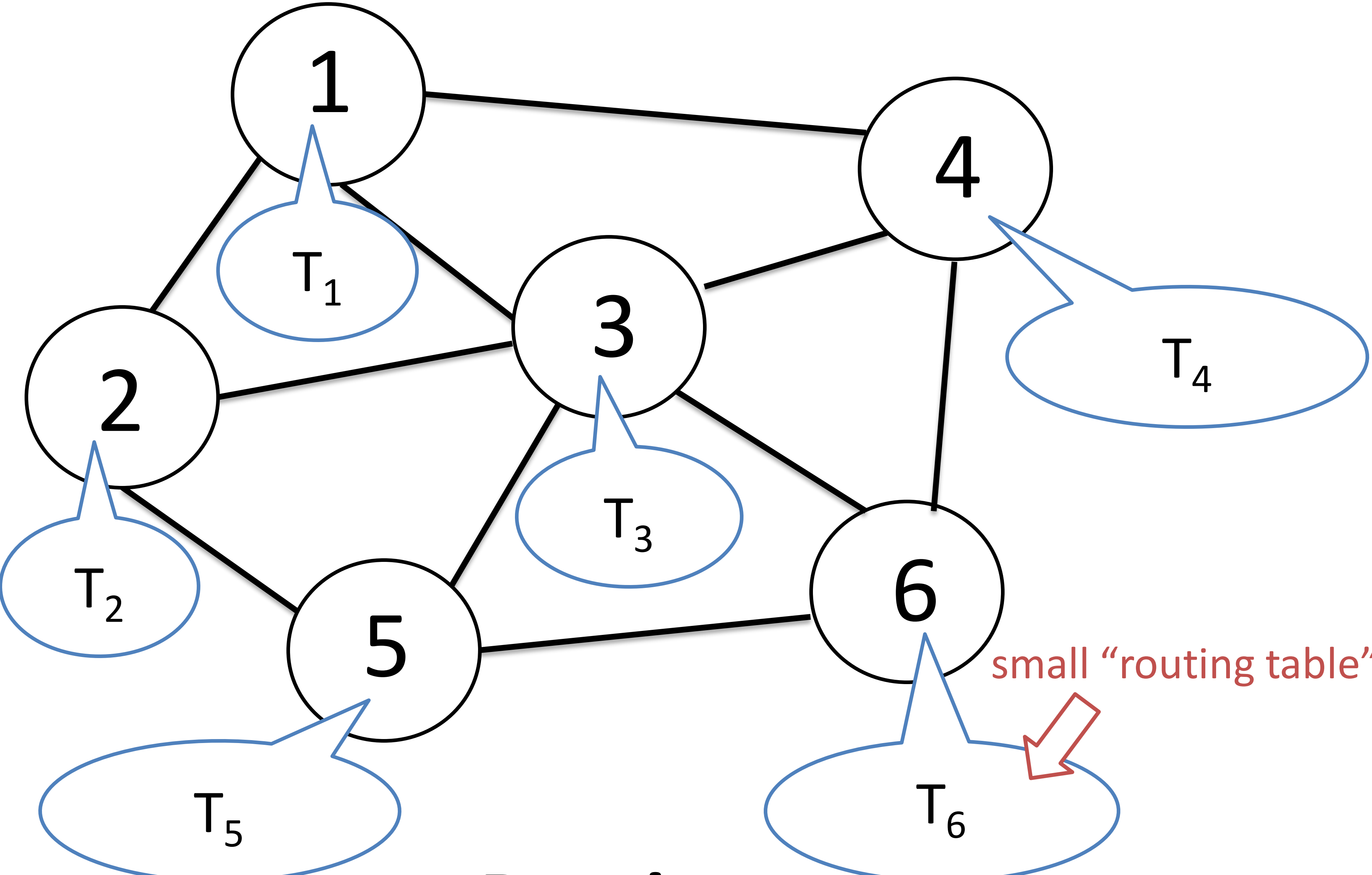
Algorithm	Time	Approximation
BFS	D	2
Holzer et al. [PODC'12] for small D	$\Omega(n)$	$3/2-\varepsilon$
[Censor-Hillel et al. DISC'16]: Result holds for any D and even for sparse graph		
Holzer et al. [PODC'12] Peleg et al. [ICALP'12]	$O(n)$	exact
Frischknecht et al. [SODA'12]	$\Omega((n/D)^{1/2}+D)$	$3/2-\varepsilon$
Lenzen-Peleg [PODC'13]	$O(n^{1/2}+D)$	$3/2$
Holzer et al. [DISC'14]	$O((n/D)^{1/2}+D)$	$3/2+\varepsilon$
Open (By Holzer)	$\Omega(n/D+D)$	$1+\varepsilon$

Also: Eccentricity, radius, etc.

Diameter (weighted)

Algorithm	Time	Approximation
Holzer et al. [PODC'12]	$\Omega(n)$	$2-\varepsilon$
Becker et al. [2016]	$O(n^{1/2} + D)$	$2+\varepsilon$
Open	sublinear	2

Intermediate problem to exact SSSP
(Getting a sublinear-time exact algorithm for SSSP will resolve this)



Routing

Some open problems

Thanks: Christoph Lenzen

- Eliminate $n^{o(1)}$ term as in the case of s-t shortest path.
  - Techniques from s-t SP usually transfer to the routing problem.
  - Exception: Becker et al [2016]
- Lower bounds on the construction time for stateful routing.
- Further read: Elkin, Neiman [PODC'16] & Lenzen, Patt-Shamir [STOC'13]

Summary of Part 2

Open problem
1. Exact $O(n^{1-\varepsilon}+D)$ time
2. Directed $O(n^{1/2}+D)$ time

