Online Submodular Matroid Maximization with Free Disposal

Zhihao Tang

The University of Hong Kong

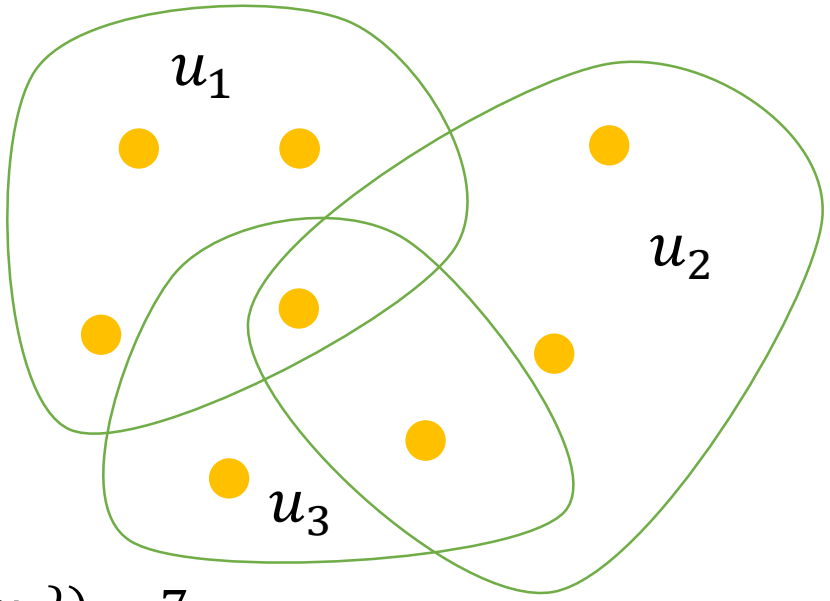
Joint work with Hubert Chan, Zhiyi Huang, Shaofeng Jiang and Ning Kang

Submodular functions

Ω: Set of elements. $f: 2^{\Omega} \to R_+$.

- **Submodularity**: for all $X \subseteq Y, u \in \Omega \backslash Y$,
$$f(X+u) - f(X) =: f(u|X) \geq f(u|Y)$$
- **Monotonicity**: for all $X \subseteq Y$,
$$f(X) \leq f(Y)$$

Example: Coverage functions



$f(\{u_1, u_2\}) = 7$
$f(\{u_2, u_3\} = 5$

Submodular functions

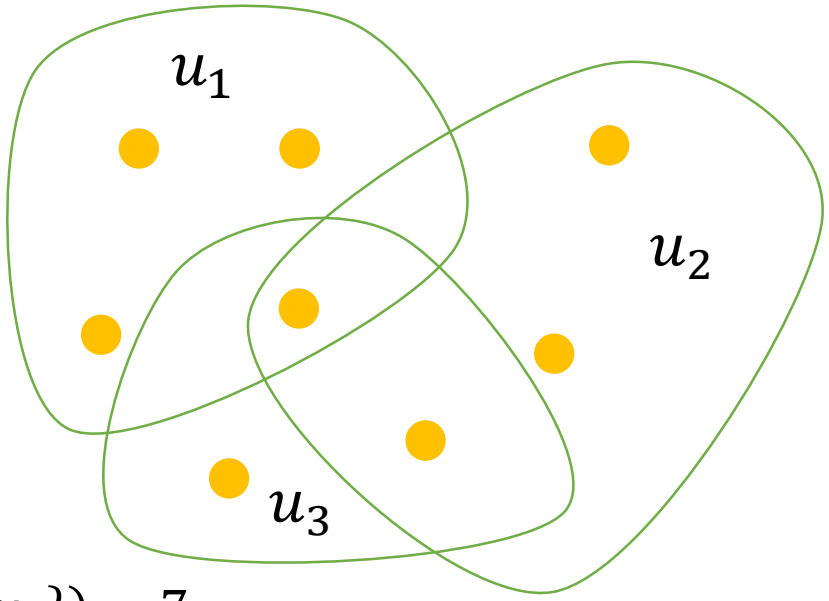
Ω: Set of elements. $f: 2^{\Omega} \to R_+$.

