

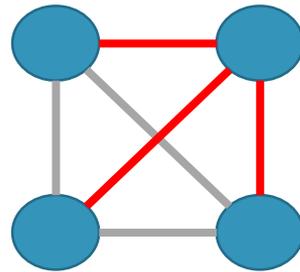
Distributed Spanner Approximation

Michal Dory, Technion

Joint work with: Keren Censor-Hillel, Technion

Spanners

A **k -spanner** of a graph G is a subgraph of G that preserves distances up to a multiplicative factor of k .



Spanners

- There are many constructions which give a **global guarantee** on the size of the spanner:
 $(2k - 1)$ -spanners with $O(n^{1+1/k})$ edges
- This is optimal in the worst case assuming Erdős's girth conjecture.

Spanner Approximation

- What about approximating the **minimum** k -spanner?

$$\text{Number of edges} \leq \alpha \cdot OPT$$

- There are graphs where any **2-spanner** has $\Omega(n^2)$ edges, this is also true for k -spanners in **directed** graphs.

In the sequential setting:

- 2-spanner: $O\left(\log \frac{|E|}{|V|}\right)$ -approximation [Kortsarz and Peleg 1994]
- Directed k -spanner: $O(\sqrt{n} \log n)$ -approximation [Berman, Bhattacharyya, Makarychev, Raskhodnikova and Yaroslavtsev 2013]

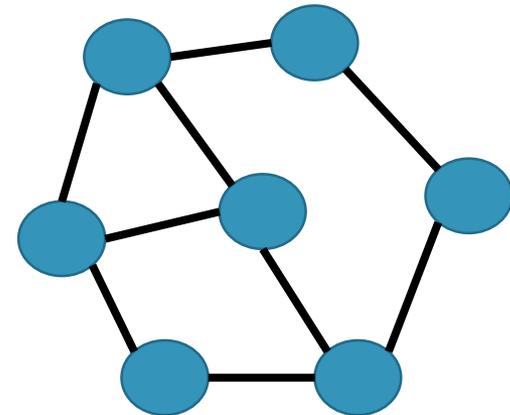
Hardness Results:

- 2-spanner: $\Omega(\log n)$ [Kortsarz 2001]
- Directed k -spanner: $\Omega(2^{(\log^{1-\varepsilon} n)})$ [Elkin and Peleg 2007]
- Undirected k -spanner: $\Omega(2^{(\log^{1-\varepsilon} n)/k})$ [Dinitz, Kortsarz and Raz 2016]

The Distributed Models

n vertices exchange messages in **synchronous** rounds

The model	Message size
LOCAL	unbounded
CONGEST	$\Theta(\log n)$ bits



In the LOCAL model

2-spanners:

Approximation	Number of rounds	
$O(\log n)$	$O(\log n)$	[Dinitz and Krauthgamer, 2011]

Directed k -spanners:

Approximation	Number of rounds	
$O(\sqrt{n} \log n)$	$O(k \log n)$	[Dinitz and Nazari, 2017]
$O(n^\epsilon)$	constant	[Barenboim, Elkin and Gavoille, 2016]
$(1 + \epsilon)$	$O(\text{poly}(\log n / \epsilon))$	Our Results

In the CONGEST model

Undirected $(2k - 1)$ -spanners:

There are **global constructions** of spanners with $O(n^{1+1/k})$ edges



Approximation	Number of rounds	
$O(n^{1/k})$	k	[Elkin and Neiman, 2017]

Spanner Approximation

- Can we give efficient approximations also in the **CONGEST** model?

Spanner Approximation

- Can we give efficient approximations also in the **CONGEST** model?

Approximating k -spanners in **directed** or **weighted** graphs is hard in the CONGEST model.

Spanner Approximation

Approximating k -spanners in **directed** or **weighted** graphs is hard in the CONGEST model.

This gives a strict **separation** between:

- The **CONGEST** and **LOCAL** models
- The undirected and directed variants

Hardness of Approximation

Directed k -spanner for $k \geq 5$:

- Randomized algorithms - $\tilde{\Omega}(\sqrt{n/\alpha})$ rounds for an α -approximation.
- Deterministic algorithms - $\tilde{\Omega}(n/\sqrt{\alpha})$

Weighted k -spanner for $k \geq 4$:

- Directed graphs - $\tilde{\Omega}(n)$
- Undirected graphs - $\tilde{\Omega}(n/k)$

How to show the results?

- Learning if two input strings of size N are **disjoint** requires exchanging $\Omega(N)$ bits.

Cartes de Noël



Cartes de Noël
www.milano.com
Lado 22

© 2010 Disney. Tutti i diritti sono riservati. Alice è un personaggio di Alice in Wonderland, opera di Lewis Carroll. Alice è un marchio registrato di Disney.

0	1	1	0	0	1	1
---	---	---	---	---	---	---

0	0	1	1	1	0	1
---	---	---	---	---	---	---



How to show the results?

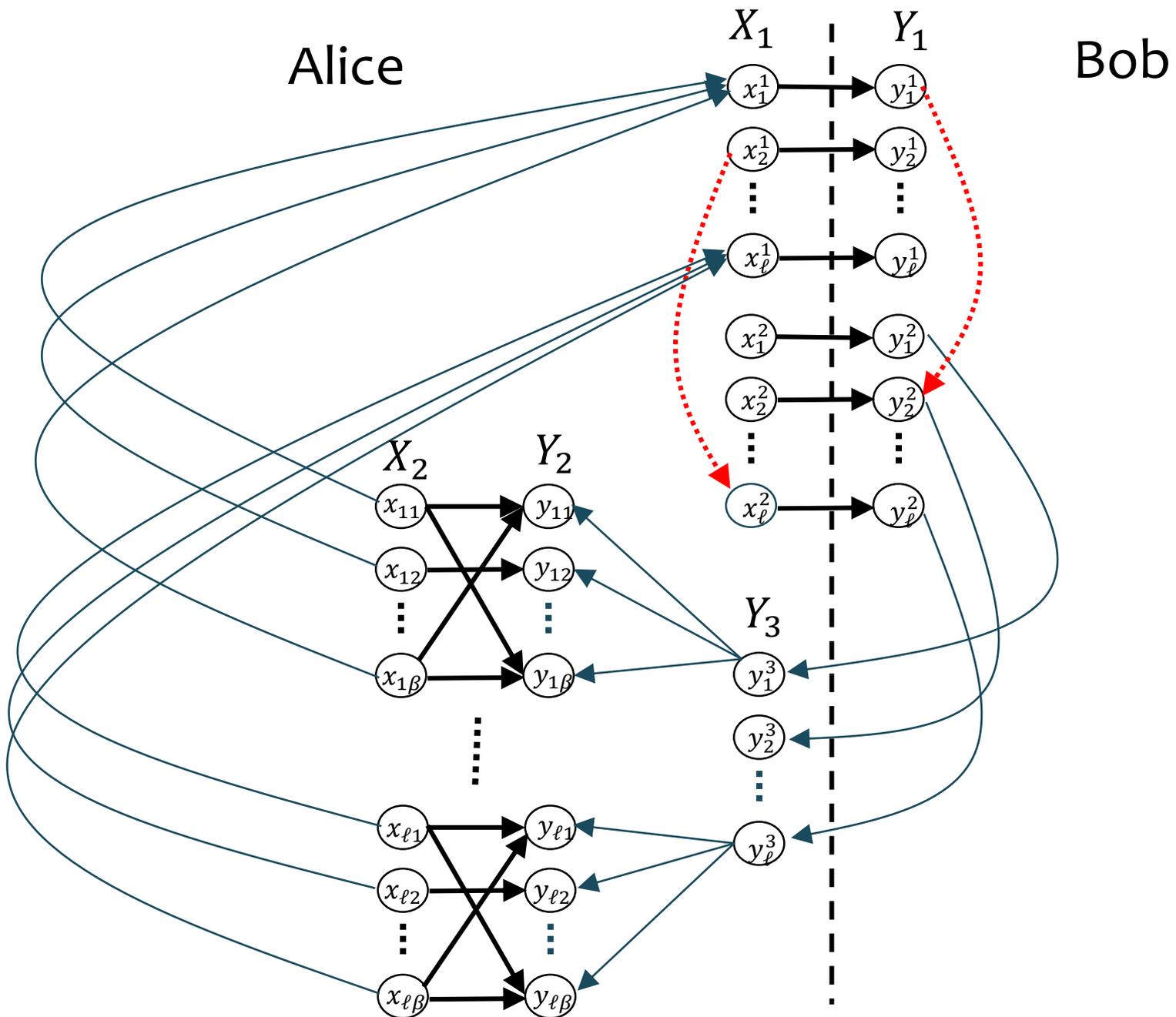
- **The goal:** create a graph G that depends on the inputs of Alice and Bob, such that

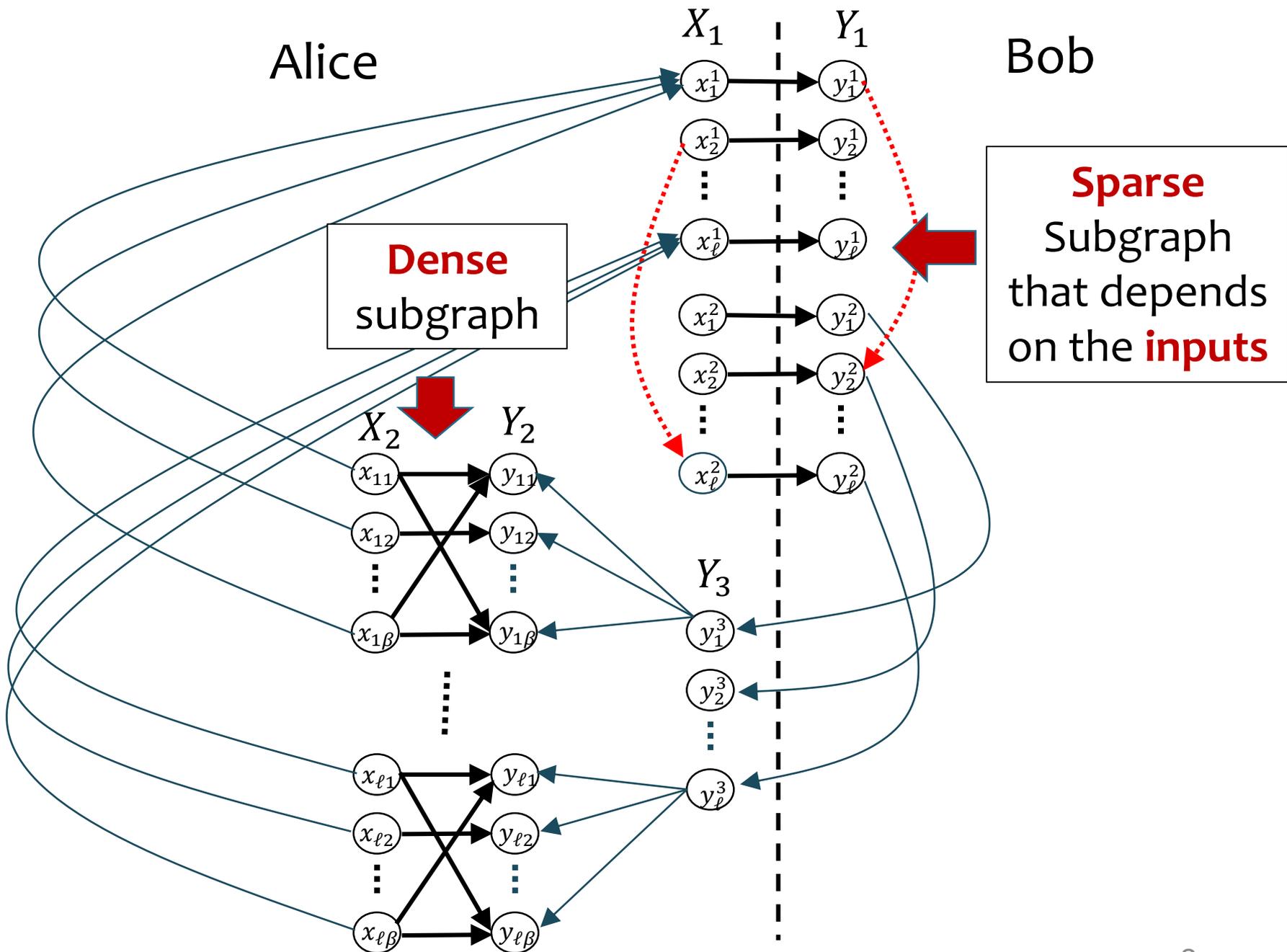
G has a **sparse spanner** \iff the inputs satisfy some property