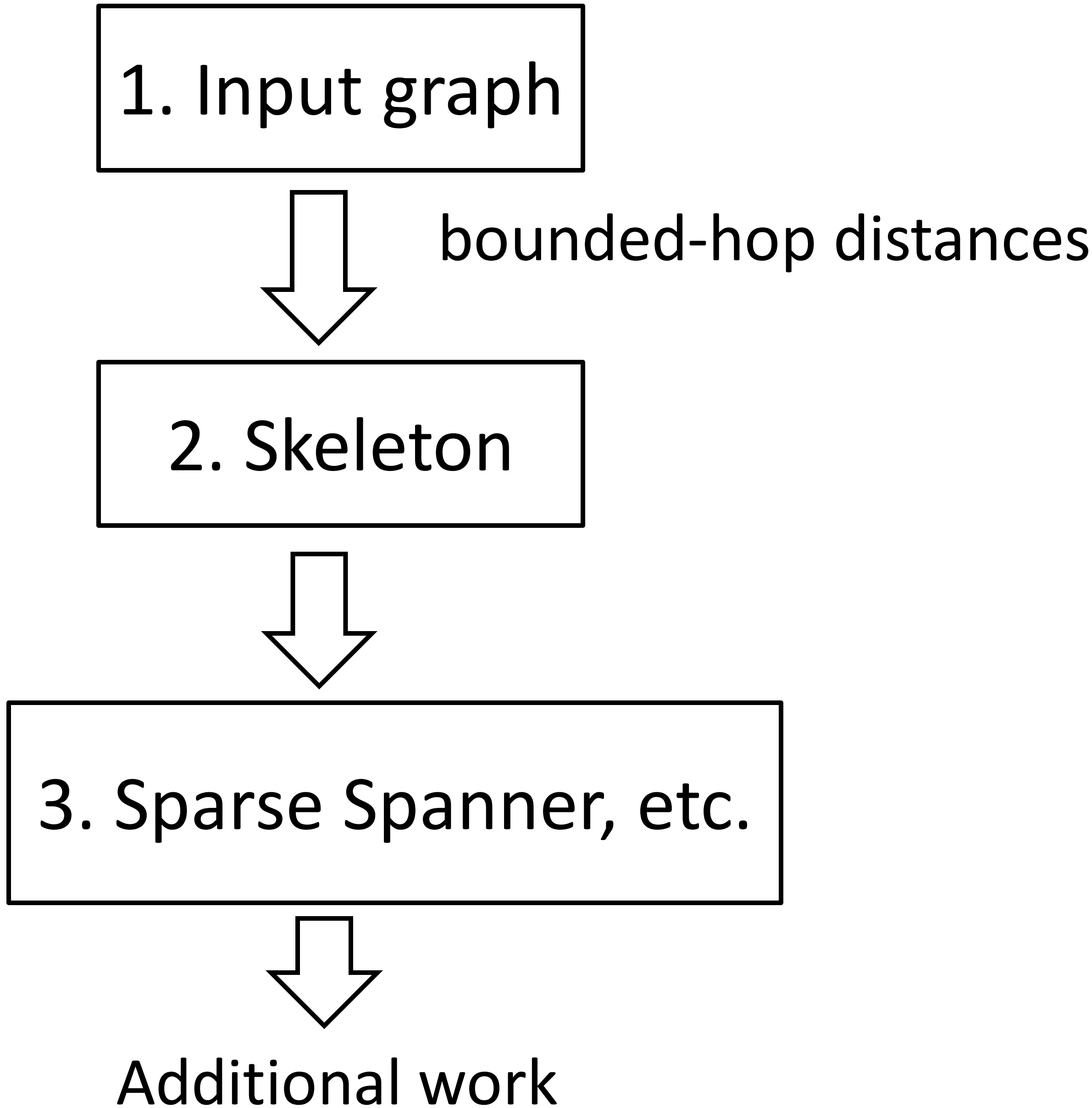
Plan

1. Problem & Known Results

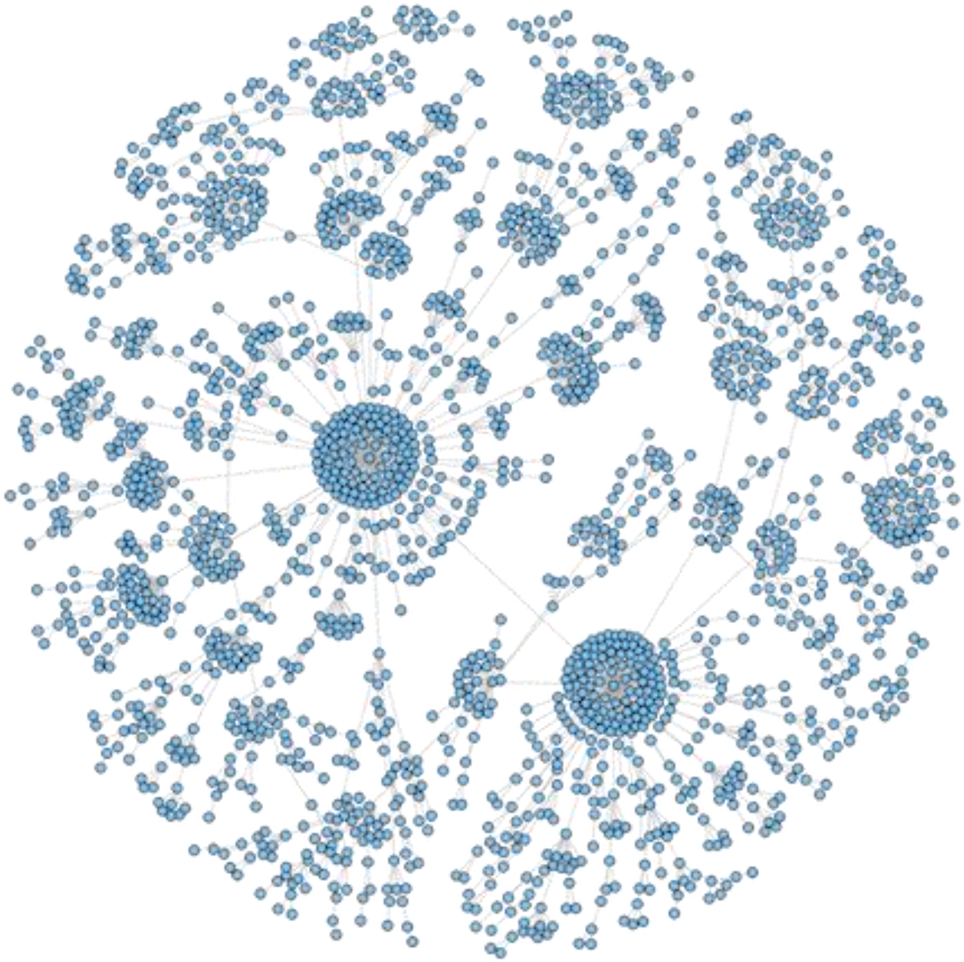
2. Open Problems

3. Technical Challenges

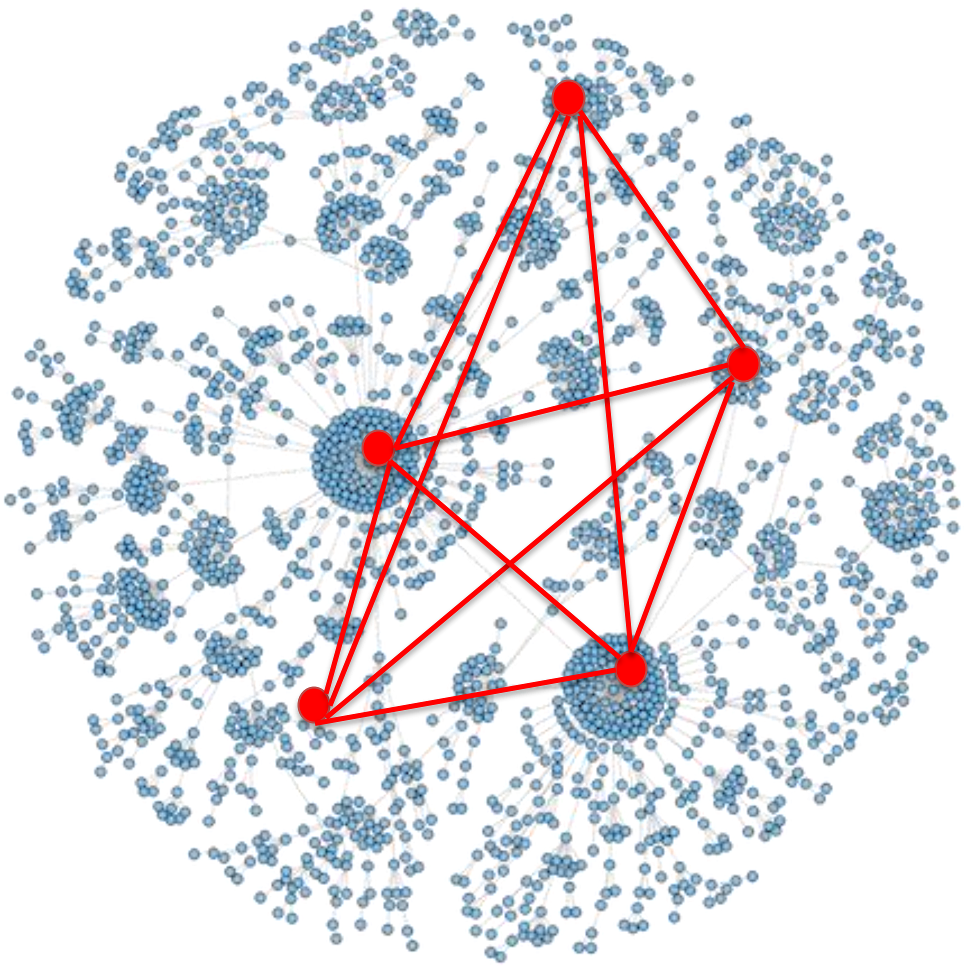
Framework for approximate s-t shortest paths


