Improved Lower Bounds for Approximating Parameterized Nearest Codeword and Related Problems under ETH

Shuangle Li — NJU

Bingkai Lin — NJU

Yuwei Liu — **SJTU**

Contents

Linear Code

A linear code $\mathcal{C} \subseteq \mathbb{F}_p^n$ is a linear space over $\mathbb{F}_p$.

- For basis $\{\vec{v}_1, \cdots, \vec{v}_d\} \subseteq \mathbb{F}_p^n$, $\mathcal{C} = \{\Sigma_{i \in [d]} c_i \vec{v}_i \mid c_1, \cdots, c_d \in \mathbb{F}_p\}$

- Minimum Distance $:= \min_{x \neq y \in \mathcal{C}} |\{i \in [n]: x[i] \neq y[i]\}|$

  $= \min_{\vec{0} \neq x \in \mathcal{C}} |\{i \in [n]: x[i] \neq 0\}|$ (by linearity)

- Num of errors can be corrected $\approx \frac{1}{2}$ Minimum Distance

Minimum Distance Problem

k-Minimum Distance Problem (k-MDP):
Given a linear code $\mathcal{C} \subseteq \mathbb{F}_p^n$, determine whether there exists a non-zero $\vec{x} \in \mathcal{C}$ with $\leq k$ non-zero entries.

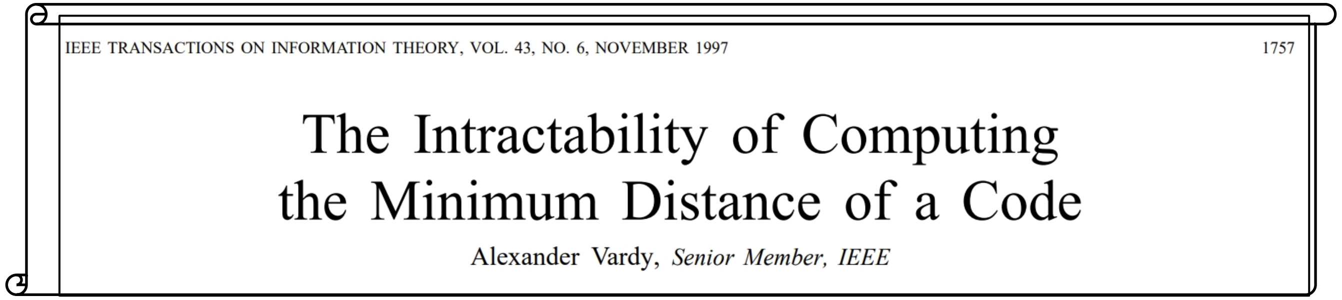
- Related problems:
  - Maximum Likelihood Decoding
  - Nearest Codeword Problem
  - Closest Vector Problem
  - Shortest Vector Problem

Finding the shortest vector in a *lattice*.
- Fundamental problem in post-quantum cryptography.

Minimum Distance Problem

k-Minimum Distance Problem (k-MDP):
Given a linear code $\mathcal{C} \subseteq \mathbb{F}_p^n$, determine whether there exists a non-zero $\vec{x} \in \mathcal{C}$ with $\leq k$ non-zero entries.

- NP-Complete! [cf. Vardy 1997]


IEEE TRANSACTIONS ON INFORMATION THEORY, VOL. 43, NO. 6, NOVEMBER 1997 1757

The Intractability of Computing the Minimum Distance of a Code

Alexander Vardy, *Senior Member, IEEE*

Approximation: γ-Gap-k-MDP Problem

$\boldsymbol{\gamma}$-Gap-$\boldsymbol{k}$-MDP:

Given a linear code $\mathcal{C} \subseteq \mathbb{F}_p^n$, distinguish between:

(YES) Exists a non-zero $\vec{x} \in \mathcal{C}$ with $\leq k$ non-zero entries.

(NO) All non-zero $\vec{x} \in \mathcal{C}$ have $> \gamma k$ non-zero entries.

- NP-hard for any constant $\gamma > 1$! cf. [DMS03][CW12][AK14][Mic14]
- $O(n \,/\, \log n)$-approximable in polynomial time for k-NCP, a variant of MDP [APY09]

Parameterized k-MDP

- Can be solved by brute-force in time $n^{O(k)}$.

- Question from parameterized complexity:

Does k-MDP have an $\boldsymbol{f(k) \cdot n^{O(1)}}$ **algorithm**?

FPT-algorithm

- (Combinatoric view) k-MDP over binary field: k-Even Set

Parameterized k-MDP

A *long standing* open problem in parameterized complexity…

Open problems for FPT School 2014

Marek Cygan, Fedor Fomin, Bart M.P. Jansen, Łukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michał Pilipczuk, Saket Saurabh

Bedlewo, 17-22 August 2014

Last update: September 1, 2014.
This list contains a compilation of open problems from recent Dagstuhl Seminars or Workshop on Kernels, as well as some problems mentioned in some recent papers. We tried to rank them (with stars) depending on their importance and possible hardness, but please do not take the ratings too seriously.
Change log:

21 Aug 2014	open problem list made public
1 Sep 2014	Knapsack problem reported to be solved

Even Set aka Minimum Codeword ($\star\star\star$)

Long standing; appeared, e.g., in [37].

In the Even Set problem the input consists of a family $\mathcal{F}$ of subset of a universe U and an integer k; the question is to find a nonempty set $A \subseteq U$ of size at most k such that $|A \cap F|$ is even for every $F \in \mathcal{F}$. Alternatively, the question can be stated as finding a non-zero codeword of Hamming weigth at most k in a linear code over $\mathbb{F}_2$. The question of parameterized complexity of this problem, parameterized by k, remains open.

