Running-Time Lower Bound 101

Bundit Laekhanukit
Institute for Theoretical Computer Science
Shanghai University of Finance and Economics

列卡努奇男，班迪（王班迪）
上海财经大学，理论计算机科学研究中心

(Based on Works that are not totally mine.)

This Talk

- The sole purpose of this talk is to advertise & recruit (prospective) PhD/Master students & Interns.

Rank institutions in the world ▾ by publications from 2008 ▾ to 2018 ▾

All Areas [off | on]

AI [off | on]

- ▶ Artificial intelligence ☐
- ▶ Computer vision ☐
- ▶ Machine learning & data mining ☐
- ▶ Natural language processing ☐
- ▶ The Web & information retrieval ☐

Systems [off | on]

- ▶ Computer architecture ☐
- ▶ Computer networks ☐
- ▶ Computer security ☐
- ▶ Databases ☐
- ▶ Design automation ☐
- ▶ Embedded & real-time systems ☐
- ▶ High-performance computing ☐
- ▶ Mobile computing ☐
- ▶ Measurement & perf. analysis ☐
- ▶ Operating systems ☐
- ▶ Programming languages ☐
- ▶ Software engineering ☐

Theory [off | on]

- ▶ Algorithms & complexity ☑
- ▶ Cryptography ☐
- ▶ Logic & verification ☐

Interdisciplinary Areas [off | on]

- ▶ Comp. bio & bioinformatics ☐
- ▶ Computer graphics ☐
- ▶ Economics & computation ☑

#	Institution	Count	Faculty
1	▶ Carnegie Mellon University	37.4	15
2	▶ Tel Aviv University	35.8	17
3	▶ Stanford University	28.2	18
4	▶ Cornell University	24.4	16
5	▶ Harvard University	21.6	10
6	▶ Technion	21.4	27
7	▶ Massachusetts Institute of Technology	21.2	22
8	▶ University of Pennsylvania	16.6	8
9	▶ Hebrew University of Jerusalem	15.9	17
10	▶ University of Southern California	15.5	8
11	▶ University of Michigan	15.1	11
12	▶ Duke University	14.6	5
13	▼ Shanghai Univ. of Finance and Economics	12.2	5

Faculty	# Pubs	Adj. #
Pinyan Lu THEORY,ECOM	31	11.7
Bundit Laekhanukit THEORY	14	5.1
Nick Gravin THEORY	13	4.1
Tsz Chiu Kwok THEORY	5	1.4
Zihe Wang AI,ECOM	4	1.7

#	Institution	Count	Faculty
14	▶ Columbia University	12.1	9
15	▶ Princeton University	11.6	11
16	▶ Rutgers University	11.4	10
17	▶ University of Washington	10.7	8

The Real Talk

- Address the question of why we cannot break running-time barriers of some classical problems.

Textbook Problems

that we have already hit the barriers

- SAT, k-SUM, k- Clique, Closest-Pair,
- Longest-Common Subsequence,
- All-Pair-Shortest-Paths,
- Maximum-Flow

Classical Techniques

difficult and have a lot of limitations

- Decision Tree Model

Sorting, 2D Closest-Pair, Shortest-Path

limited to comparison based machines

- Circuit Models

k-Cliques, k-Set cover, k-Dominating Set

limited to some class of circuits

- Adversarial Models

Data Structures, Online Algorithms

limited to some classes of algorithms

Modern Techniques

give up improving the known algorithms

- Assume that the known algorithm are the best possible ones!
- Deduce running-time lower-bounds from these given up problems!

Popular Hypotheses

sources of hardness for others problems

- **ETH:** SAT has no $2^{o(n)}$-time algorithm.
- **SETH:** SAT has no $2^{(1-\varepsilon)n}$-time algorithm.
- **k-SUM:** Find k integers from $S \subseteq [-n^2, n^2]$ that sum to zero requires $n^{k/2-\varepsilon}$-time.
- **FPT ≠ W[1]:** k-Clique has no T(k) poly(n)-time algorithm.
- **ATSP:** All-Pair Shortest-Path has no $n^{(3-\varepsilon)}$-time algorithm.

Warming-Ups

lower-bounds based on SAT (SETH)

- Reducing NP-Complete problems to P!

SAT → k-Set-Cover & (Bichromatic) Closest Pairs.

Warming-Ups 1

- SAT → k-Set Cover

Satisfiability Problem (SAT)

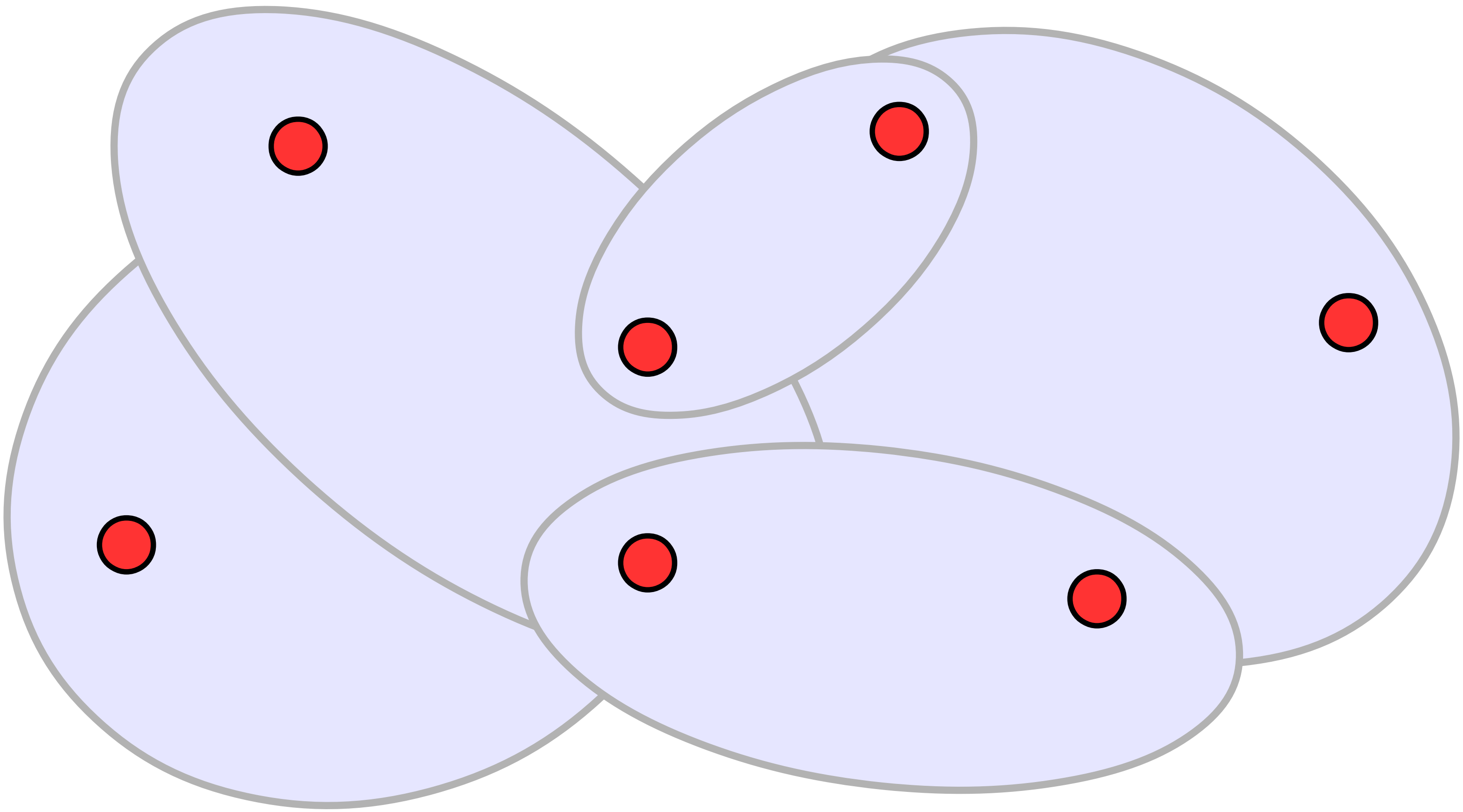
Given a CNF formula I (formula: (... or ...) and (... or ...) and (... or ...) .)

e.g., (x_1 or (**not** x_2) or x_3) and (x_2 or x_3 or (**not** x_4))

Goal: decide whether I is satisfiable.

k-Set Cover (k-SCP)

Given a collection of sets $S_1, \ldots, S_n$ and elements $e_1, \ldots, e_m$
Decide whether there are k sets that covers all the elements.

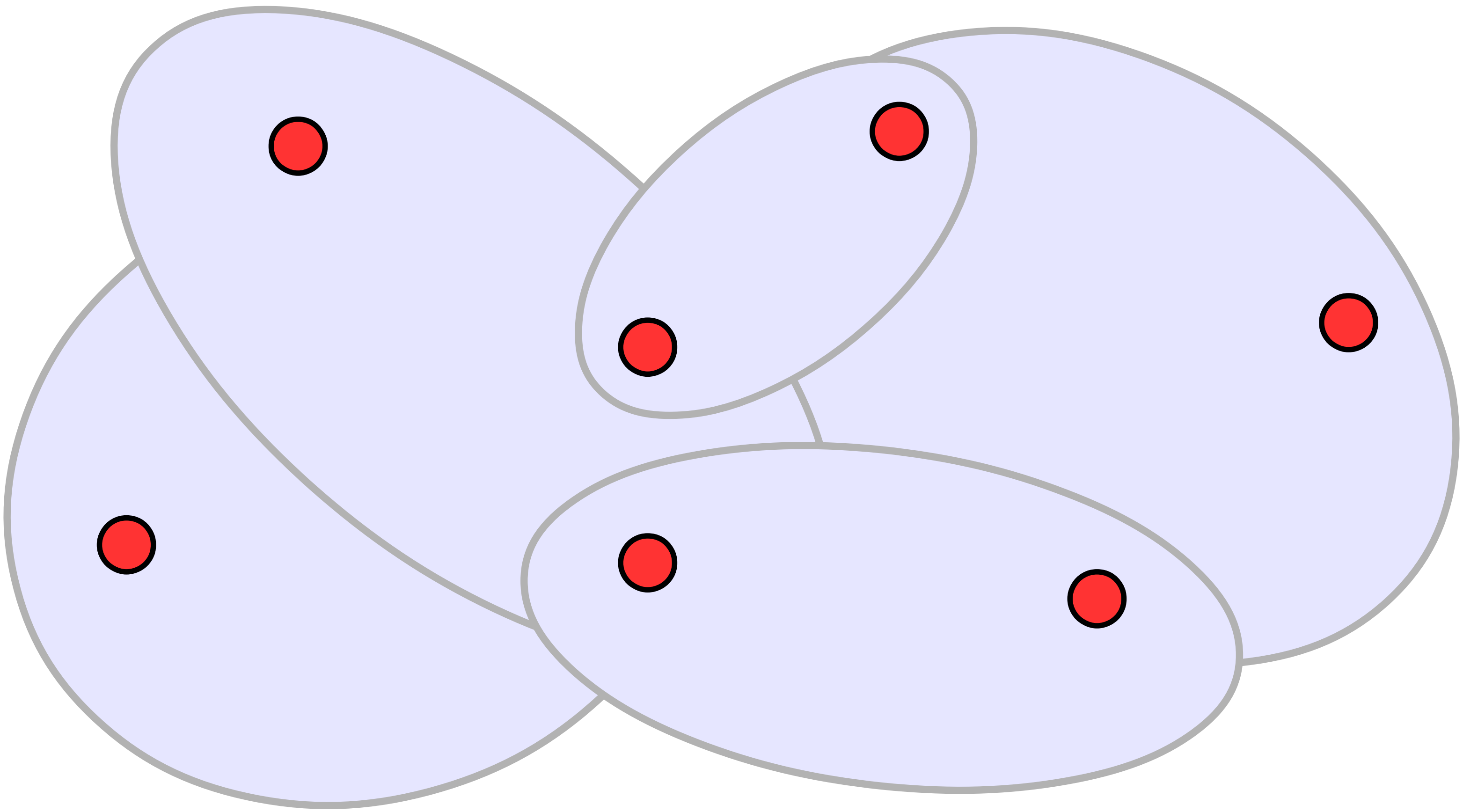


SAT → k-Set Cover

We want to show k-SCP cannot be solved in $n^{(0.99)k}$ time unless SAT can be solved in $2^{0.99n}$ time.

SAT → (n)-Set Cover

List all possible settings of each variables as sets.
List all clauses as elements.



SAT → (n)-Set Cover

List all possible settings of each variables as sets.
List all clauses as elements.

SAT → k-Set Cover

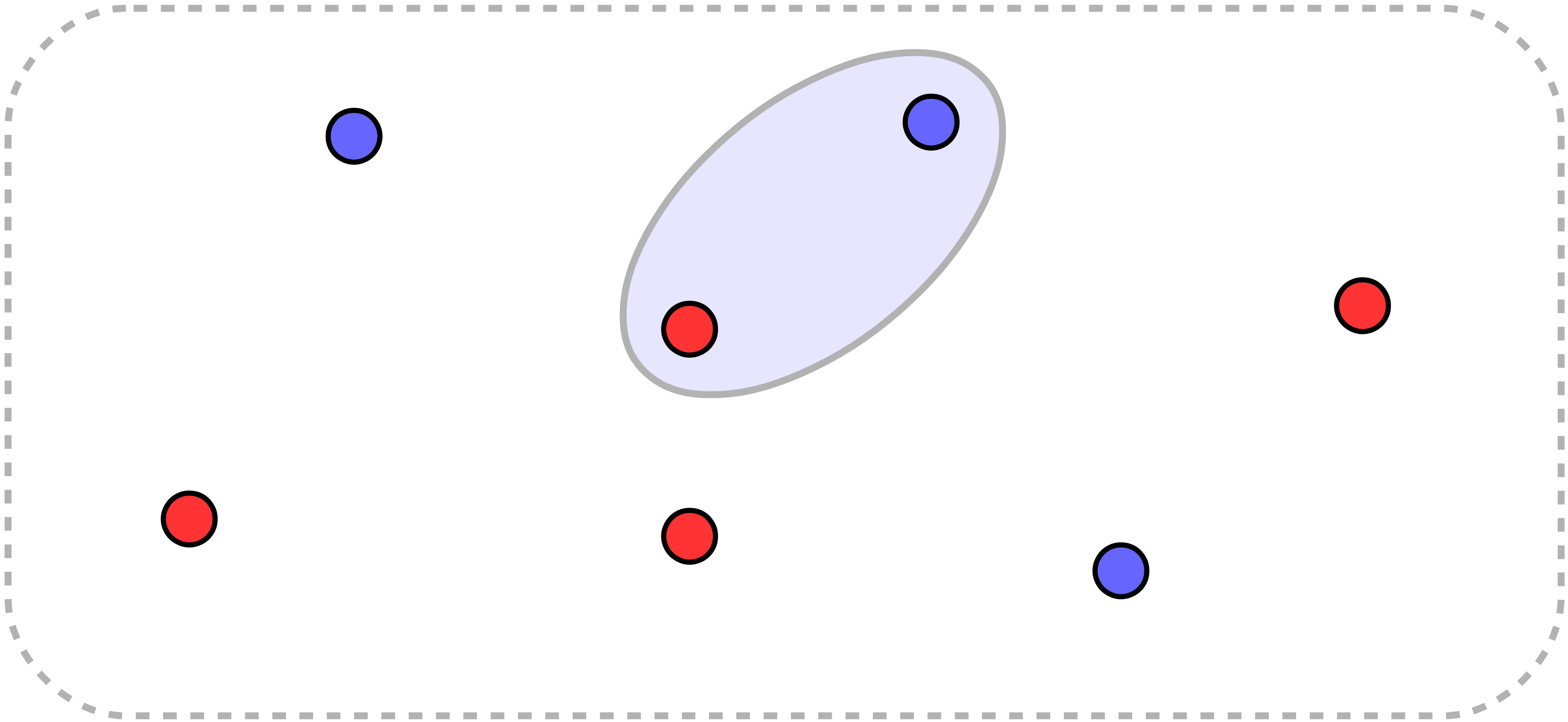
We want to show k-SCP cannot be solved in $n^{(0.99)k}$ time
unless SAT can be solved in $2^{0.99n}$ time.

Warming-Ups 2

- SAT → Bi-Chromatic Closest Pair

Bi-Chromatic Closest Pair (BCP)

Given a collection of n red and n blue points in a d-dimensional metric, find a pair of red-blue points with minimum-distance.



Satisfiability Problem (SAT)

Given a CNF formula I (formula: (... or ...) and (... or ...) and (... or ...) .)

e.g., (x_1 or (**not** x_2) or x_3) and (x_2 or x_3 or (**not** x_4))

Goal: decide whether I is satisfiable.

SAT → BCP

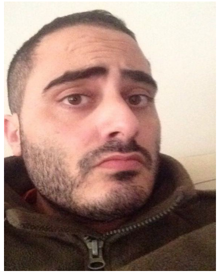
Goal: prove n^2 time lower bound for BCP.
Hint: Similar to SAT → 2-SCP

Second Part of the Talk

- Complexity of Closest Pair
- Geometric Representation of Graphs

Complexity of Closest Pair via Polar Pair of Point-Sets

Joint work with

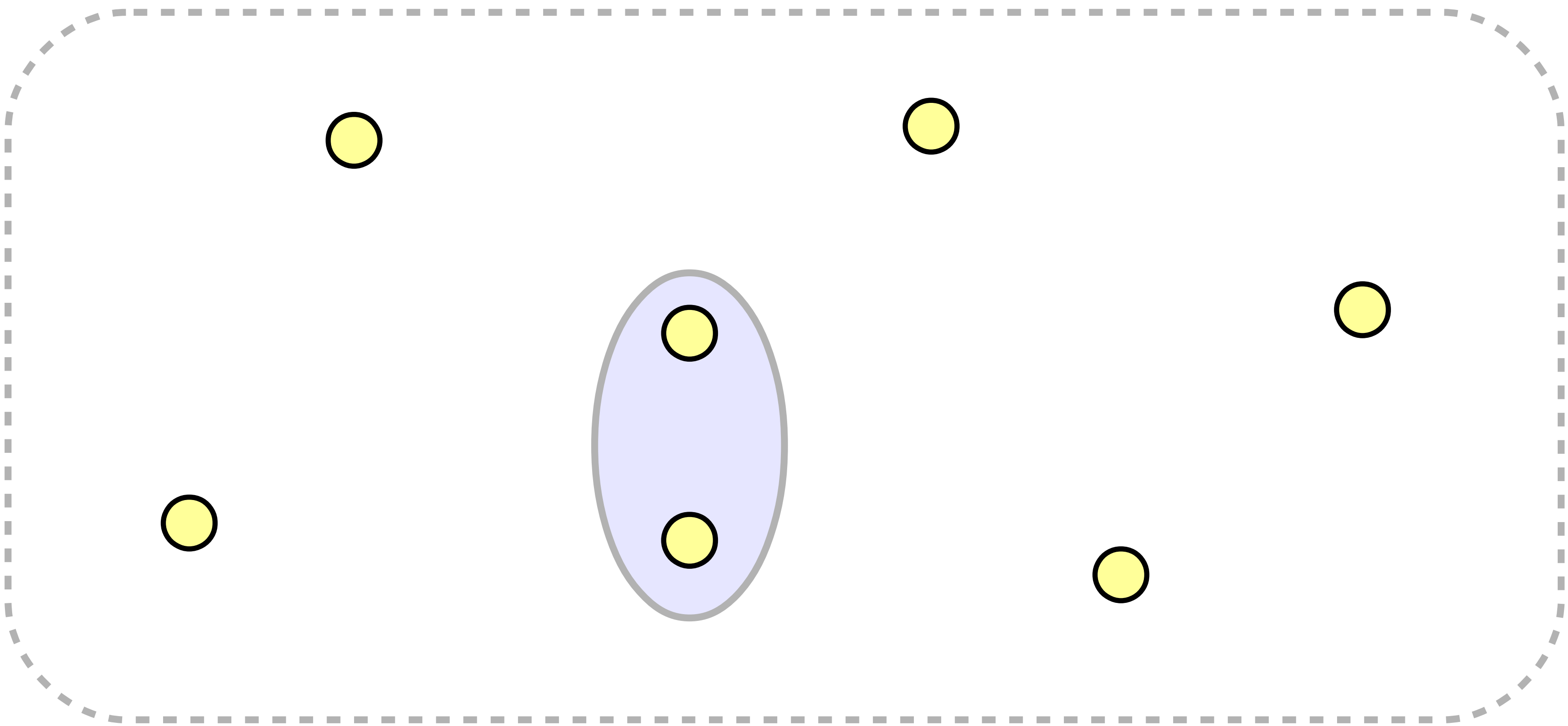


Karthik C. S.
Weizmann Institute

Roee David
Datorama

Closest Pair (CP)

Given a collection of n points in a d-dimensional metric,
find a pair of points with minimum-distance.