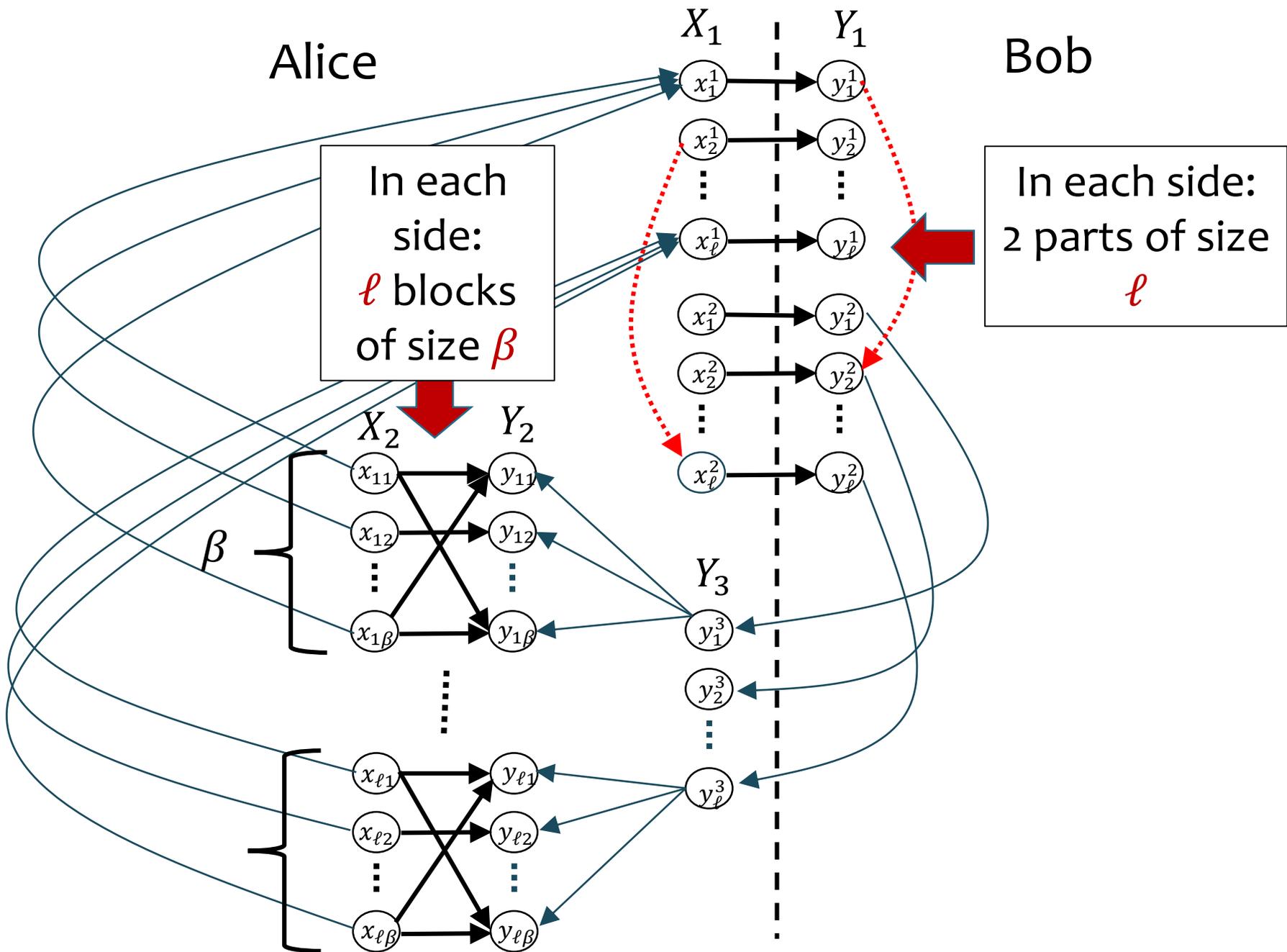
Creating gaps

We show several approaches to **create gaps**:

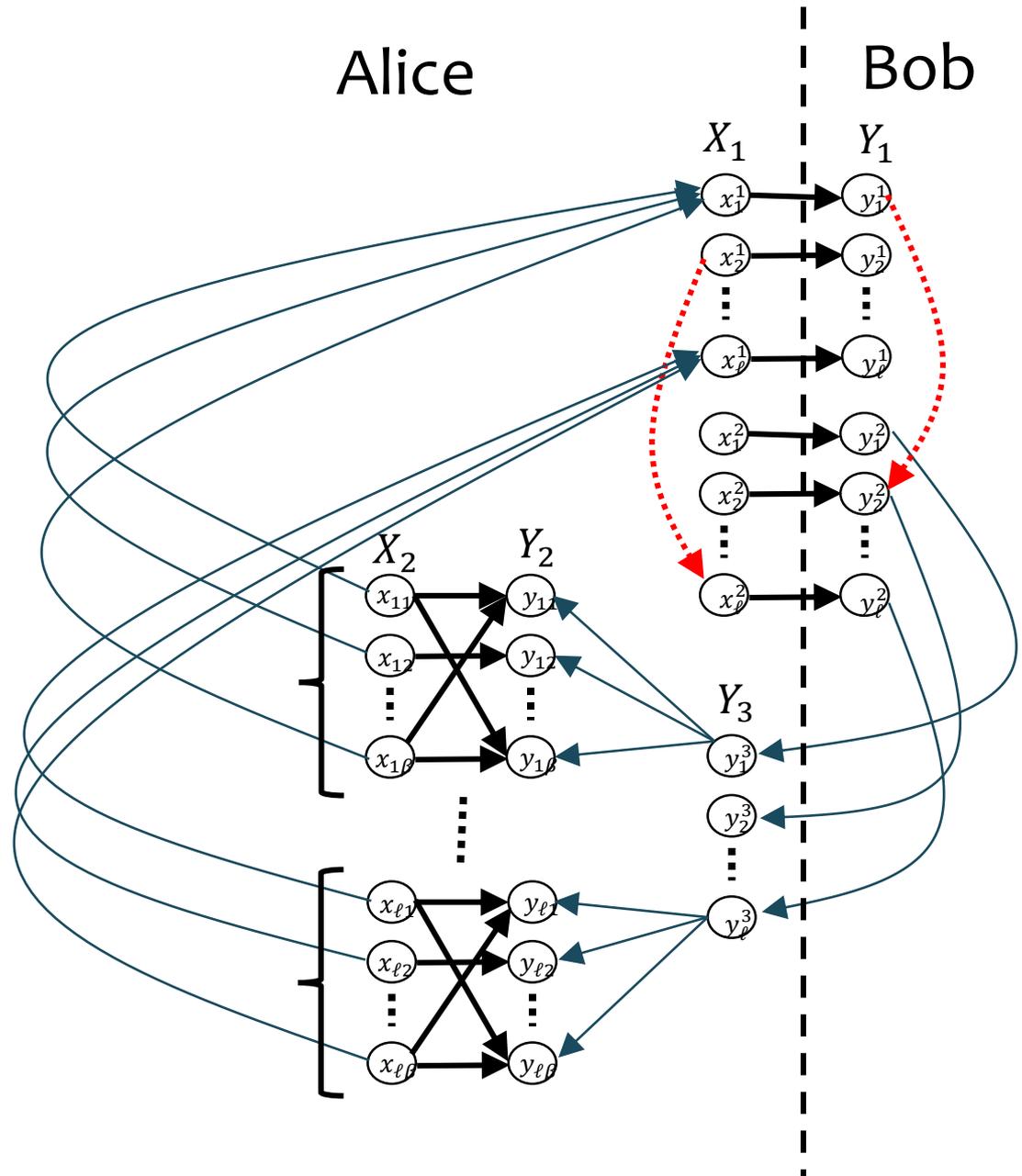
- A **construction** where each input bit affects $\Omega(\alpha n)$ edges of the spanner
- Using the **gap-disjointness** problem
- Using the **weights**



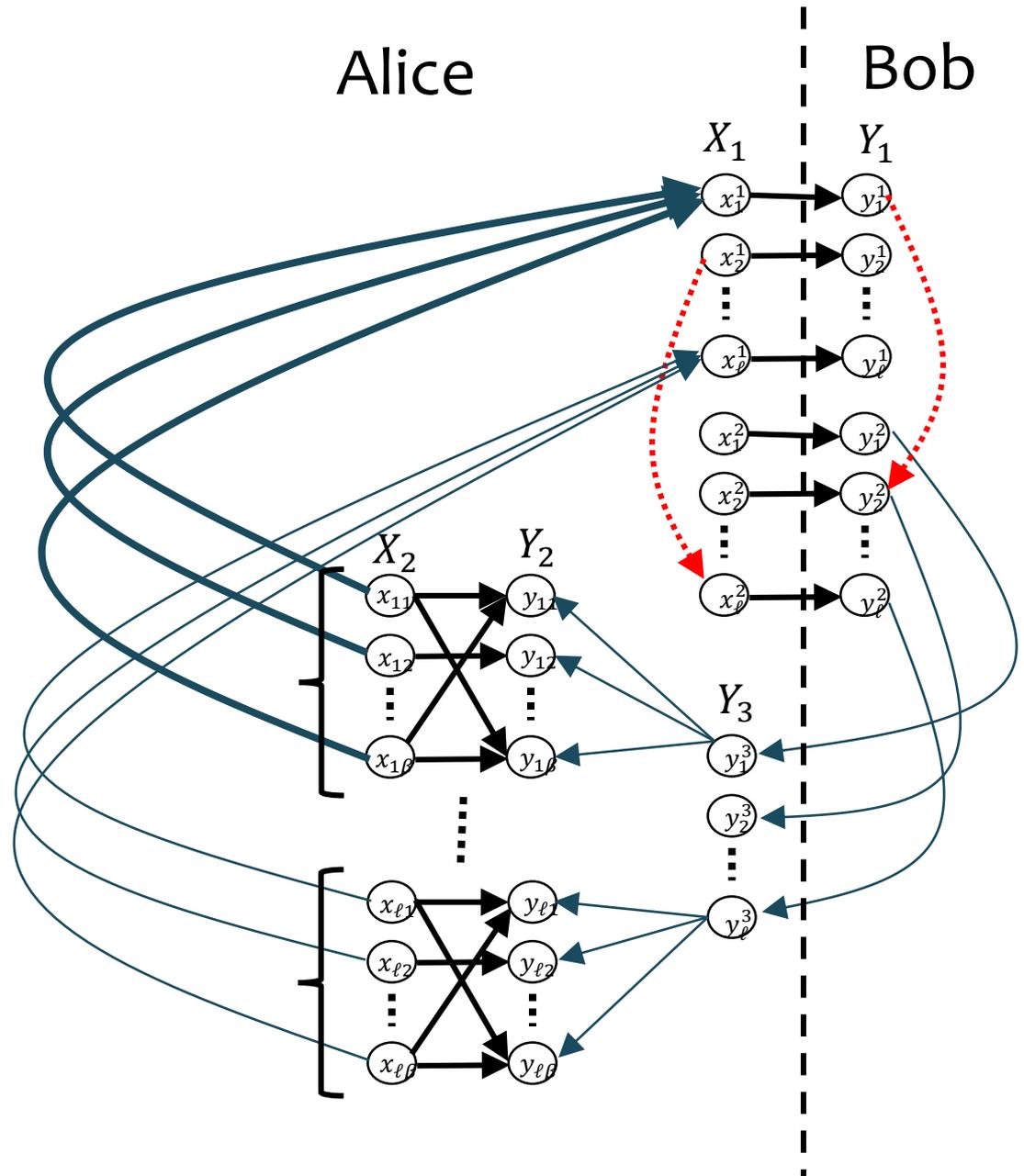




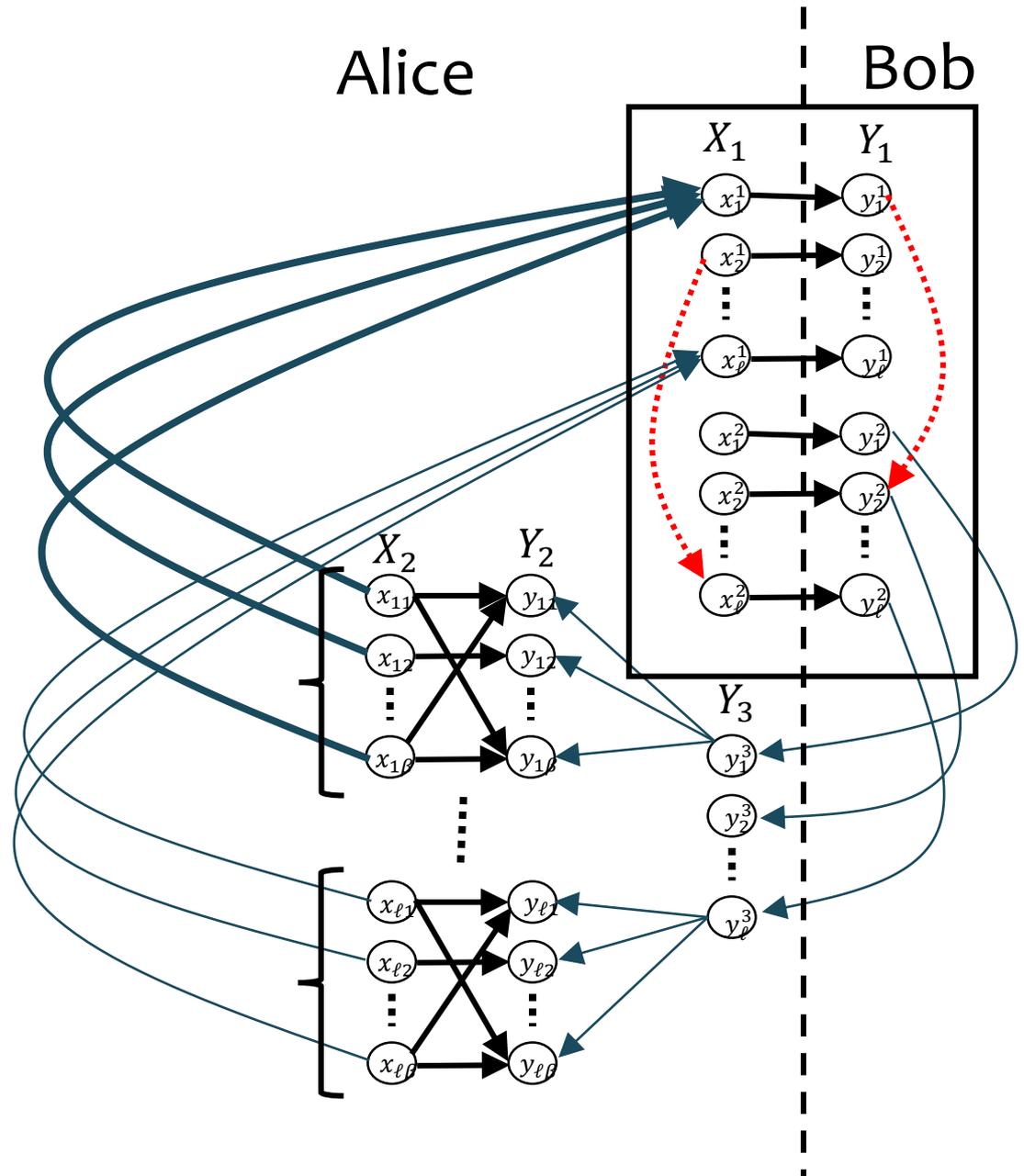
Each **block** is connected to one vertex outside the block