- **Submodularity**: for all $X \subseteq Y, u \in \Omega \backslash Y$,
$$f(X+u) - f(X) =: f(u|X) \geq f(u|Y)$$
- **Monotonicity**: for all $X \subseteq Y$,
$$f(X) \leq f(Y)$$

Example: Coverage functions

Other examples:

- Cut functions (non-monotone)
- Influence functions



$f(\{u_1, u_2\}) = 7$
$f(\{u_2, u_3\} = 5$

Submodular Maximization

- Given:
  - Ω: Set of elements
  - f: Submodular function **monotone for this talk**
  - $I \subseteq 2^{\Omega}$: Collection of feasible subsets of Ω
- Goal:
  - Find $S \in I$ that (approximately) maximize $f(S)$
  - In this work, we assume I forms a **matroid**

Submodular Maximization

- Given:
  - Ω: Set of elements
  - f: Submodular function **monotone for this talk**
  - $I \subseteq 2^{\Omega}$: Collection of feasible subsets of Ω
- Goal:
  - Find $S \in I$ that (approximately) maximize $f(S)$
  - In this work, we assume I forms a **matroid**
  - **Example:** k-uniform matroid

    I includes all subsets with cardinality at most k

    partition matroid

    direct sum of uniform matroids

Online Submodular Maximization

- Ω initially unknown
- Elements in Ω arrives one by one in arbitrary order
- Value Oracle for f: can query subsets of arrived elements
- Maintain $S \in I$
- When $u \in \Omega$ comes, algorithm decides:
  - Add u in S?
  - Remove elements from S? Maintain feasibility, **Free Disposal**
- Discarded elements cannot be retrieved back
- **Benchmark:** (offline) optimal in hindsight $\mathrm{OPT} \coloneqq \max_{S \in I} f(S)$

What's known?

Matroid	Objective f	Algorithm	Hardness
k-uniform	Monotone	4 [Buchbinder et al.]	2 [Buchbinder et al.]
	General	11.2 [Buchbinder et al.]	2.28 [Buchbinder et al.]
General	Monotone	4 [Chakrabarti, Kale] [Chekuri et al.]	2 [Buchbinder et al.]
	General	16 [Chakrabarti, Kale] [Chekuri et al.]	2.28 [Buchbinder et al.]

k-uniform: all subsets with cardinality at most k

Our results

Matroid	Objective f	Algorithm	Hardness
k-uniform	Monotone	**3.15**	2 [Buchbinder et al.]
	General	**8.63**	2.28 [Buchbinder et al.]
Partition	Monotone	**3.15**	**2.61**
	General	16 [Chakrabarti, Kale] [Chekuri et al.]	**2.61**

Partition: direct sum of uniform matroids

$\Omega = \Omega_1 \cup \cdots \cup \Omega_m$, pick $\leq k_j$ elements from Ω_j

What goes wrong with existing approaches?

[Buchbinder et al.]

- **Algorithm:** Measure ‘value’ of u by $f(u|S)$
  - Add u to S if $f(u|S)$ is ‘large enough’, i.e.,

$$f(u|S) \geq \frac{2}{k} f(S) \qquad \ldots(*)$$

  - Discard least ‘valuable’ elements u' from S if needed

[Buchbinder et al.]

- **Algorithm:** Measure 'value' of u by $f(u|S)$
  - Add u to S if $f(u|S)$ is 'large enough', i.e.,

$$f(u|S) \geq \frac{2}{k} f(S) \qquad \ldots(*)$$

  - Discard least 'valuable' elements u' from S if needed
- **Analysis:**
  - A: set of elements **ever** chosen
  - $f(OPT) \leq f(OPT \cup A)$

$$\leq f(A) + \sum_{u \in OPT - A} f(u|A)$$

[Buchbinder et al.]

- **Algorithm:** Measure 'value' of u by $f(u|S)$
  - Add u to S if $f(u|S)$ is 'large enough', i.e.,

$$f(u|S) \geq \frac{2}{k} f(S) \qquad \ldots(*)$$

  - Discard least 'valuable' elements u' from S if needed
- **Analysis:**
  - A: set of elements **ever** chosen
  - $f(OPT) \leq f(OPT \cup A)$

$$\leq f(A) + \sum_{u \in OPT - A} f(u|A)$$

$\leq 2f(S)$
Also due to (*)

[Buchbinder et al.]

- **Algorithm:** Measure 'value' of u by $f(u|S)$
  - Add u to S if $f(u|S)$ is 'large enough', i.e.,

$$f(u|S) \geq \frac{2}{k} f(S) \qquad \ldots(*)$$

  - Discard least 'valuable' elements u' from S if needed
- **Analysis:**
  - A: set of elements **ever** chosen
  - $f(OPT) \leq f(OPT \cup A)$

$$\leq f(A) + \sum_{u \in OPT - A} f(u|A)$$

$\leq 2f(S)$
Also due to (*)

$\leq k \cdot \frac{2}{k} f(S) \leq 2f(S)$
Due to (*)

[Chakrabarti, Kale] and [Chekuri et al.]