Known Algorithms

- Euclidean Closest Pair
  - Dimension $d=O(1)$:
    - $O(2^D n \log n)$ (deterministic) [Bently-Shamos'76]
    - $O(2^D n)$ (randomzied) [Rabin'76, Khuller-Mattias'95]
  - Dimension $d=\Theta(\log n)$
    - $O(d n^2)$ (trivial algorithm)
  - Dimension $d=n$: $O(n^{3-\varepsilon})$, for some $\varepsilon > 0$

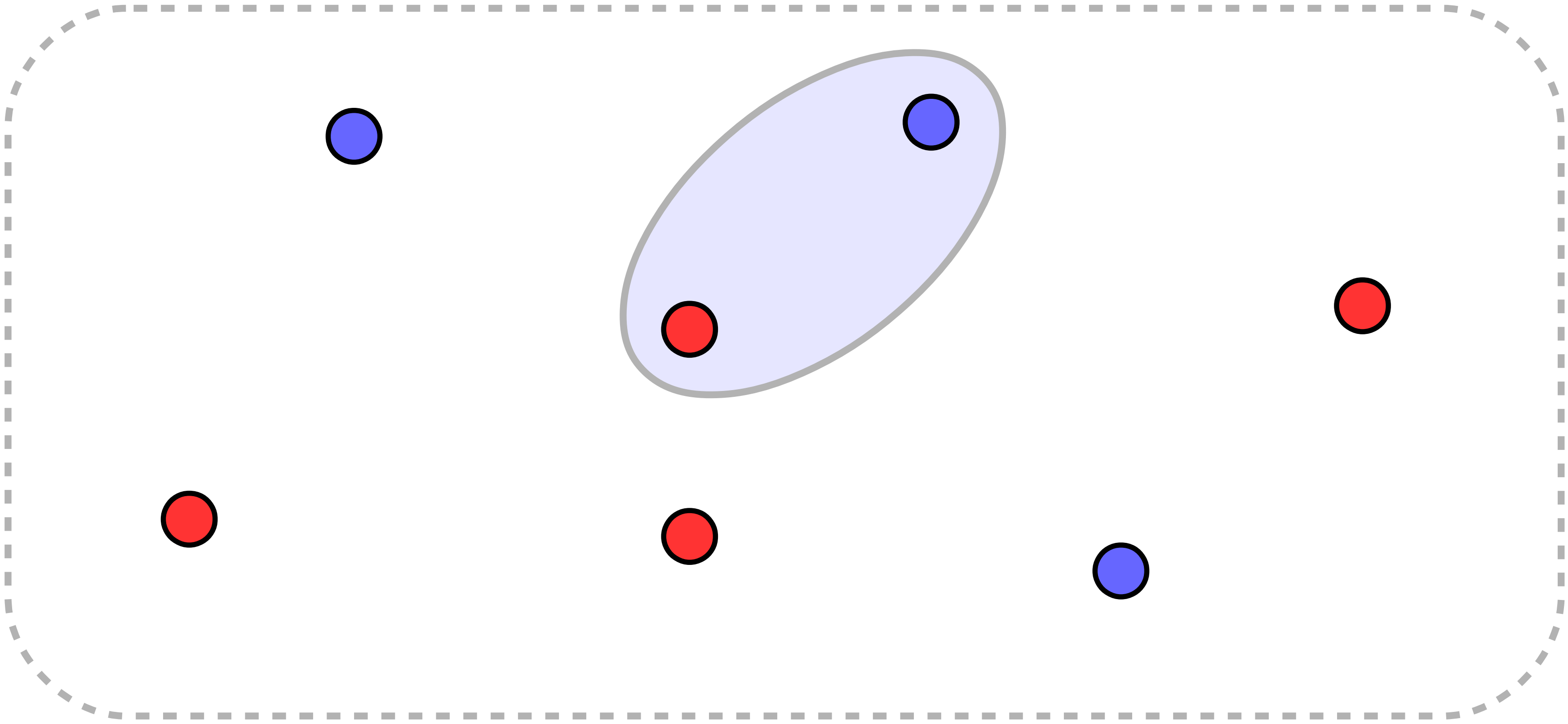
Is there an $O(n^{1.9})$-time algorithm when dimension $d=\Omega(\log n)$?

Is there an $O(n^{1.9})$-time algorithm when dimension $d=\Omega(\log n)$?

Don't know for Euclidean Closest Pair.
No for the bichromatic variant.

Bi-Chromatic Closest Pair (BCP)

Given a collection of n red and n blue points in a d-dimensional metric, find a pair of red-blue points with minimum-distance.



Bi-Chromatic Closest Pair
has no $O(n^{1.9})$-time algorithm
for any L^P-metric
unless SAT has $(2^{1.9n})$-time algorithm

Alman-Williams 2015

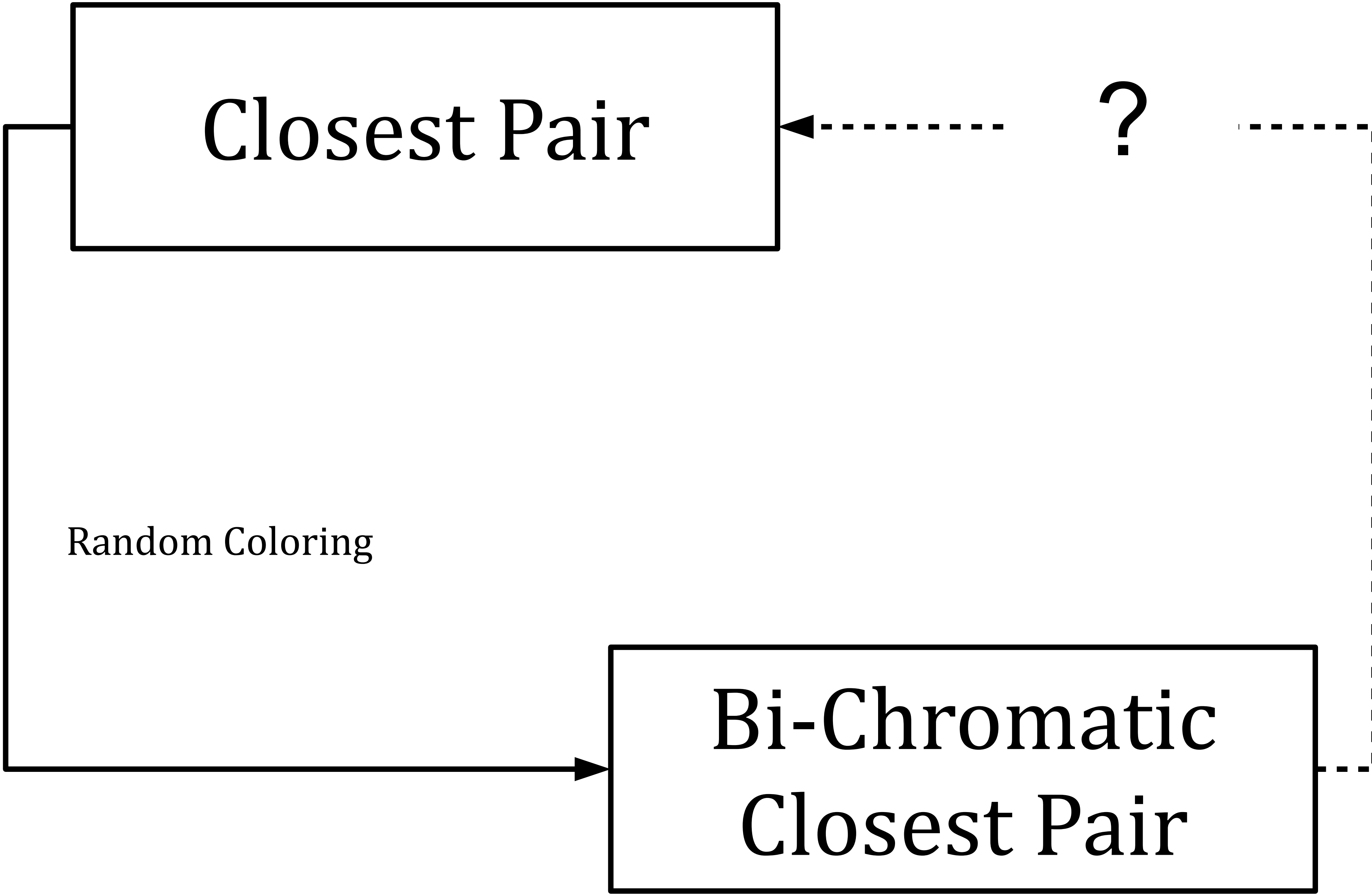
Bi-Chromatic Closest Pair has no $O(n^{1.9})$-time algorithm for any L^P-metric

unless SAT has $(2^{1.9n})$-time algorithm

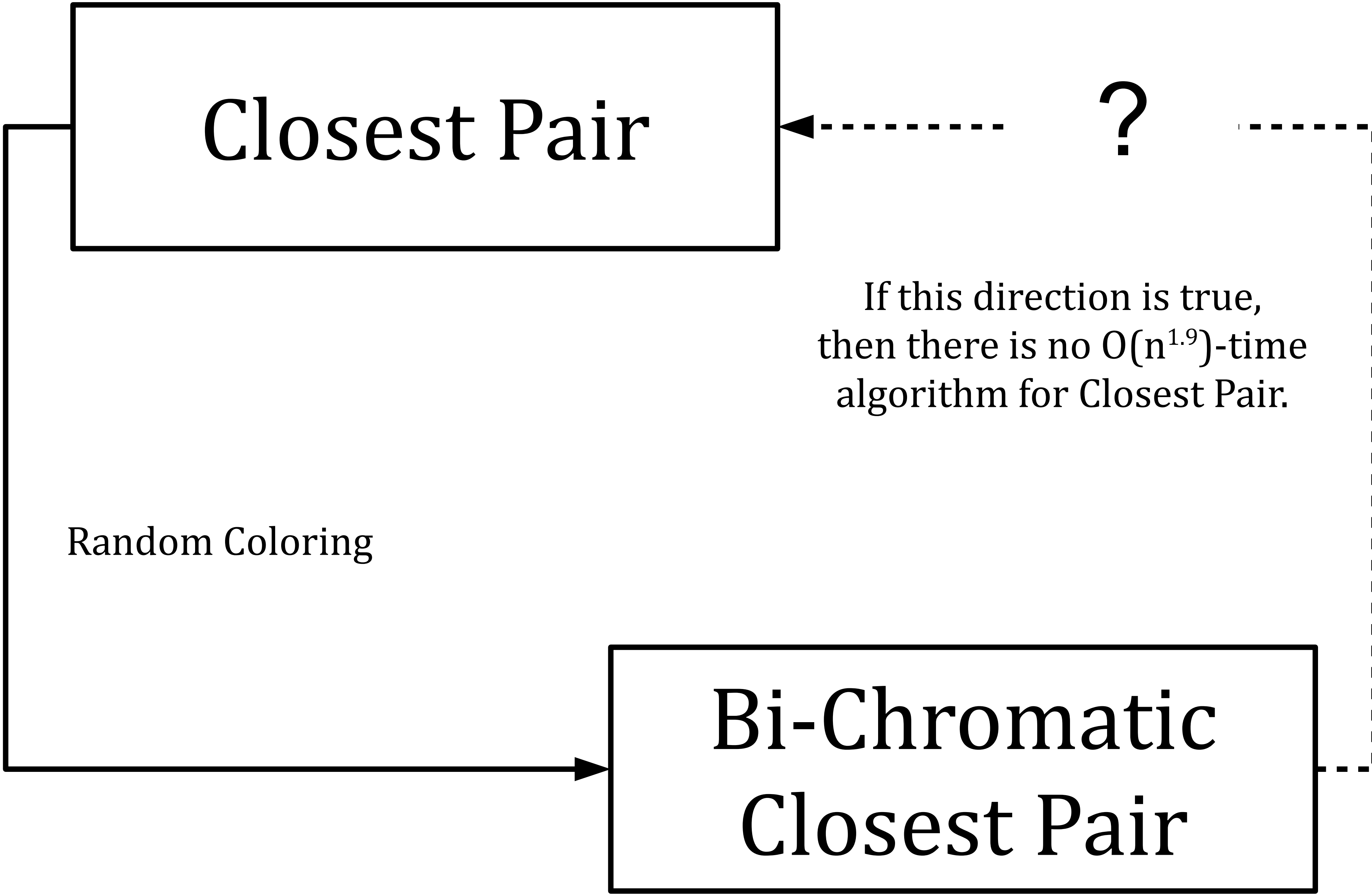
Alman-Williams 2015

Even $(1+o(1))$-approx in $O(n^{1.9})$-time is not possible.

Rubinstein 2018

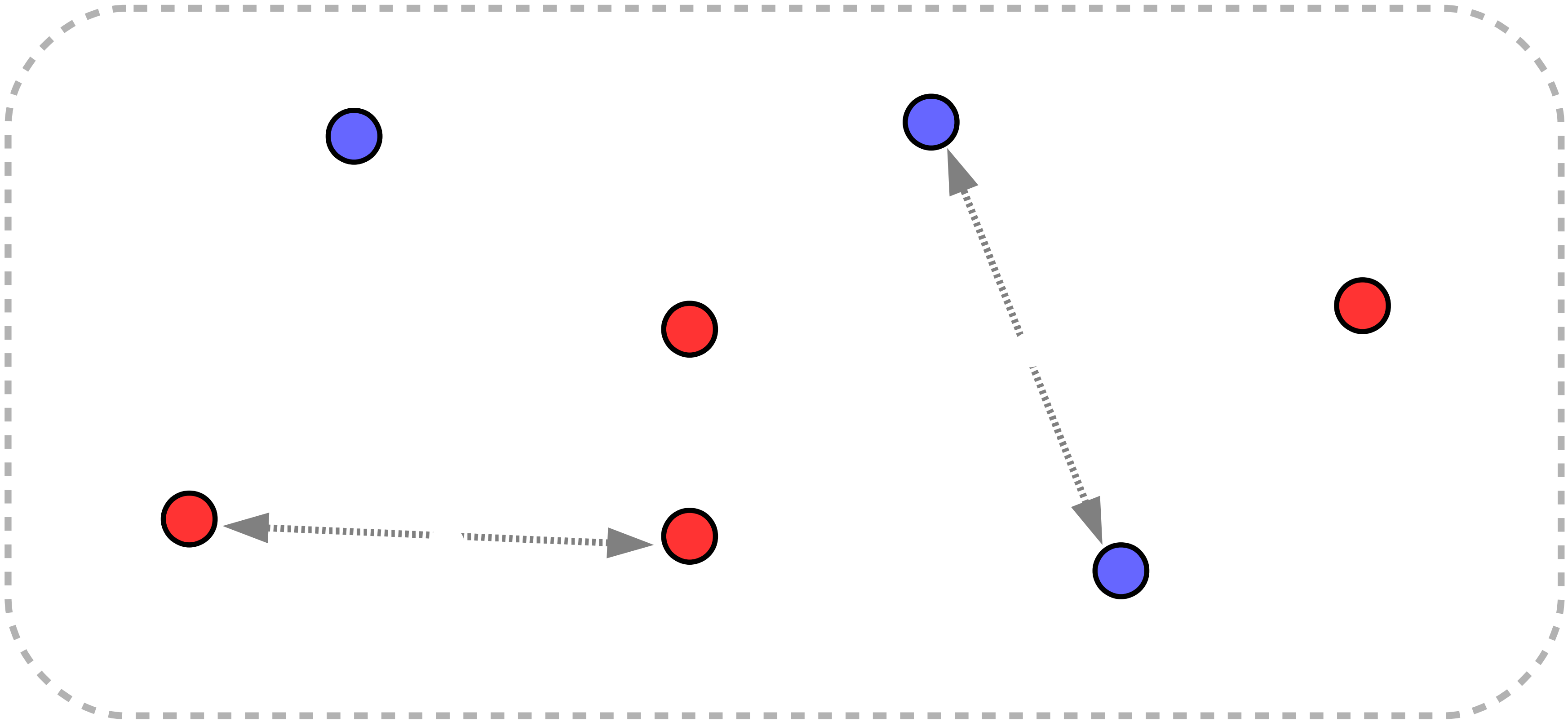


Closest Pair
?
Random Coloring
Bi-Chromatic
Closest Pair


Closest Pair
?
If this direction is true, then there is no $O(n^{1.9})$-time algorithm for Closest Pair.
Random Coloring
Bi-Chromatic Closest Pair

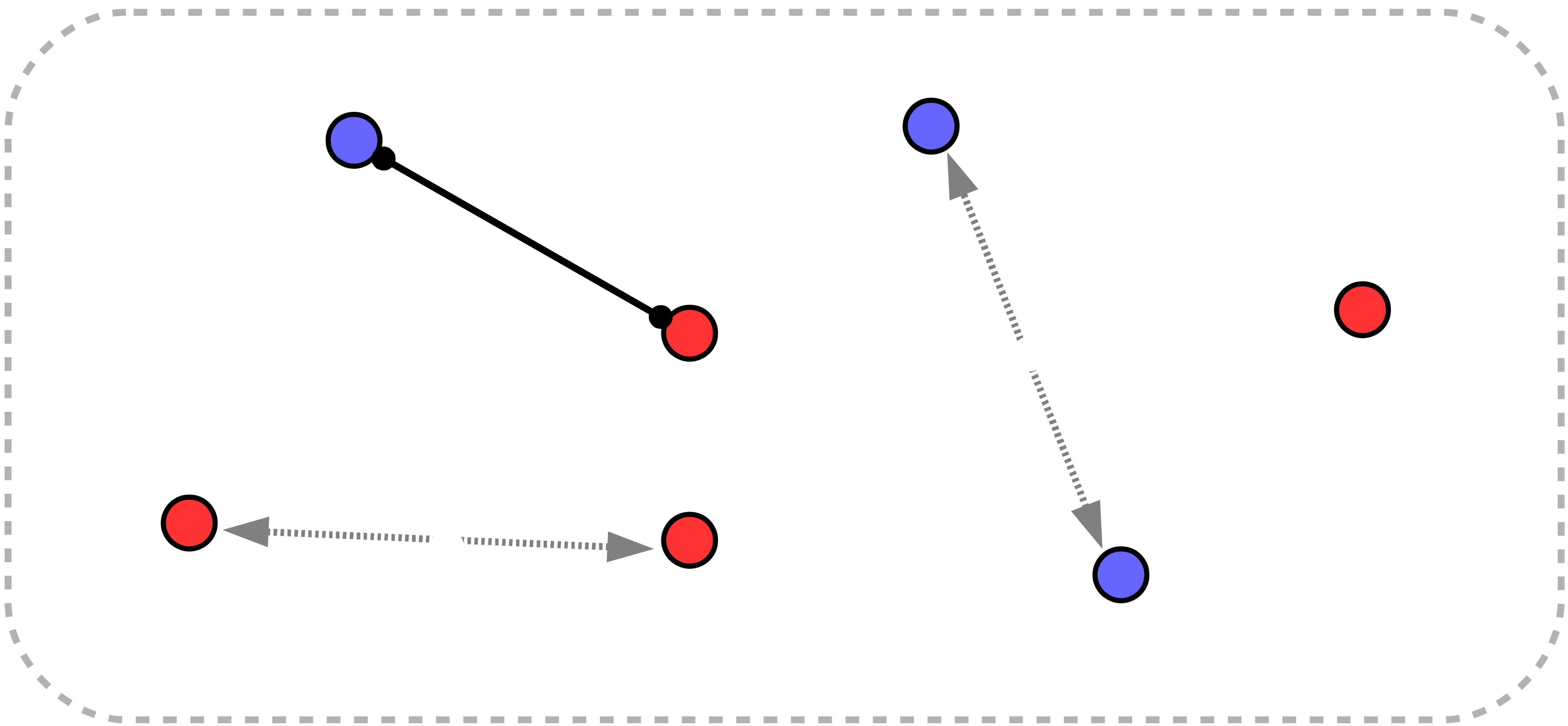
Reduction BCP → CP

Enlarge Red-Red, Blue-Blue distances.



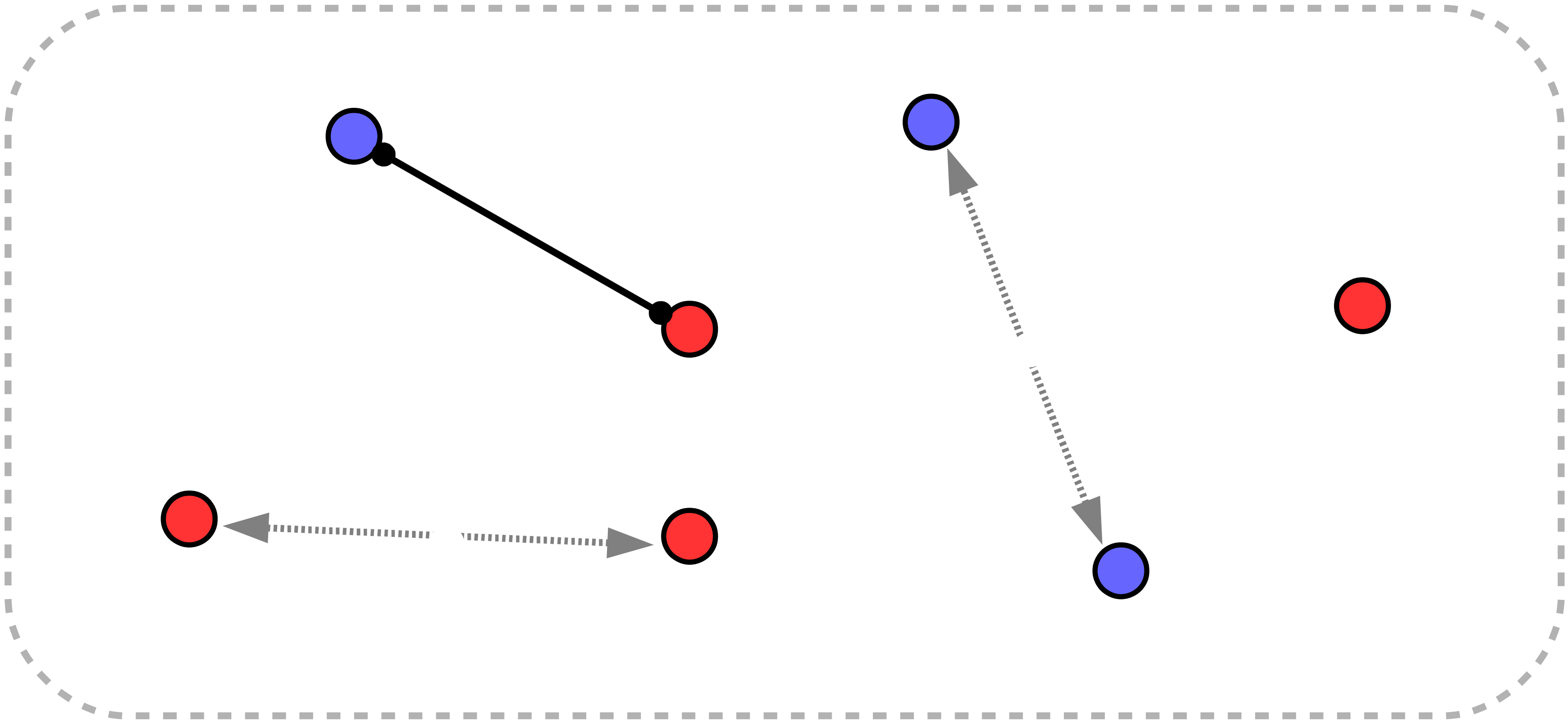
Reduction BCP → CP

Enlarge Red-Red, Blue-Blue distances.