Note that if we require the set A to be of size exactly k, or we require the intersections to be odd, the problem becomes W[1]-hard.

Framework for refuting Turing kernels ($\star\star\star$)

Long-standing, appeared, e.g., in [37, 27].

Parameterized Complexity of k-MDP

Parameterized Inapproximability Hypothesis[LRSZ20]:
No $f(k)n^{O(1)}$ time $(1-\epsilon)$-approximation algorithm for 2CSP with k variables and n-sized alphabet.

[Bhattacharyya, Ghoshal, Karthik, Manurangsi, ICALP'18]:

No $f(k)n^{O(1)}$ time algorithm for k-MDP, under PIH.

The reduction:

Gap-2CSP $\longrightarrow$ Gap-k-NCP $\longrightarrow$ Gap-k-MDP

[Bonnet, Egri, Lin, Marx, arXiv pre-print, 2018]:

No $f(k)n^{O(1)}$ time algorithm for k-NCP, under W[1]≠FPT.

The reduction:

k-Clique $\longrightarrow$ One-sided Gap k-Biclique $\longrightarrow$ Gap-k-NCP

Parameterized Complexity of k-MDP

Combining together,

[Bhattacharyya, Bonnet, Egri, Ghoshal, Karthik, Lin, Manurangsi, Marx, J.ACM'21]:
No $f(k)n^{O(1)}$ time algorithm for Gap-k-MDP over $\mathbb{F}_2$, under W[1]$\neq$FPT.

Extending to all $\mathbb{F}_p$,

[Bennett, Cheraghchi, Guruswami, Ribeiro, STOC'23]:
No $f(k)n^{O(1)}$ time algorithm for Gap-k-MDP **over all $\mathbb{F}_p$**, under W[1]$\neq$FPT.

Parameter Blow-up in Reduction From [BBE+21]

k-Clique → (s, h, l)-One-Sided Gap Biclique, with $h = (k+6)! \cdot (\gamma k^2)^{k^2}$, $l = (k+1)!$, $s = k(k-1)/2$

(s, h, l)-One-Sided Gap Biclique → k_1-Gap-Linear Dependent Set, with $k_1 = hs = \mathbf{2^{\Omega(k^2 \log k)}}$

k_1-Gap-Linear Dependent Set → $(3-\varepsilon)$-Gap-k_2-MLD, with $k_2 = k_1 = \mathbf{2^{\Omega(k^2 \log k)}}$

$(3-\varepsilon)$-Gap-k_2-MLD → γ-Gap-k_3-MLD, with $k_3 = poly(k_2)$

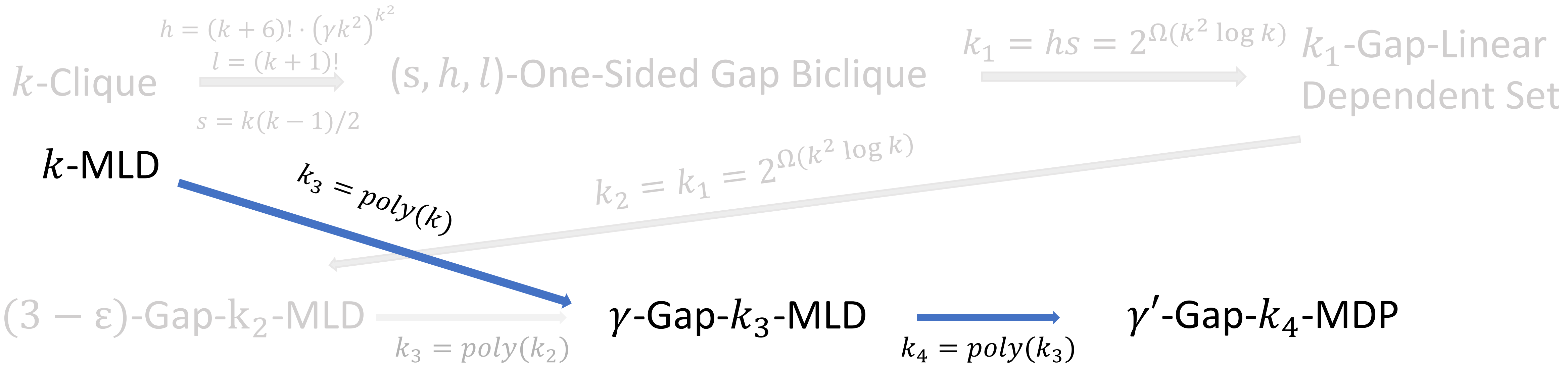
γ-Gap-k_3-MLD → γ'-Gap-k_4-MDP, with $k_4 = poly(k_3)$

Huge parameter blow-up in reduction from k-Clique to Gap-k-MLD!

k-Clique: $n^{\Omega(k)}$ time Lower bound → Gap-k-MDP: Only $n^{\Omega((\log k)^{1/c})}$ time Lower bound

(under ETH)

Our Improvement



Polynomial parameter growth from k-MLD to Gap-k-MLD!

k-MLD: $n^{\Omega(k)}$ time Lower bound → Gap-k-MDP: $n^{\Omega(k^{\epsilon})}$ time Lower bound

(under ETH)

Summary on Gap-k-MDP Results

Work	Assumption	Time Lower Bound	Field	Approx. Ratio
BGKM, ICALP'18	PIH	No FPT Algorithm	Binary	Constant
BBE+, J.ACM'21	W[1]≠FPT	No FPT Algorithm	Binary	
	ETH	$n^{(\log k)^{1/c}}$		
BCGR, STOC'23	W[1]≠FPT	No FPT Algorithm	All $p > 1$	
	ETH	$n^{(\log k)^{1/c}}$		
This Work	**ETH**	$\boldsymbol{n^{k^{\Omega(1)}}}$	**All $p > 1$**	
Manurangsi, SODA'20	*Gap*-ETH	$n^{\Omega(k)}$ **(Tight!)**	Binary	

A step toward closing this gap!