Each **block** is connected to one vertex outside the block



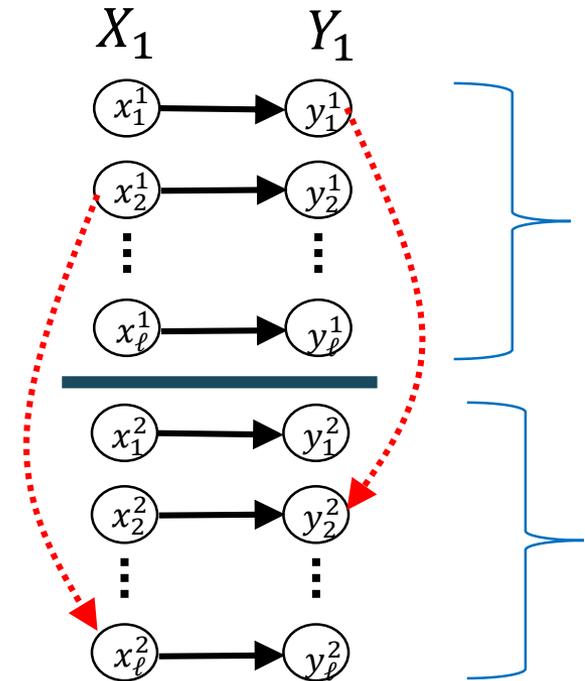
Each **block** is connected to one vertex outside the block



There is a directed path of length 2 from x_i^1 to y_j^2

\Leftrightarrow

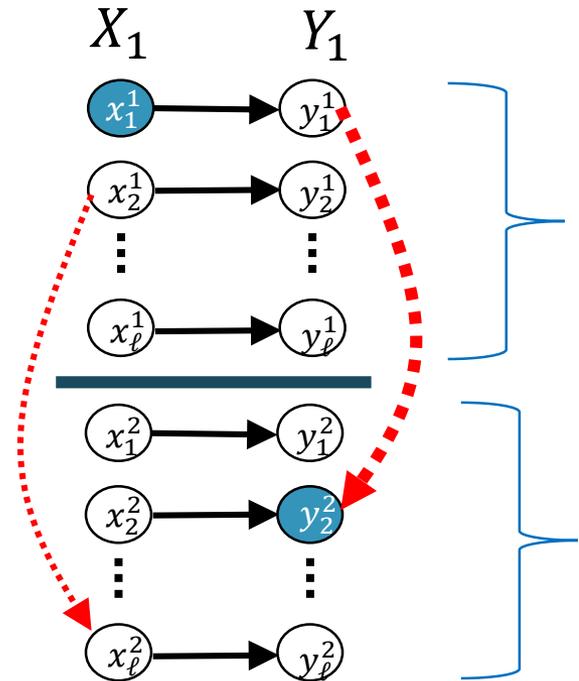
$$a_{ij} = 0 \text{ or } b_{ij} = 0$$



There is a directed path of length 2 from x_i^1 to y_j^2

\Leftrightarrow

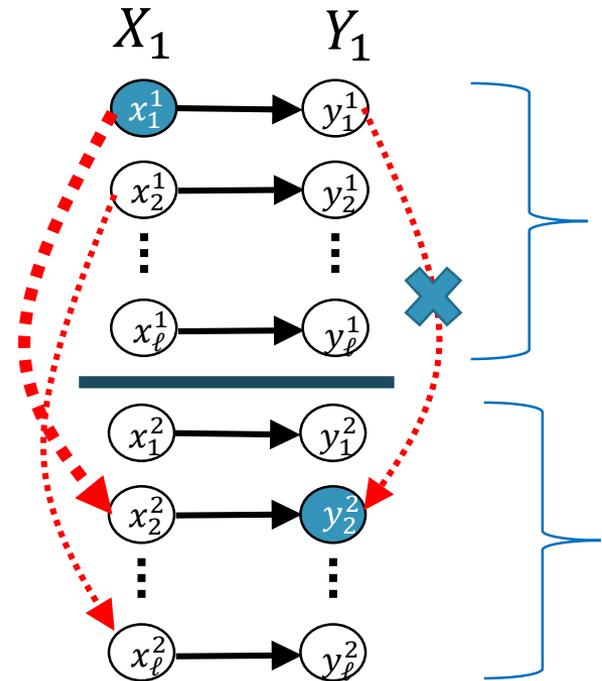
$$a_{ij} = 0 \text{ or } b_{ij} = 0$$



There is a directed path of length 2 from x_i^1 to y_j^2

\Leftrightarrow

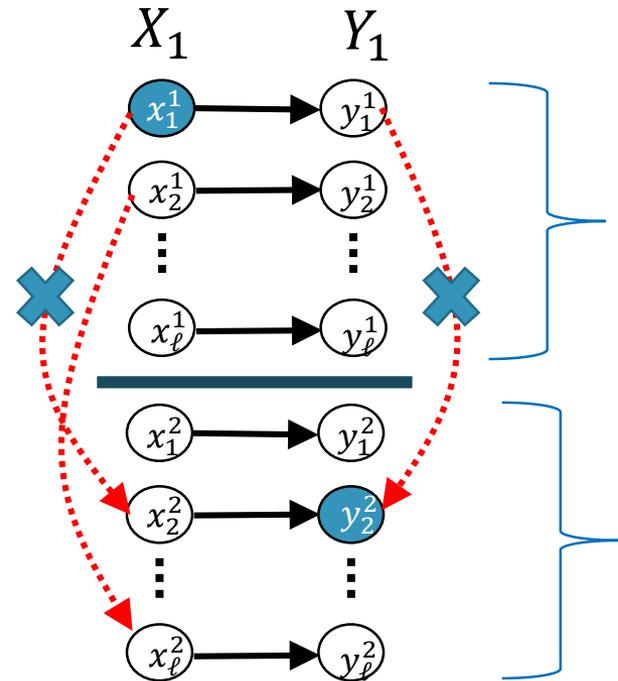
$$a_{ij} = 0 \text{ or } b_{ij} = 0$$



There is a directed path of length 2 from x_i^1 to y_j^2

\Leftrightarrow

$$a_{ij} = 0 \text{ or } b_{ij} = 0$$

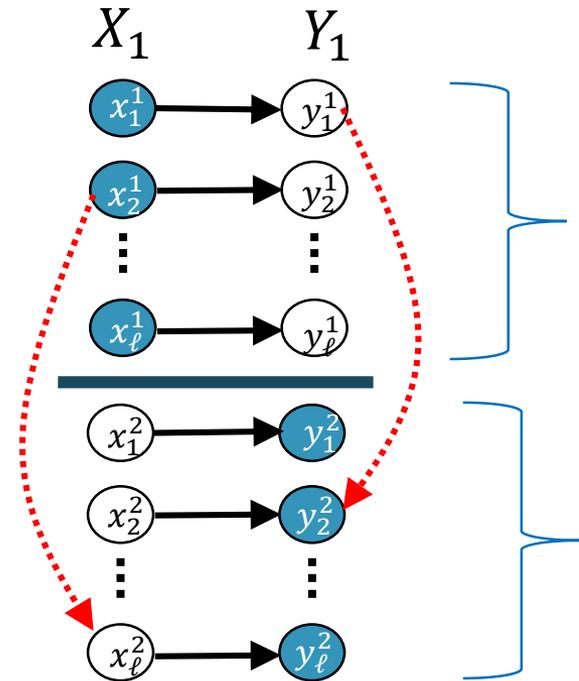


Conclusion:

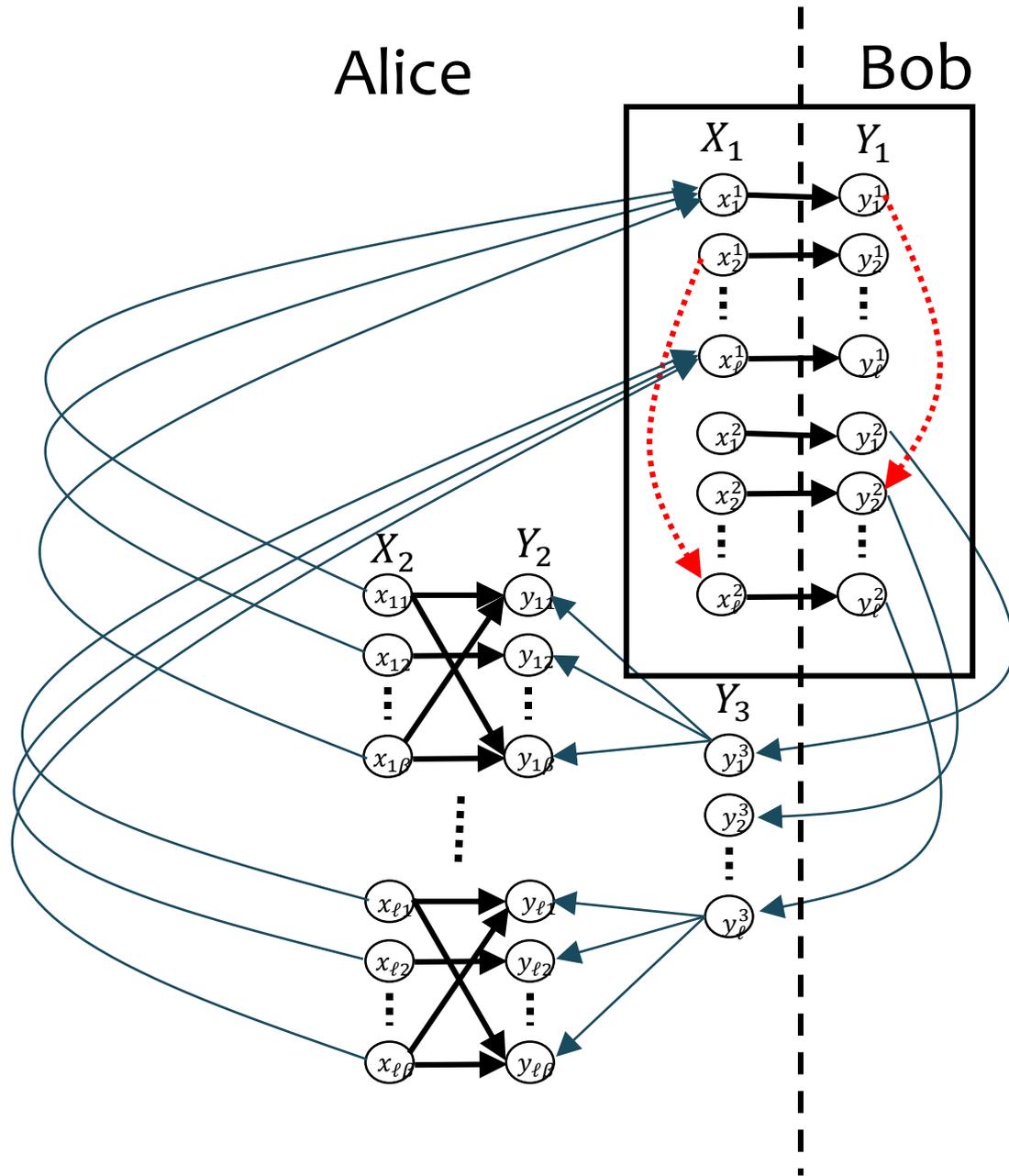
a, b are *disjoint*

\Leftrightarrow

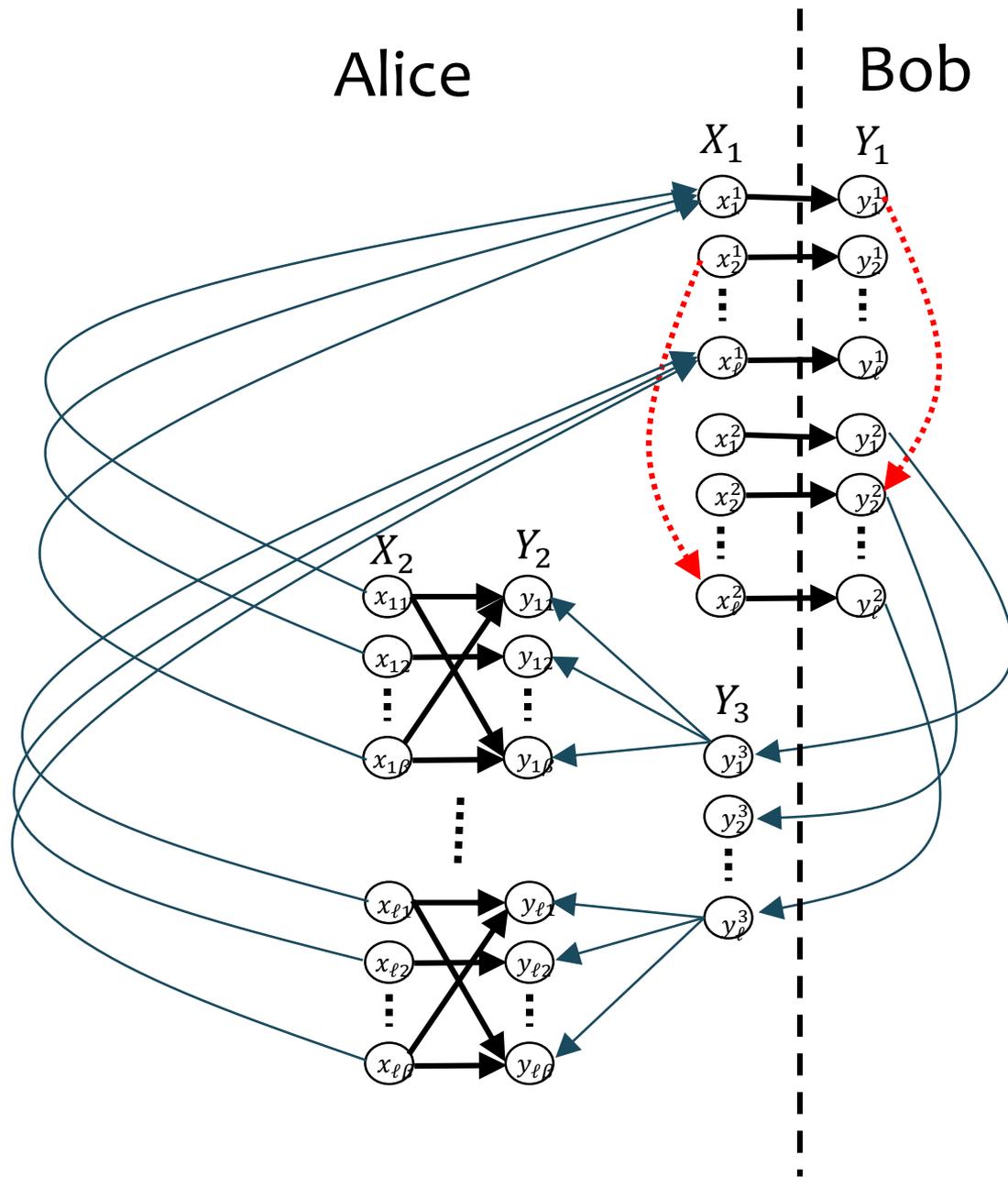
there is a path of length 2
from x_i^1 to y_j^2 for all i, j