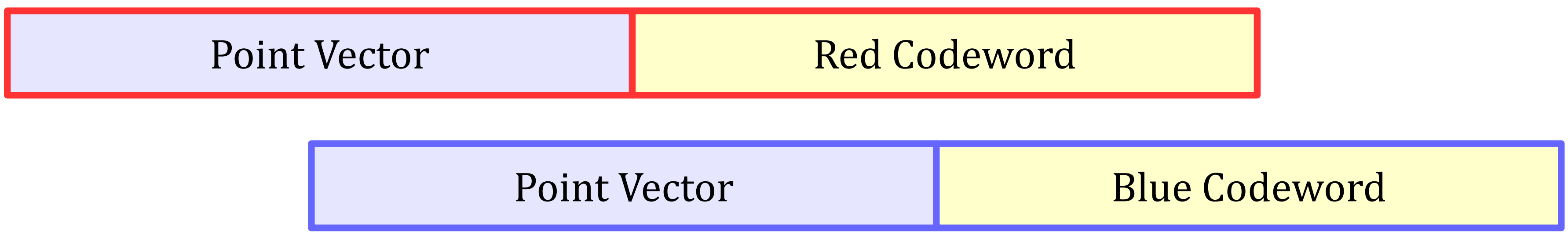
Reduction BCP → CP

Concatenate point-vectors with codewords



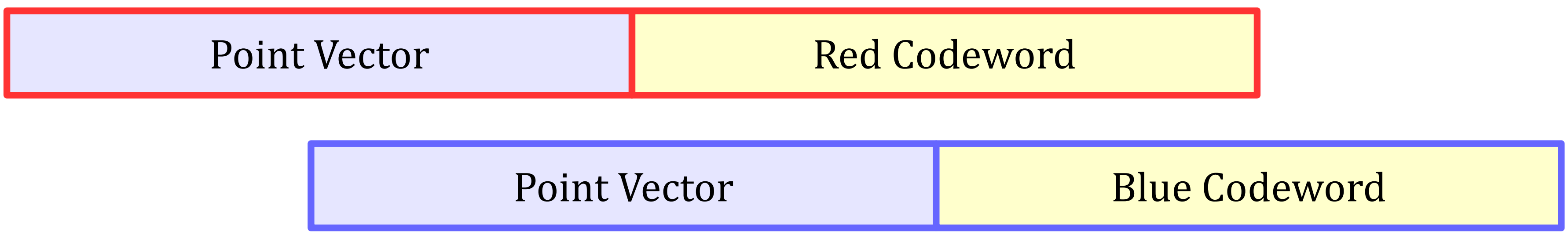
Reduction BCP → CP

Concatenate point-vectors with codewords



Reduction BCP → CP

Concatenate point-vectors with codewords



Needed Properties of The Codewords
(Bi-Clique Property)

Distance(Red-Code, Red-Code') ≥ R + 1/n
Distance(Blue-Code,Blue-Code') ≥ R + 1/n
Distance(Red-Code, Blue-Code) = R

The existence of Codewords with Bi-Clique Property implies BCP → CP

(that runs in $O(n^{1.9})$-time)

The existence of Codewords with Bi-Clique Property implies BCP → CP

(that runs in $O(n^{1.9})$-time)

Proof

Scale each point so that maximum distance $\leq 1/n$.

Concatenate each red/blue point with red/blue codewords.

For any (new) red-points $red = r_1 \cdot rc_1$ and $red' = r_2 \cdot rc_2$

$$dist(red, red') = (dist(r_1, r_2)^2 + dist(rc_1, rc_2)^2)^{1/2} > R + 1 / n$$

$$(\text{since } dist(\text{red-code}, \text{red-code}') > R + 1/n)$$

For any (new) red & blue point $red = r \cdot rc$ and $blue = b \cdot bc$

$$dist(red, blue)^2 = (dist(r, b)^2 + dist(rc, rb)^2)^{1/2}$$

$$= (dist(r, b)^2 + R^2)^{1/2} < (1/n + R^2)^{1/2} < dist(red, red')$$

⟹ Any red-blue closest-pair is the closest-pair in the new instance.

■

The existence of Codewords with Bi-Clique Property implies BCP → CP

(that runs in $O(n^{1.9})$-time)

Proof

Scale each point so that maximum distance ≤ 1/n.

Concatenate each red/blue point with red/blue codewords.

For any (new) red-points red = $r_1 \cdot rc_1$ and red' = $r_2 \cdot rc_2$

$$\text{dist}(\text{red}, \text{red}') = (\ \text{dist}(r_1, r_2)^2 + \text{dist}(rc_1, rc_2)^2\ )^{1/2} > R + 1/n$$

(since dist(red-code, red-code') > R + 1/n)

For any (new) red & blue point red = $r \cdot rc$ and blue = $b \cdot bc$

$$\text{dist}(\text{red}, \text{blue})^2 = (\ \text{dist}(r, b)^2 + \text{dist}(rc, rb)^2\ )^{1/2}$$

$$= (\text{dist}(r, b)^2 + R^2)^{1/2} < (1/n + R^2)^{1/2} < \text{dist}(\text{red}, \text{red}')$$

⟹ Any red-blue closest-pair is the closest-pair in the new instance.

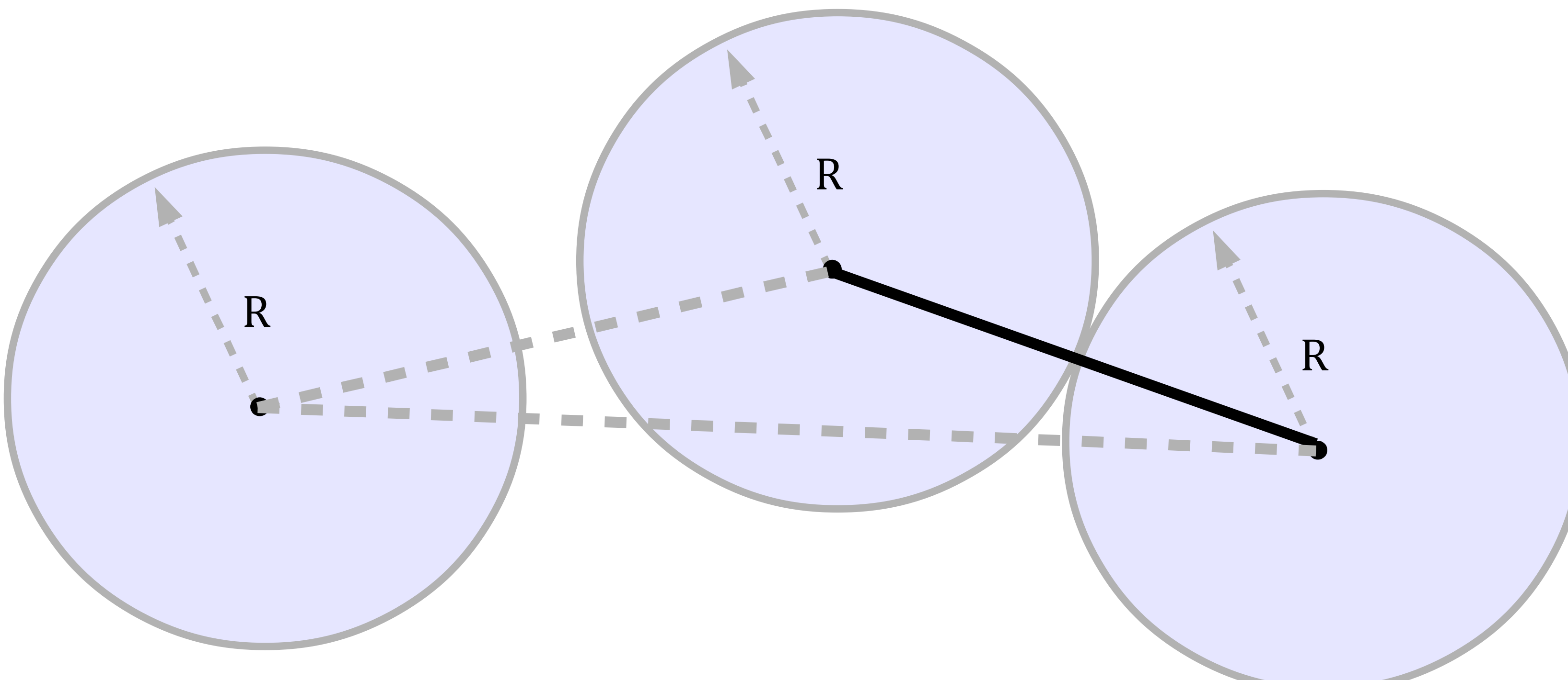
■

The existence of Codewords with Bi-Clique Property implies BCP → CP

(that runs in $O(n^{1.9})$-time)

Requires codewords construction with running time $O(n^{1.9})$ & dimension $O(\log n)$

Complexity Question of CP reduces to Geometric Representation of Bi-Clique

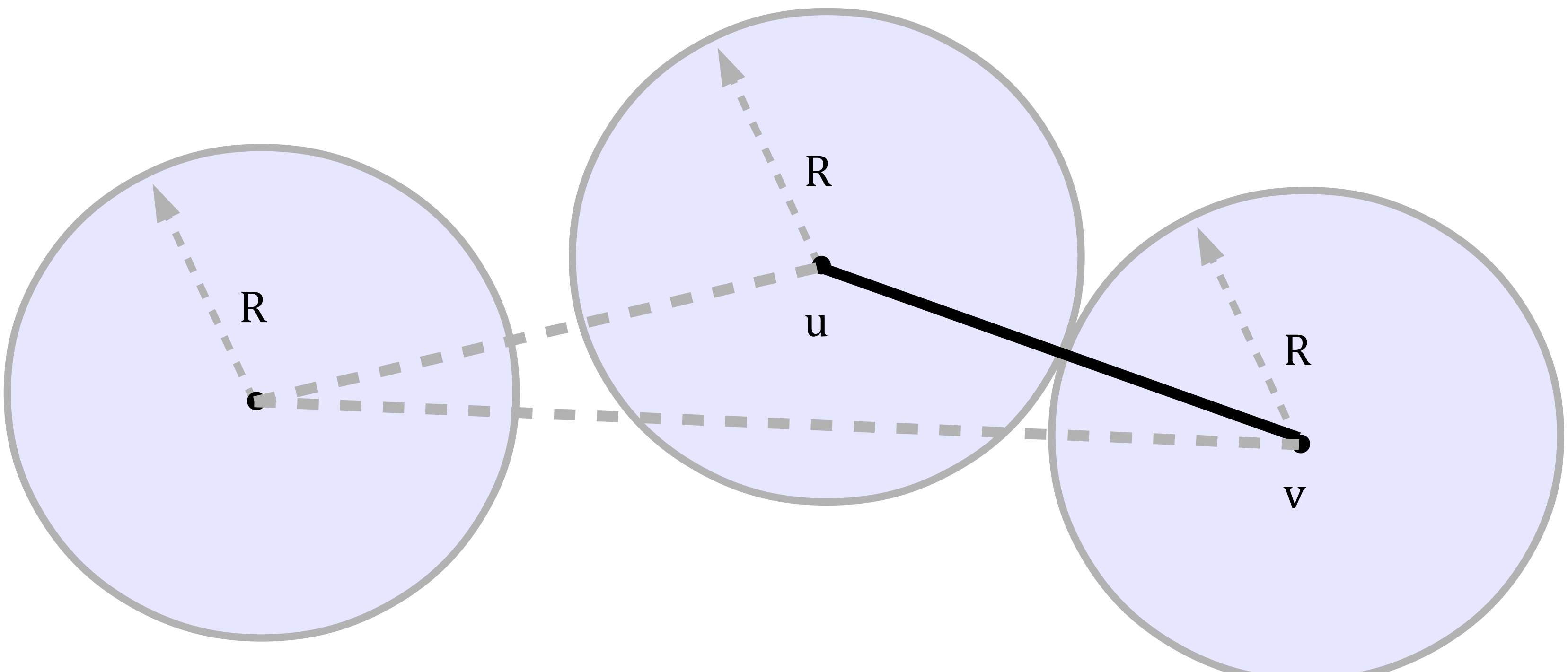


Complexity Question of CP reduces to Geometric Representation of Bi-Clique

What is the smallest dimension to represent a bi-clique in L^p-metric?

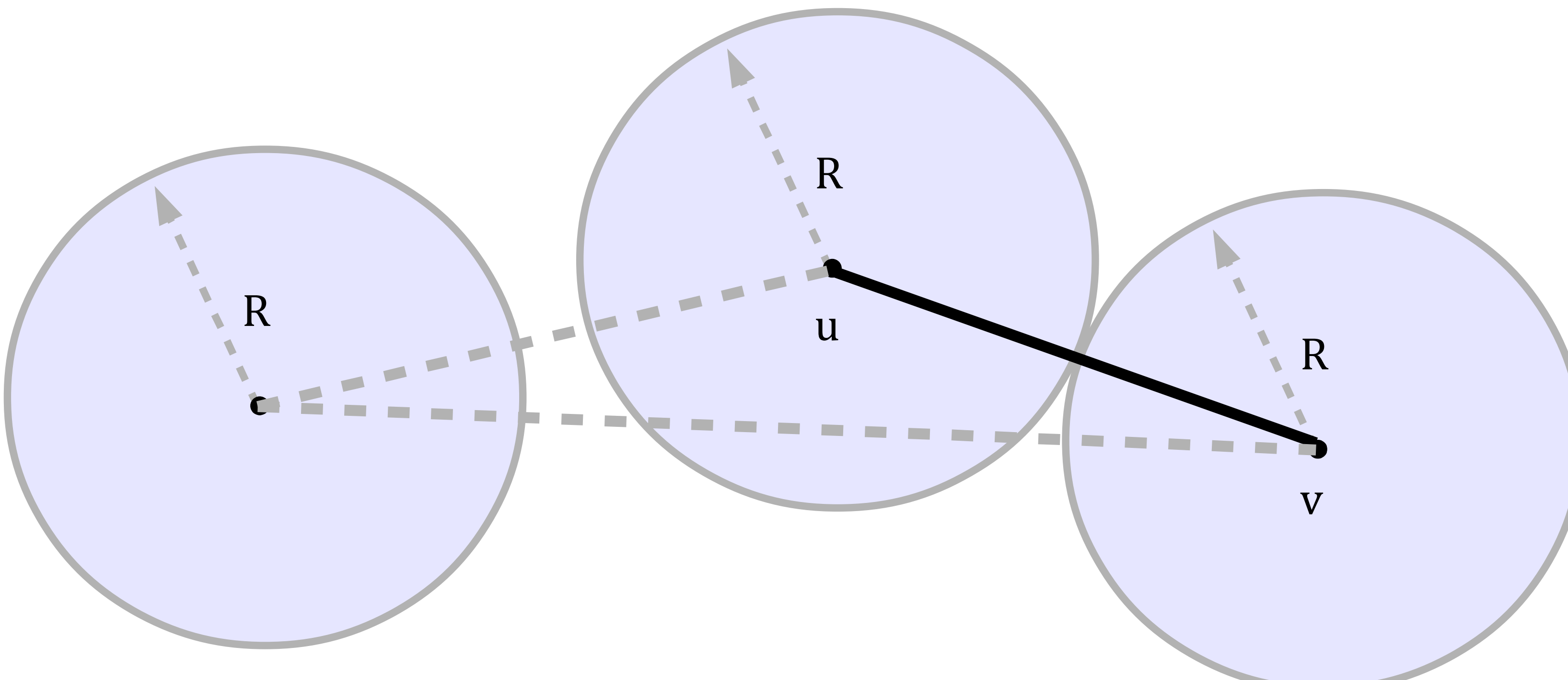
Contact-Graph

Vertices are non-overlapping d-dimensional spheres of radius R.
There is an edge uv if Sphere(u) touches Sphere(v).



Contact-Dimension of Bi-Clique in L^p

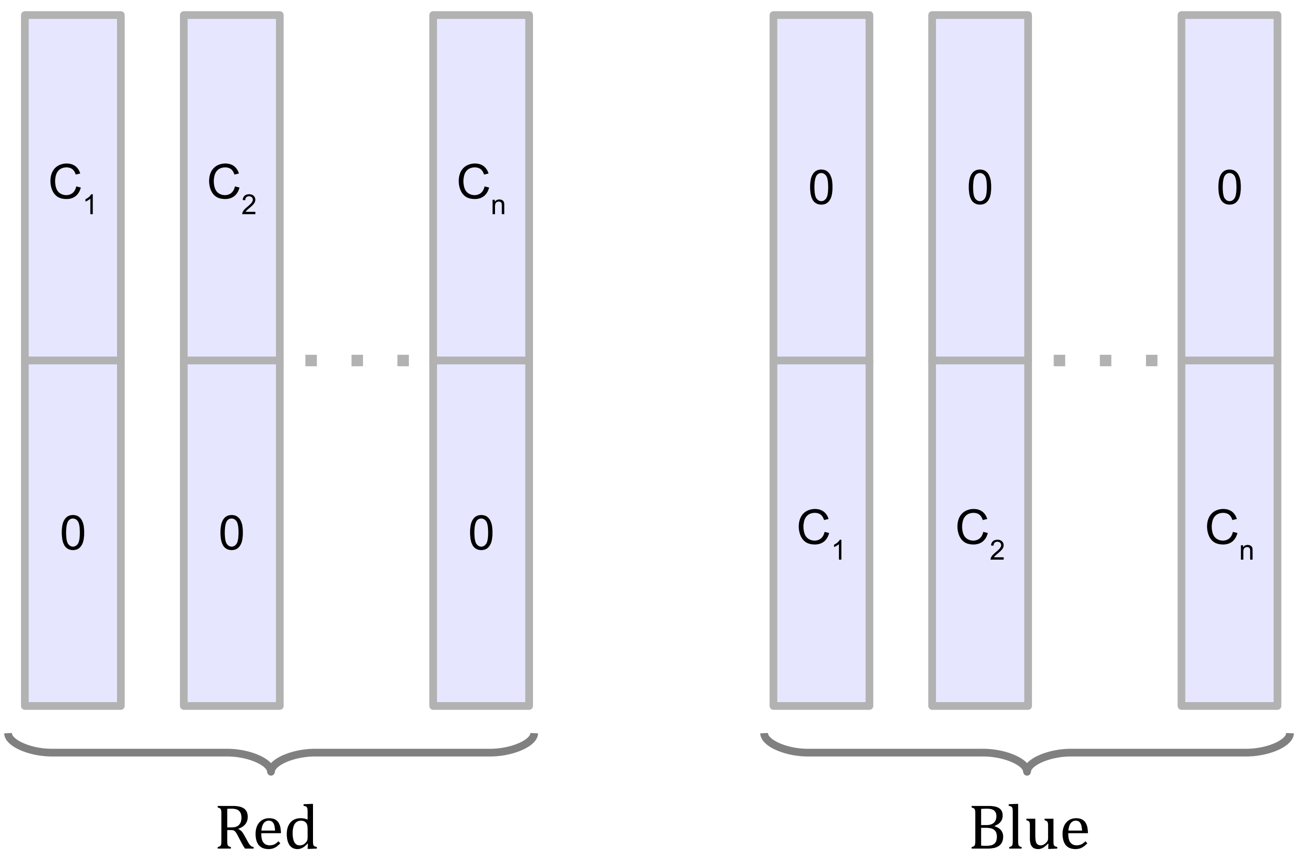
bicd(L^p) = the smallest d such that $K_{n,n}$ can be represented by a contact graph in the d-dimensional L^p metric.



bicd(L^p) ≤ O(log n) for p > 2

Concatenate random codewords C in $\{1,-1\}^n$ with 0^n

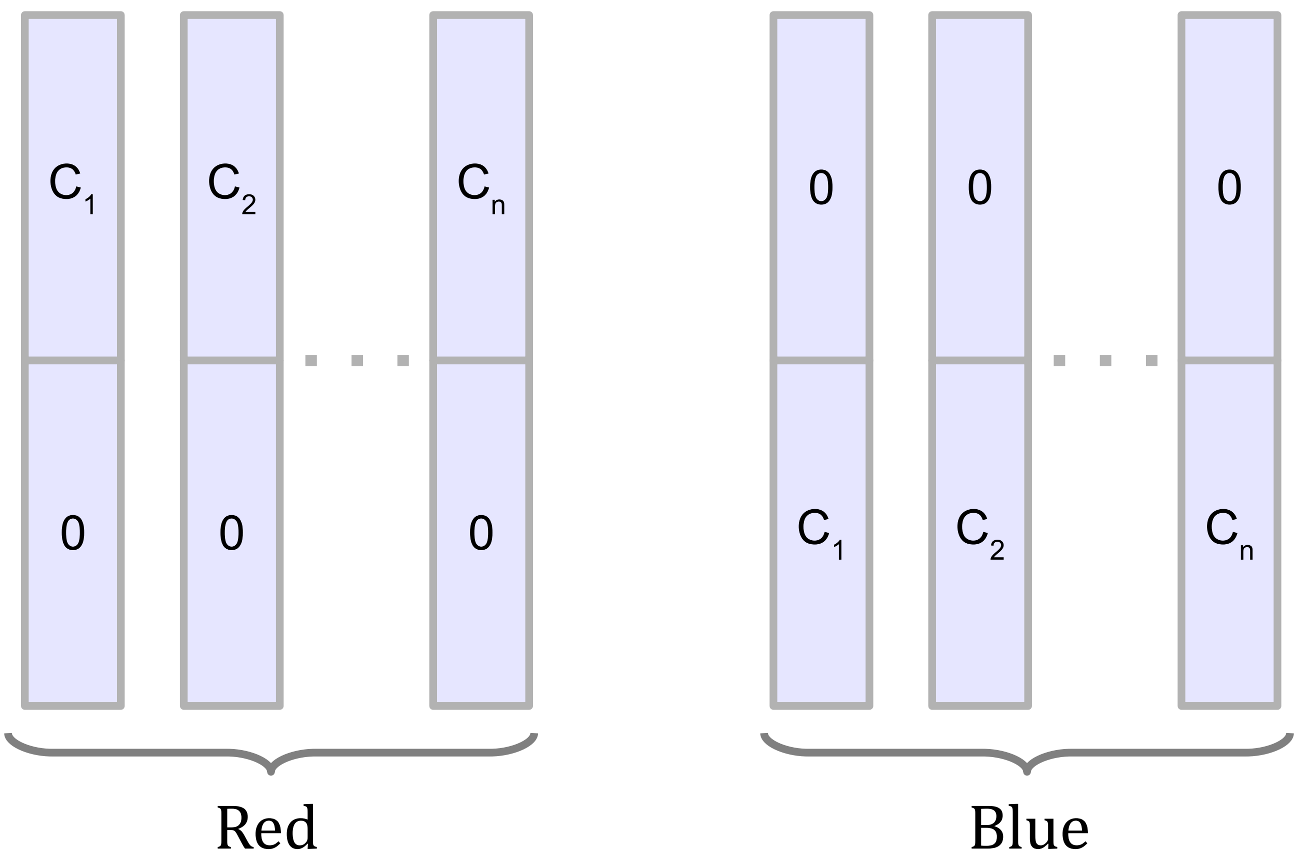
Red-codeword = C · 0^n, Blue-codeword = 0^n · C