1. Input graph
bounded-hop distances
2. Skeleton
3. Sparse Spanner, etc.
Additional work

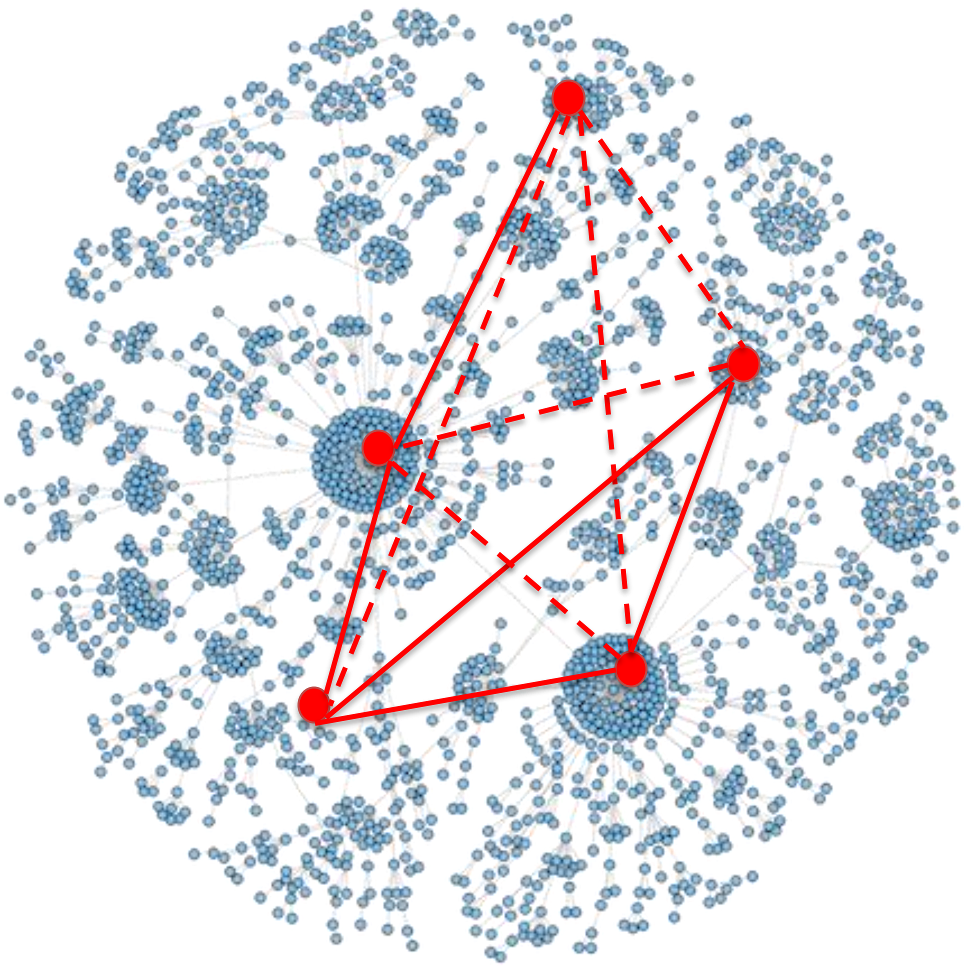
1. Input graph



2. Skeleton



3. Sparse spanner, etc.



1. Input graph

bounded-hop distances

Challenge for **exact** case

2. Skeleton

Challenge for **directed** case

3. Sparse Spanner, etc.

Additional work

Part 3.1

Exact case challenge: bounded-hop distance

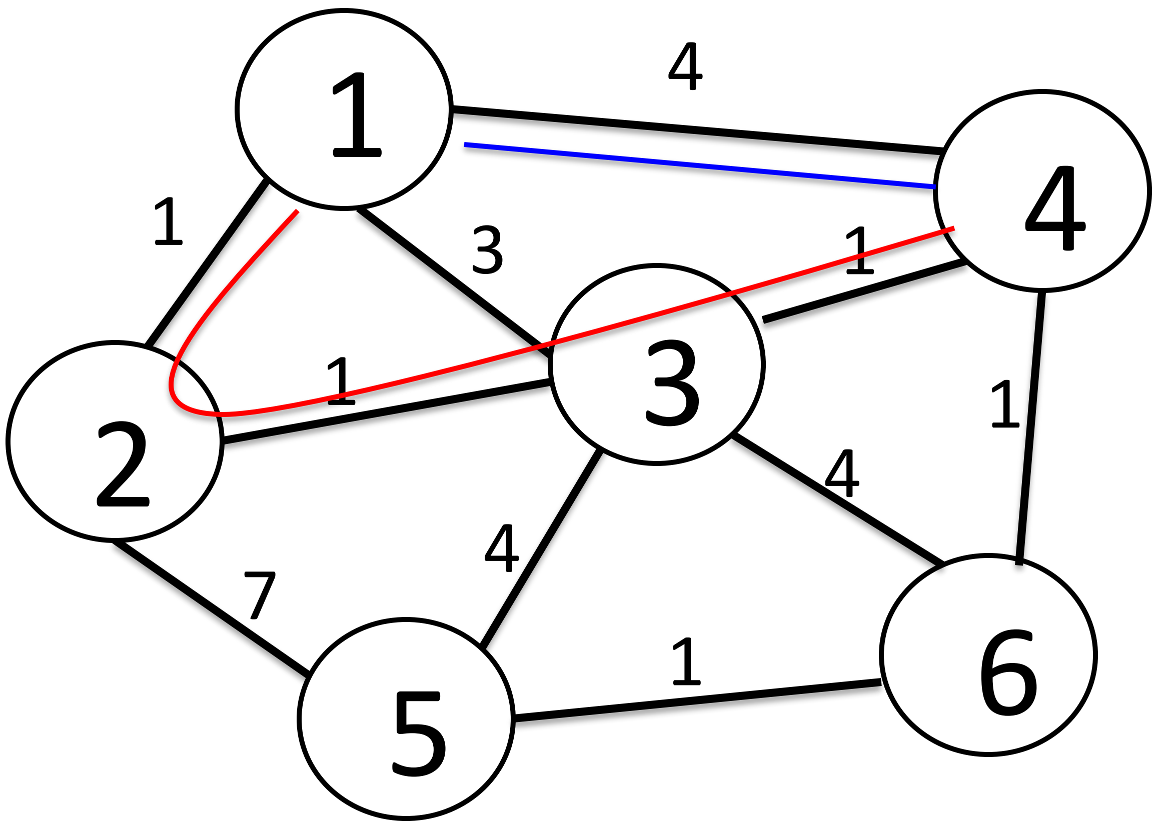
Recall

Reference	**Time**	**Approximation**
Bellman&Ford [1950s]	$O(n)$	exact
Das Sarma et al [STOC 2011]	$\Omega(n^{1/2} + D)$	any α
OPEN	$\mathbf{O(n^{1-\varepsilon}+D)}$	**exact**

- polylog(n/ε) factors are hidden

Definition: h-hop distance

- $dist^h(u,v)$:= smallest total weight among u-v paths containing at most h edges.



$dist(1, 6) = 3$

$dist^1(1, 6) = 4$

Definition: h-hop distance

- $\text{dist}^h(u,v)$:= smallest total weight among u-v paths containing at most h edges.
- k-sources h-hop distances: find $\mathbf{dist^h(s_i, v)}$ for all k sources $s_1, \ldots, s_k$, and all nodes v.

Theorem

We can find k-sources $(1+\varepsilon)$-approx. h-hop distances in $O(k+h/\varepsilon)$ time