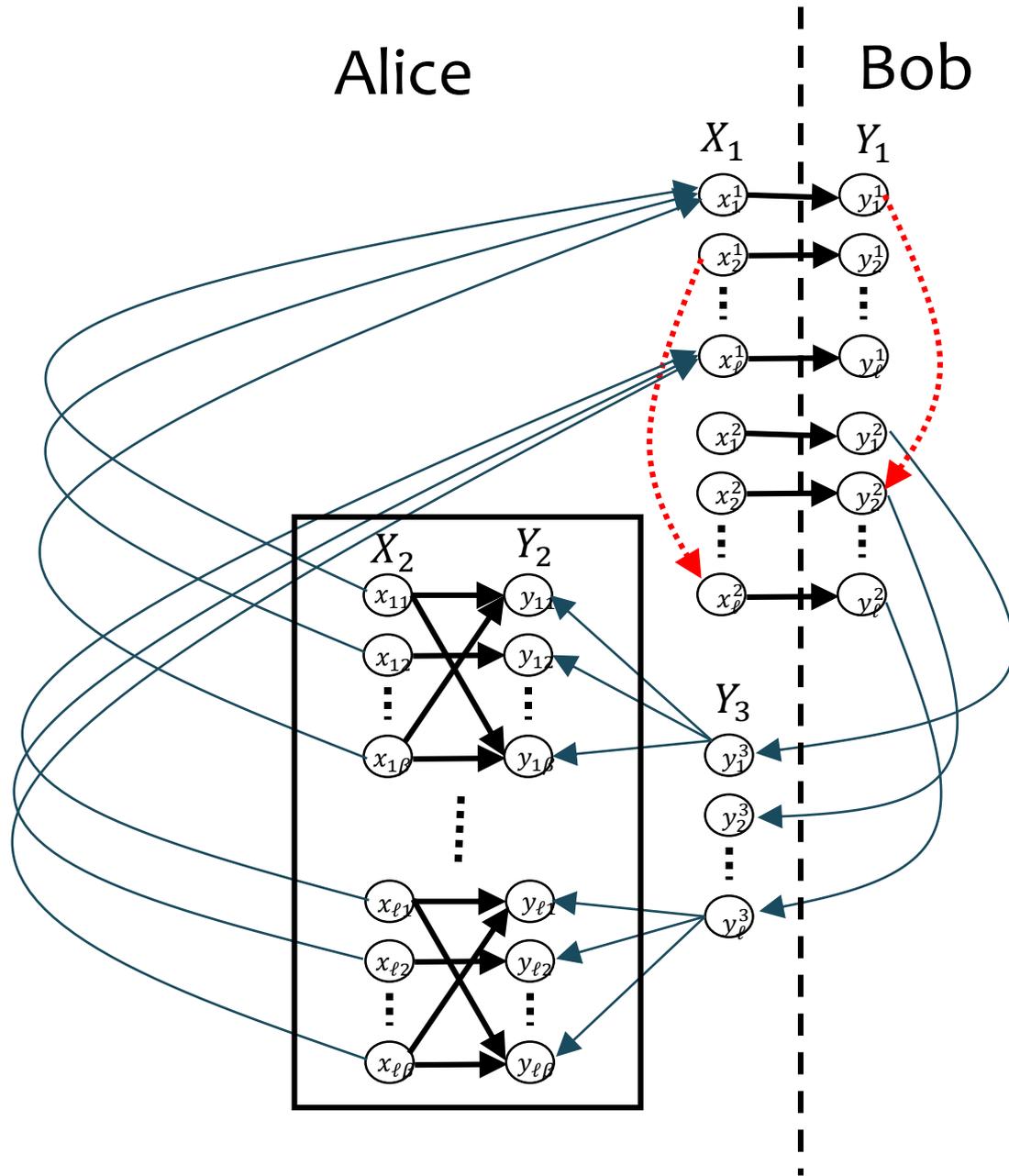
G contains a
 sparse 5-spanner
 \Leftrightarrow
 a, b are disjoint



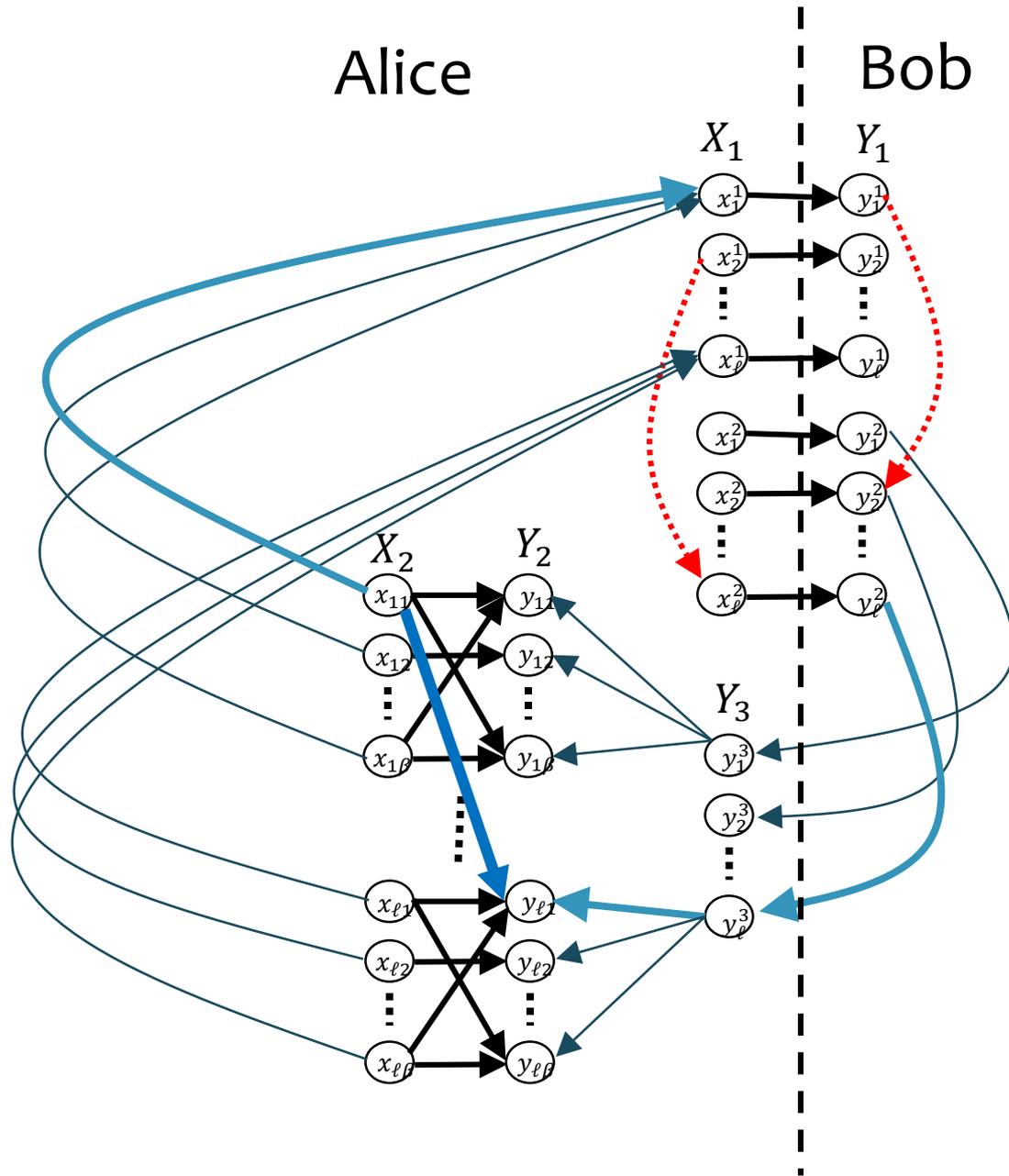
G contains a
 sparse 5-spanner
 \iff
 a, b are disjoint



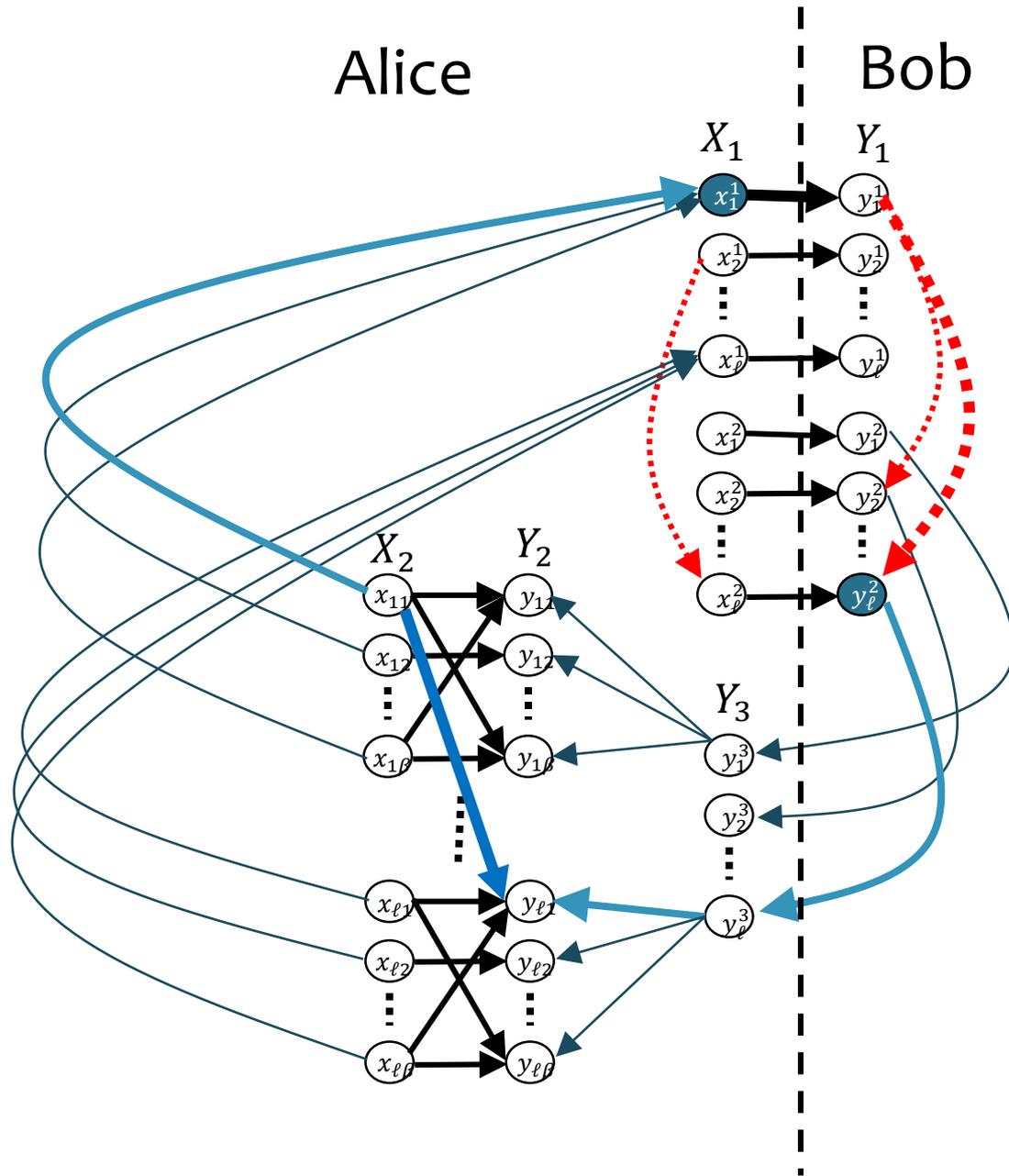
G contains a
 sparse 5-spanner
 \Leftrightarrow
 a, b are disjoint



