Spanner, Distance Oracle, and Compact Routing for Unweighted Graphs

Cyril Gavoille
(Joint work with Ittai Abraham,
Microsoft, Mountview, CA)

University of Bordeaux

ALADDIN Meeting - Arcachon
May 2011

Topic

Given a graph G with n nodes compute an **efficient** data structure supporting **approximate** distance and/or routing queries in G.

- **Efficient**: polynomial time pre-processing, constant or poly-log(n) query time
- **Approximate**: guarantees on the distance $\hat{d}$ or route length returned w.r.t. the shortest path in G.

  Ex: $d_G(u,v) \leqslant \hat{d}(u,v) \leqslant f(d_G(u,v))$

For instance

2010 IEEE 51st Annual Symposium on Foundations of Computer Science

Distance Oracles Beyond the Thorup–Zwick Bound

Mihai Pătraşcu Liam Roditty

Theorem 1. *For any unweighted graph, there exists a distance oracle of size* $O(n^{5/3})$ *that, given any nodes* u *and* v *at distance* d*, returns a distance of at most* $2d+1$ *in constant time. A path of this length can be listed in* $O(1)$ *time per hop.*

Targets

Ideal/realistic:

- data structures of linear size? $o(n^2)$ space?
- constant time query? poly-log time?
- linear pre-processing time? polynomial time?

Targets

Ideal/realistic:

- data structures of linear size? $o(n^2)$ space?
- constant time query? poly-log time?
- linear pre-processing time? polynomial time?

Extra:

- data structure can be split into n balanced labels?

Compression
(to get $o(n^2)$ space data structures)

Hints: sparsify the graph!

Compute a spanning subgraph (spanner) preserving distances.

Compression
(to get $o(n^2)$ space data structures)

Hints: sparsify the graph!

Compute a spanning subgraph (spanner) preserving distances.

Good news: Every graph G has a spanner H of $O(n^{1+1/k})$ edges with *stretch* function $(2k-1)d$.

(i.e., $d_H(u,v) \leqslant (2k-1)d_G(u,v)$ for all u, v)

Compression: stretch $d+2$ spanner of size $O(n^{3/2})$

Proof:

1. $H := \varnothing$
2. While $\exists\, u \in V$, $\deg(u) \geqslant \sqrt{n}$:
   1. $H := H \cup \mathrm{BFS}(u, G)$
   2. $G := G \setminus N(u)$
3. $H := H \cup G$

Compression: stretch $d+2$ spanner of size $O(n^{3/2})$

Proof:

1. $H := \varnothing$
2. While $\exists\, u \in V$, $\deg(u) \geqslant \sqrt{n}$:
   1. $H := H \cup \mathrm{BFS}(u, G)$
   2. $G := G \setminus N(u)$
3. $H := H \cup G$

Size: $O(n\sqrt{n}\,)$

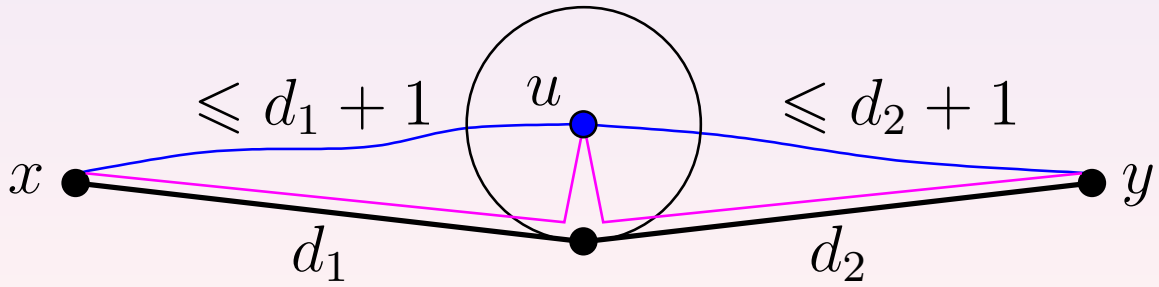
Compression: stretch $d+2$ spanner of size $O(n^{3/2})$

Proof:

1. $H := \varnothing$
2. While $\exists\, u \in V$, $\deg(u) \geqslant \sqrt{n}$:
   1. $H := H \cup \mathrm{BFS}(u, G)$
   2. $G := G \setminus N(u)$
3. $H := H \cup G$

Size: $O(n\sqrt{n}\,)$

Stretch: if an xy-shortest path does not intersect a neighbor of any selected node u (or cuts u), then stretch is d, otherwise $d_H(x, y) \leqslant d_1 + d_2 + 2 = d + 2$.



Compression: ✔

Query time?

How do we get $d_H(u, v)$?

We can do in $O(n + m_H)$ time instead of $O(n + m_G)$ but it's far from $\mathrm{polylog}(n)$.

Compression: ✔
Query time?

How do we get $d_H(u, v)$?

We can do in $O(n + m_H)$ time instead of $O(n + m_G)$
but it's far from $\mathrm{polylog}(n)$.

What about if G is already sparse? say G is a cubic graph?

Compression: ✔
Query time?

How do we get $d_H(u, v)$?

We can do in $O(n + m_H)$ time instead of $O(n + m_G)$ but it's far from $\mathrm{polylog}(n)$.

What about if G is already sparse? say G is a cubic graph?