Results for Other Related Problems

Under ETH:

Problem	Inapprox. Ratio	Time Lower Bound	Constant Dependency	Specification
k-NCP (k-MLD)	Any $\gamma \in (1, \frac{3}{2})$	$f(k)n^{\Omega(\sqrt{k/\log k})}$		Any finite field $\mathbb{F}_p$
	Any $\gamma > 1$	$f(k)n^{\Omega(k^{\epsilon})}$	$\epsilon = \frac{1}{poly \log \gamma}$	Any finite field $\mathbb{F}_p$
k-CVP	Any $\gamma > 1$	$f(k)n^{\Omega(k^{\epsilon})}$	$\epsilon = \Theta(\frac{1}{poly \log \gamma})$	Any ℓ_p-norm, $p \geq 1$
k-SVP	Any $\gamma > 1$	$f(k)n^{\Omega(k^{\epsilon})}$	$\epsilon = \epsilon(p, \gamma)$	Any ℓ_p-norm, $p > 1$
	Any $\gamma \in [1,2)$	$f(k)n^{\Omega(k^{\epsilon})}$	$\epsilon = \epsilon(p, \gamma)$	Any ℓ_p-norm, $p \geq 1$

Contents

- Introduction
- **Technical overview**
- Future direction

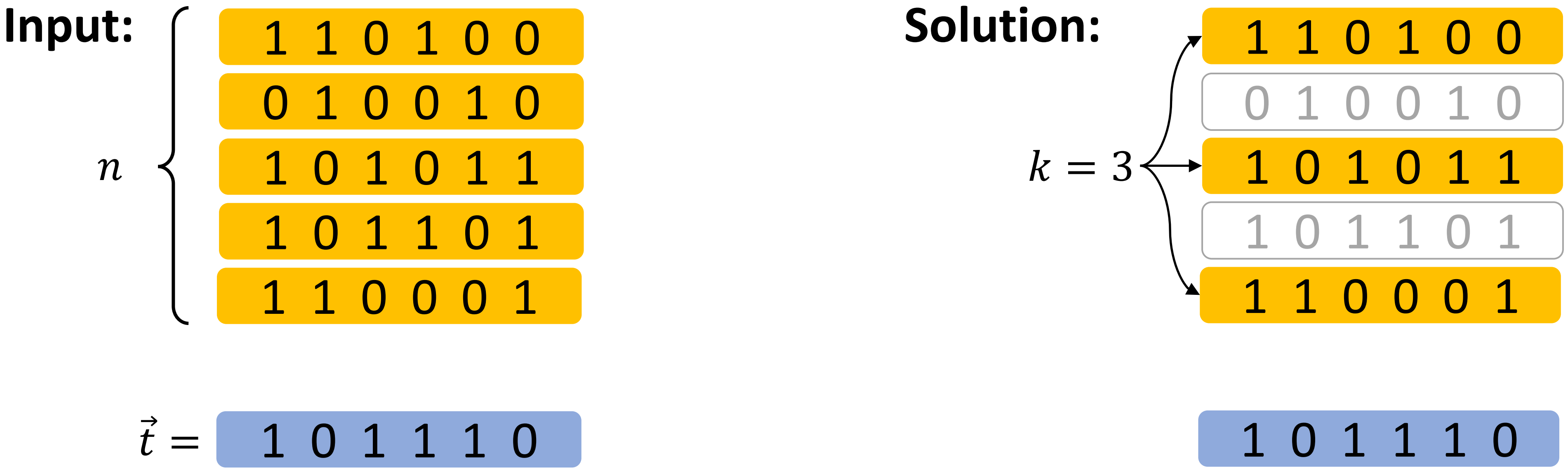
k-Maximum Likelihood Decoding (k-MLD)

a.k.a. k-NCP

Input: n vectors $\vec{v}_1, \vec{v}_2 \dots, \vec{v}_n \in \mathbb{F}_p^d$ and a target vector $\vec{t} \in \mathbb{F}_p^d$

Goal: find k vectors $\vec{v}_{i_1}, \vec{v}_{i_2}, \dots, \vec{v}_{i_k}$ s.t.