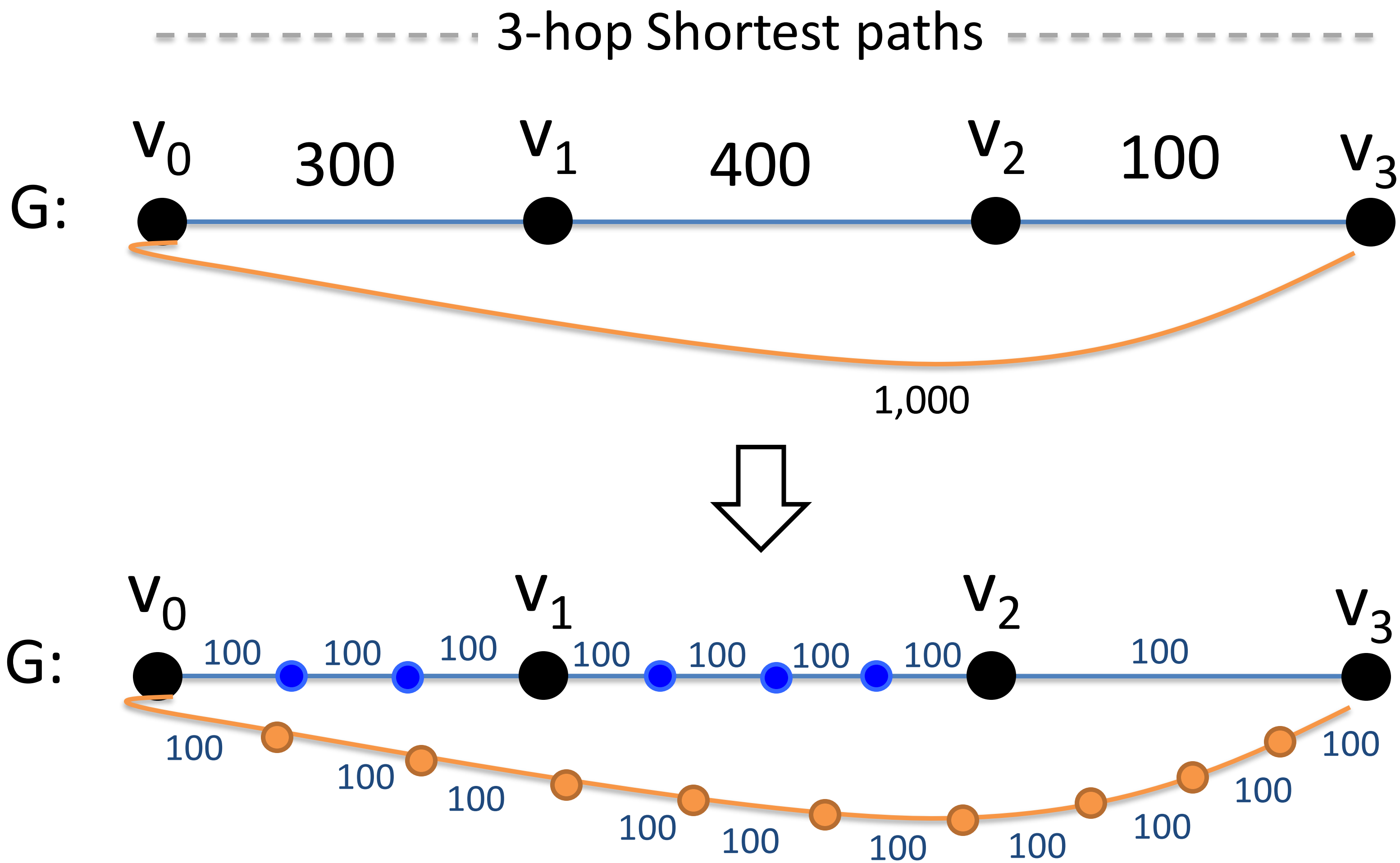
\begin{technical}

Skip

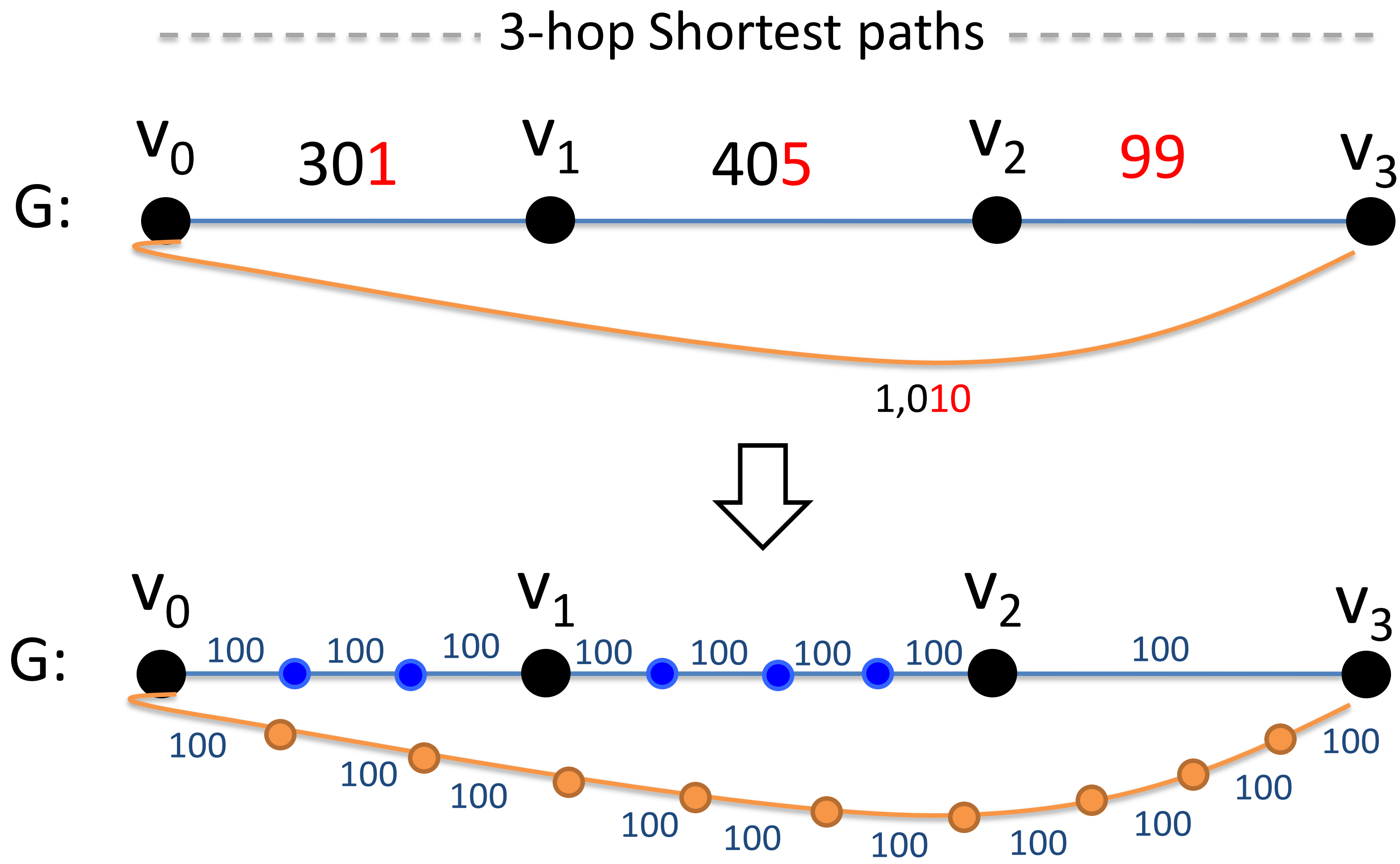
Approximating k-sources h-hop distances
in the weighted case is as easy as
computing a BFS tree on unweighted graphs

Key idea: **Weight rounding**

1. Pretend that the graph is unweighted



2. Round weight -- ignore small errors



Approximating k-sources h-hop distances

1. Pretend that the graph is unweighted.
2. Round weight -- ignore small errors.
3. With appropriate rounding, we get distance $O(h)$ and $(1+\varepsilon)$ approximation.
4. Run BFS algorithms from k sources in parallel.

See N [STOC'14] for more details.

\end{technical}

Theorem (recall)

We can find k-sources $(1+\varepsilon)$-approx. h-hop distances in $O(k+h/\varepsilon)$ time

Question

Can we find k-sources