Lower bound [Sommer et al. (FOCS'09)] Every stretch $O(k) \cdot d$ query time t distance oracle for graphs with $\tilde{O}(n)$ edges must have a size $n^{1+\Omega(1/(kt))}$.

Space limitation of $n^{1+\Omega(1/k)}$ comes from not only from compression but also from *constant* query time.

State-of-the-art

Thorup-Zwick distance oracle (J.ACM '05) Every weighted graph has a stretch $(2k-1)d$ distance oracle of size $\tilde{O}(n^{1+1/k})$ with query time $O(k)$, and polynomial pre-processing. Moreover, the oracle can be represented as a distance labeling.

For k=2:
$\Rightarrow$ stretch $3d$, space $n^{3/2}$, constant query time

Can we do better for unweighted graphs?

Pătraşcu-Roditty (FOCS '10) Every unweighted graph has stretch $2d+1$ distance oracle of size $\tilde{O}(n^{5/3})$ with constant query time. (Label approach fails because of use a global hash table of size $n^{5/3}$)

Can we do better for unweighted graphs?

Pătraşcu-Roditty (FOCS '10) Every unweighted graph has stretch $2d+1$ distance oracle of size $\tilde{O}(n^{5/3})$ with constant query time. (Label approach fails because of use a global hash table of size $n^{5/3}$)

Theorem (This talk)

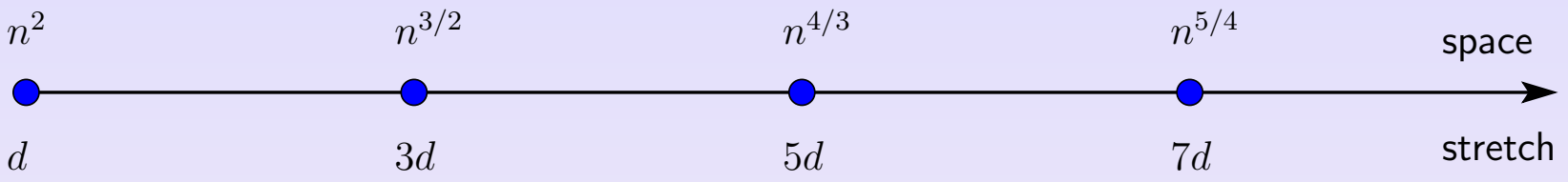
Let $k \geqslant 2$. Every unweighted graph has stretch $(2k-2)d+1$ distance oracle of size $\tilde{O}(n^{1+2/(2k-1)})$ with query time $O(k)$. Moreover it can be represented as a distance labeling.

For k=2:

$\Rightarrow$ stretch $2d+1$, space $n^{1+2/3} = n^{5/3}$, constant query time

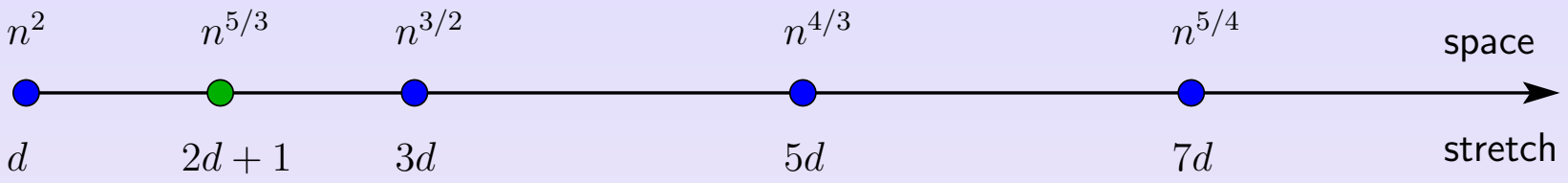
Different trade-offs

[TZ05]



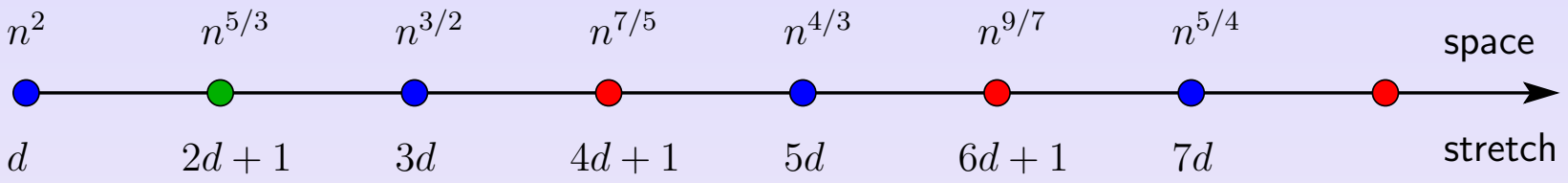
Different trade-offs

[TZ05][PR10]



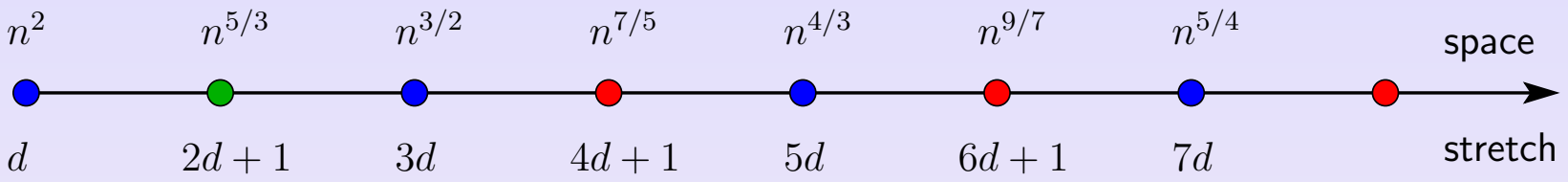
Different trade-offs

[TZ05][PR10][us]