$$\vec{v}_{i_1} + \vec{v}_{i_2} + \cdots + \vec{v}_{i_k} = \vec{t}$$

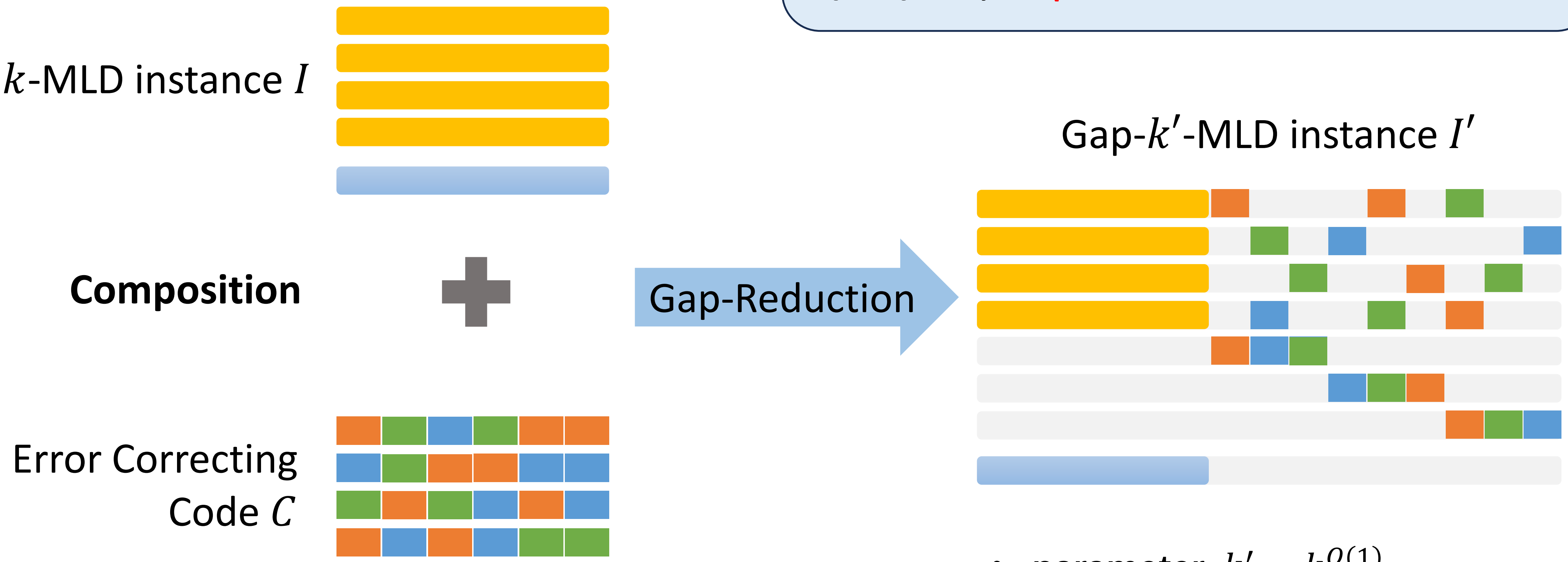


Main Contribution

γ-Gap-k-MLD is to distinguish between:

(YES) there are k vectors adding to $\vec{t}$

(NO) any $\leq \gamma k$ vectors don't add to $\vec{t}$

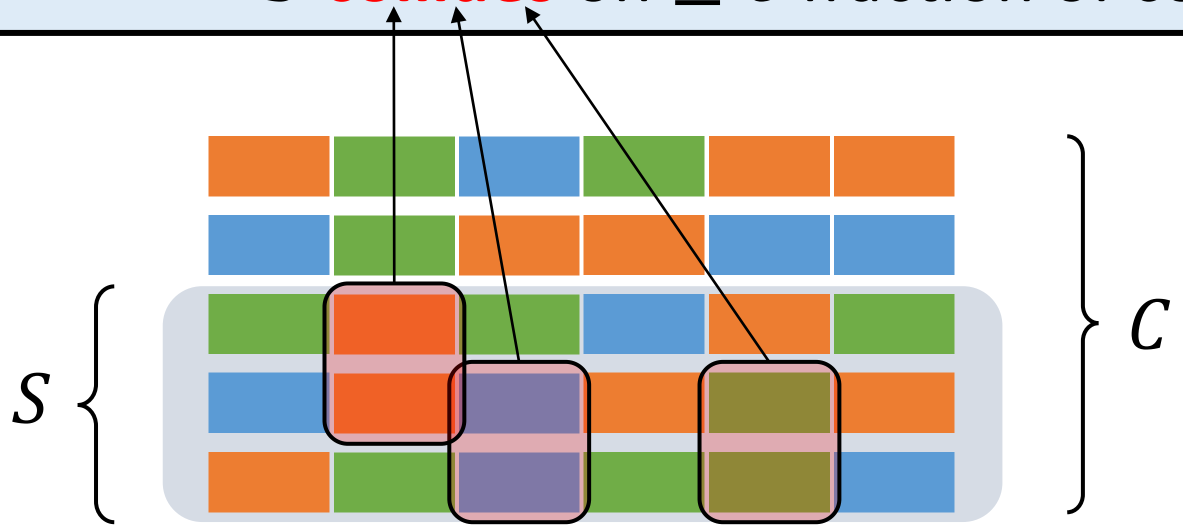


- **parameter** $k' = k^{O(1)}$
- **gap** $\gamma = 3/2 - \varepsilon$

Tool: Collision Number

An error correcting code C has **ε-collision number h** if for any S that is a collection of codewords of C,

S **collides** on $\geq \varepsilon$ fraction of coordinates $\longrightarrow$ $|S| \geq h$