$\mathrm{bicd}(L^p) \le O(\log n)$ for $p > 2$

Concatenate random codewords C in $\{1,-1\}^n$ with 0^n

Red-codeword = $C \cdot 0^n$, Blue-codeword = $0^n \cdot C$



$\mathrm{dist}(\text{red}, \text{blue})^p = 2n$

$\mathrm{dist}(\text{red}, \text{red}')^p$

$> (\frac{1}{2} - \varepsilon)\, n \cdot 2^p$

$> 2n$

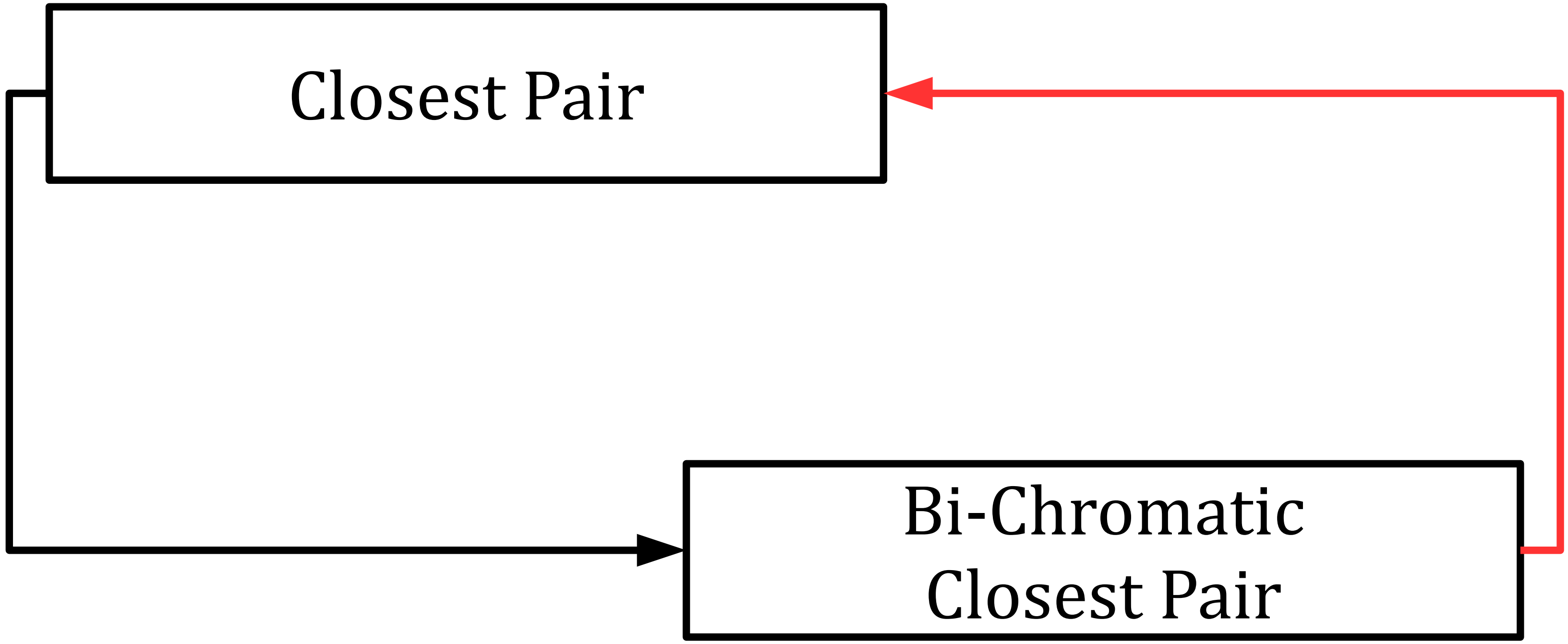
(with high probability)

CP & BCP are equivalent in L^p-metrics.

(under subquadratic-time reduction)

CP & BCP are equivalent in L^p-metrics.

(under subquadratic-time reduction)



How about other L^p-metrics?

How about other L^p-metrics?

Yes for L^∞, **No** for L^0 and L^2, **Don't know** for L^1

$$1.286\, n \leq \mathrm{bicd}(L^2) \leq 1.5\, n$$

Maehara 1985, Frankl-Maehara 1988

$$1.286\, n \le \mathrm{bicd}(L^2) \le 1.5\, n$$

Maehara 1985, Frankl-Maehara 1988

Biclique has no contact-graph
in $O(\log n)$-dimensional L^2-metric.

$$n \leq \mathrm{bicd}(L^0) \leq n$$ (hamming metric)

$$? \leq \mathrm{bicd}(L^1) \leq n^2$$

$$? \leq \mathrm{bicd}(L^p) \leq n$$ for $1 < p < 2$

$$\Omega(\log n) \leq \mathrm{bicd}(L^\infty) \leq 2 \log_2 n$$

David, Karthik CS, **L.** 2018

$$n \le \mathrm{bicd}(L^0) \le n \quad \text{(hamming metric)}$$

$$? \le \mathrm{bicd}(L^1) \le n^2$$

$$? \le \mathrm{bicd}(L^p) \le n \quad \text{for } 1 < p < 2$$

$$\Omega(\log n) \le \mathrm{bicd}(L^\infty) \le 2 \log_2 n$$

David, Karthik CS, **L.** 2018

Biclique has no contact-graph
in O(log n)-dimensional L^0-metric.

Current Bounds

for Biclique Contact Dimension

n	$\leq \text{bicd}(L^0)$	$\leq n$	
?	$\leq \text{bicd}(L^1)$	$\leq n^2$	
?	$\leq \text{bicd}(L^p)$	$\leq n$	for $1 < p < 2$
$1.286\,n$	$\leq \text{bicd}(L^2)$	$\leq 1.5\,n$	
$\Omega(\log n)$	$\leq \text{bicd}(L^p)$	$\leq O(\log n)$	for $p > 2$
$\Omega(\log n)$	$\leq \text{bicd}(L^\infty)$	$\leq 2 \log_2 n$	

Open Problems

- Better Lower / Upper Bounds for L^1 and L^2 ?
  - An alternative way to reduce BCP → CP ?

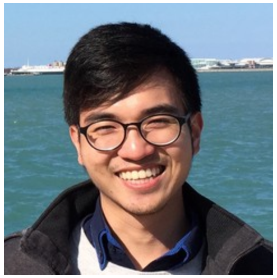
Open Problems

- Better Lower / Upper Bounds for L^1 and L^2 ?
  - An alternative way to reduce BCP → CP ?

Open problem solved:

Bichromatic & Classical Closest Pair are equivalent.

Recent work of



Karthik C. S.
Weizmann Institute

Pasin Manurangsi
UC Berkeley

The End

Thank you for your attention.

Questions?

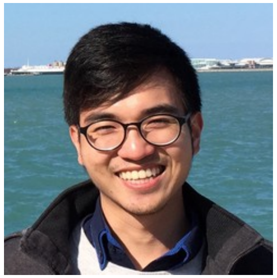
Third Part of the Talk

(extended part)

- Getting Running-Time Lower bound for Approximation Algorithms

On the Parameterized Complexity of Approximating Dominating Set

Joint work with



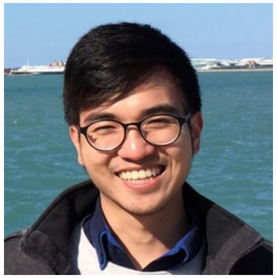
Karthik C. S.
Weizmann Institute

Pasin Manurangsi
UC Berkeley

On the Parameterized Complexity of Approximating Dominating Set

Equivalent to Set Cover

Joint work with

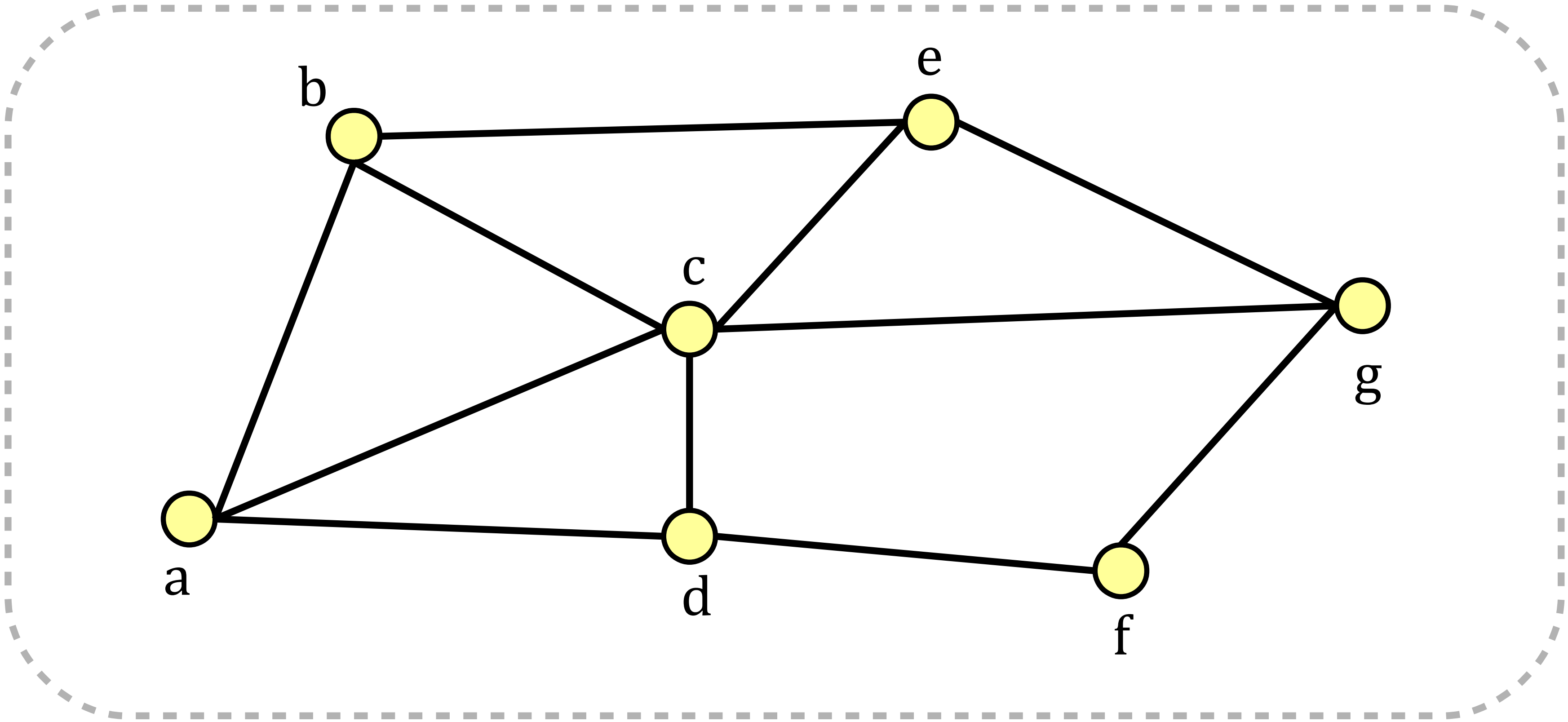


Karthik C. S.
Weizmann Institute

Pasin Manurangsi
UC Berkeley

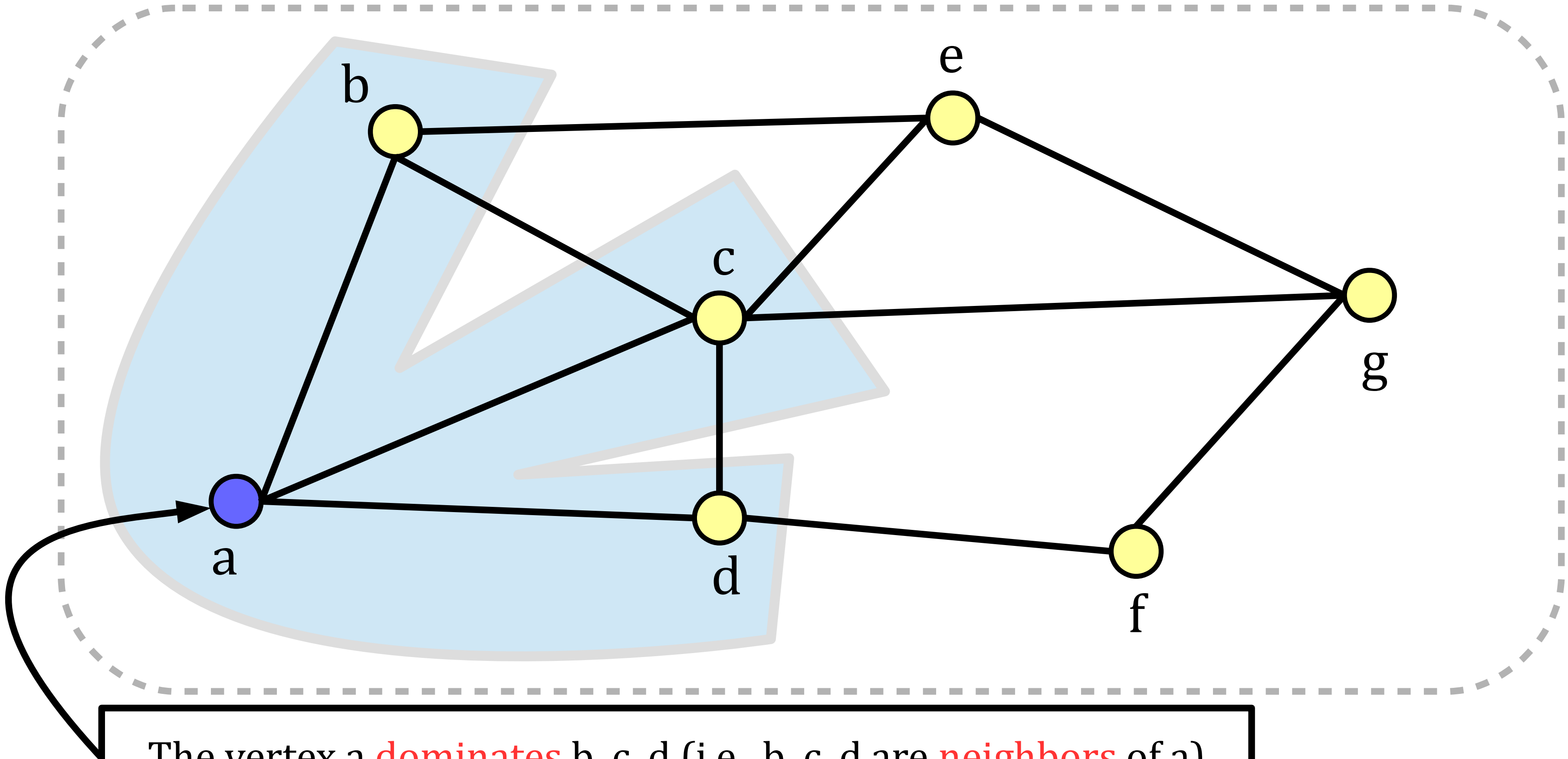
k-Dominating Set (k-DS)

Input: A graph G=(V, E) and an integer k > 0.
Goal: Find k vertices that dominates all other vertices.



k-Dominating Set (k-DS)

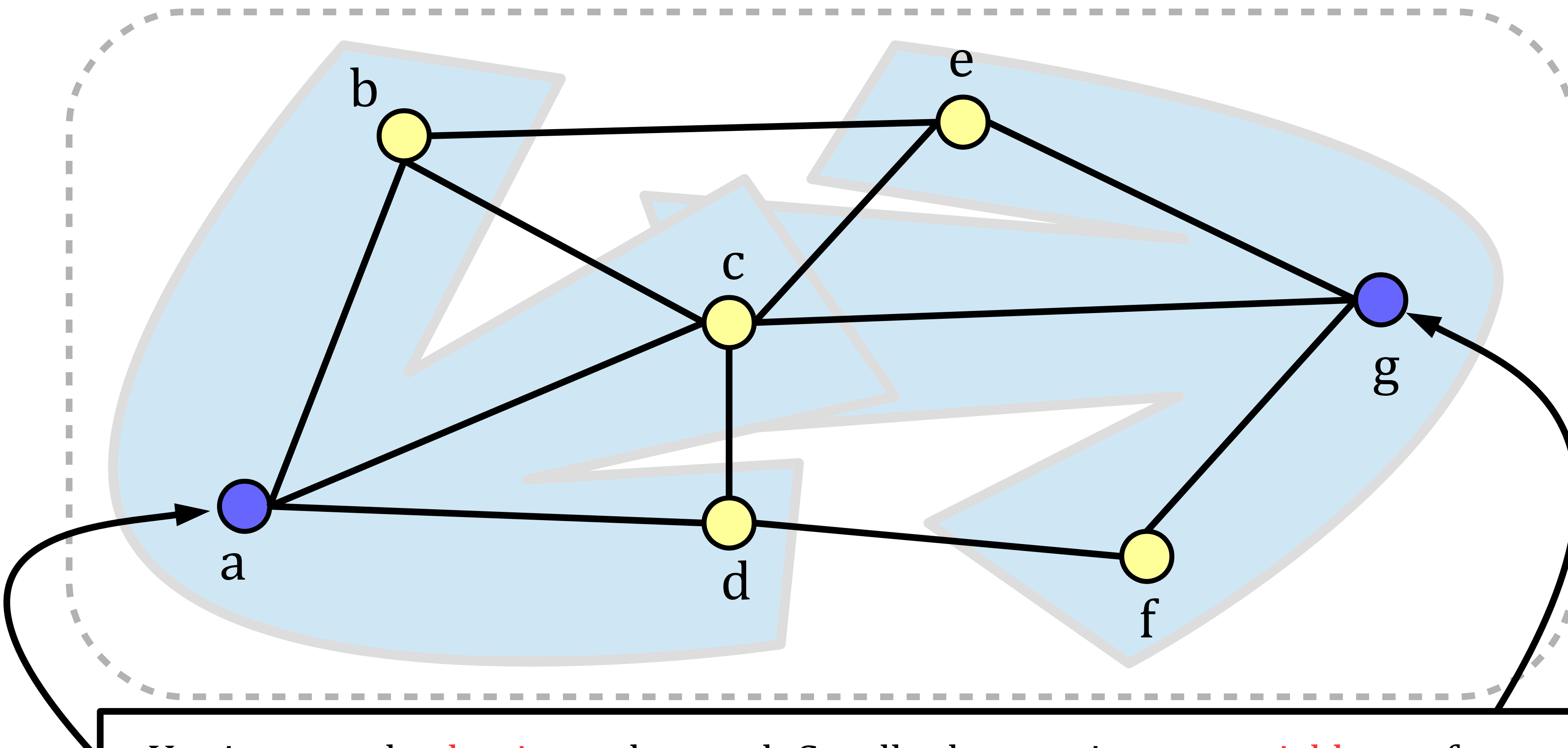
Input: A graph G=(V, E) and an integer k > 0.
Goal: Find k vertices that dominates all other vertices.



The vertex a dominates b, c, d (i.e., b, c, d are neighbors of a)

k-Dominating Set (k-DS)

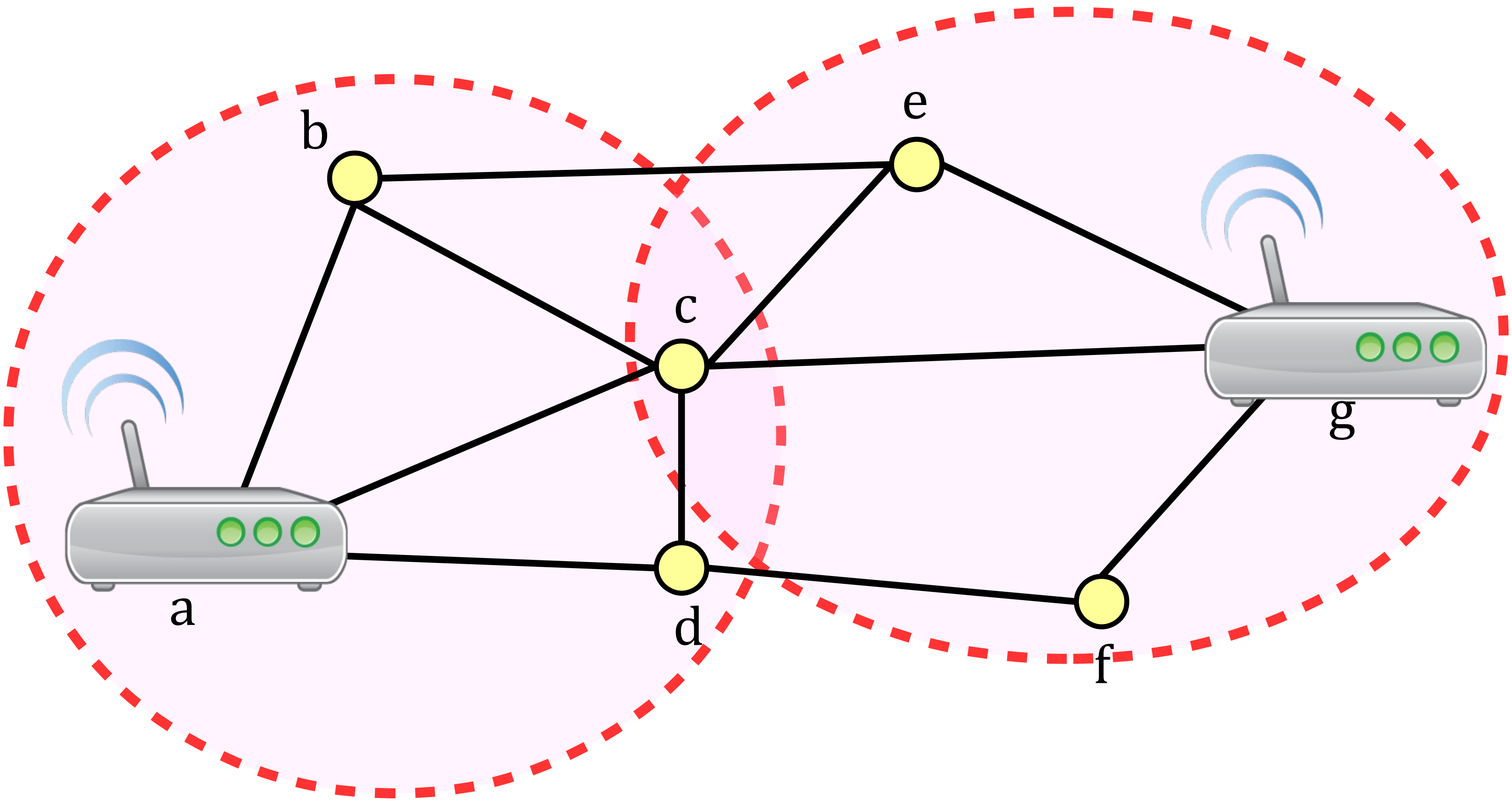
Input: A graph G=(V, E) and an integer k > 0.
Goal: Find k vertices that dominates all other vertices.