Different trade-offs

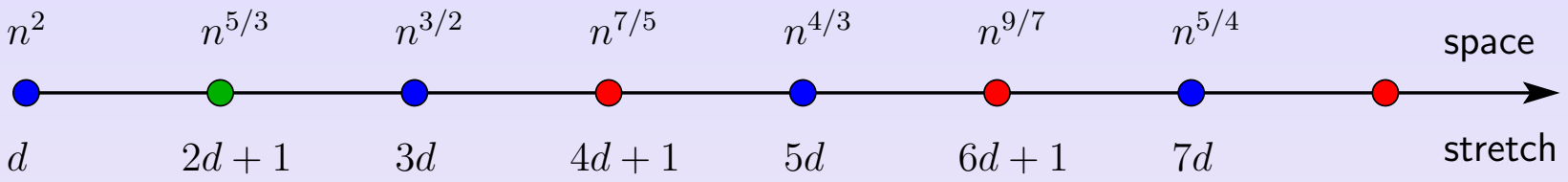
[TZ05][PR10][us]



The **Best Solution** depends on the question:

Different trade-offs

[TZ05][PR10][us]

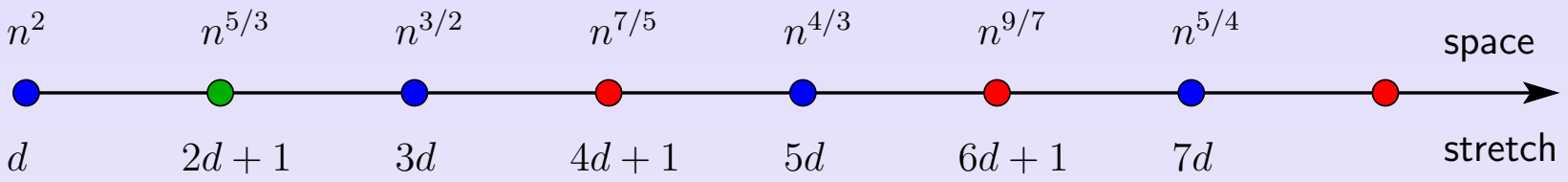


The **Best Solution** depends on the question:

1. What is the best space complexity with stretch $3d$?
   $\Rightarrow n^{3/2}$ [TZ05]

Different trade-offs

[TZ05][PR10][us]

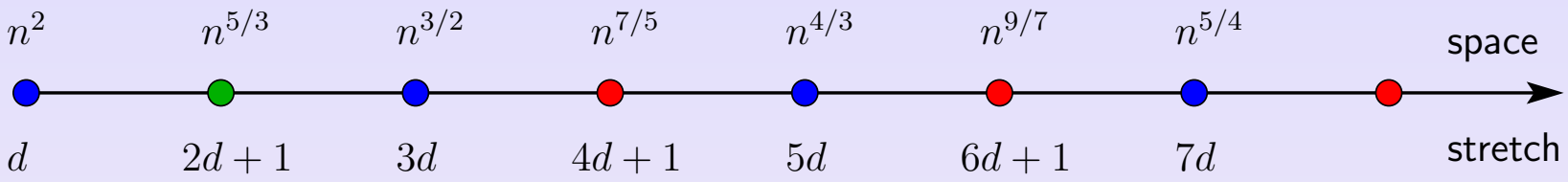


The **Best Solution** depends on the question:

1. What is the best space complexity with stretch $3d$?
   $\Rightarrow n^{3/2}$ [TZ05]
2. What is the best stretch with space complexity $o(n^2)$?
   $\Rightarrow$ at most $2d+1$ [PR10][us]

Different trade-offs

[TZ05][PR10][us]



The **Best Solution** depends on the question:

1. What is the best space complexity with stretch $3d$?
   $\Rightarrow n^{3/2}$ [TZ05]
2. What is the best stretch with space complexity $o(n^2)$?
   $\Rightarrow$ at most $2d+1$ [PR10][us]

Under a conjecture about hardness of sparse intersecting set data structures, [PR10] believe that stretch $(2-\varepsilon)d$ requires space $\tilde{\Omega}(n^2)$, and stretch $2d+1$ requires $\tilde{\Omega}(n^{3/2})$.

Proof for $k = 2$.

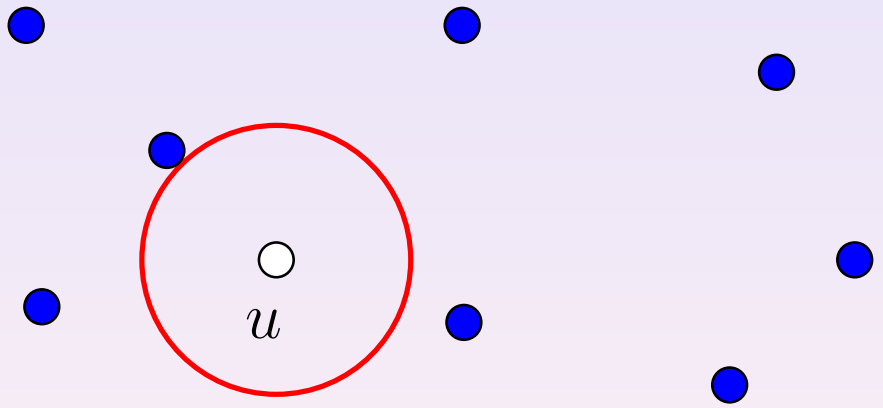
(we want n labels of $\tilde{O}(n^{2/3})$ bits and stretch $2d + 1$)

Proof for $k = 2$.