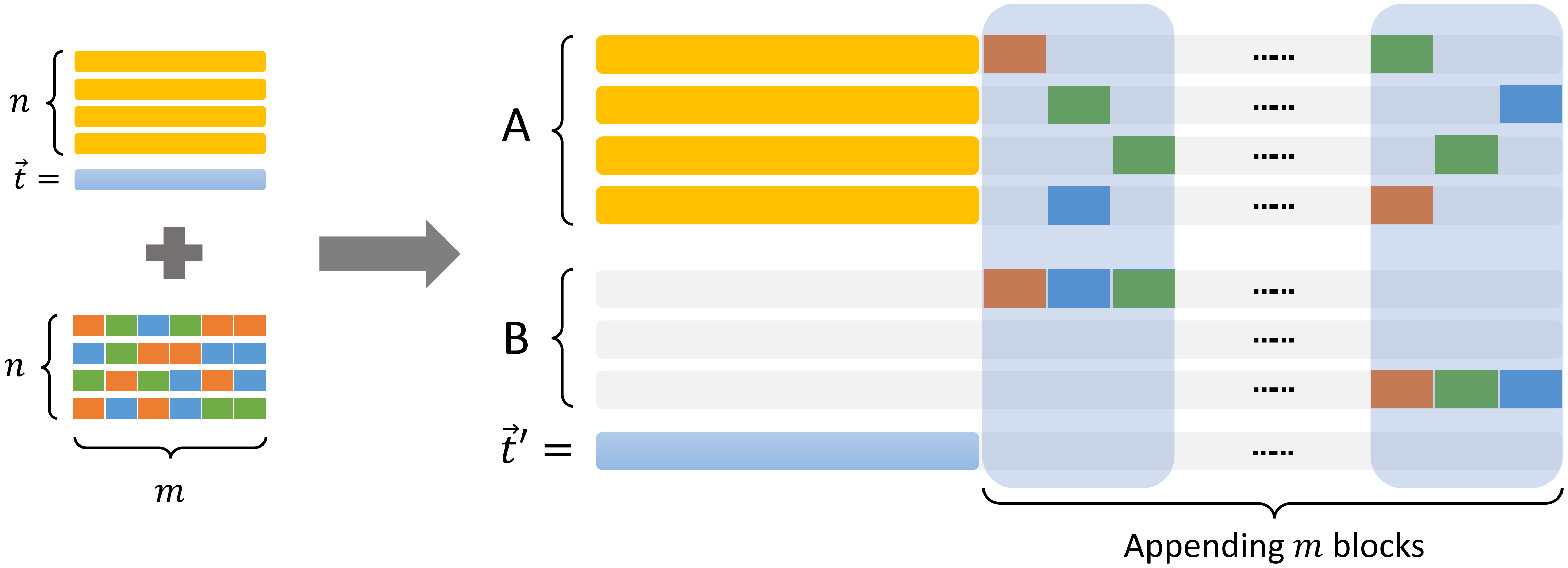


S collides on i-th coordinate if
$\exists x, y \in S$ s.t. $x[i] = y[i]$

Gap Reduction

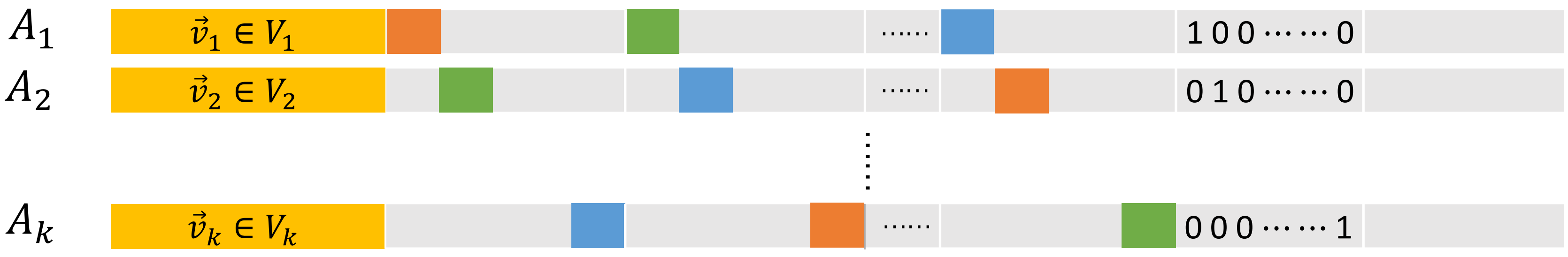
1. Associate each vector with an unique codeword and construct **vector set A**;

2. Construct **vector set B** to force collision to happen on many coordinates if $k+1$ vectors needed.

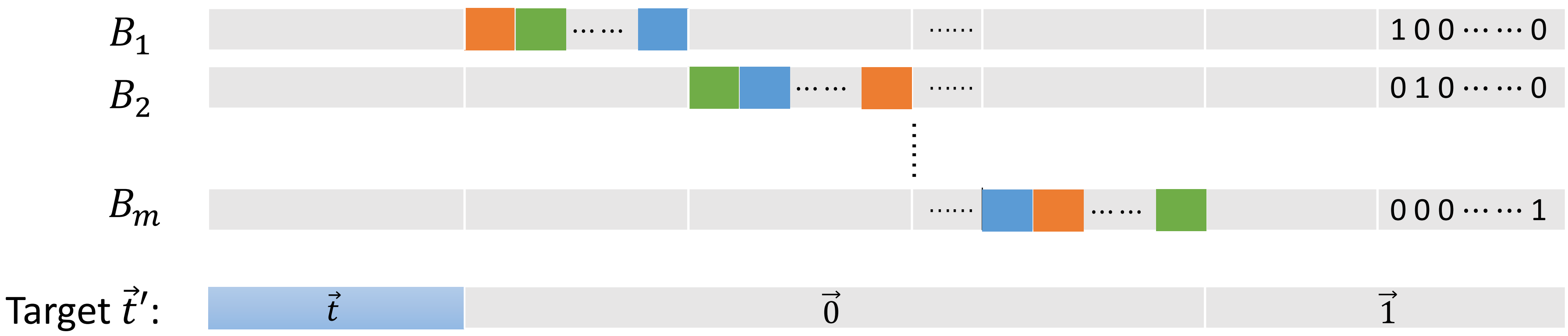


Our Construction

For each $\vec{v}_i \in V_i$, associate it with an ECC codeword, construct a vector like:

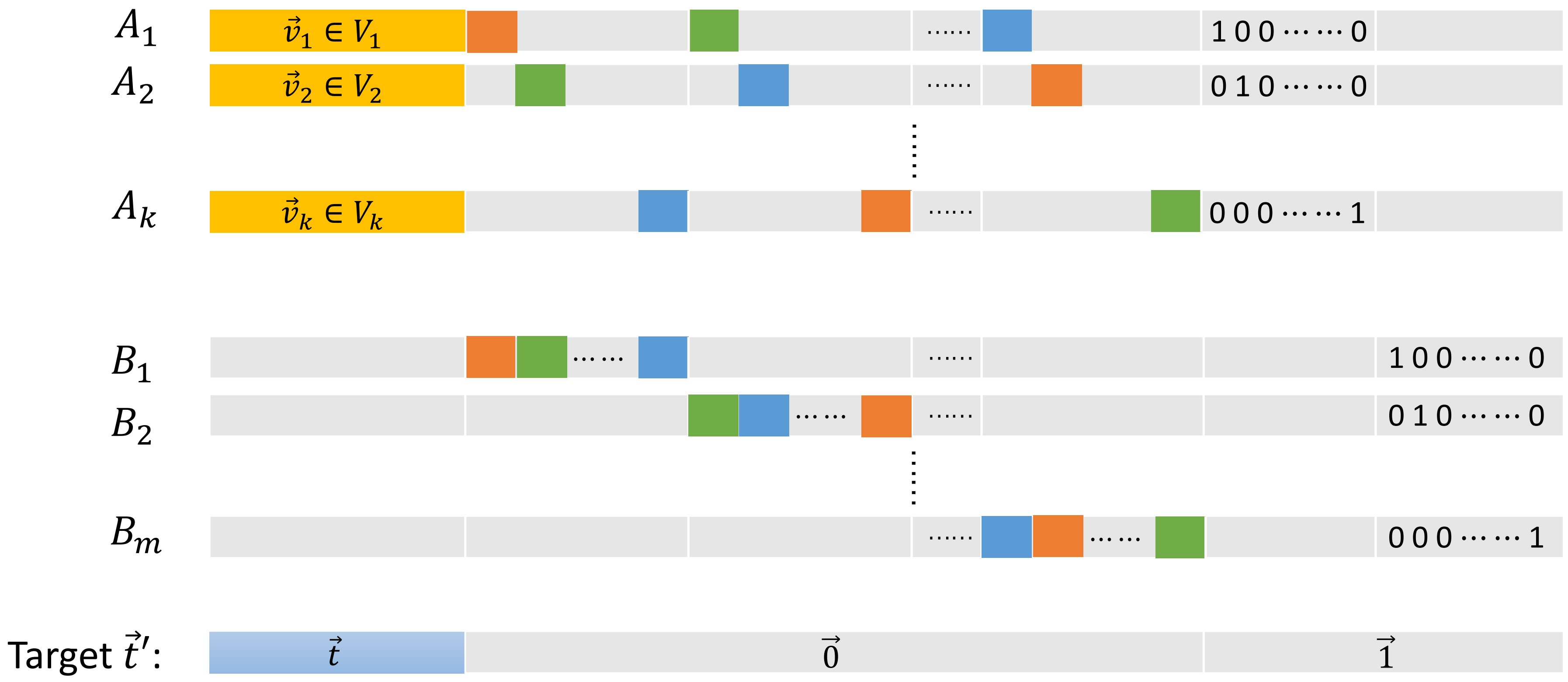


For each entry $j \in [m]$, enumerate all k-tuples, to "guess" contents in A, construct a vector like:



Completeness

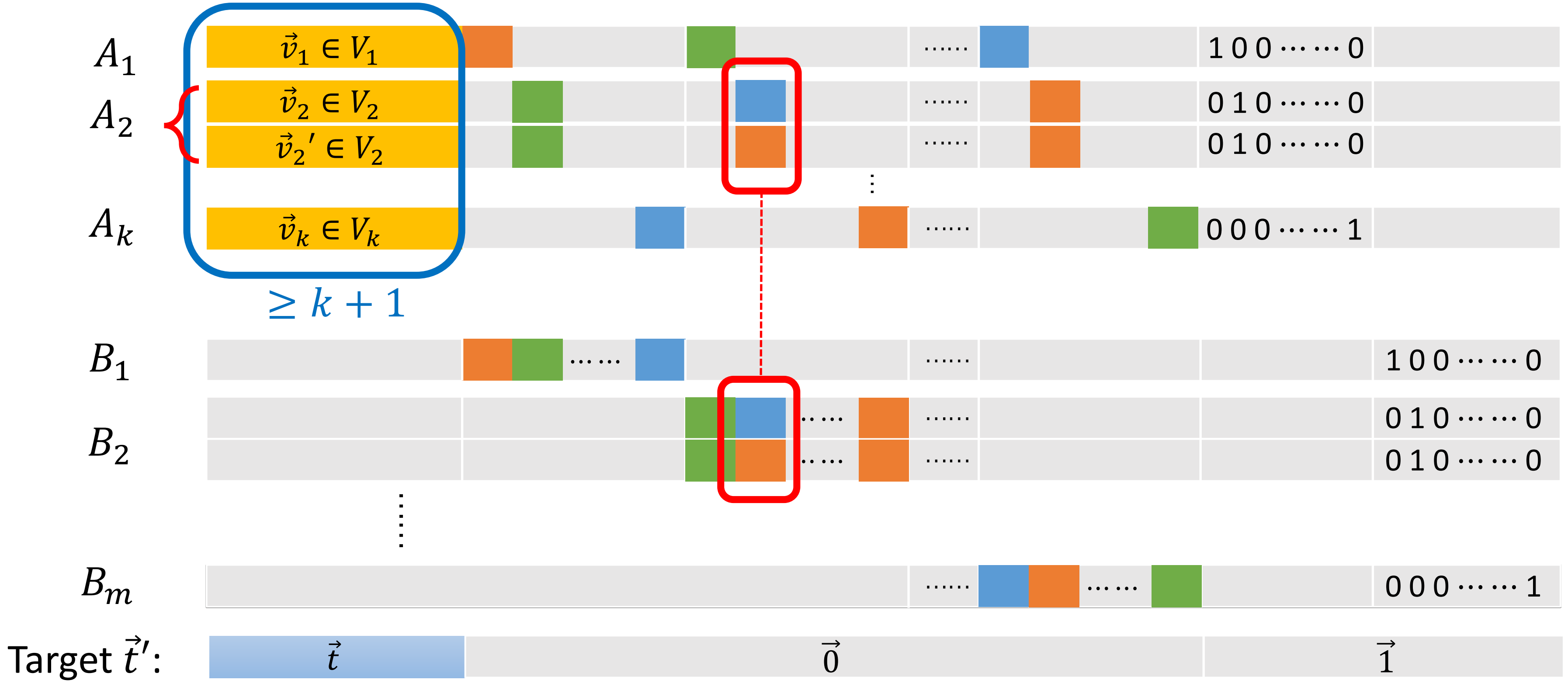
If $\vec{v}_1 + \cdots + \vec{v}_k = \vec{t}$, then the corresponding $k + m$ new vectors sum to $\vec{t}'$.