~~(1+ε)-approx~~. exact h-hop distances in

O(k+h) time?

If so, we will be able to solve **SSSP** exactly in sublinear time **and APSP** exactly in linear time

Answer (Lenzen, Patt-Shamir [PODC'15]):

No. o(kh) time is impossible.

<u>Challenge for exact computation</u>

k-sources h-hop distances – avoid it to get $O(n^{1-\varepsilon}+D)$ time!

Part 3.2

Directed case challenge: sparse spanner

Recall: Directed s-t & single-source distances

Reference	Time	Approximation
N [STOC'14]	$O(n^{1/2}D^{1/2}+D)$	$1+\varepsilon$
Ghaffari, Udwani [PODC'15]	$O(n^{1/2}D^{1/4}+D)$	Reachability
OPEN	$\mathbf{O(n^{1/2}+D)}$	**$1+\varepsilon$ or just reachability**

1. Input graph

bounded-hop distances

2. Skeleton

Challenge for **directed** case

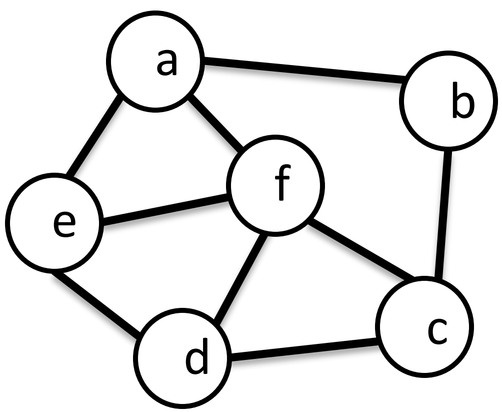
3. Sparse structure

(spanner/hopset)