(we want n labels of $\tilde{O}(n^{2/3})$ bits and stretch $2d + 1$)

Definitions: Given a set of landmarks $L \subset V$:

$$B_L(u) = \{v \in V : d(u, v) < d(u, L)\}$$

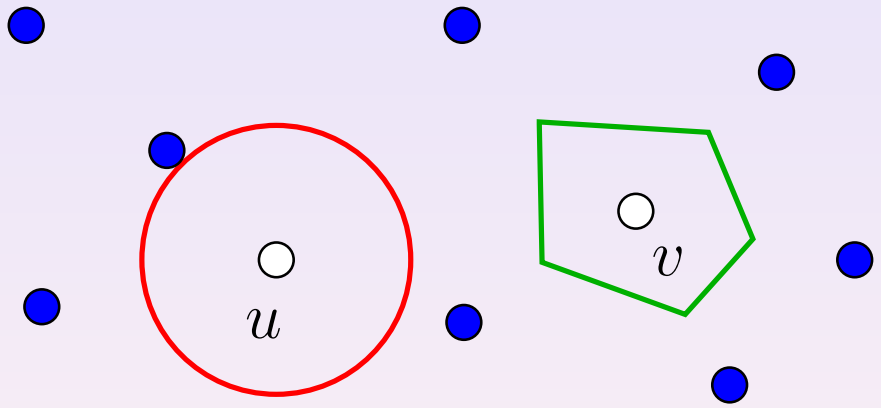


Proof for $k = 2$.

(we want n labels of $\tilde{O}(n^{2/3})$ bits and stretch $2d + 1$)

Definitions: Given a set of landmarks $L \subset V$:

$B_L(u) = \{v \in V : d(u, v) < d(u, L)\}$
$C_L(v) = \{u \in V : v \in B_L(u)\}$



☞ $v \in B_L(u)$ iff $u \in C_L(v)$

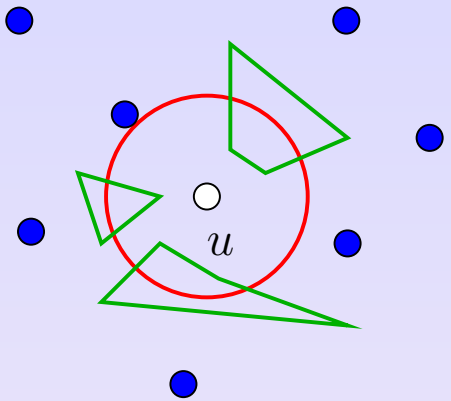
Sampling Lemma [TZ - SPAA '01]

Select: $|L| \sim n^{2/3}$ such that $\forall u$, $|B_L(u)|$ & $|C_L(u)| \sim n^{1/3}$

Lemma. Given s $(= n^{2/3})$, one can construct in polynomial time a landmark set L such that for every node u of G, $|B_L(u)|$ & $|C_L(u)| \leqslant 4n/s$, and in expectation, $|L| \leqslant 2s \log n$.

Proof idea. Sample nodes of V with probability s/n. $|B_L(u)|$ is ok whp. Compute set W of w having **large** $C_L(w)$. If $|W| \leqslant s$, add W to L. Otherwise, sample this W with probability $s/|W|$. Then, half of **large** w becomes **small** (double counting + Markov).

Storage

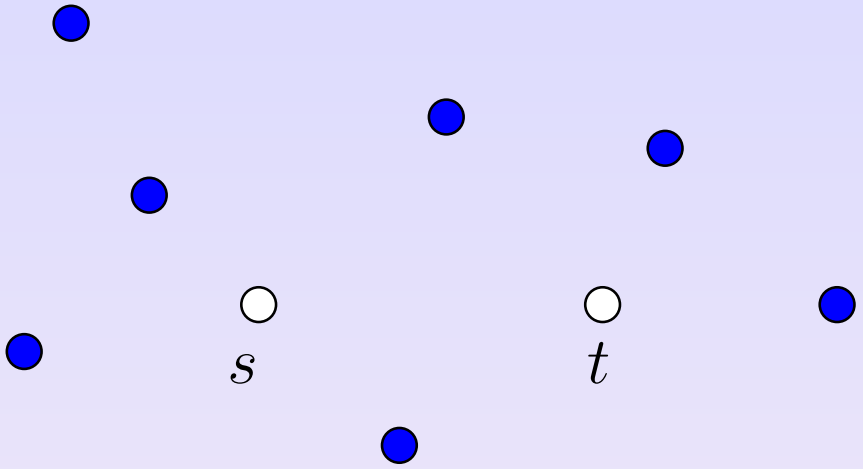


$$I(u) \;:=\; L \cup B_L(u) \cup \left(\bigcup_{v \in B_L(u)} C_L(v) \right)$$

Storage for u: $I(u)$ and the distance from u to every $v \in I(u)$, plus its closest landmarks l_v.