Vertices a and g dominate the graph G – all other vertices are neighbors of a or g

k-Dominating Set (k-DS)

Input: A graph G=(V, E) and an integer k > 0.
Goal: Find k vertices that dominates all other vertices.



The Importance of Dominating Set

- W[2]-Complete Problem

Analogue of SAT in Parameterized Complexity

- Fine-Grained Lower Bound

Conjecture: No n^{k-1}-time algorithm for k-DS

The Importance of Dominating Set

- W[2]-Complete Problem

Analogue of SAT in Parameterized Complexity

- Fine-Grained Lower Bound

Conjecture: No n^{k-1}-time algorithm for k-DS

Inapproximability k-DS implies running-time bound of approximating many problems.

Known Algorithms

- Exact $n^{k+o(1)}$-time [PW'10]
- $O(\log n)$-approx in $O(n^3)$-time [J74]
- $(\ln n - \ln \ln n - \Theta(1))$-approx in $n^{O(1)}$-time [S96]

Known Algorithms

- Exact $n^{k+o(1)}$-time [PW'10]
- $O(\log n)$-approx in $O(n^3)$-time [J74]
- $(\ln n - \ln \ln n - \Theta(1))$-approx in $n^{O(1)}$-time [S96]

We don't know how to find a dominating set of size 2^k in time n^{k-1} even if we know that the graph has dominating set of size $k \ll n$.

Is there an $O(n^{k-1})$-time algorithm that gives $O(2^k)$-approximation?

Is there an $O(n^{k-1})$-time algorithm that gives $O(2^k)$-approximation?

Our answer is NO, and it is even WORSE!

(assuming popular hypotheses)

Our Results

Parameterized Inapproximability of k-Dominating Set

Hypothesis	Inapprox	Running Time	
$W[1] \neq FPT$	$\Omega(\log^{1/poly(k)} n)$	FPT-Time	(T(k) poly(n))
ETH	$\Omega(\log^{1/poly(k)} n)$	$n^{o(k)}$	
SETH	$\Omega(\log^{1/f(k)} n)$	$n^{k-\varepsilon}$	for any $\varepsilon > 0$
k-SUM	$\Omega(\log^{1/poly(k)} n)$	$n^{k/2-\varepsilon}$	for any $\varepsilon > 0$

Our Results

Parameterized Inapproximability of k-Dominating Set

Hypothesis	Inapprox	Running Time	
W[1]≠FPT	$\Omega(\log^{1/\text{poly}(k)} n)$	FPT-Time (T(k) poly(n))	
ETH	$\Omega(\log^{1/\text{poly}(k)} n)$	$n^{o(k)}$	
SETH	$\Omega(\log^{1/f(k)} n)$	$n^{k-\varepsilon}$	for any $\varepsilon > 0$
k-SUM	$\Omega(\log^{1/\text{poly}(k)} n)$	$n^{k/2-\varepsilon}$	for any $\varepsilon > 0$

Even exp(exp(exp(k)))-Approx in $(n^{k-0.1})$-Time is unlikely!

Our Results

Parameterized Inapproximability of k-Dominating Set

Hypothesis	Inapprox	Running Time	
~~W[1]≠FPT~~	$\Omega(\log^{1/\mathrm{poly}(k)} n)$	FPT-Time	(T(k) poly(n))
~~ETH~~	$\Omega(\log^{1/\mathrm{poly}(k)} n)$	$n^{o(k)}$	
~~SETH~~	$\Omega(\log^{1/f(k)} n)$	~~$n^{k-\varepsilon}$~~	for any $\varepsilon > 0$
~~k-SUM~~	$\Omega(\log^{1/\mathrm{poly}(k)} n)$	~~$n^{k/2-\varepsilon}$~~	for any $\varepsilon > 0$

Hardness of these forms (i.e., $\Omega(\log^{1/k} n)$-inapprox in $n^{o(k)}$-time) were known only under Gap-ETH. [CCK**L**MNT, FOCS'17]

Our Techniques

Parameterized Inapproximability via Communication Complexity

Our results follow by the connection between Communication Protocols & Label Cover.

Our Techniques

Parameterized Inapproximability via Communication Complexity

Our results follow by the connection between Communication Protocols & Label Cover.

(Precisely, the variant of Label Cover defined in [CCK**L**MNT'17], namely Max-Cover.)

Our Techniques

Parameterized Inapproximability via Communication Complexity

Our results follow by the connection between Communication Protocols & Label Cover.

(Precisely, the variant of Label Cover defined in [CCK**L**MNT'17], namely Max-Cover.)

This connection was first discovered by [ARW, FOCS'17] for Inapprox of Problems in P.
We extend their works to the area of Parameterized Inapproximability.

Our Techniques

Parameterized Inapproximability via Communication Complexity

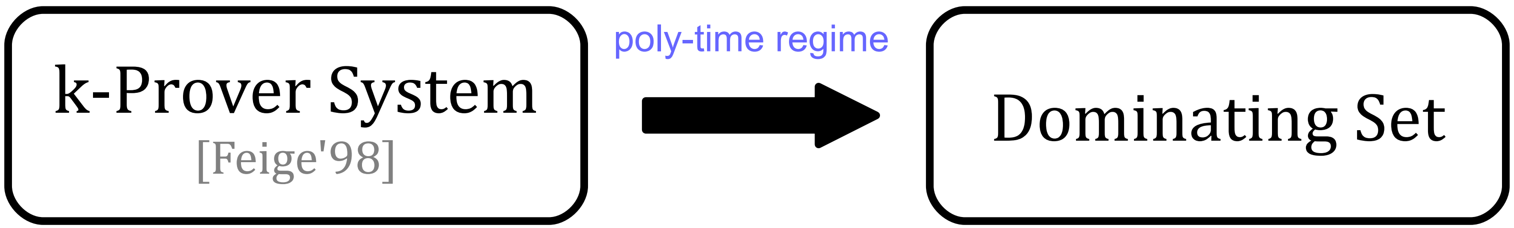
Our results follow by the connection between Communication Protocols & Label Cover.

(Precisely, the variant of Label Cover defined in [CCK**L**MNT'17], namely Max-Cover.)

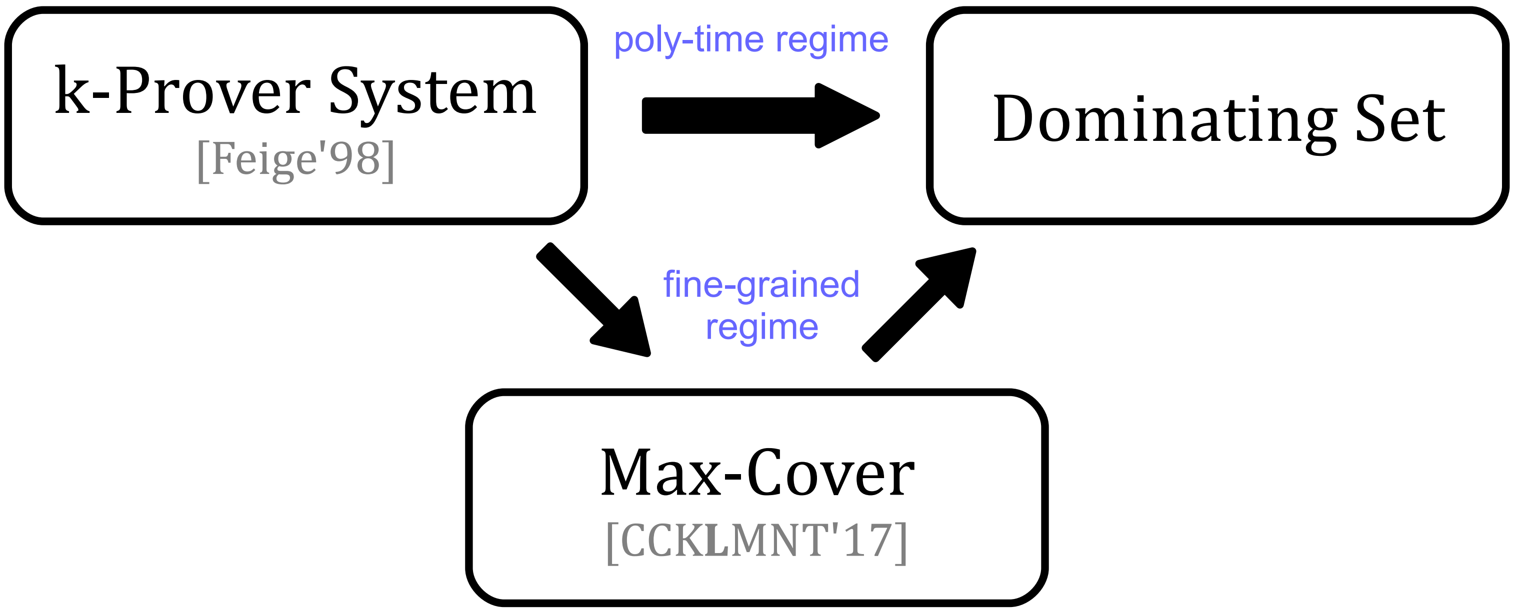
We devise communication protocols for problems corresponding to the popular hypotheses FPT≠W[1], ETH, SETH, k-SUM.

This connection was first discovered by [ARW, FOCS'17] for Inapprox of Problems in P.
We extend their works to the area of Parameterized Inapproximability.