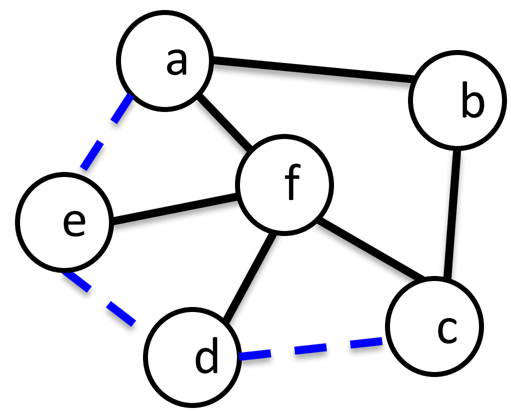
Additional work

Definition: Spanner

- p-spanner: Subgraph that preserves distances with multiplicative error



input graph



2-spanner

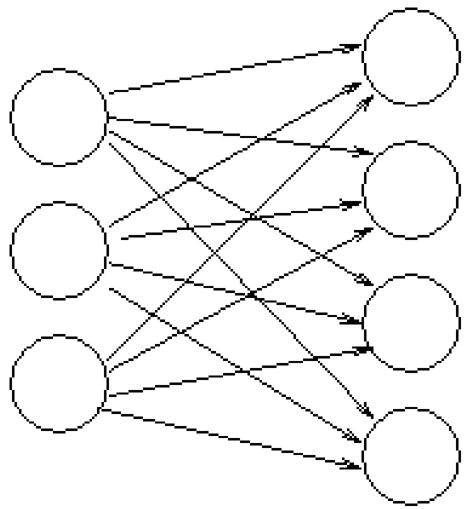
Computing spanner on distributed networks

- Baswana-Sen [Rand. Struct & Alg. 2007]: $(2p-1)$-spanner of size $O(n^{1+1/p})$ in $O(p)$ rounds for any p.
- There's a huge literature on this.
  - See, e.g., Pettie [Dist. Comp. 2010]

* It was pointed out by Pettie that the size of Baswana-Sen's spanner is $O(kn+(\log n)n^{1+1/k})$

Exists sparse directed spanner?

No:

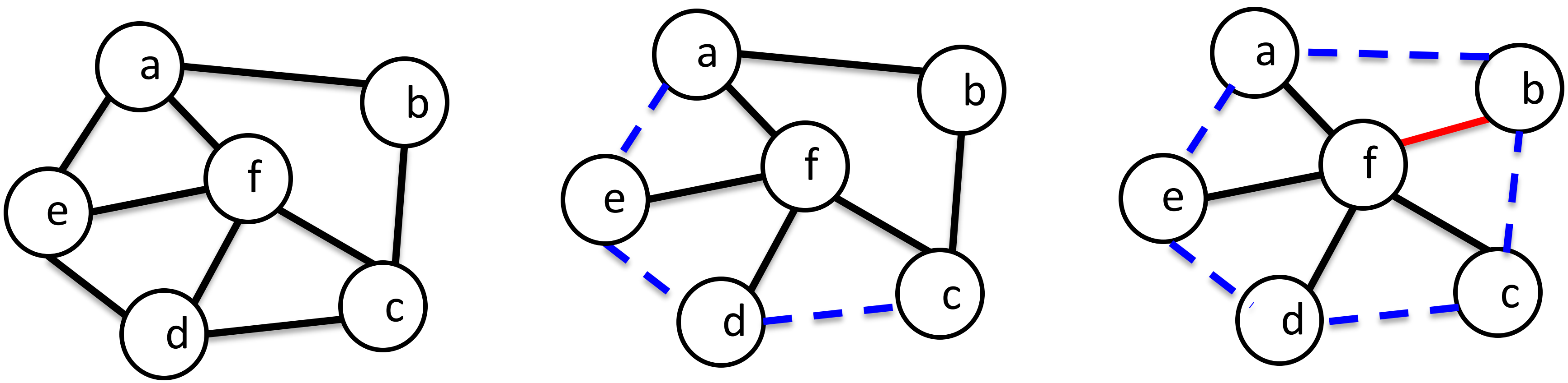


\begin{technical}

Skip

Definitions

- p-spanner: Subgraph that preserves distances with multiplicative error p
- p-emulators: Graph on the same set of vertices that preserves distances



input graph 2-spanner 2-emulator

Hopset [Cohen, JACM'00]