G contains a
 sparse 5-spanner
 \Leftrightarrow
 a, b are disjoint



G contains a
 sparse 5-spanner
 \Leftrightarrow
 a, b are disjoint



If a, b are **not** disjoint



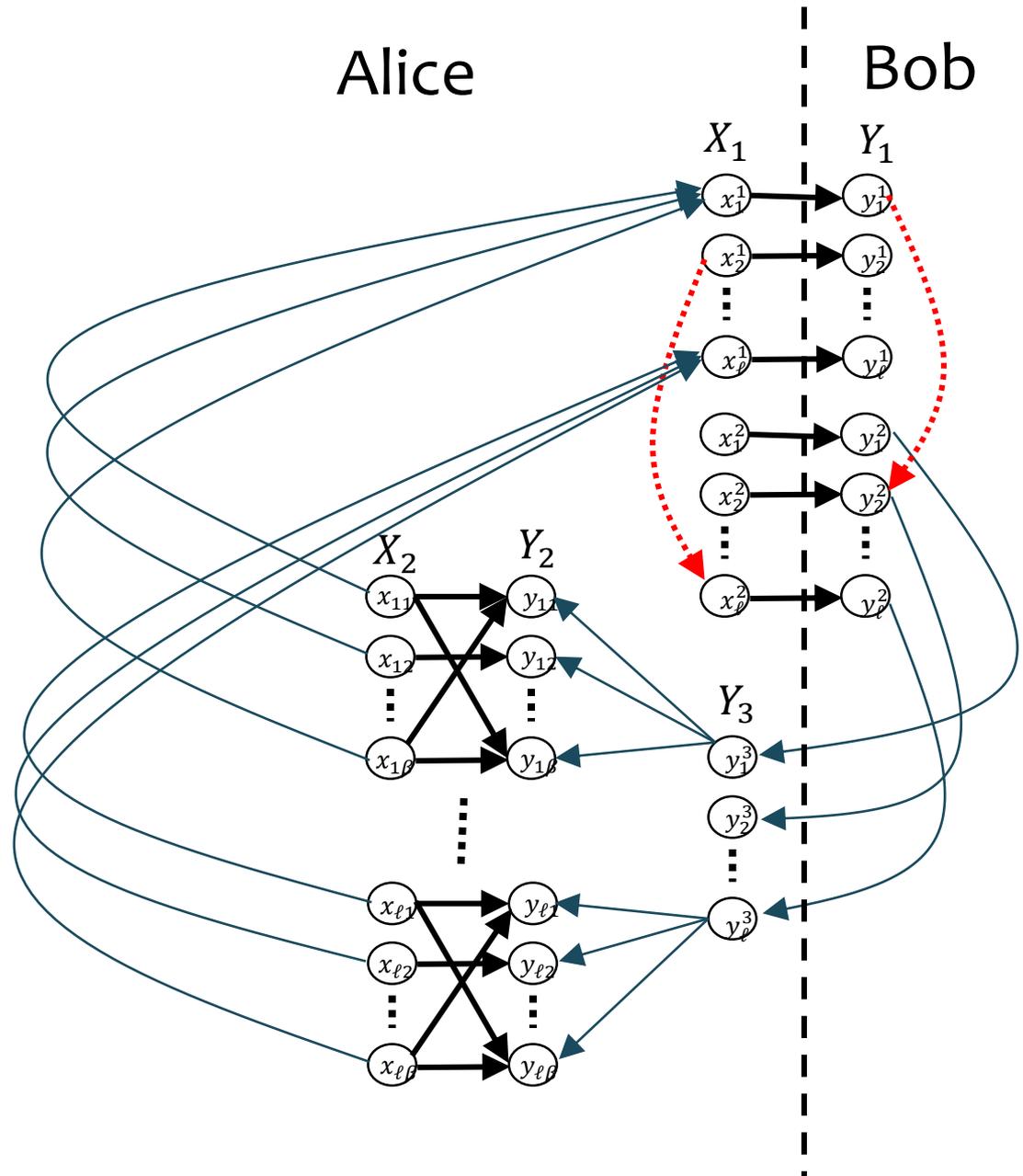
There are i, j where there is **no** directed path of length 2 from

x_i^1 to y_j^2



We need to take at least **β^2 edges** to the spanner

We choose $\beta \approx \sqrt{\alpha n}$



If a, b are **not** disjoint



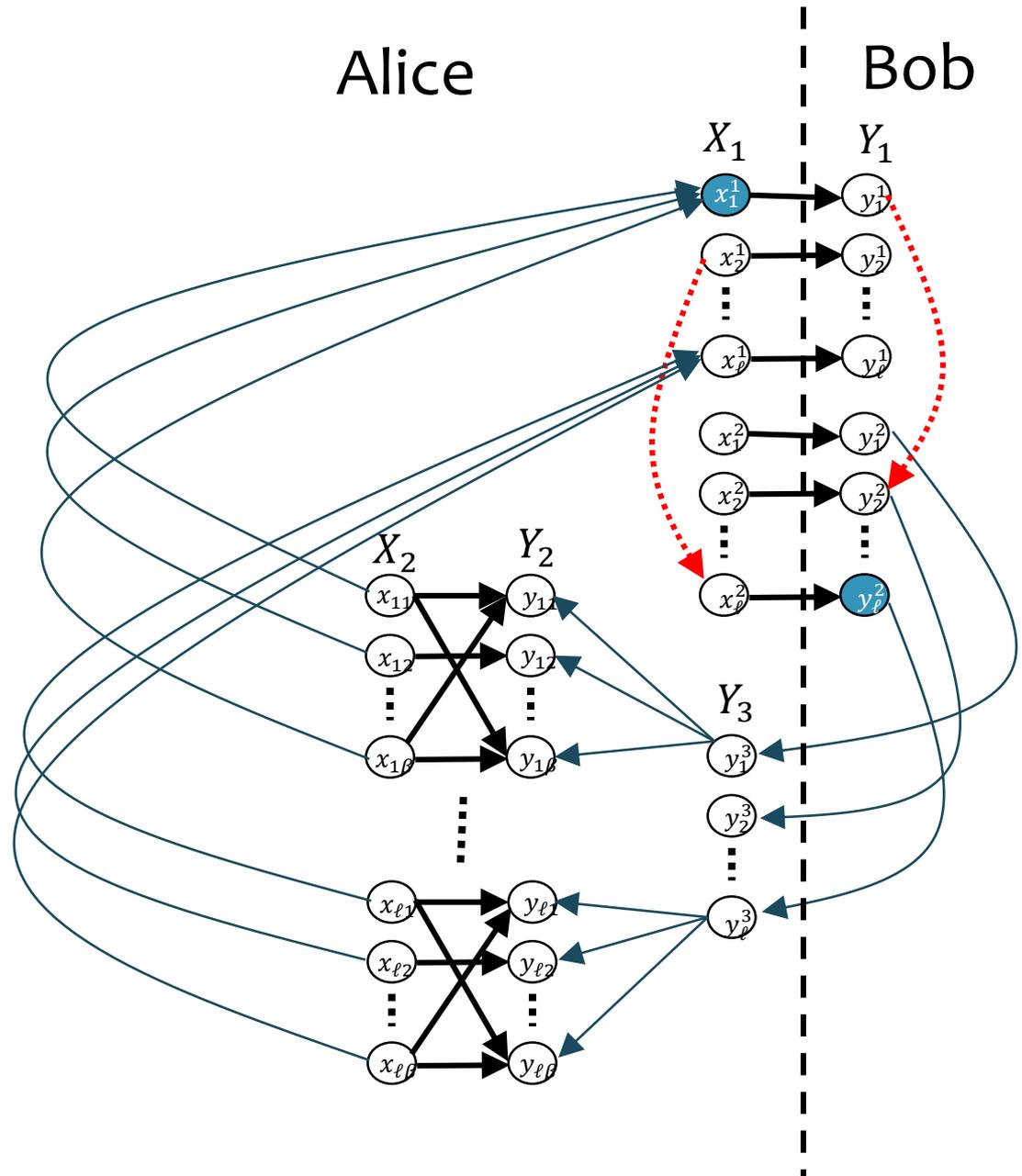
There are i, j where there is **no** directed path of length 2 from

x_i^1 to y_j^2



We need to take at least **β^2 edges** to the spanner

We choose $\beta \approx \sqrt{\alpha n}$



If a, b are **not** disjoint



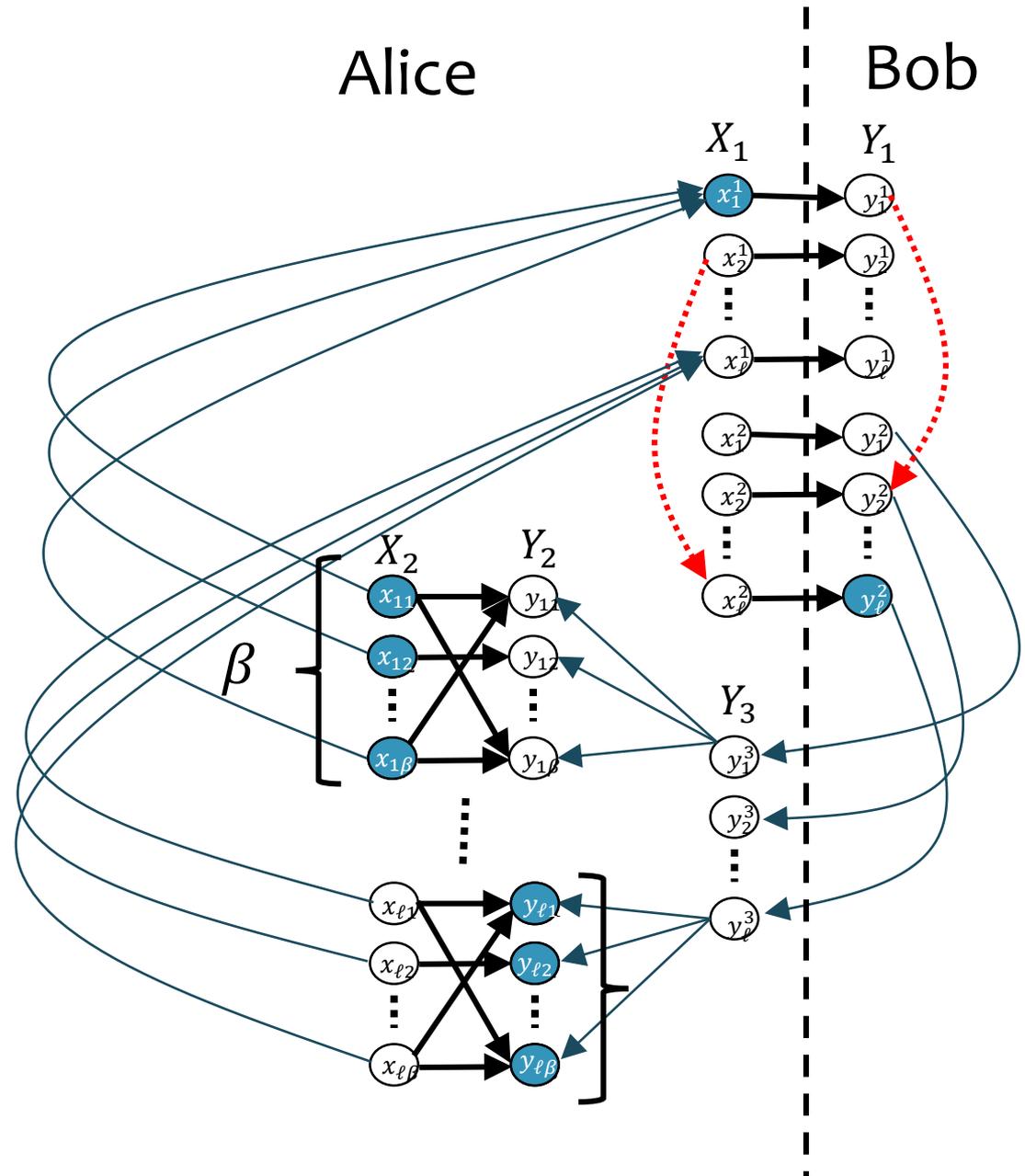
There are i, j where there is **no** directed path of length 2 from