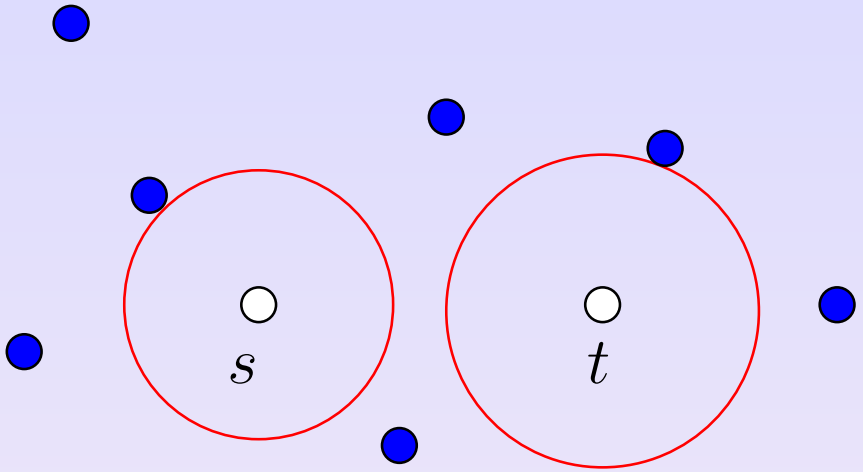
[$|I(u)| = \tilde{O}(n^{2/3})$, storage ✔]

Querying between s and t



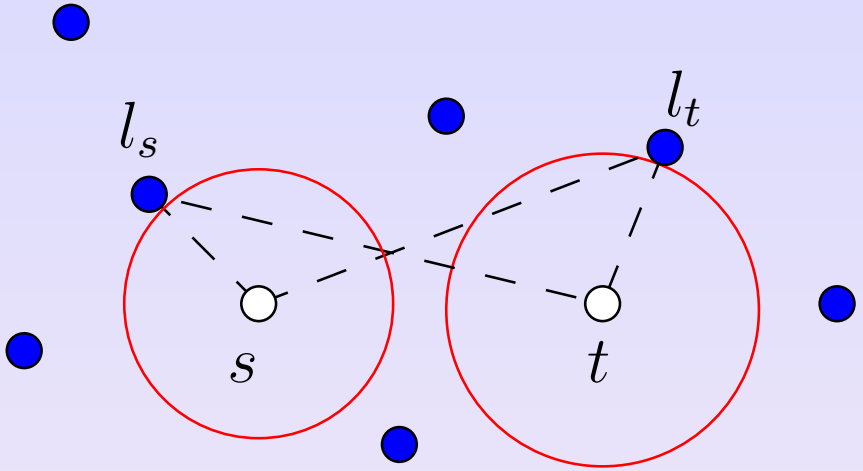
If $t \in I(s)$, **then** returns $d(s,t)$

Querying between s and t



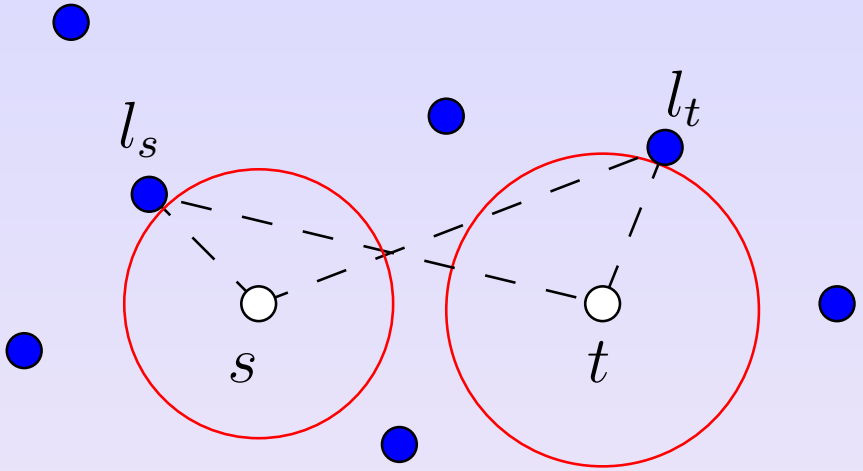
If $t \in I(s)$, **then** returns $d(s,t)$

Querying between s and t



If $t \in I(s)$, **then** returns $d(s,t)$
else returns $\min\{d(s,l_s)+d(l_s),t), d(t,l_t)+d(l_t,t)\}$

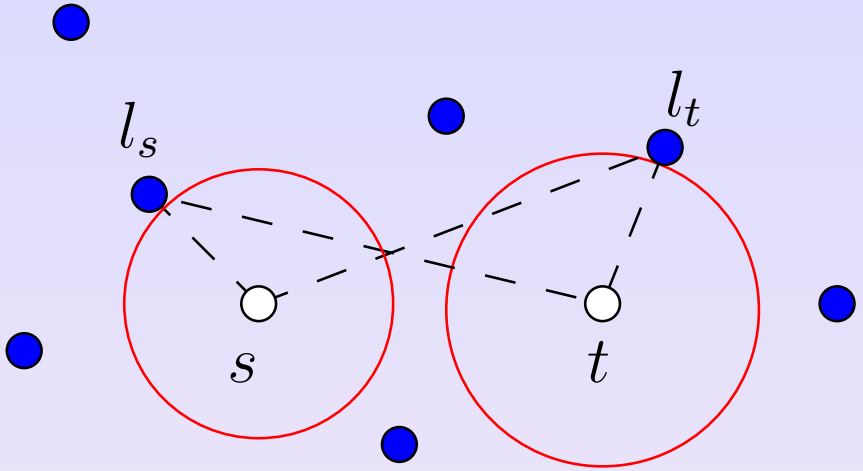
Querying between s and t



If $t \in I(s)$, **then** returns $d(s,t)$
else returns $\min\{d(s,l_s)+d(l_s),t), d(t,l_t)+d(l_t,t)\}$