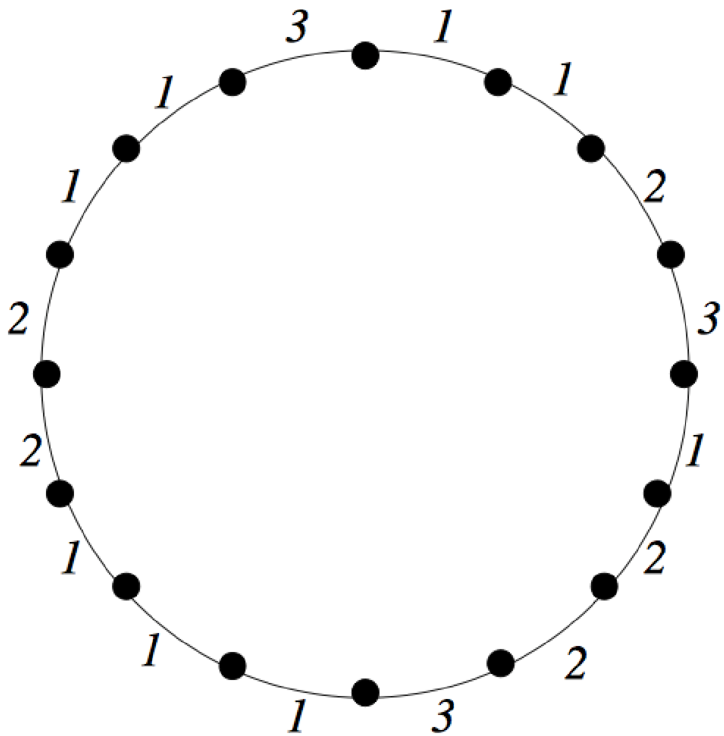
(h,ε)-hopset of a network $G = (V,E)$

is a set E^* of new weighted edges such that

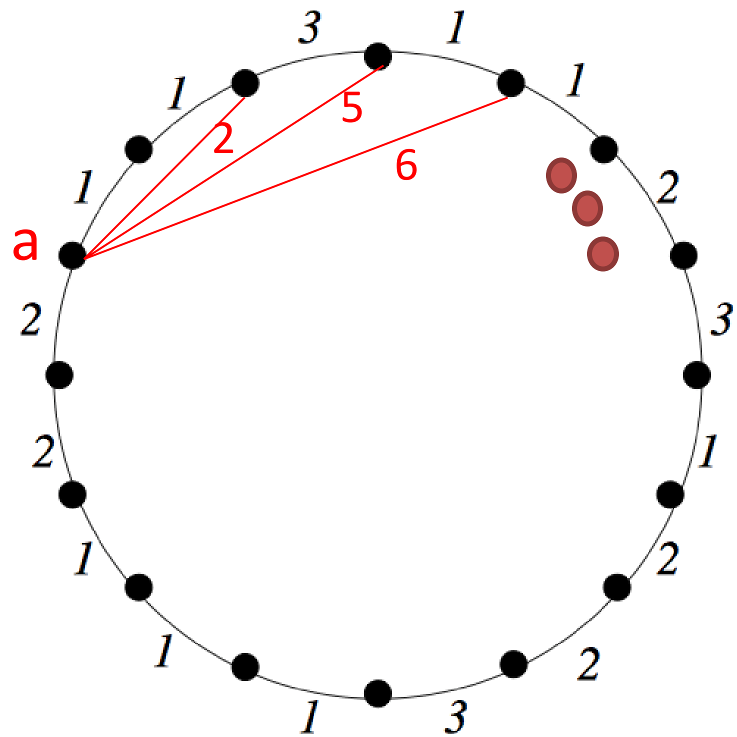
h-edge paths in $H=(V, E \cup E^*)$

give $(1+\varepsilon)$ approximation to distances in G.

Example (1)



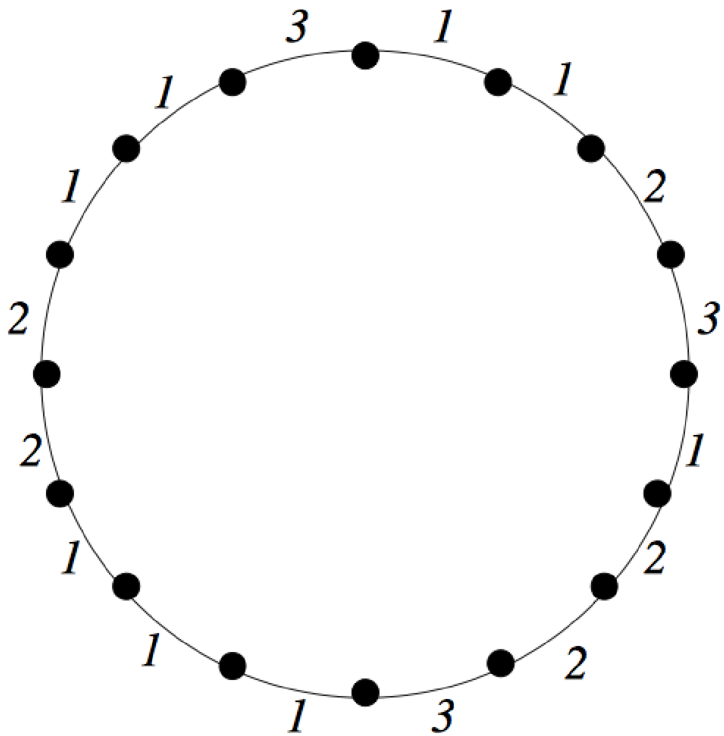
Input graph



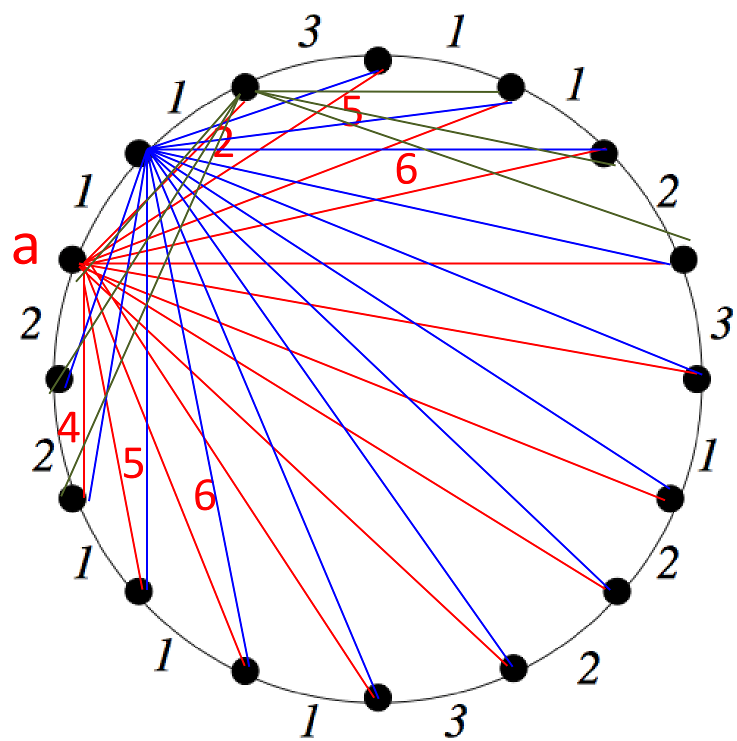
Add shortcuts between every pair

Picture from Cohen [JACM'00]

Example (1)