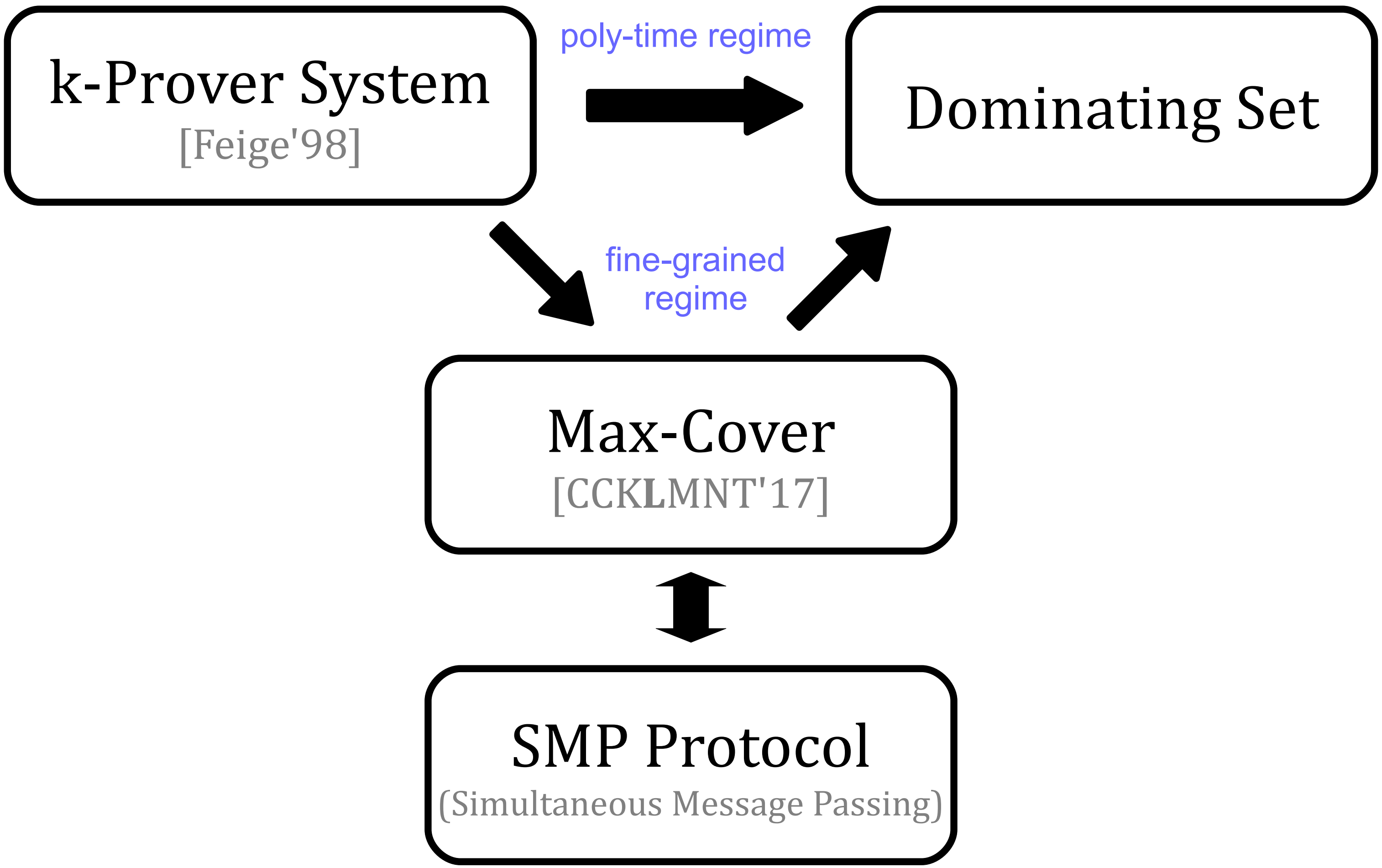
k-DS & Communication Protocols



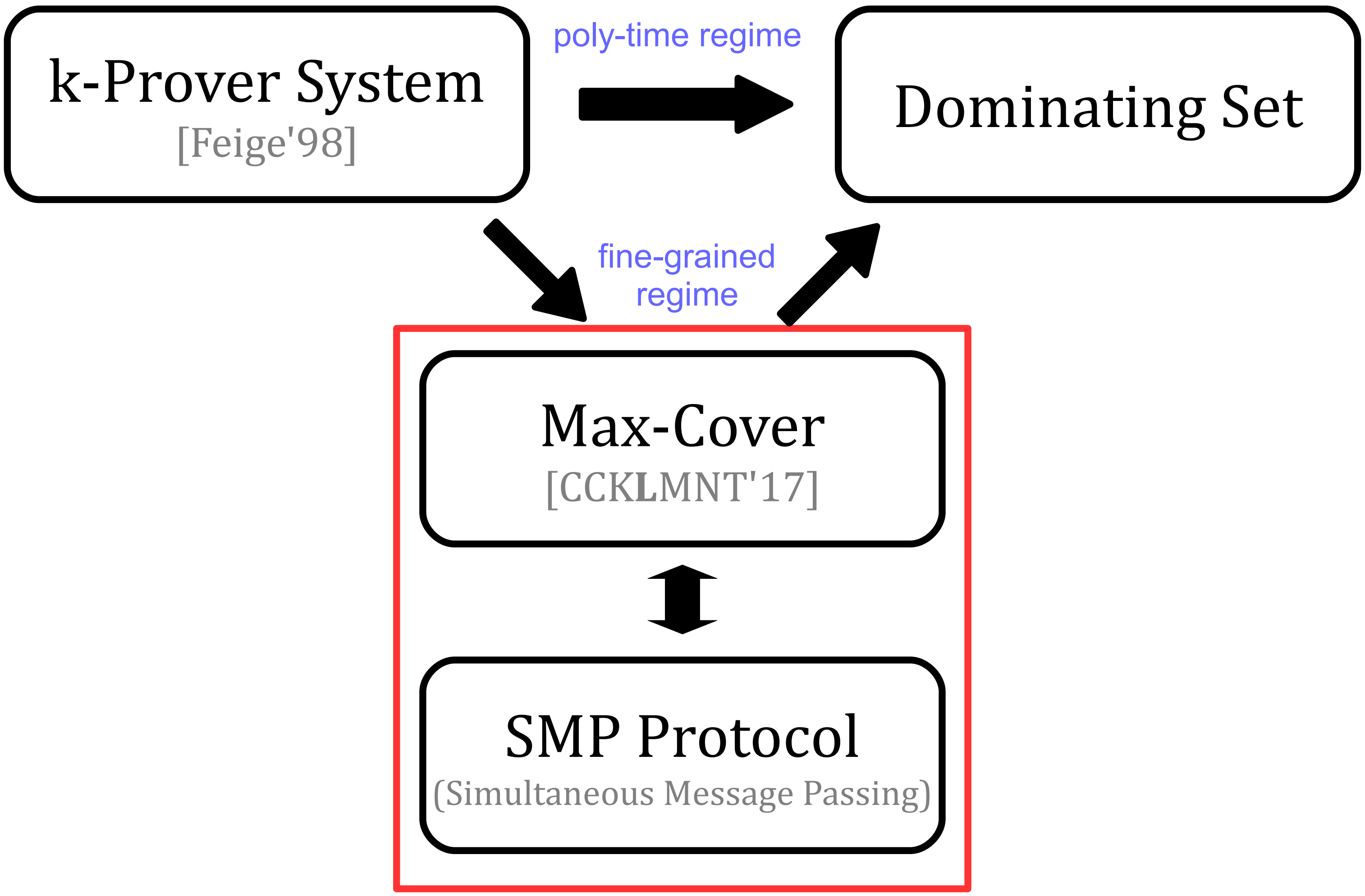
k-DS & Communication Protocols



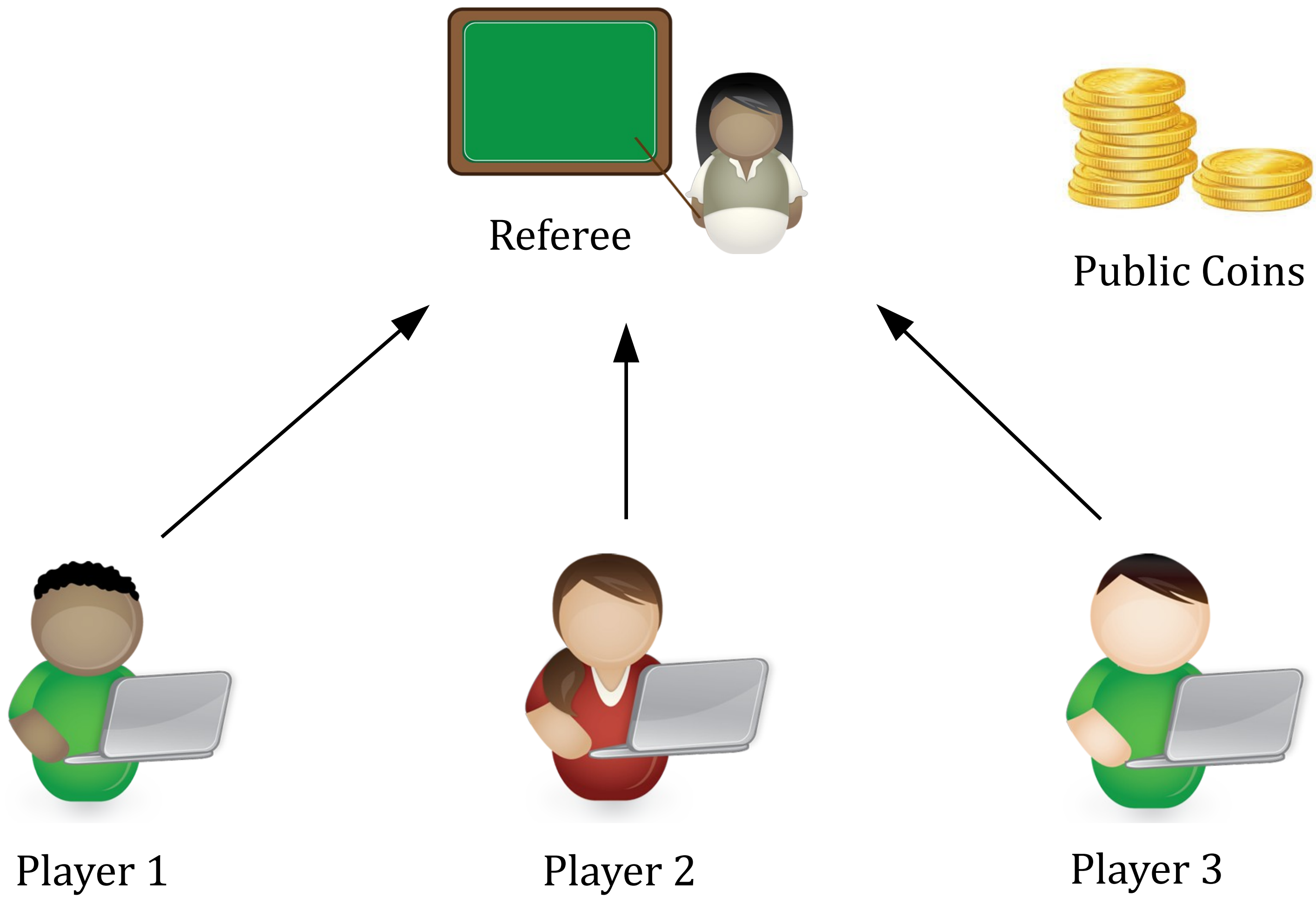
k-DS & Communication Protocols



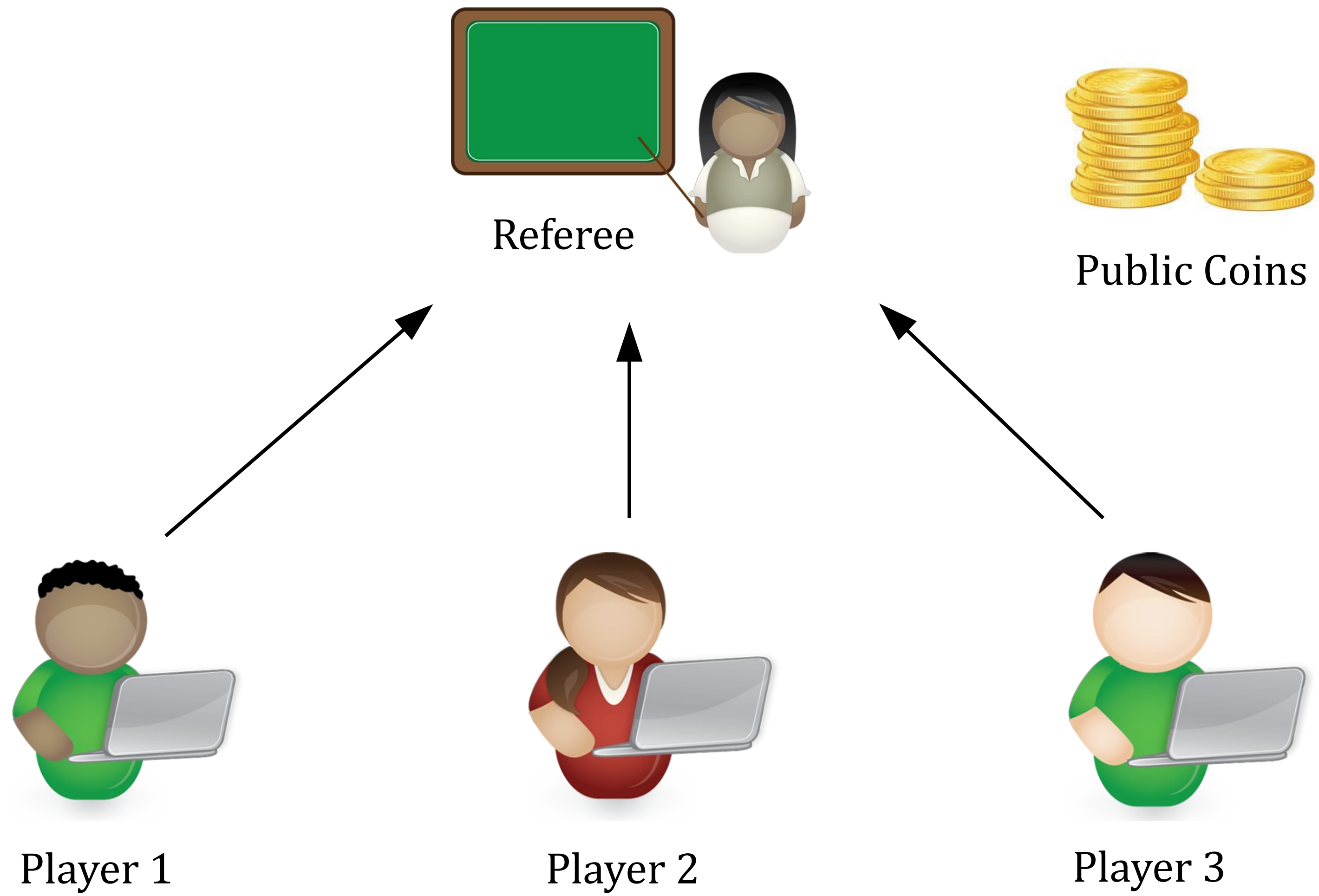
k-DS & Communication Protocols



Single Message Passing Protocol

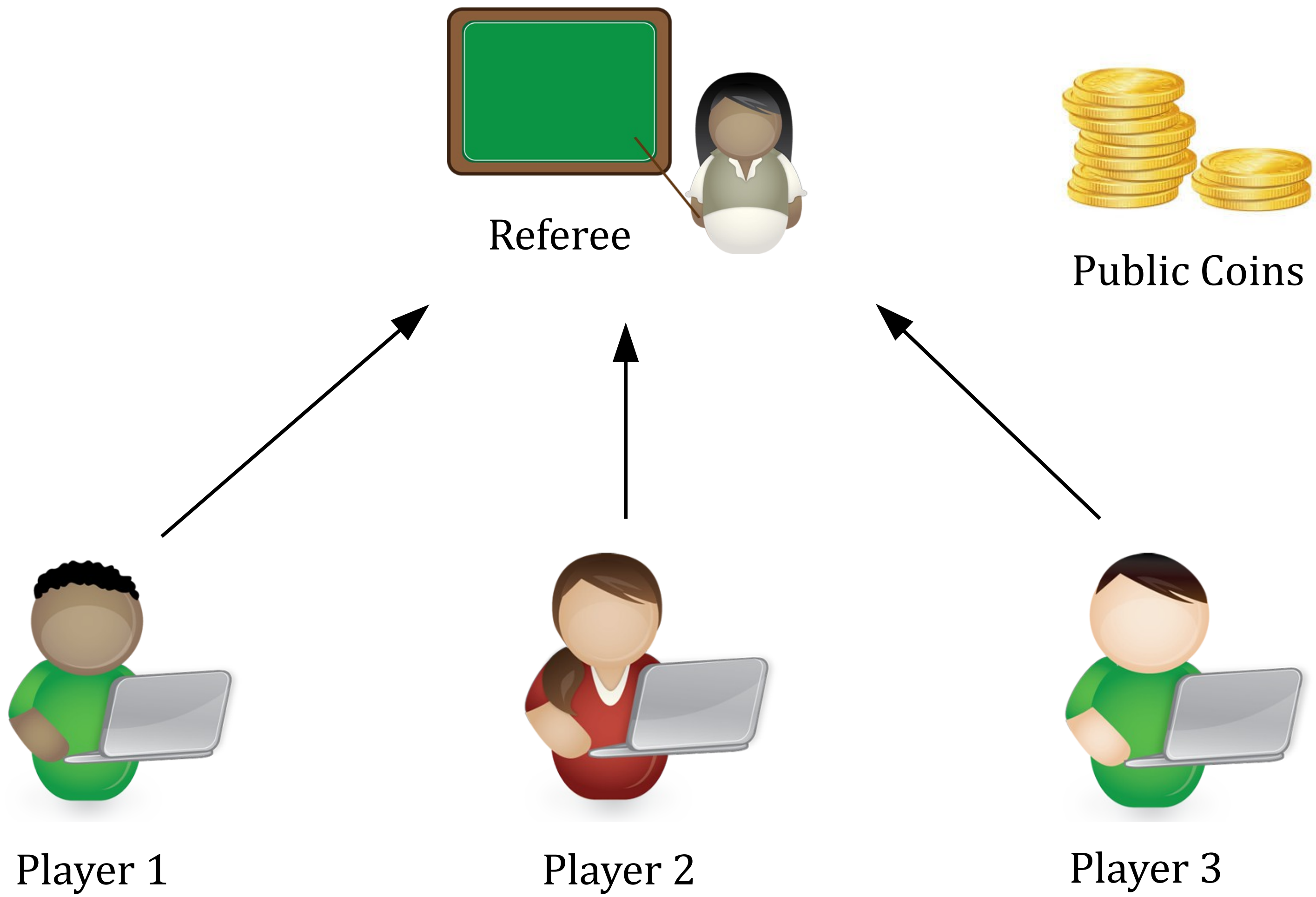


Single Message Passing Protocol



The goal is to use the smallest communication bits.

Single Message Passing Protocol



For our purpose, we want to use $\leq O(1)$ bits.

Known SMP Protocols

that captures popular conjectures

Hypothesis	Protocols	Running Time	
W[1]≠FPT	Agreement	FPT-Time	(T(k) poly(n))
ETH	Set Disjointness	$n^{o(k)}$	
SETH	Set Disjointness	$n^{k-\varepsilon}$	for any $\varepsilon > 0$
k-SUM	Sum-Zero	$n^{k/2-\varepsilon}$	for any $\varepsilon > 0$

All protocols give inapprox ratio $\Omega(\log n)^{1/\mathrm{poly}(k)}$.

Popular Hypotheses

- **ETH:** SAT has no $2^{o(n)}$-time algorithm.
- **SETH:** SAT has no $2^{(1-\varepsilon)n}$-time algorithm.
- **k-SUM:** Find k integers from $S \subseteq [-n^2, n^2]$ that sum to zero requires $n^{k/2-\varepsilon}$-time.
- **FPT ≠ W[1]:** k-DS has no T(k) poly(n)-time algorithm.

FPT≠W[1] Hypothesis

NP-hardness-like assumption in Parameterized Complexity

"k-Clique has no T(k) poly(n) time algorithm."

FPT≠W[1] Hypothesis

NP-hardness-like assumption in Parameterized Complexity

"k-Clique has no T(k) poly(n) time algorithm."

FPT is a class of problems that have T(k) poly(n) time (i.e., FPT) algorithms.
W[1]-complete is a class of problems that have no known FPT algorithm.
It is known that FPT ⊆ W[1], but the converse is believed to be false.

FPT≠W[1] Hypothesis

NP-hardness-like assumption in Parameterized Complexity

"k-Clique has no T(k) poly(n) time algorithm."

FPT is a class of problems that have T(k) poly(n) time (i.e., FPT) algorithms.
W[1]-complete is a class of problems that have no known FPT algorithm.
It is known that FPT ⊆ W[1], but the converse is believed to be false.

Like SAT is a representative of NP-complete problems ,
k-Clique is a representative of W[1]-complete.

FPT≠W[1] Hypothesis

NP-hardness-like assumption in Parameterized Complexity

"~~k-Clique~~ has no $T(k)$ poly(n) time algorithm."
k-SUM

FPT is a class of problems that have $T(k)$ poly(n) time (i.e., FPT) algorithms.
W[1]-complete is a class of problems that have no known FPT algorithm.
It is known that FPT ⊆ W[1], but the converse is believed to be false.

Like SAT is a representative of NP-complete problems ,
~~k-Clique~~ is a representative of W[1]-complete.
k-SUM

SMP-Protocol $\leftrightarrow$ Max-Cover $\rightarrow$ k-DS

We will skip the formal definition of
the **S**imultaneous **M**essages **P**assing protocol.

SMP-Protocol $\leftrightarrow$ Max-Cover $\rightarrow$ k-DS

(for k-SUM)

We will skip the formal definition of
the **S**imultaneous **M**essages **P**assing protocol.

SMP-Protocol $\leftrightarrow$ ~~Max-Cover~~ $\rightarrow$ k-DS

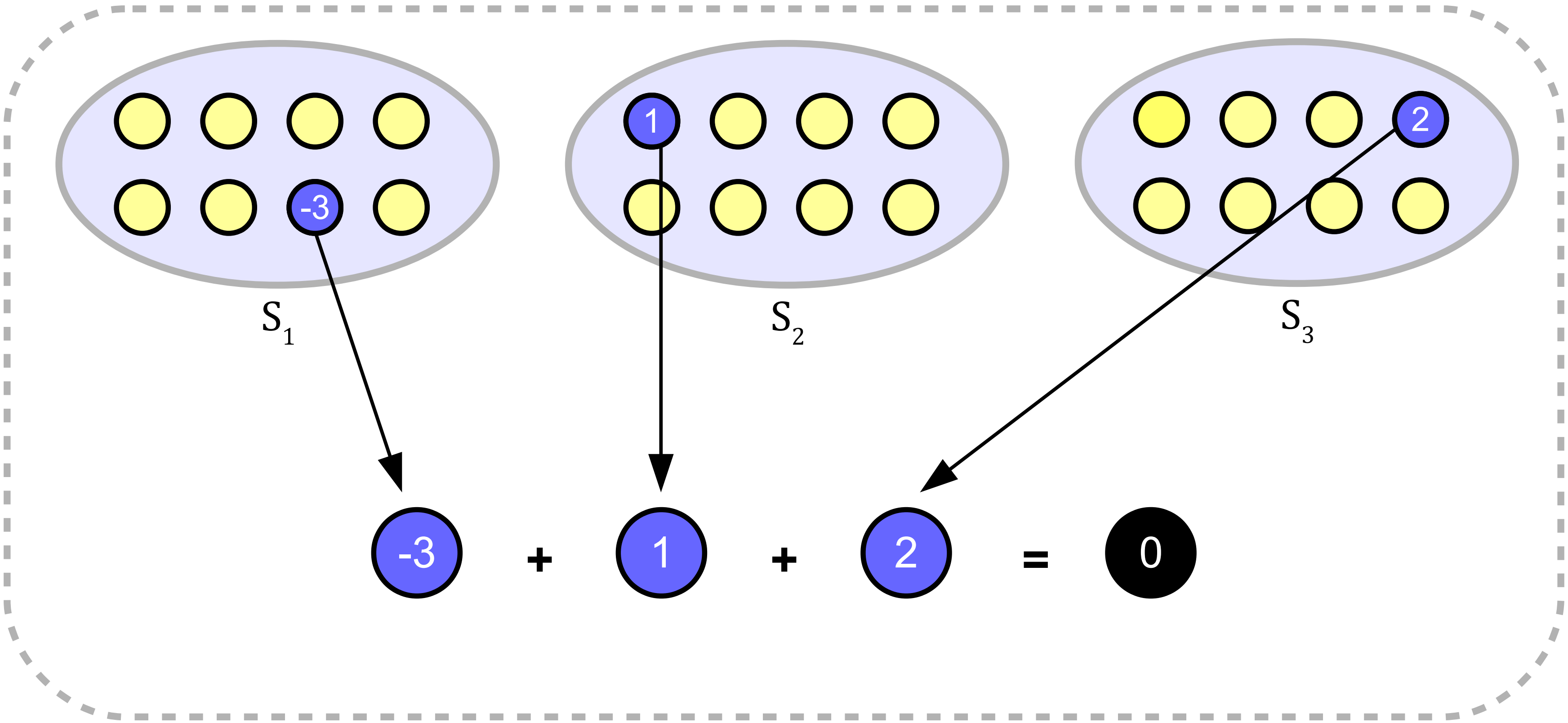
(for k-SUM)

We will skip the formal definition of
the **S**imultaneous **M**essages **P**assing protocol.

k-SUM (multi-set version)

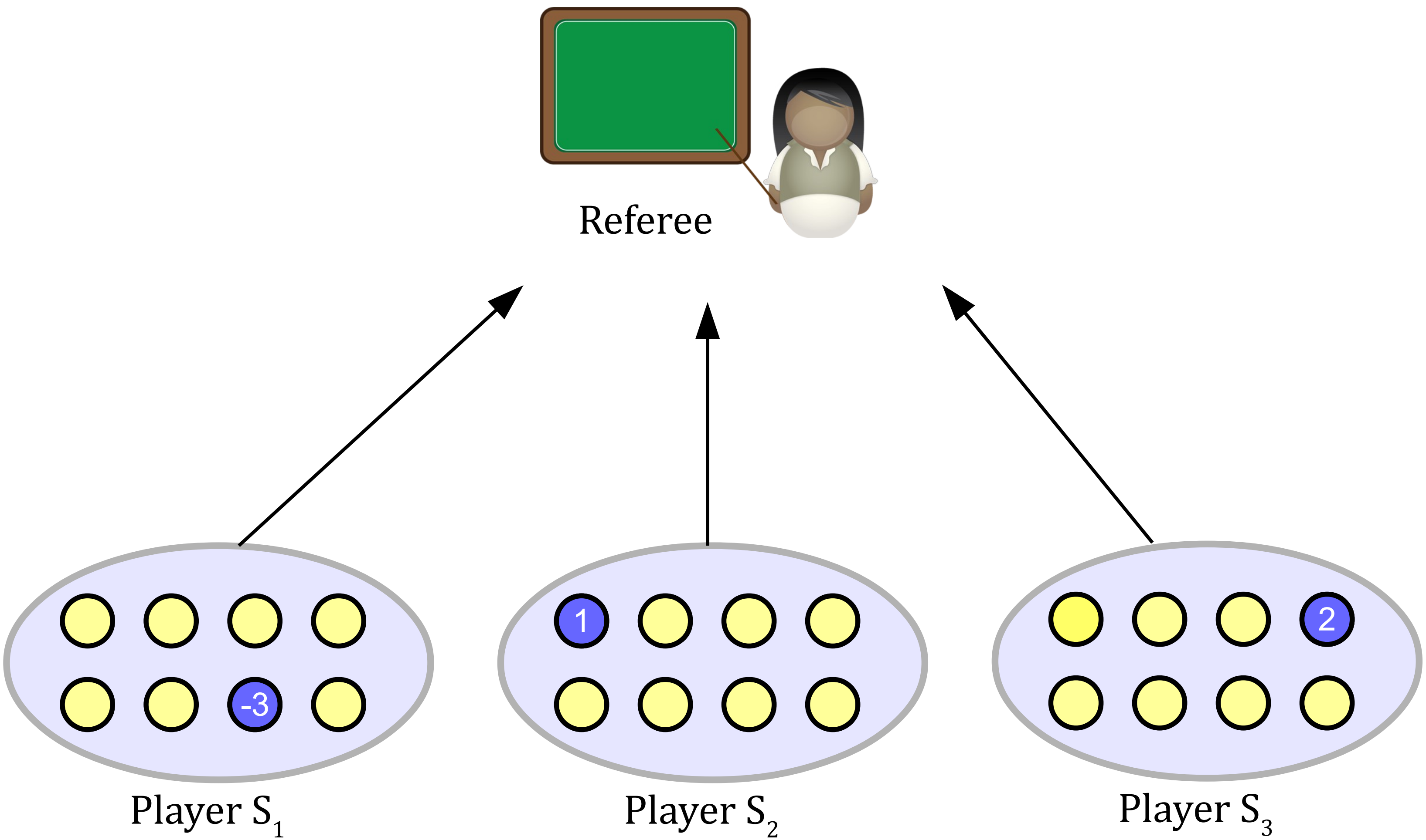
Input: k collections of Integers $S_1, ..., S_k$.

Goal: find k integers $a_1 \in S_1, ..., a_k \in S_k$ s.t. $a_1 + ... + a_k = 0$.



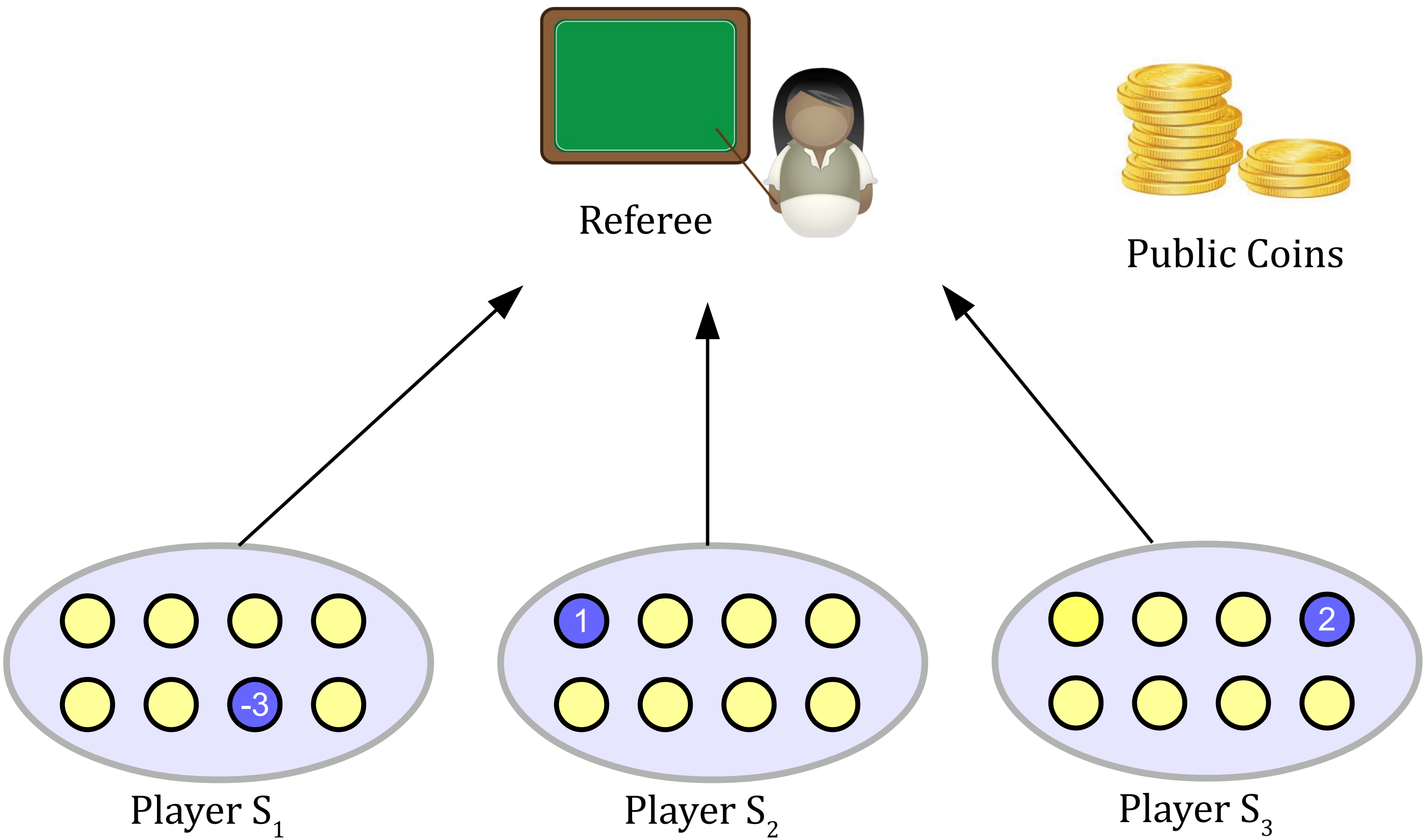
Now we devise a protocol for k-SUM. Let 's assume that each play has one number (presumably given by k-SUM algorithm).