[dictionary and 2-level hash table, query time ✔]

Querying between s and t



If $t \in I(s)$, **then** returns $d(s,t)$
else returns $\min\{d(s,l_s)+d(l_s),t), d(t,l_t)+d(l_t,t)\}$

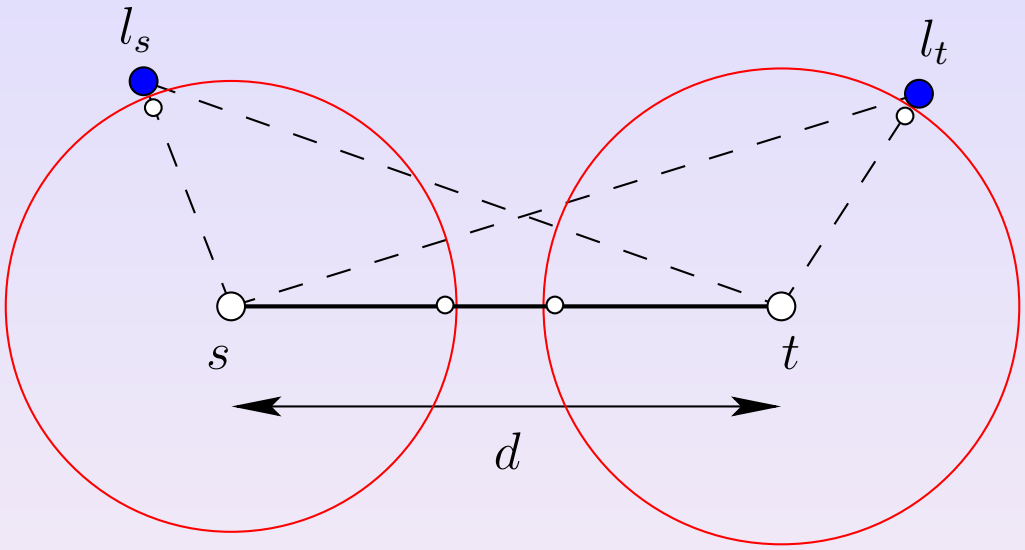
[stretch $2d+1$?]

☞ If $t \notin I(s)$, then $B_L(s) \cap B_L(t) = \varnothing$

[otherwise $\exists w \in B_L(t) \cap B_L(s) \Rightarrow t \in C_L(w)$ and $w \in B_L(s) \Rightarrow t \in I(s)$]

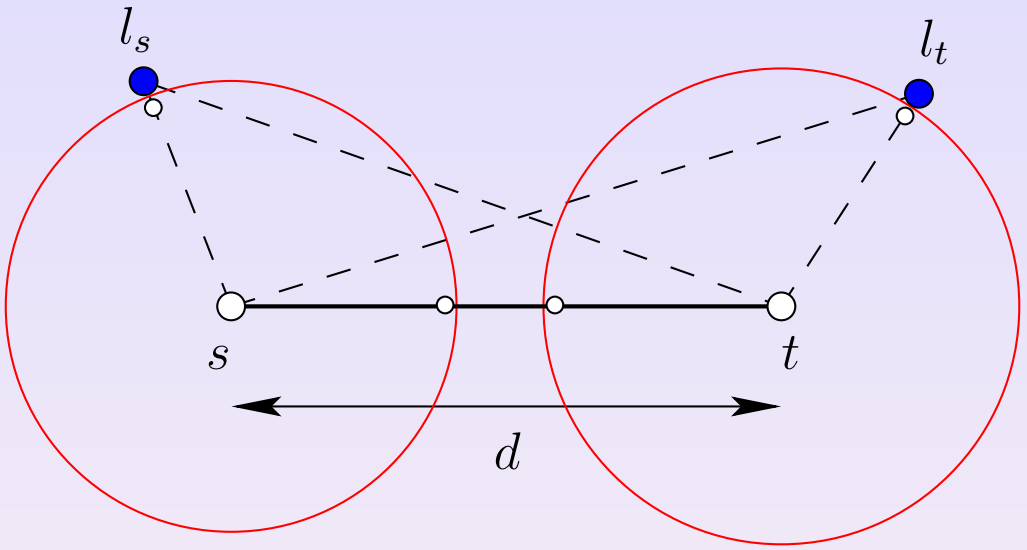
If $t \notin I(s)$, i.e., $B_L(s) \cap B_L(t) = \varnothing$