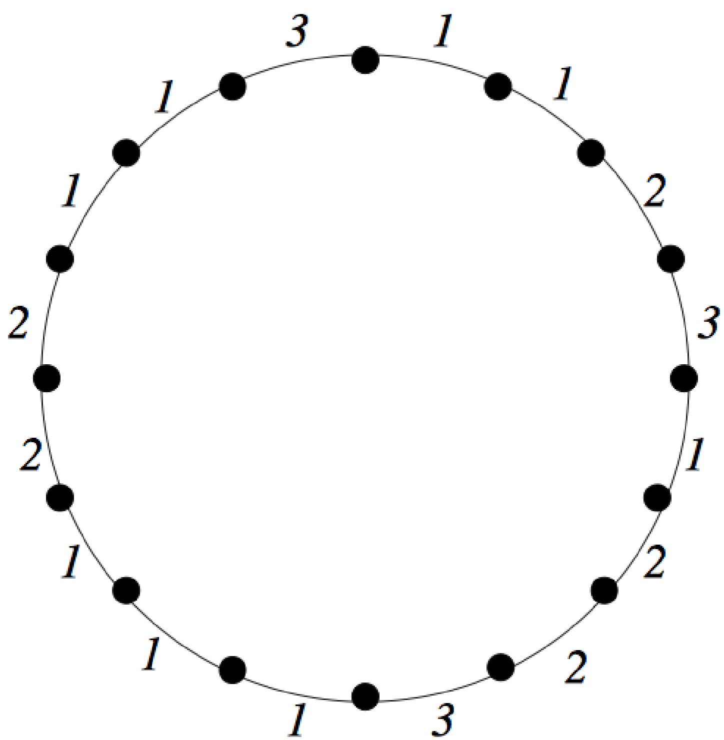
Input graph



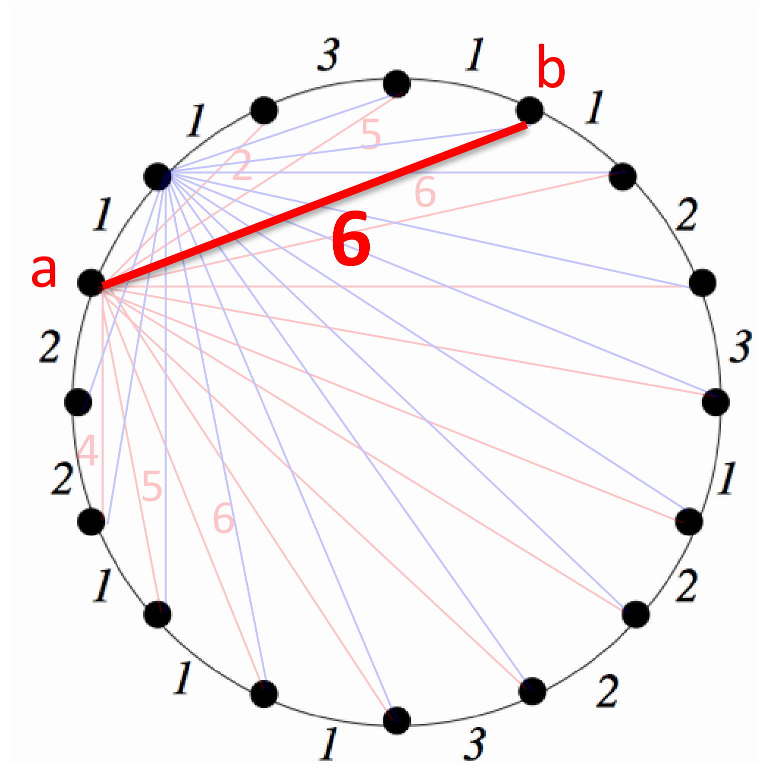
Add shortcuts between every pair

Picture from Cohen [JACM'00]

Example (1)



Input graph



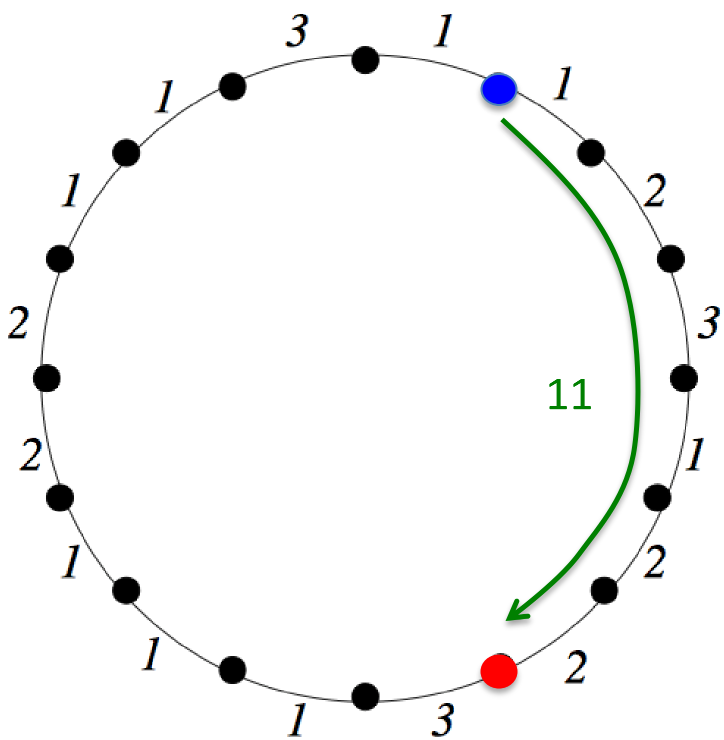
(1, 0)-hopset

one edge is enough to get distance

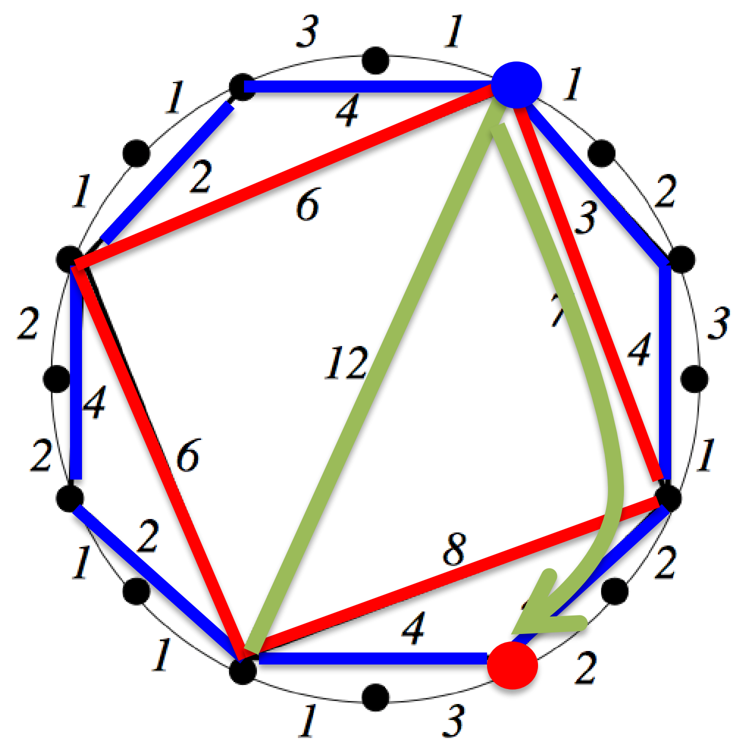
no error

Picture from Cohen [JACM'00]

Example (2)



Input graph



Input graph with (5, 0)-hopset

Picture from Cohen [JACM'00]

Exists sparse **directed** emulator/hopset?
(No)

\end{technical}

Challenge for directed case

Can we avoid the use of sparse spanner and related structures?

Summary

Main focus

s-t Shortest Paths

Known

$(1+\varepsilon)$-approx. in $\Theta(n^{1/2}+D)$ time

Open problem	Technical challenge
1. Exact $O(n^{1-\varepsilon}+D)$ time	Avoid bounded-hop distances!
2. Directed $O(n^{1/2}+D)$ time	Avoid sparse spanner, etc.!

Thank you