Soundness

If summing to $\vec{t}$ requires at least $k+1$ vectors, then for each $\boldsymbol{B_i}$

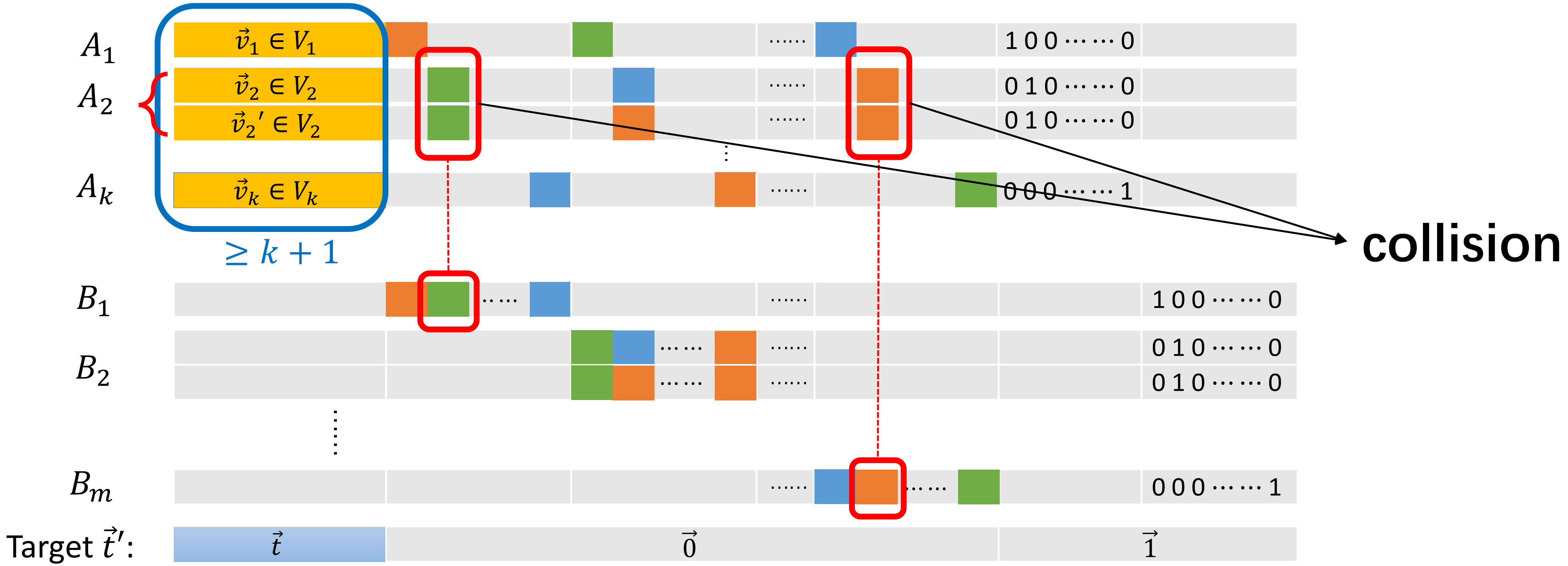
➢ **Either choose ≥ 2 vectors (e.g., $\boldsymbol{B_2}$) cancelling different encoding,**



Soundness

If summing to $\vec{t}$ requires at least $k+1$ vectors, then for each $\boldsymbol{B_i}$

- ➢ Either choose ≥ 2 vectors cancelling different encoding,
- ➢ **or choose 1 vector (e.g., $\boldsymbol{B_1}, \boldsymbol{B_m}$), indicating a collision in this entry.**



Soundness

If summing to $\vec{t}$ requires at least $k+1$ vectors, then for each $\boldsymbol{B_i}$

- Either choose ≥ 2 vectors cancelling different encoding,
- or choose 1 vector indicating a collision in this entry.