x_i^1 to y_j^2

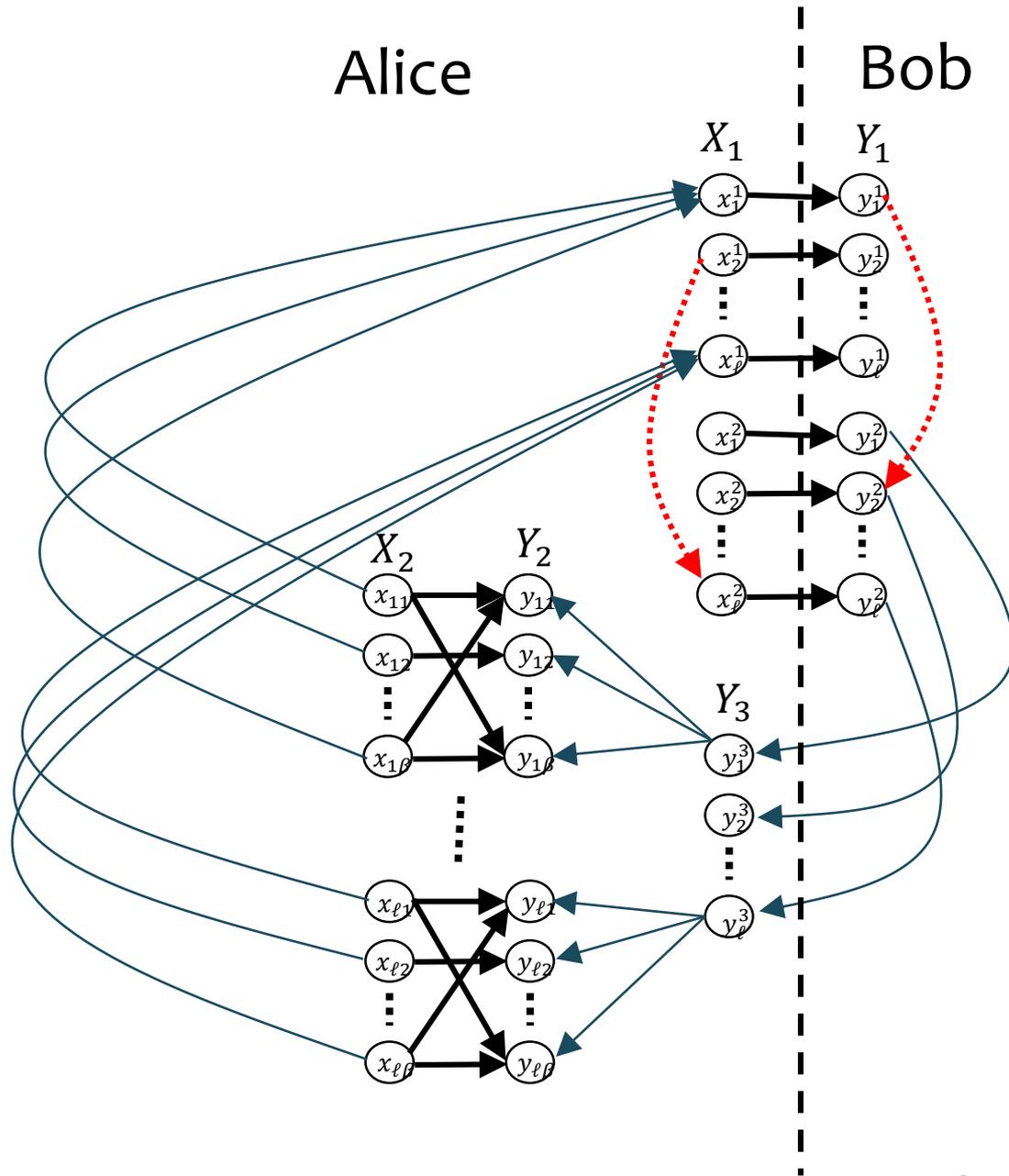


We need to take at least **β^2 edges** to the spanner

We choose $\beta \approx \sqrt{\alpha n}$



G contains a
 sparse 5-spanner
 \Leftrightarrow
 a, b are disjoint

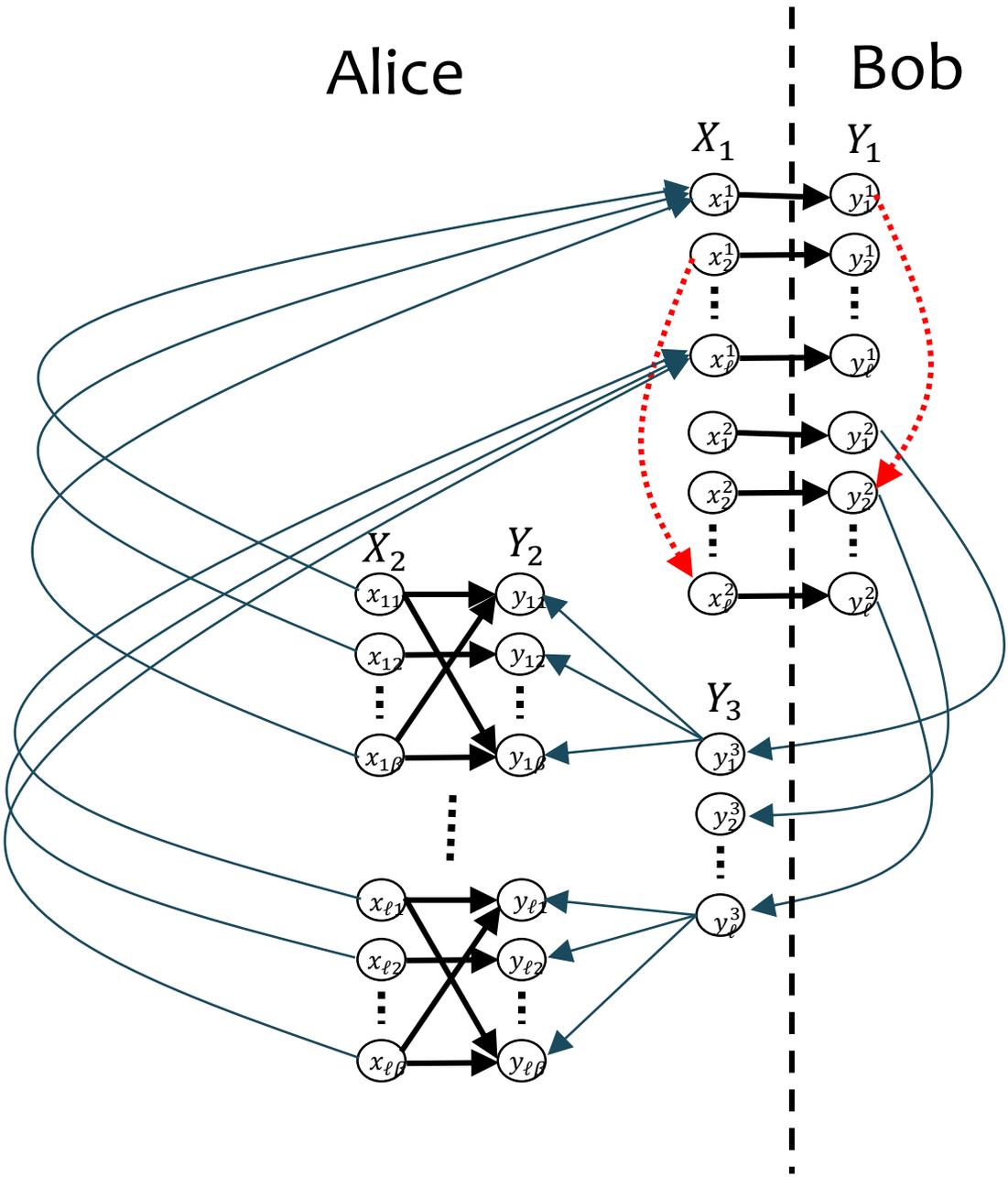


$\Omega(\ell^2)$ lower bound
for set-disjointness

Approximation
algorithm for 5-
spanners in
 $O(T(n))$ rounds



A protocol for
disjointness that
takes
 $O(T(n) \log n |CUT_{A-B}|)$
bits



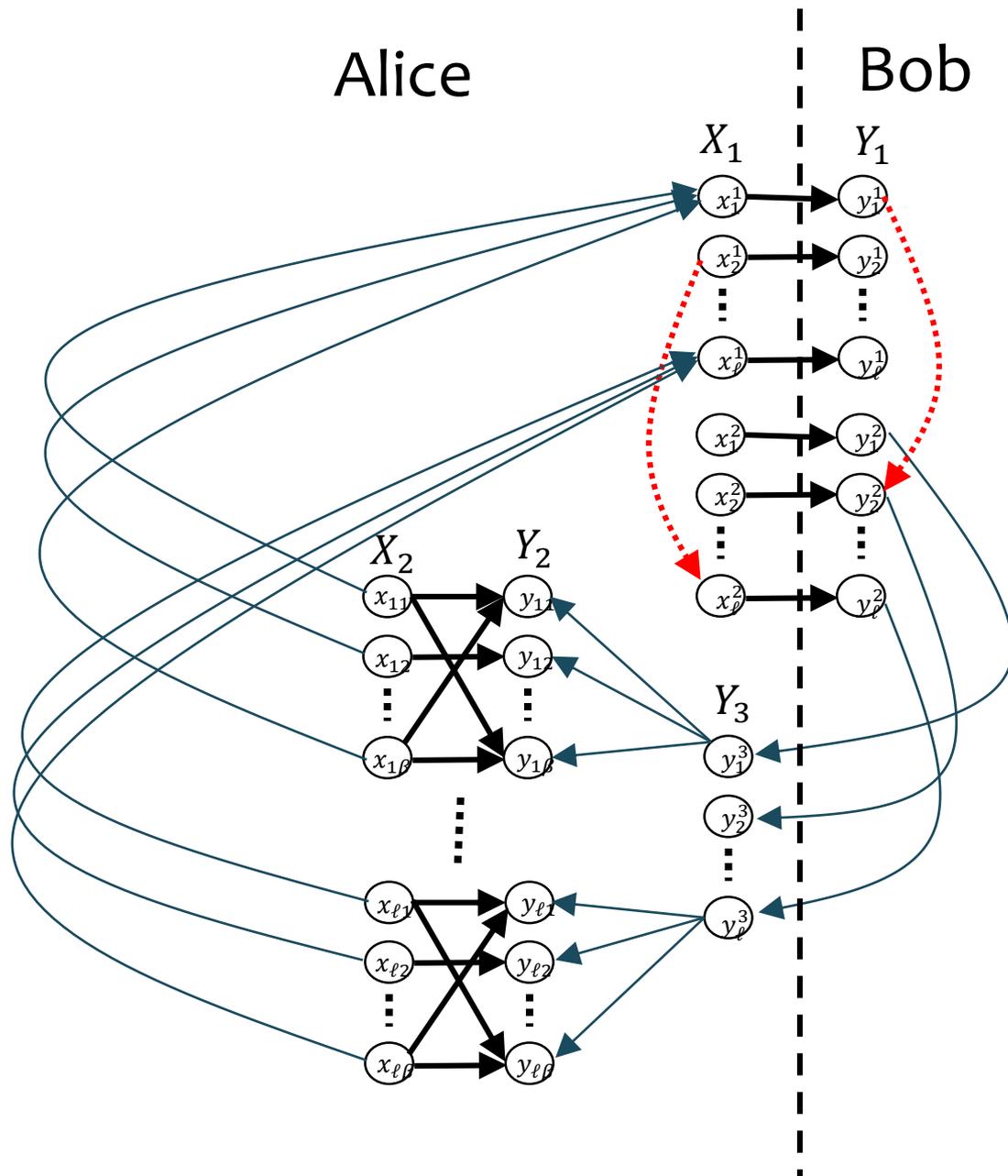
$$|CUT_{A-B}| = \theta(\ell)$$

↓

$$T(n) \cdot \log n \cdot \ell = \Omega(\ell^2)$$

↓

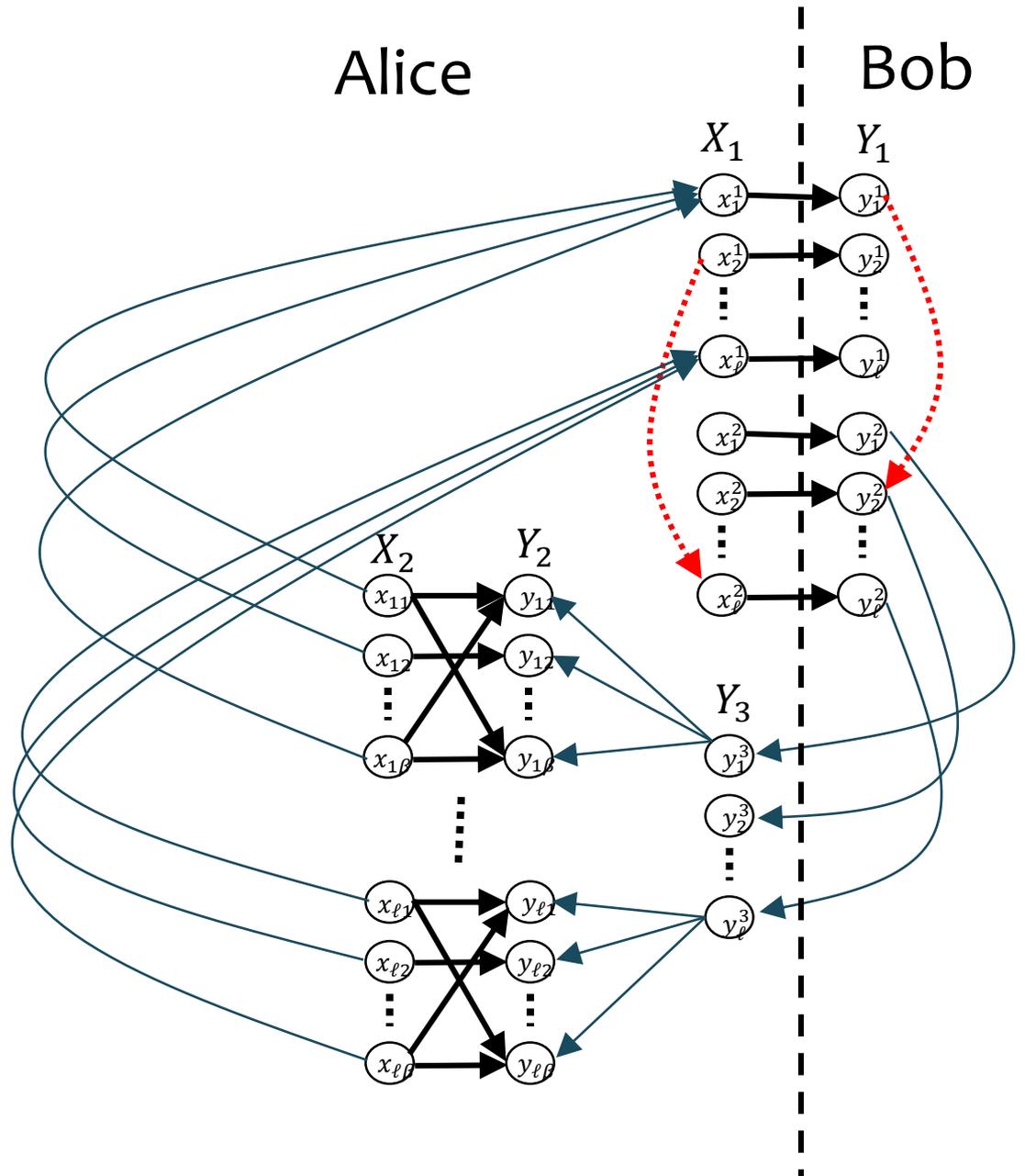
$$T(n) = \Omega\left(\frac{\ell}{\log n}\right)$$



$$T(n) = \Omega\left(\frac{\ell}{\log n}\right)$$

$$\beta \approx \sqrt{\alpha n}$$

$$\ell = \theta(n/\beta) = \theta(\sqrt{n/\alpha})$$



Conclusion

- $\tilde{\Omega}(\sqrt{n/\alpha})$ rounds are required for constructing an α -approximation for directed k -spanner (holds also for $k \geq 5$).