W.l.o.g. $d(s, l_s) \leqslant d(t, l_t)$



If $t \notin I(s)$, i.e., $B_L(s) \cap B_L(t) = \varnothing$

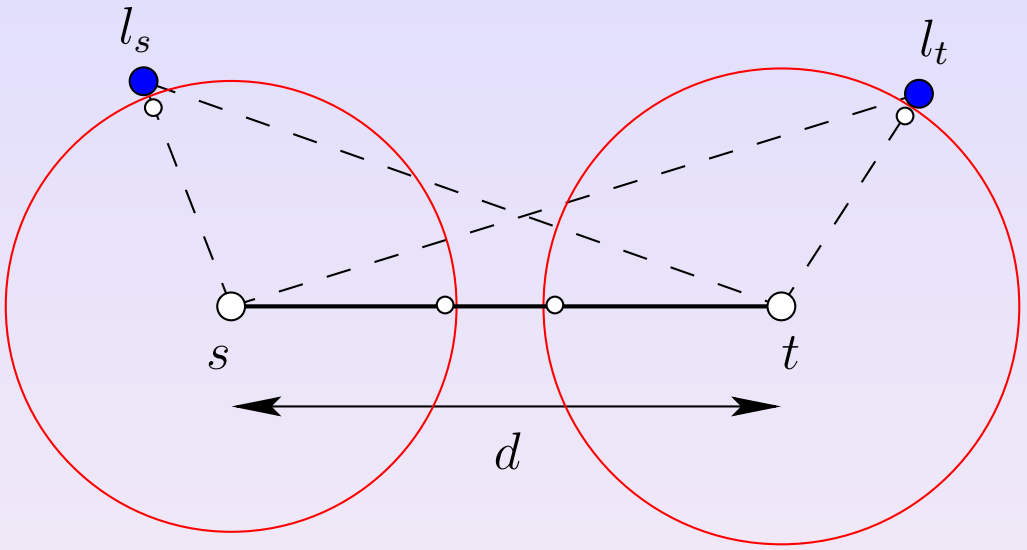
W.l.o.g. $d(s, l_s) \leqslant d(t, l_t)$