Recall:

An error correcting code C has $\boldsymbol{\varepsilon}$-collision number h if for any S that is a collection of codewords of C,

S collides on $\geq \varepsilon$ fraction of coordinates $\longrightarrow$ $|S| \geq h$

Either $\geq \varepsilon m$ entries have collision $\longrightarrow$ $\geq h$ vectors in A

or $< \varepsilon m$ entries have collision $\longrightarrow$ $\geq (2-\varepsilon)m$ vectors in B

Summary: If summing to $\vec{t}$ requires at least $k+1$ vectors, then summing to $\vec{t}'$ need $\min\{h+m, k+(2-\varepsilon)m\}$ vectors.

On Choosing Good Code

We need a code C with:

- n codewords
- alphabet size $\Sigma = n^{O(1/k)}$
- ε-collision number $\geq ck$
- block length $m = k^{O(1)}$

the smaller, the better

Our best construction

- Reed-Solomon code $\rightarrow m = O(k^3)$, deterministic
- random code $\rightarrow m = O(k^2 \log k)$, randomized

Our Result

- Direct gap-creating self-reduction for k-MLD:

Theorem (informal). A reduction k-MLD to γ-Gap-k'-MLD with

(Parameter) $k' = O(k^3)$ (deterministic) or $k' = O(k^2 \log k)$ (randomized)

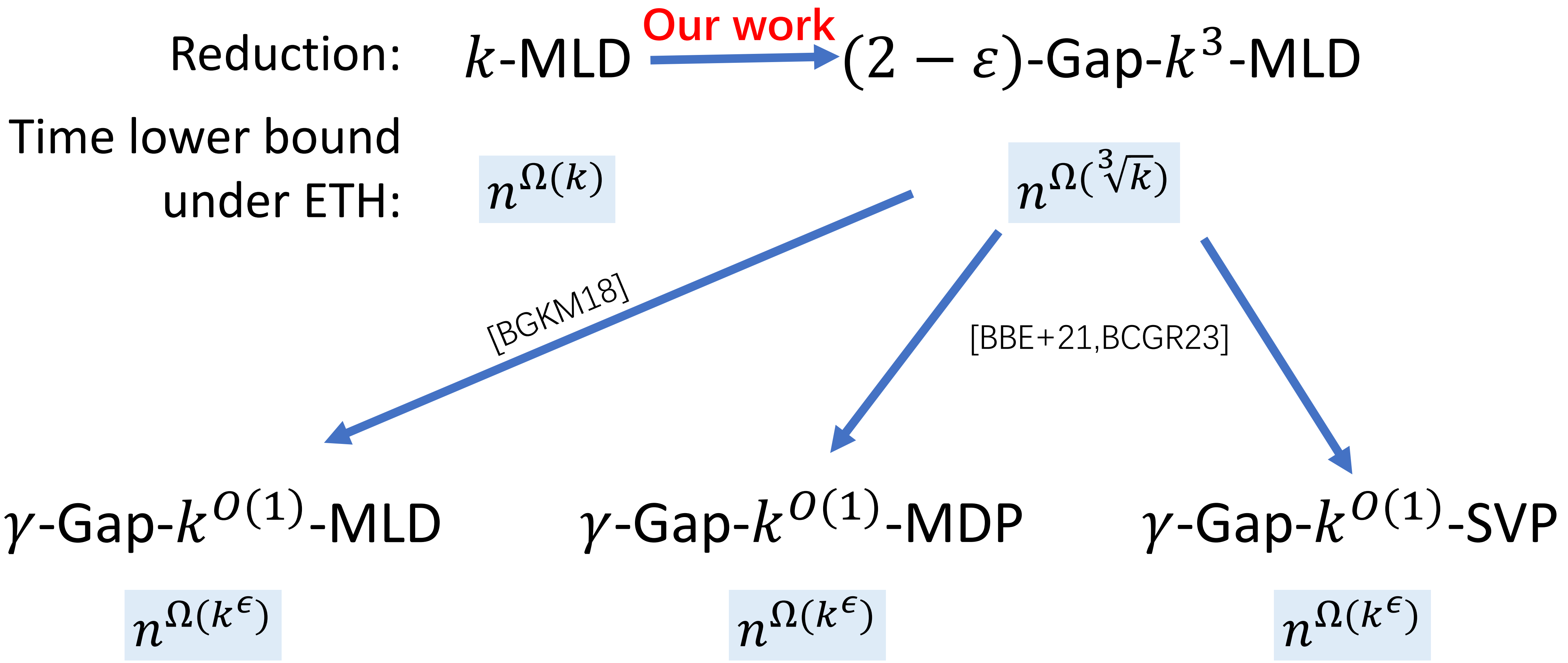
(Gap) $\gamma \to 3/2$.

- Consequence: tighter time lower bound

Corollary (informal). Assuming randomized ETH, no $f(k)n^{o\left(\sqrt{k/\log k}\right)}$ time algorithm for γ-Gap-k-MLD.

Recall: $n^{(\log k)^{1/c}}$ in previous work.

Consequences



Contents

- Introduction
- Technical overview
- **Future direction**

Future Direction

- Time lower bound

Our bound: $n^{\Omega(k^{\epsilon})}$

Brute-force algorithm: $n^{O(k)}$

Closing this gap?

$n^{\Omega(k)}$ lower bound for constant approximating k-MDP under ETH?

or

$n^{o(k)}$ algorithm for constant approximating k-MDP?

- Approximation ratio

No constant approximation for k-NCP

Closing this gap?

$O(n/\log n)$-approximation algorithm for k-NCP [APY09]

Hardness on $\omega(1)$-approximating k-NCP?

or

Algorithm for $o(n/\log n)$ approximating k-NCP?

Thank you!