Other results:

- For **deterministic** algorithms, we can improve the lower bound to $\tilde{\Omega}(n/\sqrt{\alpha})$ using gap-disjointness.
- For **weighted** graphs, we can improve the lower bound to $\tilde{\Omega}(n)$ for directed graphs, and $\tilde{\Omega}(n/k)$ for undirected graphs.

Other variants

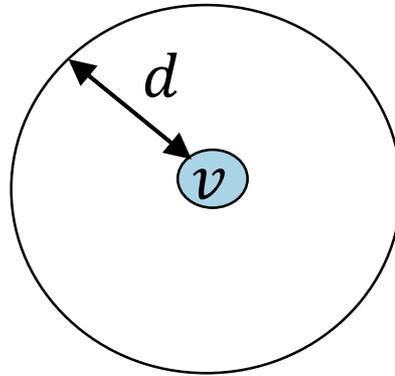
- We can use this construction to show hardness results for additional variants, such as the **client-server** variant.
- The main open question is the **undirected unweighted** case.

Algorithm in the LOCAL model

We can get $(1 + \epsilon)$ -approximation in $O(\text{poly}(\log n / \epsilon))$ rounds in the LOCAL model. [inspired by Ghaffari, Kuhn, and Maus, 2017]

A sequential algorithm

- v_1, v_2, \dots, v_n
- $B_d(v)$ = the ball of radius d around v
- $g(v, d)$ = the size of an **optimal k -spanner** for the uncovered edges in **$B_d(v)$**

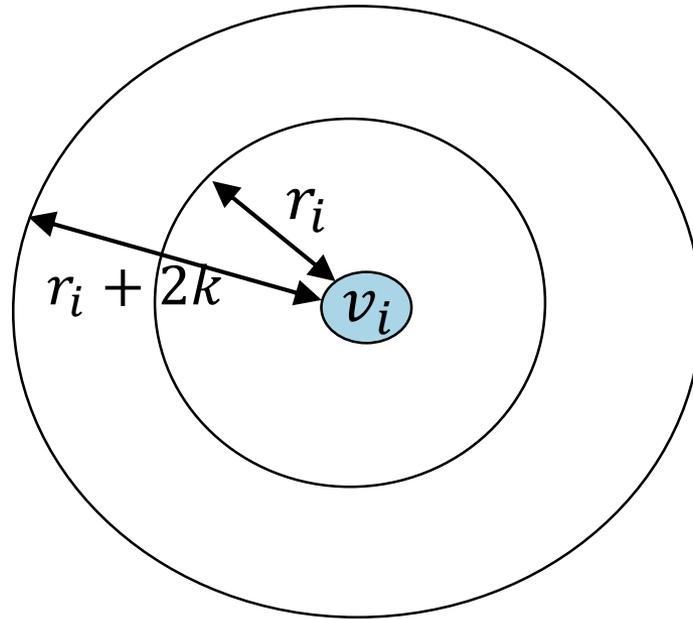


A sequential algorithm

$g(v, d)$ = the size of an **optimal k -spanner** for $B_d(v)$

In iteration i :

find r_i such that $g(v_i, r_i + 2k) \leq (1 + \epsilon)g(v_i, r_i)$



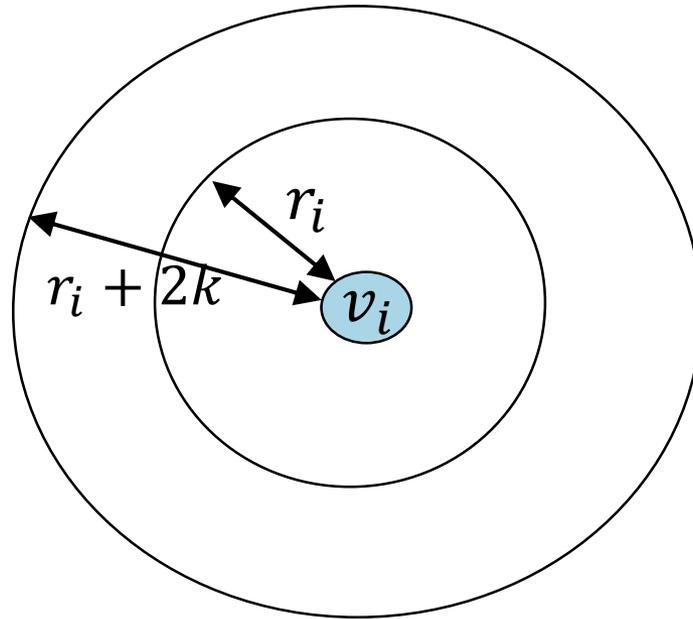
A sequential algorithm

$g(v, d)$ = the size of an **optimal k -spanner** for $B_d(v)$

In iteration i :

find r_i such that $g(v_i, r_i + 2k) \leq (1 + \epsilon)g(v_i, r_i)$

$$r_i = O(\log n / \epsilon)$$

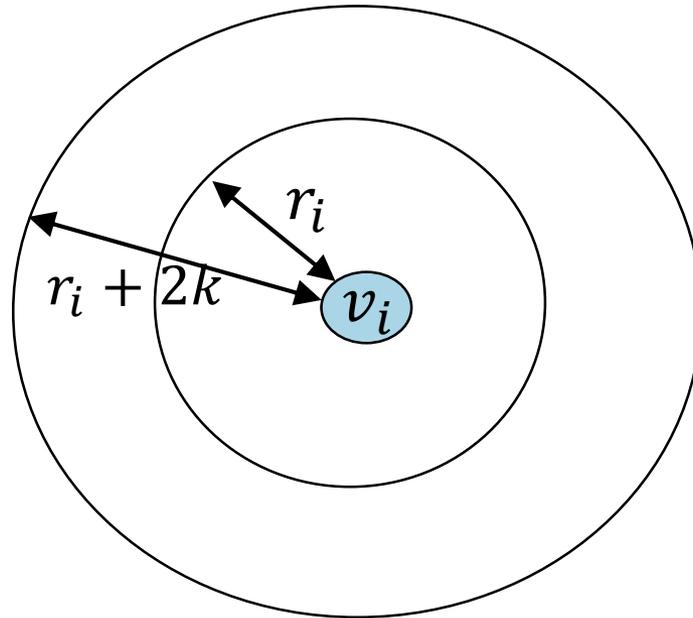


A sequential algorithm

In iteration i :

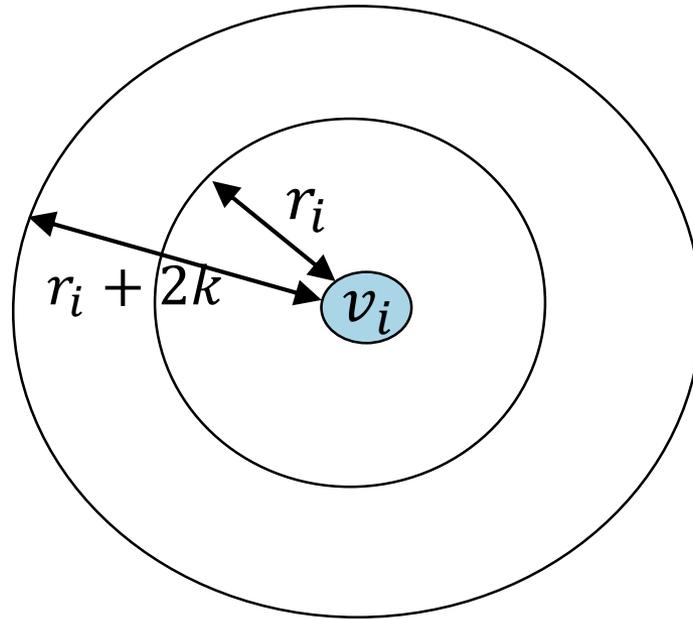
find r_i such that $g(v_i, r_i + 2k) \leq (1 + \epsilon)g(v_i, r_i)$

add an **optimal spanner** for $B_{r_i+2k}(v)$



Approximation ratio analysis

- H^* - an optimal spanner
- E_i = the uncovered edges in $B_{r_i}(v_i)$ before iteration i
- $H_i^* \subseteq H^*$ = minimum k -spanner for E_i



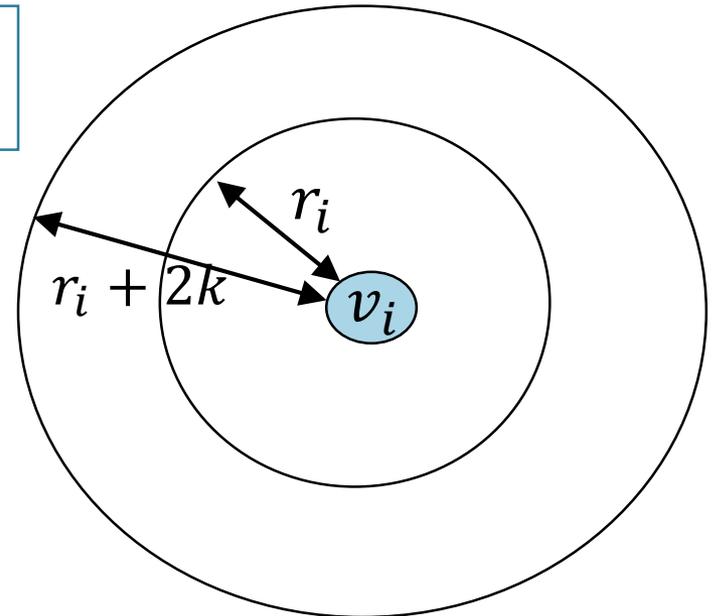
Approximation ratio analysis

- H^* - an optimal spanner
- E_i = the uncovered edges in $B_{r_i}(v_i)$ before iteration i
- $H_i^* \subseteq H^*$ = minimum k -spanner for E_i

E_i, E_j are at distance at least $2k + 1$



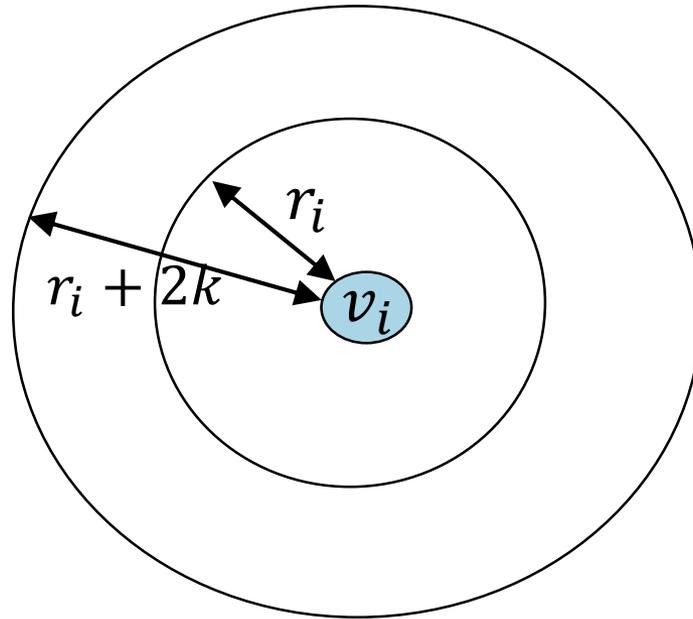
The subsets H_i^* are disjoint



Approximation ratio analysis

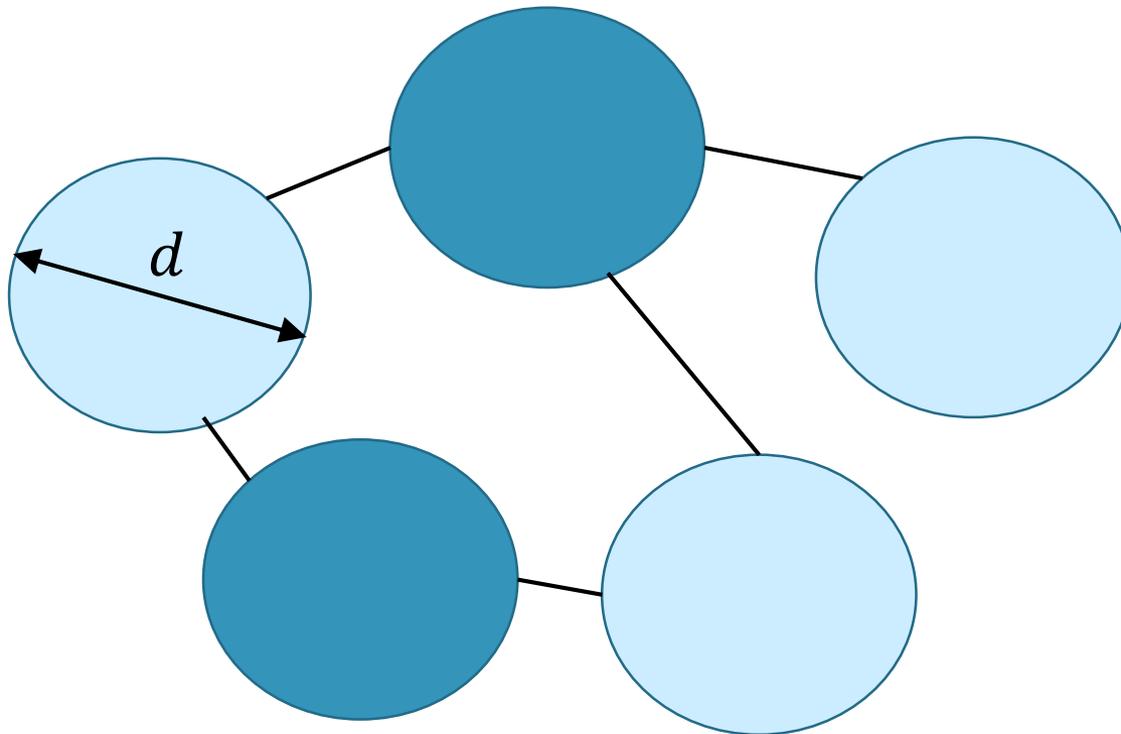
The number of edges in our spanner is at most