☞ $[d(s, l_s) - 1] + 1 + [d(t, l_t) - 1] \leqslant d$

If $t \notin I(s)$, i.e., $B_L(s) \cap B_L(t) = \varnothing$

W.l.o.g. $d(s, l_s) \leqslant d(t, l_t)$

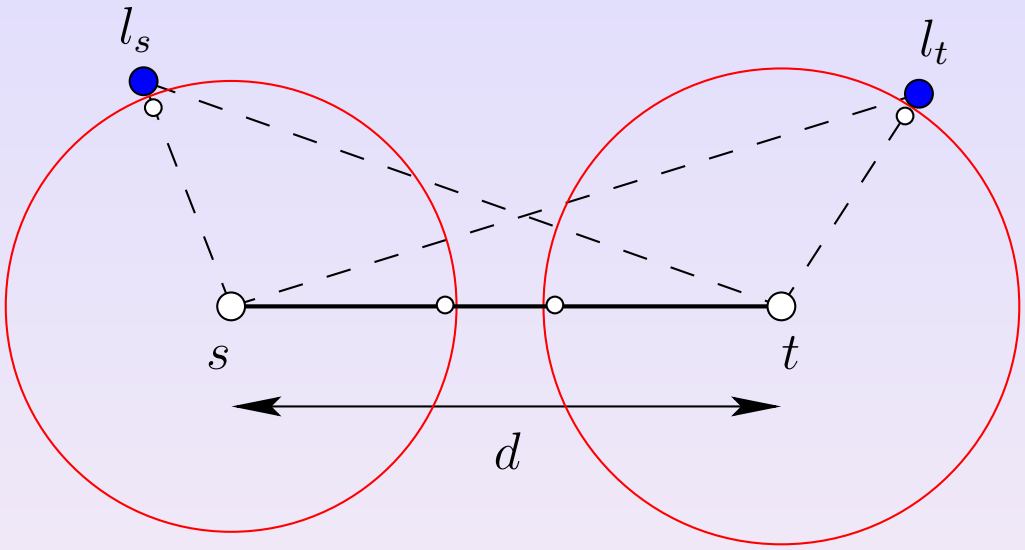


$$\boxed{\text{☞ } [d(s, l_s) - 1] + 1 + [d(t, l_t) - 1] \leqslant d}$$

$$\Rightarrow 2d(s, l_s) \leqslant d + 1$$

If $t \notin I(s)$, i.e., $B_L(s) \cap B_L(t) = \varnothing$

W.l.o.g. $d(s, l_s) \leqslant d(t, l_t)$



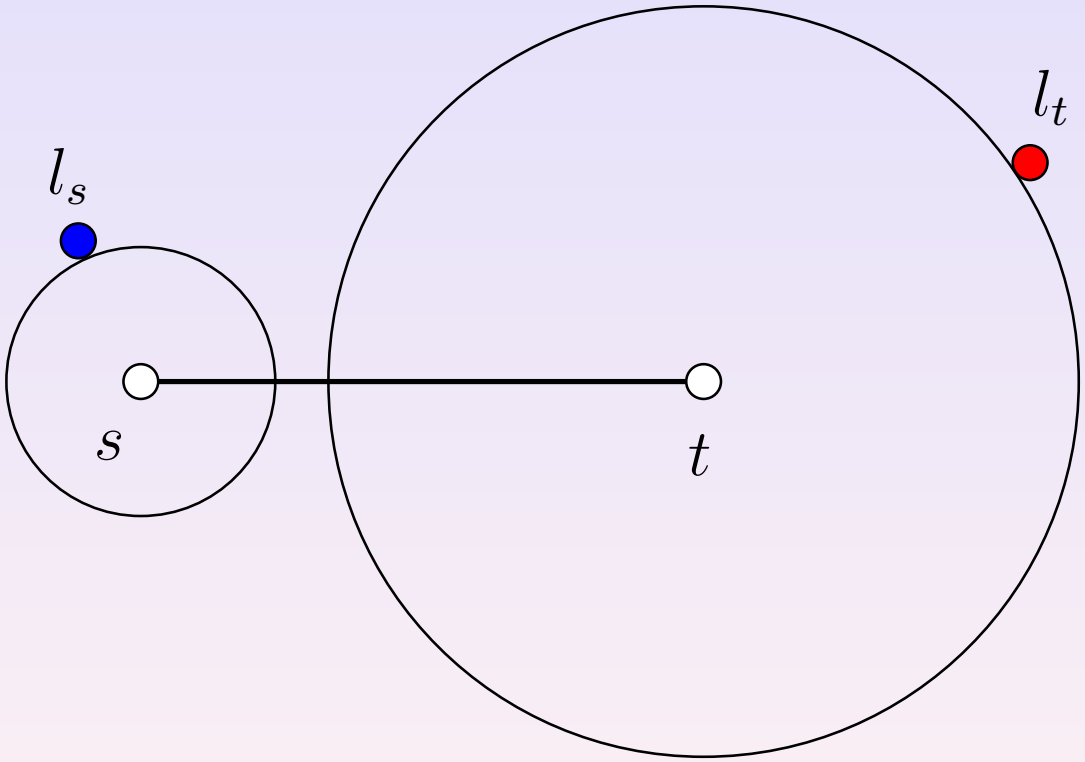
$$\boxed{\text{☞ } [d(s, l_s) - 1] + 1 + [d(t, l_t) - 1] \leqslant d}$$

$$\Rightarrow 2d(s, l_s) \leqslant d + 1$$

$$\Rightarrow \hat{d} \leqslant 2d(s, l_s) + d \leqslant 2d + 1$$

Observation

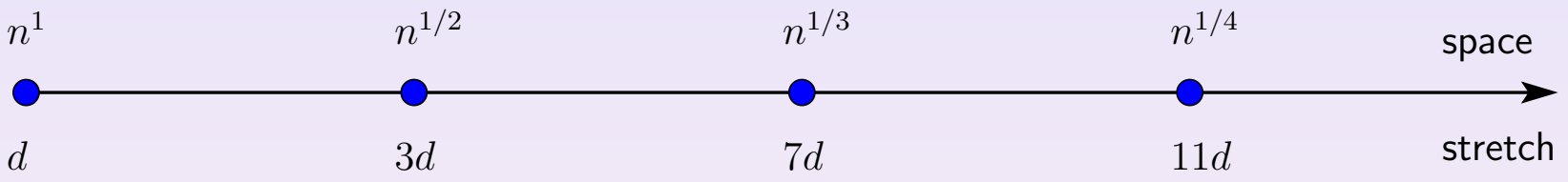
Routing with $\tilde{O}(n^{2/3})$ bit routing tables, polylog addresses and stretch $2d+1$ is not known. Routing query is not symetric!



What about Compact Routing?

Best routing scheme [Thorup-Zwick (SPAA '01)]
achieves stretch $(4k-5)d$ and routing tables of size $\tilde{O}(n^{1/k})$.

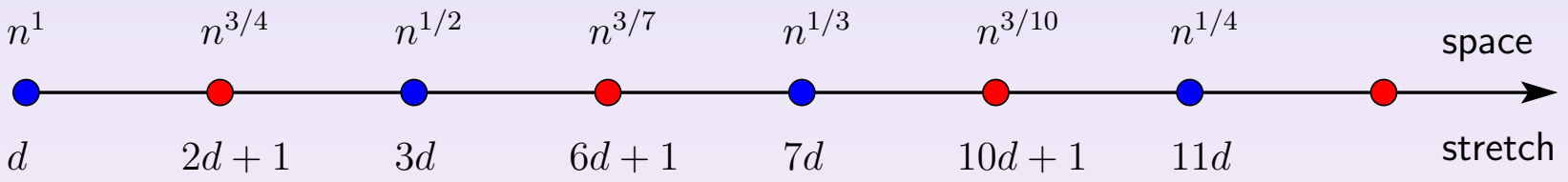
[TZ01]



What about Compact Routing?

Best routing scheme [Thorup-Zwick (SPAA '01)] achieves stretch $(4k-5)d$ and routing tables of size $\tilde{O}(n^{1/k})$.

[TZ01][us]



New construction with stretch $(4k-6)d+1$ and routing tables of size $\tilde{O}(n^{3/(3k-2)})$.

Conclusion

	stretch		size
Spanner	$d+2$		$O(n^{1/2}) \cdot n$
Distance Labeling	?	$2d+1$	$\tilde{O}(n^{2/3})$
Compact Routing		$2d+1$	$\tilde{O}(n^{3/4})$
Compact Routing	$d+\beta$		$\tilde{\Omega}(n/\beta^2)$

Conclusion

	stretch		size
Spanner	$d+2$		$O(n^{1/2}) \cdot n$
Distance Labeling	?	$2d+1$	$\tilde{O}(n^{2/3})$
Compact Routing		$2d+1$	$\tilde{O}(n^{3/4})$
Compact Routing	$d+\beta$		$\tilde{\Omega}(n/\beta^2)$

Thank You!