Protocol for k-SUM (Sum-Zero Protocol)



Each player has one number-in-hand (presumably, given by a k-SUM algorithm).
And they, honestly, want to decide if these numbers sum to zero.

Protocol for k-SUM (Sum-Zero Protocol)

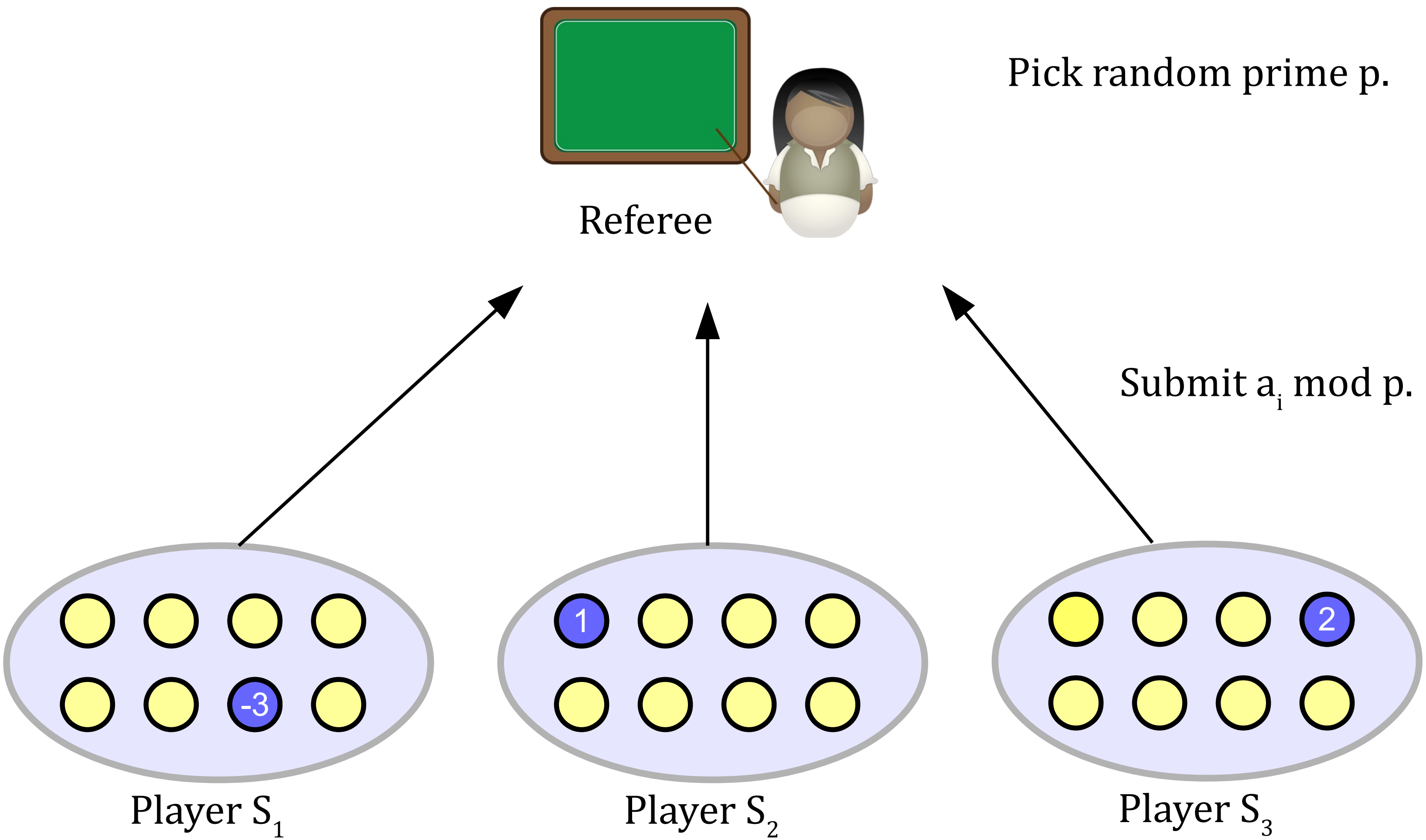


The referee may use randomness to reduce the communication bits.

Protocol for k-SUM (Sum-Zero Protocol)

Pick random prime p.

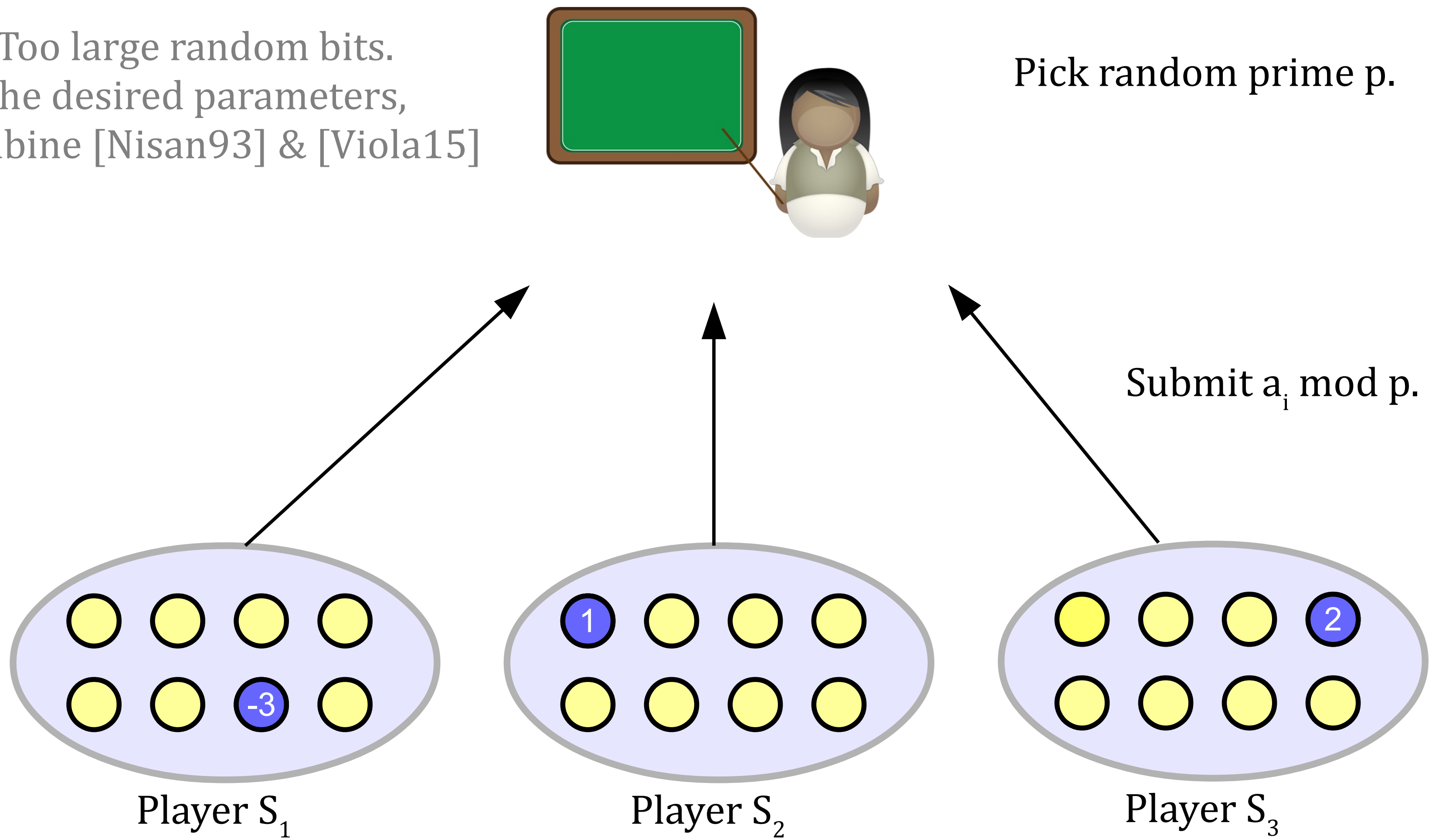
Submit a_i mod p.



$$a_1 + a_2 + a_3 = 0 \;\rightarrow\; (a_1 \bmod p) + (a_2 \bmod p) + (a_3 \bmod p) = 0$$

Protocol for k-SUM (Sum-Zero Protocol)

NOTE: Too large random bits.
To get the desired parameters,
we combine [Nisan93] & [Viola15]

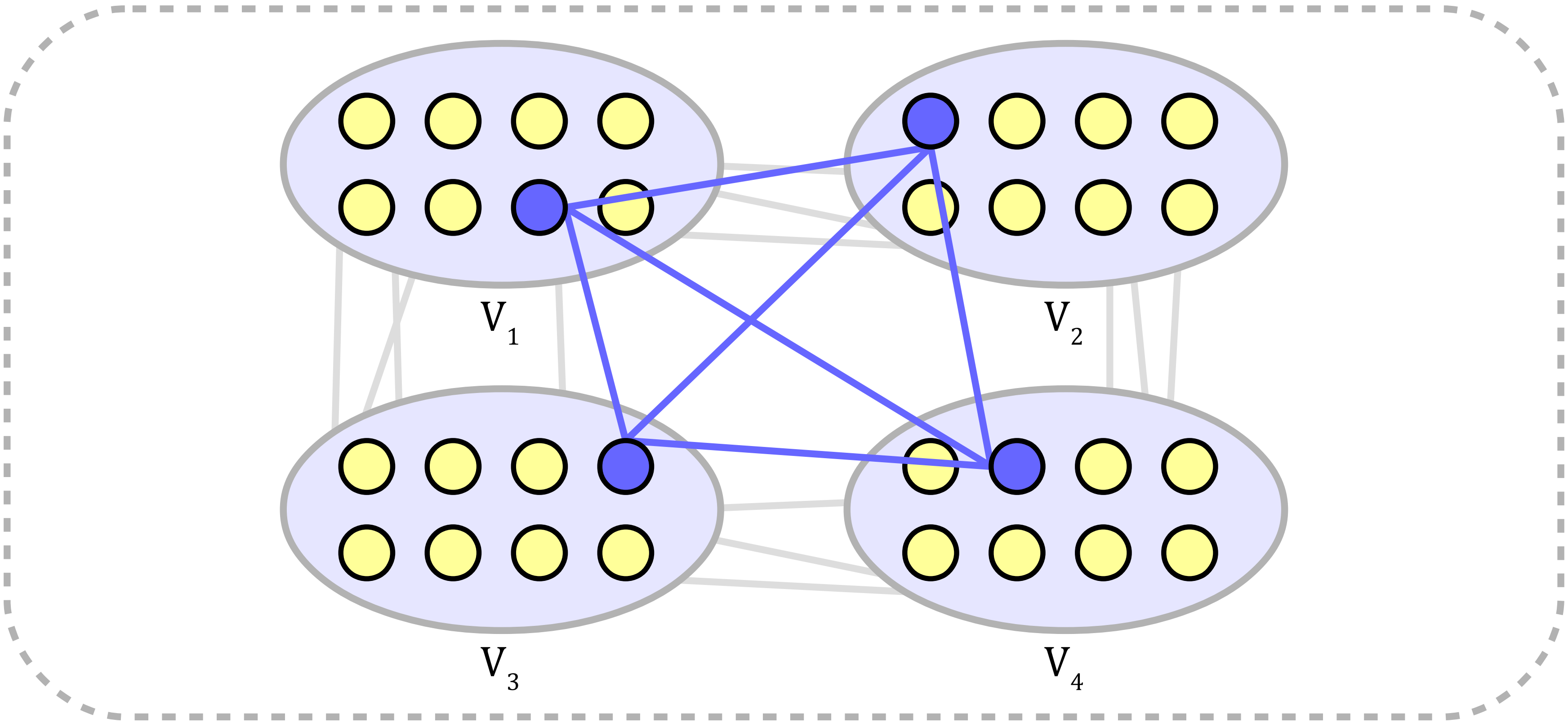


$$a_1 + a_2 + a_3 = 0 \;\rightarrow\; (a_1 \bmod p) + (a_2 \bmod p) + (a_3 \bmod p) = 0$$

k-Clique (multi-color version)

Input: a bipartite graph $G = (V, E)$, a k-partition of $V = V_1 \uplus \ldots \uplus V_k$

Goal: find a subset of vertices $S \subseteq V(G)$, one from each V_i s.t. $G[S]$ is a clique.

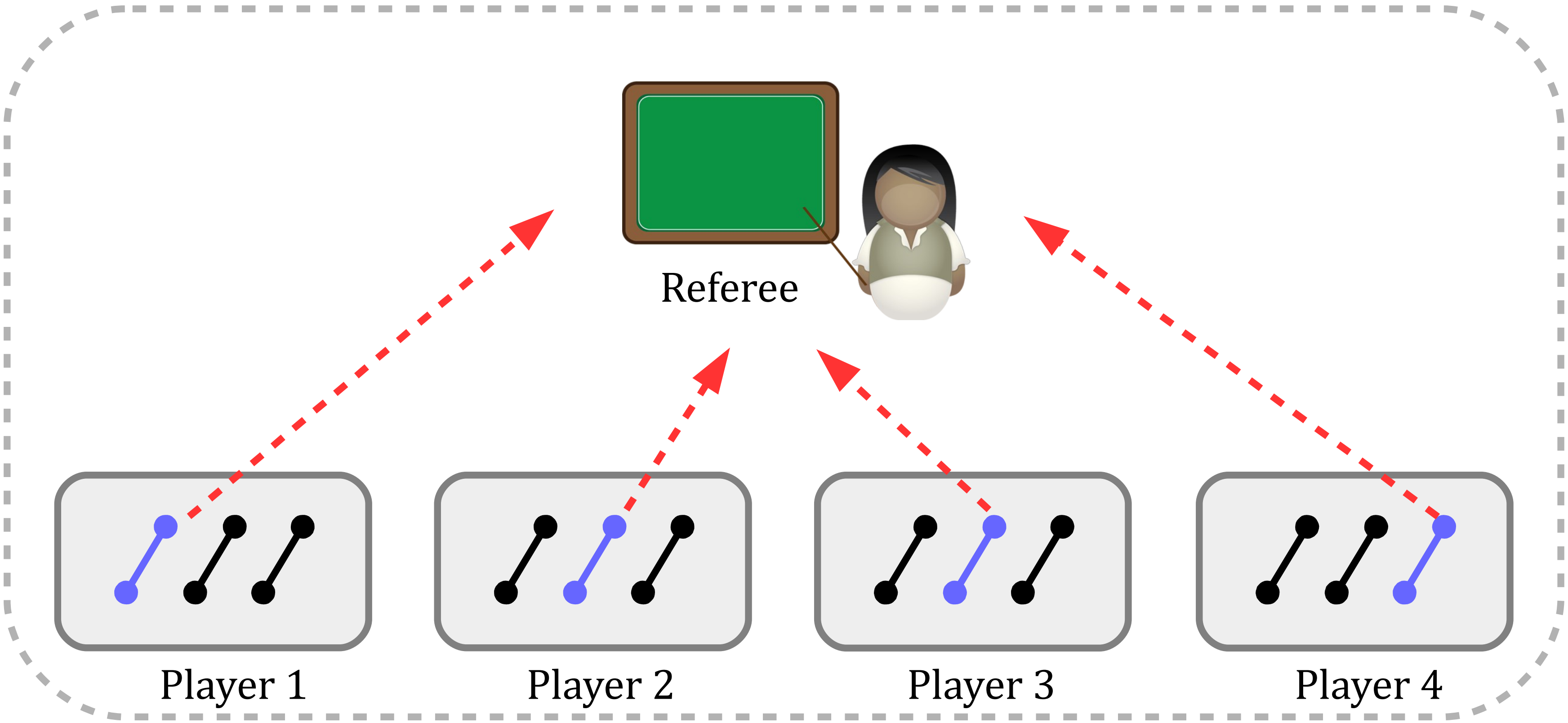


Protocol for k-Clique

(Edge-Vertex Incident Game)

Players: $O(k^2)$ $\{i,j\}$-players, each submits one edge between V_i to V_j.

Referee: Check if there exists k vertices (one from each V_i) such that each vertex is incident to k-1 submitted edges.

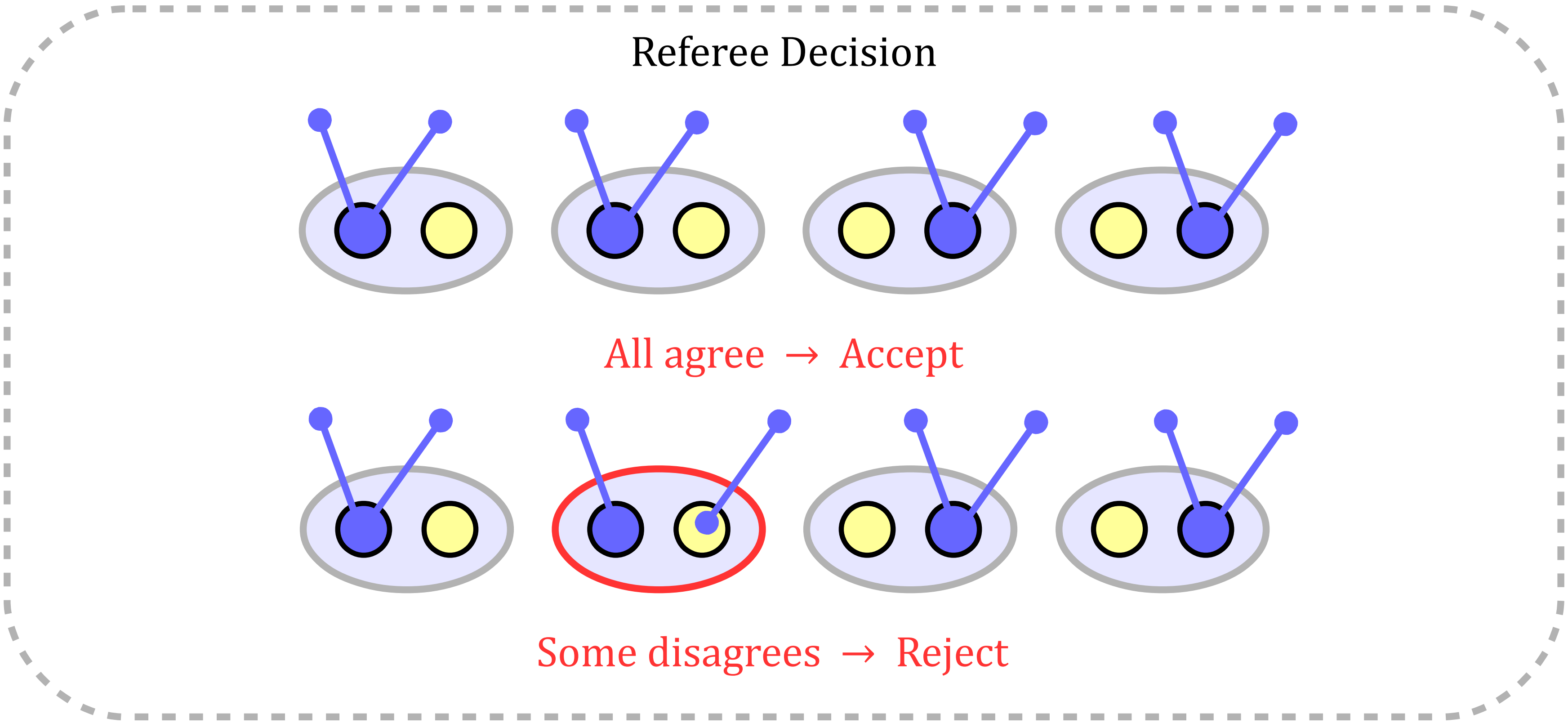


Protocol for k-Clique

(Edge-Vertex Incident Game)

Players: $O(k^2)$ $\{i,j\}$-players, each submits one edge between V_i to V_j.

Referee: Check if there exists k vertices (one from each V_i) such that each vertex is incident to k-1 submitted edges.

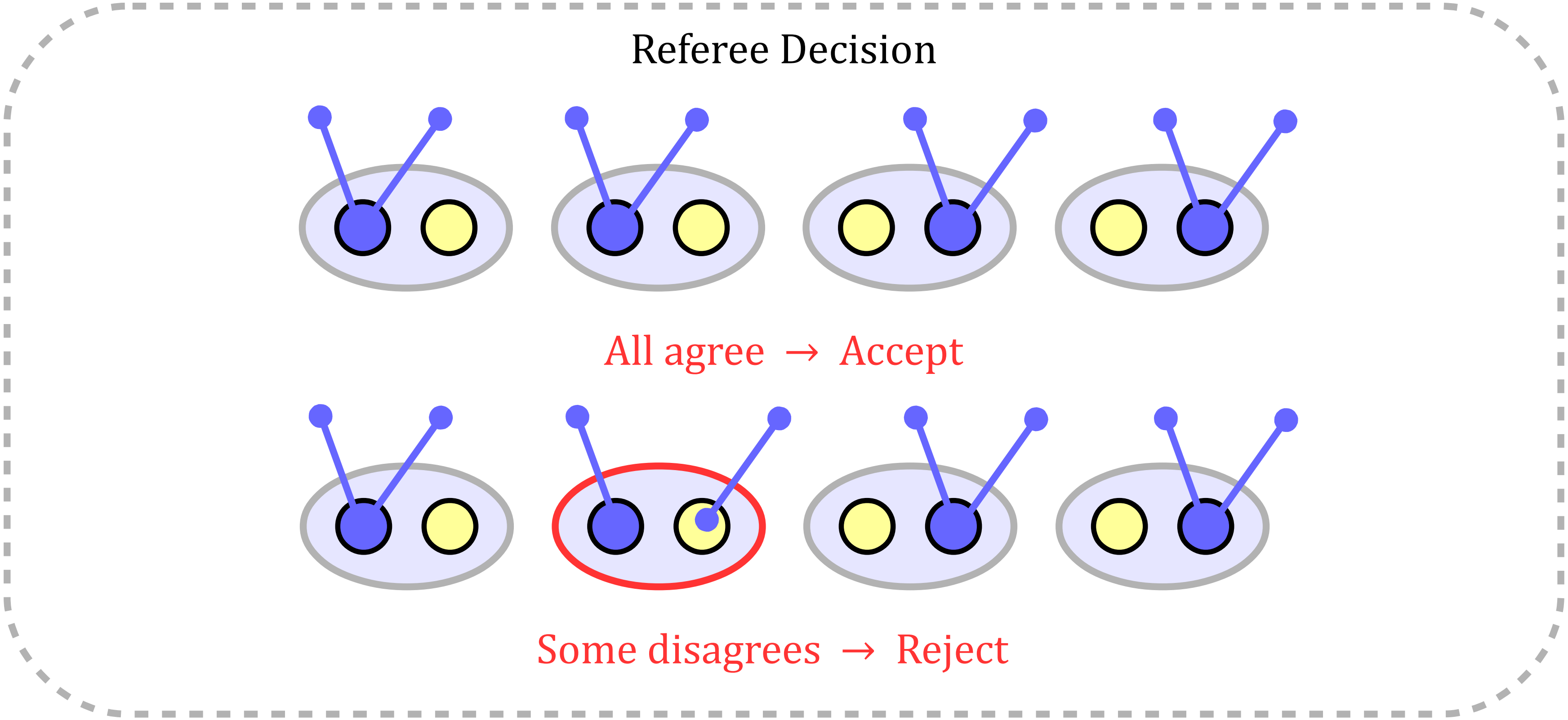


Protocol for k-Clique

(Edge-Vertex Incident Game)

Players: $O(k^2)$ {i,j}-players, each submits one edge between V_i to V_j.