$$\sum_{i=1}^n (1 + \epsilon) |H_i^*| \leq (1 + \epsilon) |H^*|$$



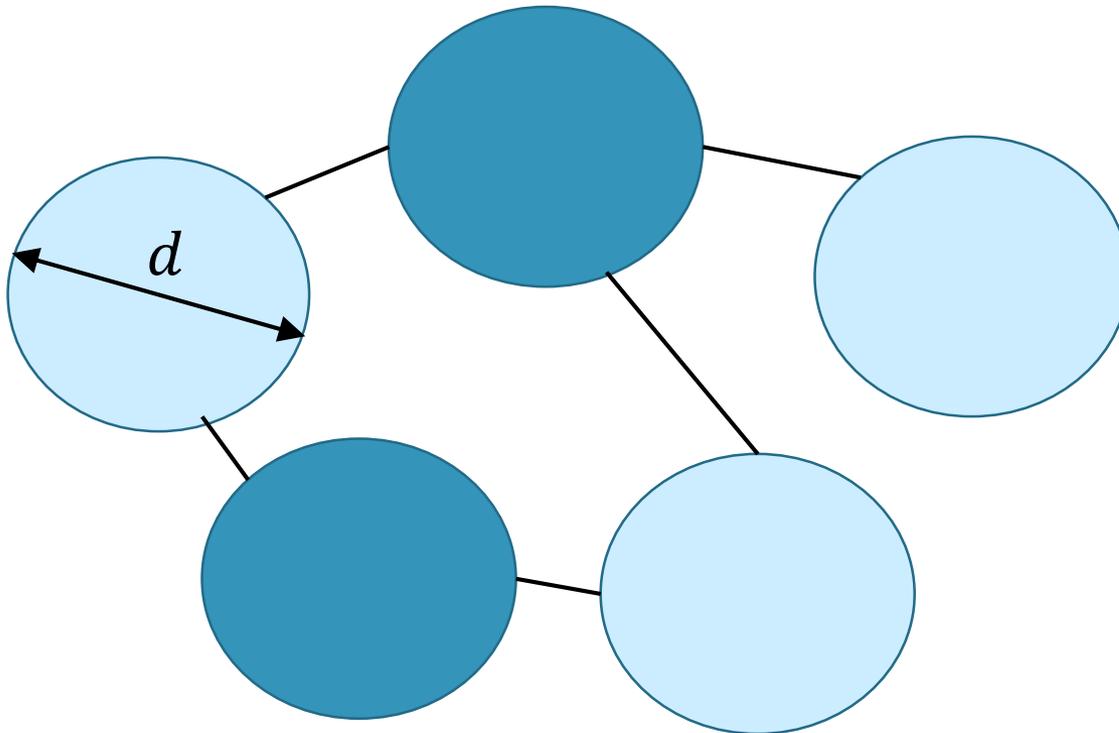
Distributed algorithm

We use (d, c) -**network decomposition**, with $d = c = O(\log n)$
[Linial and Saks, 1993]



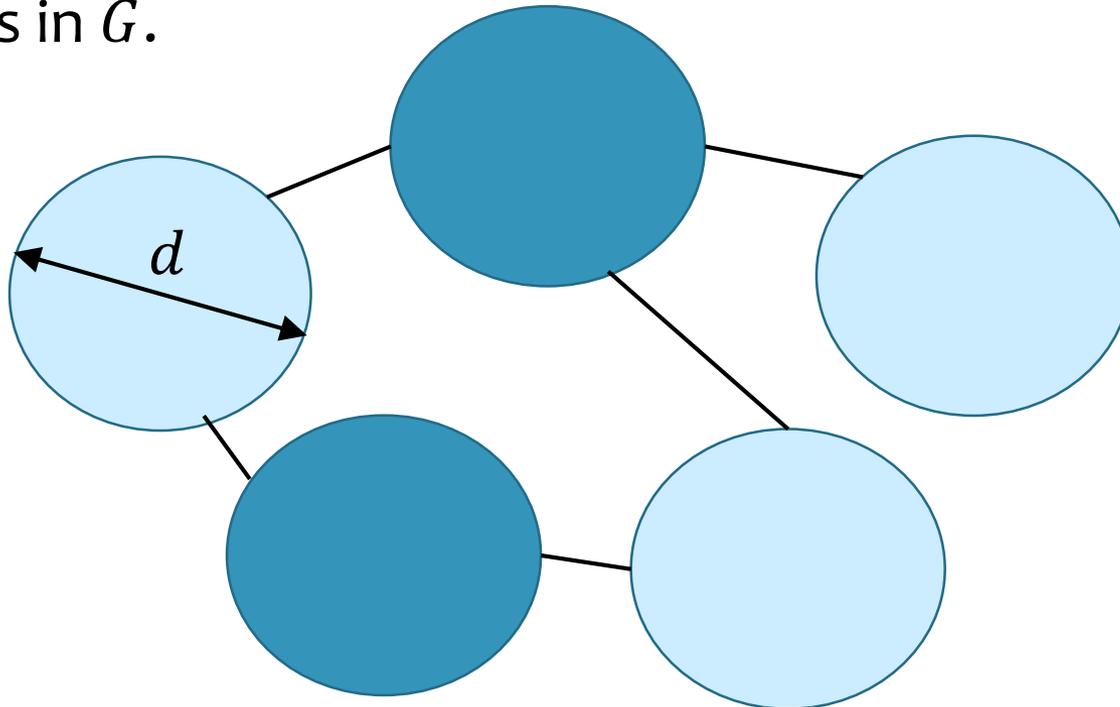
Distributed algorithm

- We choose $r = O(\log n / \epsilon)$ such that $r > r_i + 4k$ for all i
- We compute a **network decomposition of G^r** ($O(\text{poly}(\log n / \epsilon))$ rounds in the LOCAL model).



Distributed algorithm

- The label of a vertex v is $(col_v, id_v) \rightarrow$ **order** of the vertices
- We simulate the sequential algorithm according to **increasing order of the colors.**
- The computations depend only on the **r -neighborhood** of vertices in G .



Conclusion

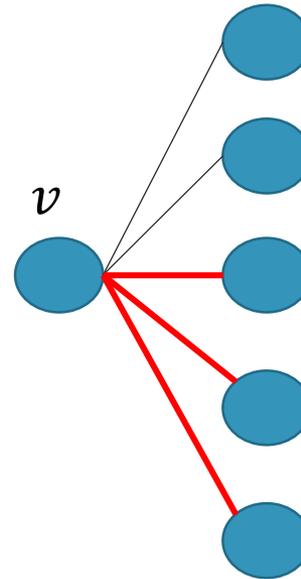
- We can get $(1 + \epsilon)$ -approximation in $O(\text{poly}(\log n / \epsilon))$ rounds in the LOCAL model.
- This algorithm is based on learning neighborhoods of polylogarithmic size and solving NP-complete problems.

2-spanners

- There is an $O(\log n)$ -round $O(\log n)$ -approximation in expectation using only **polynomial local computations** [Dinitz and Krauthgamer, 2011]
- Can we give an $O\left(\log \frac{|E|}{|V|}\right)$ -approximation?
- Can we **guarantee** the approximation ratio?
- Can we give an algorithm in the CONGEST model?
- What about lower bounds?

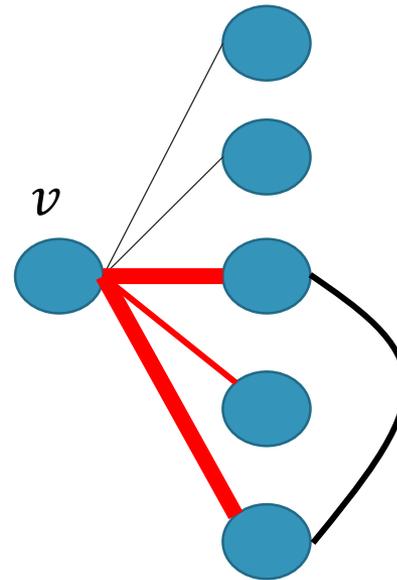
Stars and Densities

- A **star** around a vertex v , is a subset S of edges between v to some of its neighbors.
- The **density** of a star S is $\frac{C_S}{|S|}$ where C_S is the number of edges covered by the star S .



Stars and Densities

- A **star** around a vertex v , is a subset S of edges between v to some of its neighbors.
- The **density** of a star S is $\frac{C_S}{|S|}$ where C_S is the number of edges covered by the star S .



Sequential Greedy Algorithm

[Kortsarz and Peleg 1994]

- At each step, find the **densest star** in the graph, and add its edges to the spanner.
- Continue until all edges are covered.

- Achieves approximation ratio of $O\left(\log \frac{|E|}{|V|}\right)$

Distributed Algorithm – take 1

- At each step, find the **densest star** in the graph, and add its edges to the spanner.
- Continue until all edges are covered.

Distributed Algorithm – take 2

- At each step, find all the stars that are densest in their **local 2-neighborhood**, and add their edges to the spanner.
- Continue until all edges are covered.

Distributed Algorithm – take 3

- At each step, find all the stars that are densest in their local 2-neighborhood, they are the ***candidates***.
- Each candidate chooses a random number $r \in [0,1]$.
- Each uncovered edge **votes** to the first candidate that covers it.
- A star is added to the spanner if it gets **at least $\frac{1}{8}$ of the votes** of the edges it covers.

Distributed Algorithm – take 3

- At each step, find all the stars that are densest in their local 2-neighborhood, they are the ***candidates***.
- Each candidate chooses a random number $r \in [0,1]$.
- Each uncovered edge **votes** to the first candidate that covers it.
- A star is added to the spanner if it gets **at least $\frac{1}{8}$ of the votes** of the edges it covers.

Distributed Algorithm – take 3

- At each step, find all the stars that are densest in their local 2-neighborhood, they are the ***candidates***.
- Each candidate chooses a random number $r \in [0,1]$.
- Each uncovered edge **votes** to the first candidate that covers it.
- A star is added to the spanner if it gets **at least $\frac{1}{8}$ of the votes** of the edges it covers.

Distributed Algorithm – take 3

- At each step, find all the stars that are densest in their local 2-neighborhood, they are the ***candidates***.
- Each candidate chooses a random number $r \in [0,1]$.
- Each uncovered edge **votes** to the first candidate that covers it.
- A star is added to the spanner if it gets **at least $\frac{1}{8}$ of the votes** of the edges it covers.

Distributed Algorithm – take 3

- At each step, find all the stars that are densest in their local 2-neighborhood, they are the ***candidates***.
 - Each candidate chooses a random number $r \in [0,1]$.
 - Each uncovered edge **votes** to the first candidate that covers it.
- A star is added to the spanner if it gets **at least $\frac{1}{8}$ of the votes** of the edges it covers.

Approximation ratio

- For $e \in E$, we define $cost(e) = \frac{1}{\rho}$ if e is covered by a star it votes for and has density ρ , and $cost(e) = 0$ otherwise.
- We show:

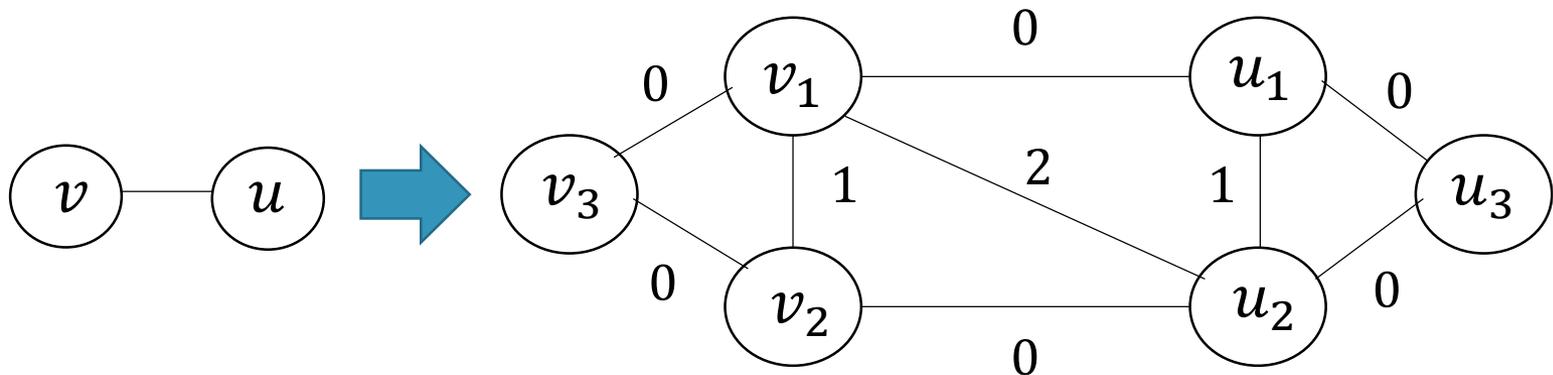
$$|H| \leq 8 \sum_{e \in E} cost(e) \leq O\left(\log \frac{|E|}{|V|}\right) |H^*|$$

Conclusion

- We can show that this approach **guarantees** an **approximation** of $O\left(\log \frac{|E|}{|V|}\right)$ in $O(\log n \log \Delta)$ rounds w.h.p.
- Extends also to the weighted, directed and client-server variants.
- We can also show that $\Omega\left(\frac{\log \Delta}{\log \log \Delta}\right)$ or $\Omega\left(\sqrt{\frac{\log n}{\log \log n}}\right)$ rounds are required for a logarithmic approximation for **weighted 2-spanner**. [reduction from [Kuhn, Moscibroda, and Wattenhofer, 2016](#)]

Lower bound for weighted 2-spanner

- We show a reduction from vertex cover



Open questions

Hardness of Approximation:

- Is it possible to show **separations** between the **LOCAL** and **CONGEST** models for other problems?
- **Undirected unweighted k -spanner**

2-spanner:

- Is it possible to show a **CONGEST** algorithm?