Referee: Check if there exists k vertices (one from each V_i) such that each vertex is incident to k-1 submitted edges.



Use **Error Correcting Codes** to reduce communications.

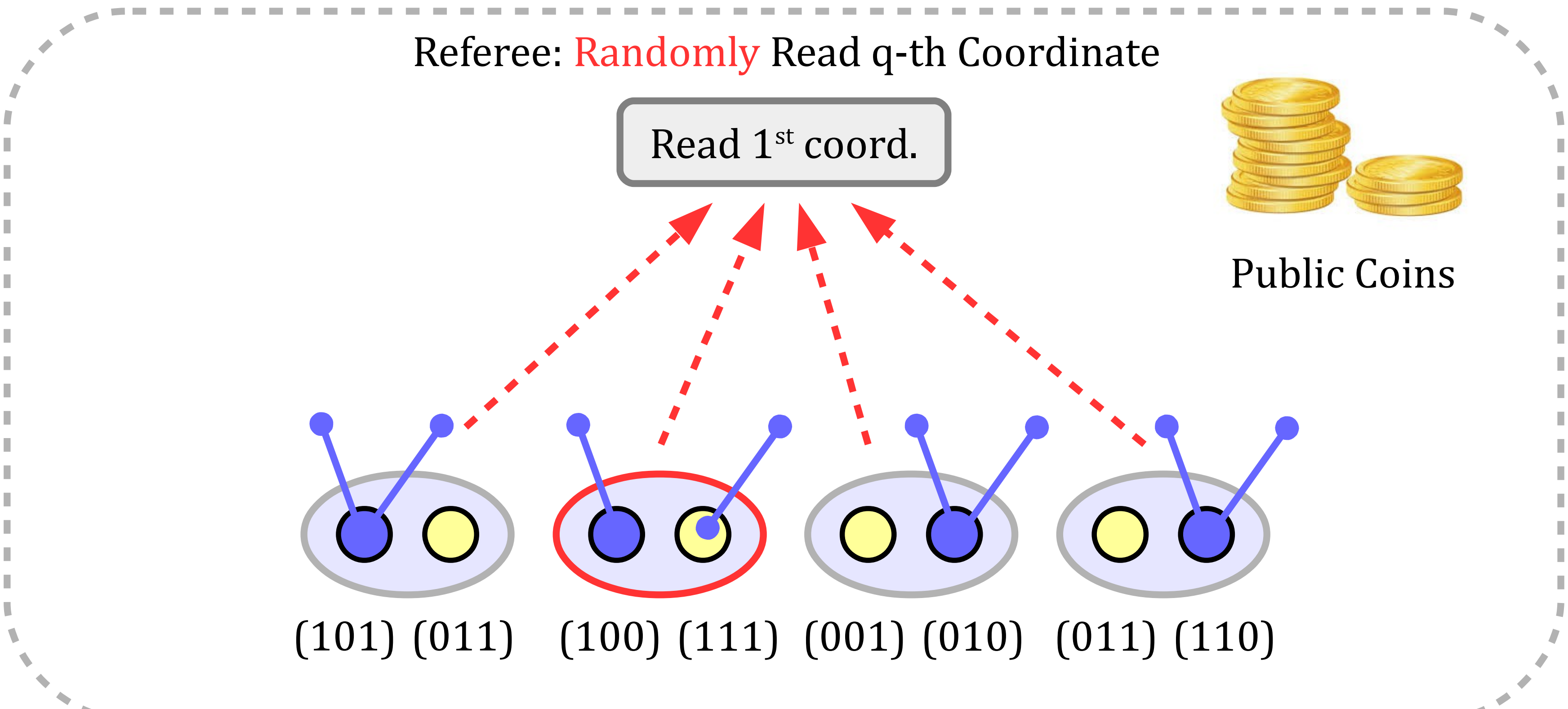
Revised Protocol for k-Clique

(Encode Vertices Using Good Codes)

Codes: Each vertex is encoded with random codes from $[2]^{\log n}$

Players: (i, j)is asked to submit encoded-endpoints of edge V_i - V_j.

Referee: Random an index q. Then check if q-th char of k-1 encoded-edges are consistent.



Use **Error Correcting Codes** to reduce communications.

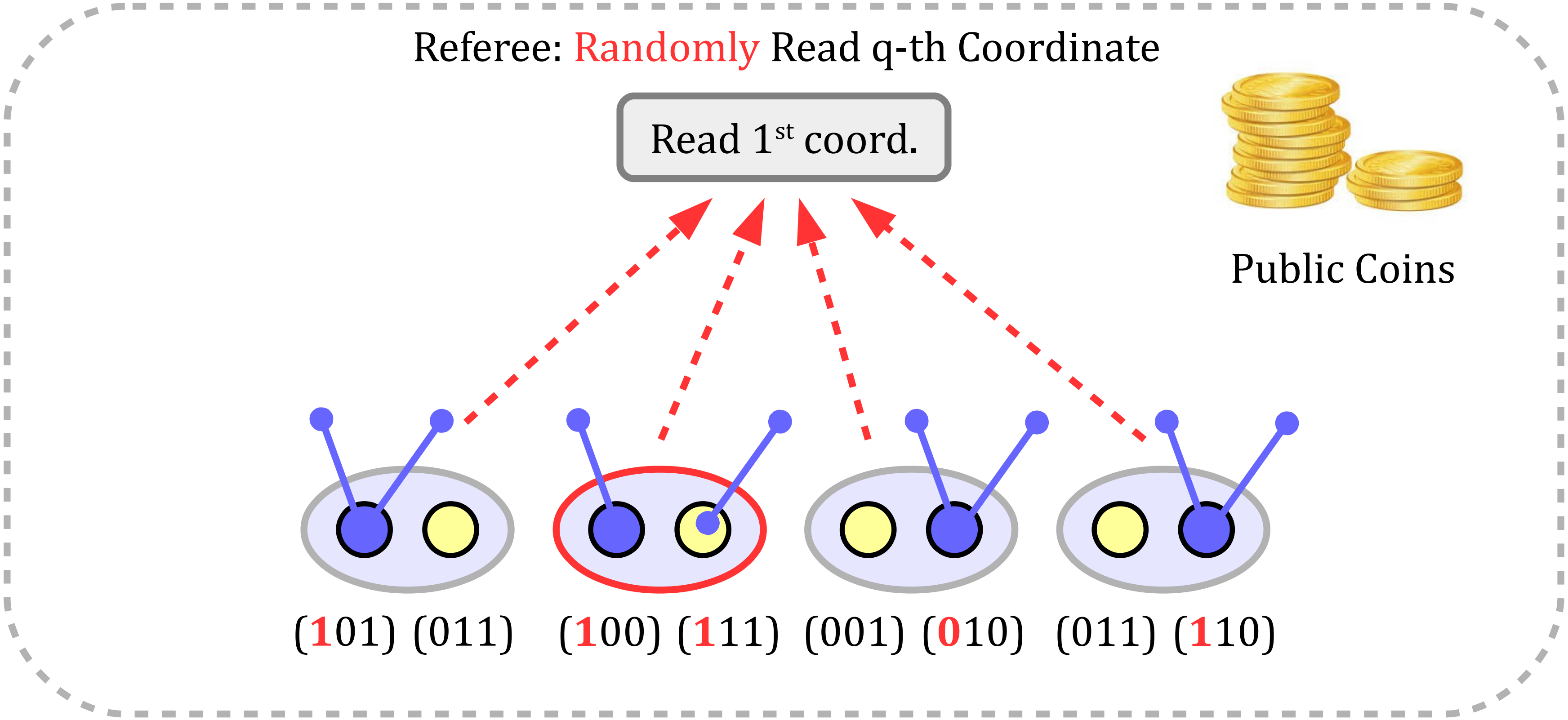
Revised Protocol for k-Clique

(Encode Vertices Using Good Codes)

Codes: Each vertex is encoded with random codes from $[2]^{\log n}$

Players: (i, j)is asked to submit encoded-endpoints of edge V_i - V_j.

Referee: Random an index q. Then check if q-th char of k-1 encoded-edges are consistent.



Use **Error Correcting Codes** to reduce communications.

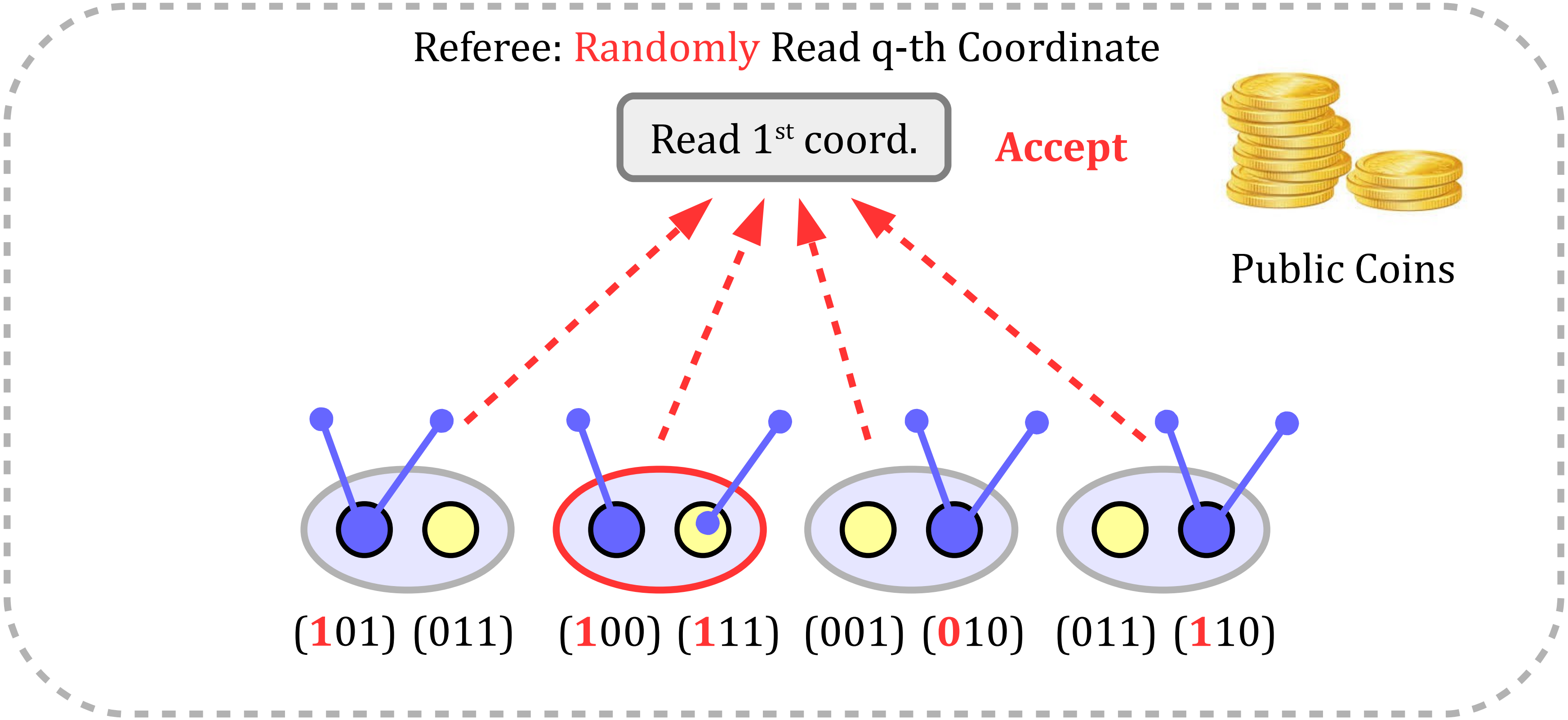
Revised Protocol for k-Clique

(Encode Vertices Using Good Codes)

Codes: Each vertex is encoded with random codes from $[2]^{\log n}$

Players: (i, j)is asked to submit encoded-endpoints of edge V_i - V_j.

Referee: Random an index q. Then check if q-th char of k-1 encoded-edges are consistent.



Use **Error Correcting Codes** to reduce communications.

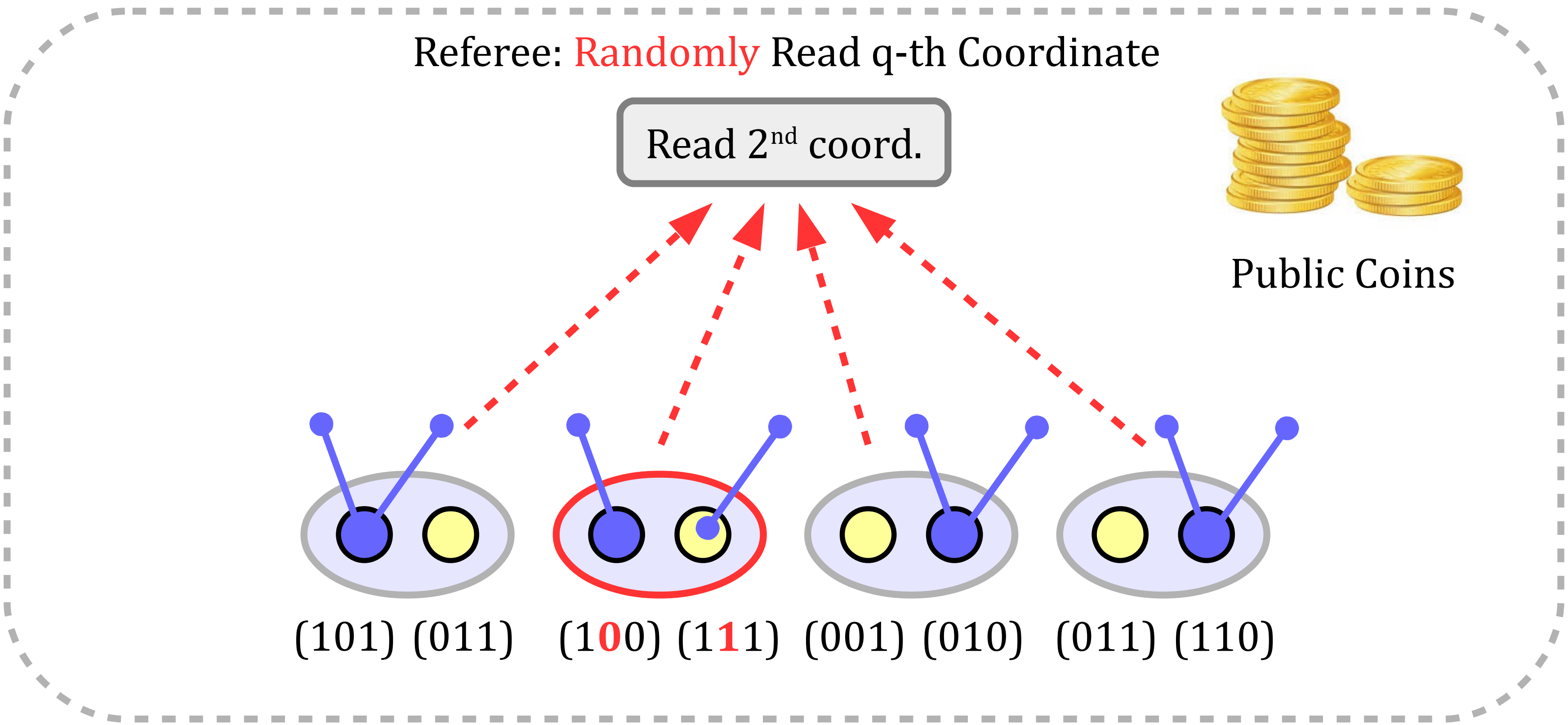
Revised Protocol for k-Clique

(Encode Vertices Using Good Codes)

Codes: Each vertex is encoded with random codes from $[2]^{\log n}$

Players: (i, j)is asked to submit encoded-endpoints of edge V_i - V_j.

Referee: Random an index q. Then check if q-th char of k-1 encoded-edges are consistent.



Use **Error Correcting Codes** to reduce communications.

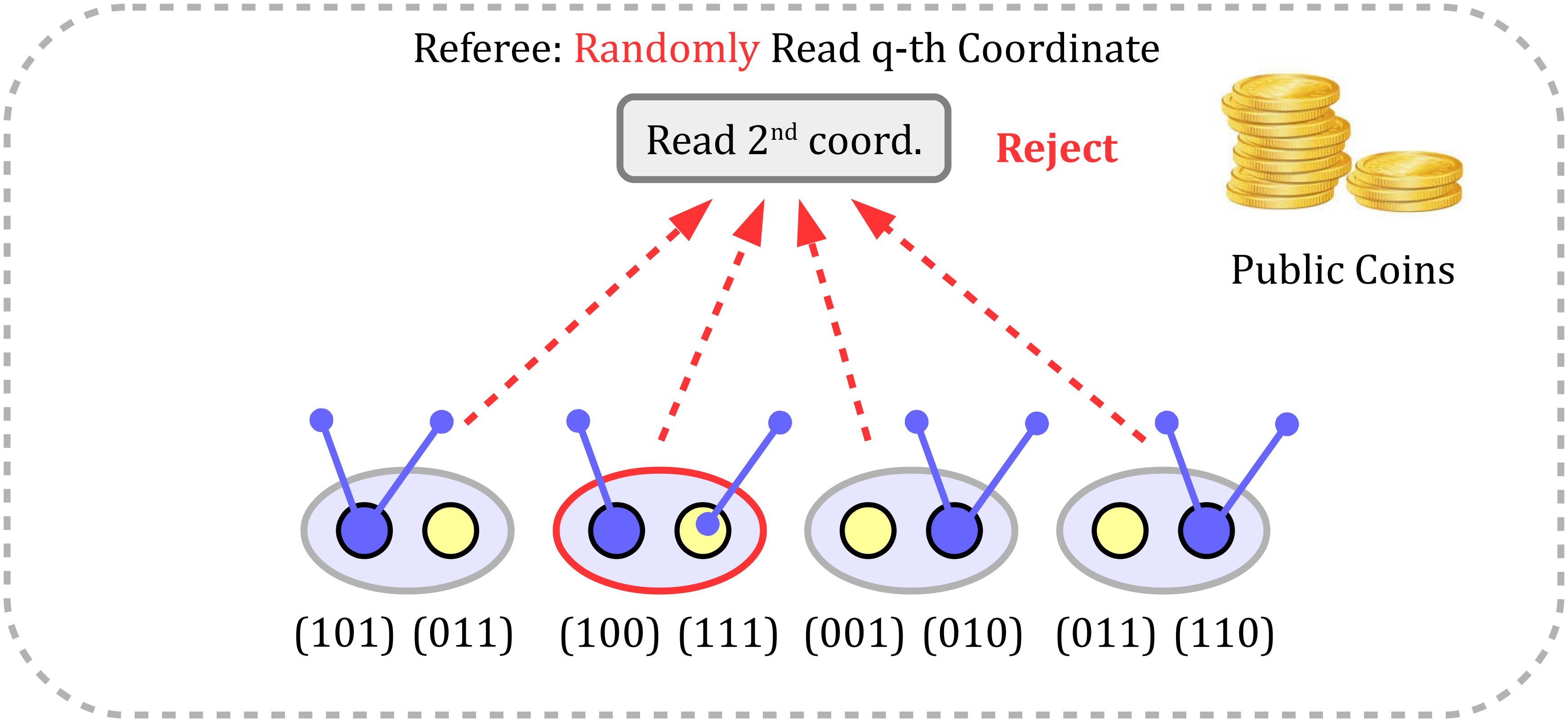
Revised Protocol for k-Clique

(Encode Vertices Using Good Codes)

Codes: Each vertex is encoded with random codes from $[2]^{\log n}$

Players: (i, j)is asked to submit encoded-endpoints of edge V_i - V_j.

Referee: Random an index q. Then check if q-th char of k-1 encoded-edges are consistent.



Use **Error Correcting Codes** to reduce communications.

Conclusion

- Parameterized Inapproximability of k-DS under FPT≠W[1], ETH, SETH, k-SUM.
- Connection between Communication Complexity and Parameterized Inapproximability.
  [SMP-Protocol → Max-Cover → k-DS]

Conclusion

- Parameterized Inapproximability of k-DS under FPT≠W[1], ETH, SETH, k-SUM.
- Connection between Communication Complexity and Parameterized Inapproximability.
  [SMP-Protocol → Max-Cover → k-DS]

Recently, Bingkai Lin gives an alternate proof of our results.
His technique is simpler but is not captured by our framework.

Open Problems

- Prove k-DS inapprox under FPT $\neq$ W[2]
  - k-DS is known to be W[2]-complete.
  - Is it possible that it is W[2]-hard even to approximate k-DS?
- Bypassing Gap-ETH for k-Clique
  - FPT-inapprox for k-Clique has been shown under Gap-ETH.
  - Is it possible to base the FPT-inapprox for k-Clique under ETH, SETH or W[1] $\neq$ FPT?

The End

(the real end of this talk)

Thank you for your attention.

Questions?