$A(u)$: Set of elements ever chosen when u comes

- **Algorithm:** Measure 'value' of u by $f(u|A(u))$
  - Add u and discard u' if

$$f(u|A(u)) \geq 2 \cdot f(u'|A(u')) \qquad \ldots(*)$$

[Chakrabarti, Kale] and [Chekuri et al.]

$A(u)$: Set of elements ever chosen when u comes

- **Algorithm:** Measure 'value' of u by $f(u|A(u))$
  - Add u and discard u' if

$$f(u|A(u)) \geq 2 \cdot f(u'|A(u')) \qquad \ldots(*)$$

- **Analysis:**
  - $f(OPT) \leq f(OPT \cup A)$

$$\leq f(A) + \sum_{u \in OPT - A} f(u|A)$$

[Chakrabarti, Kale] and [Chekuri et al.]

$A(u)$: Set of elements ever chosen when u comes

- **Algorithm:** Measure 'value' of u by $f(u|A(u))$
  - Add u and discard u' if

$$f(u|A(u)) \geq 2 \cdot f(u'|A(u')) \qquad \ldots(*)$$

- **Analysis:**
  - $f(OPT) \leq f(OPT \cup A)$

$$\leq f(A) + \sum_{u \in OPT - A} f(u|A)$$

$\leq 2f(S)$
Also due to (*)

[Chakrabarti, Kale] and [Chekuri et al.]

$A(u)$: Set of elements ever chosen when u comes

- **Algorithm:** Measure 'value' of u by $f(u|A(u))$
  - Add u and discard u' if

$$f(u|A(u)) \geq 2 \cdot f(u'|A(u')) \qquad \ldots(*)$$

- **Analysis:**
  - $f(OPT) \leq f(OPT \cup A)$

$$\leq f(A) + \sum_{u \in OPT - A} f(u|A)$$

$\leq 2f(S)$
Also due to (*)

$$\leq 2 \cdot \sum_{u \in S} f(u|A(u)) \leq 2f(S)$$
Due to (*)

Both approaches fail to beat factor 4

- [Buchbinder et al.]
  - Measure ‘value’ of u by $\boldsymbol{f(u|S)}$
  - Add u to S if $f(u|S)$ is ‘large enough’, i.e.,

$$\boldsymbol{f(u|S) \geq \frac{2}{k} f(S)}$$

- [Chakrabarti, Kale] and [Chekuri et al.]
  - Measure ‘value’ of u by $\boldsymbol{f(u|A(u))}$
  - Add u and discard u' if

$$\boldsymbol{f(u|A(u)) \geq 2 \cdot f(u'|A(u'))}$$

‘value’ of u changes as S changes

‘local’ criteria: may accept element with little benefit

Our approach

- **Algorithm:**
  - Measure ‘value’ of u by $\boldsymbol{f(u|A(u))}$
  - Add u to S if $f(u|A(u))$ is ‘large enough’, i.e.,

$$\boldsymbol{f(u|A(u)) \geq \frac{1}{k}\left(\alpha f\left(S(u)\right) - f\left(A(u)\right)\right)} \qquad \ldots(*)$$

  - Discard least ‘valuable’ u from S if needed

Our approach

- **Algorithm:**
  - Measure 'value' of u by $\boldsymbol{f(u|A(u))}$
  - Add u to S if $f(u|A(u))$ is 'large enough', i.e.,

$$\boldsymbol{f(u|A(u)) \geq \frac{1}{k}\left(\alpha f\left(S(u)\right) - f\left(A(u)\right)\right)} \qquad \ldots(*)$$

  - Discard least 'valuable' u from S if needed
- **Analysis:**
  - $f(OPT) \leq f(OPT \cup A)$

$$\leq f(A) + \sum_{u \in OPT - A} f(u|A)$$

$\leq \alpha \cdot f(S) - f(A)$ due to (*), if $\alpha \cdot f(S) - f(A)$ is monotone
Best α depends on k: $\alpha_4 \approx 3.38, \alpha_\infty \approx 3.15$

So far…

- Identify the problems of existing (two) approaches
- Propose better algorithm **within** current framework
- How good is our algorithm?
  - Using $\boldsymbol{f(OPT) \leq f(OPT \cup A) \leq f(A) + \sum_{u \in OPT - A} f(u|A)}$ cannot beat our ratio α_k (4 for partition matroid)

Deterministic and Monotone

- **Monotone:**
  - Accept an element u only if $f(S)$ strictly increase

Deterministic and Monotone

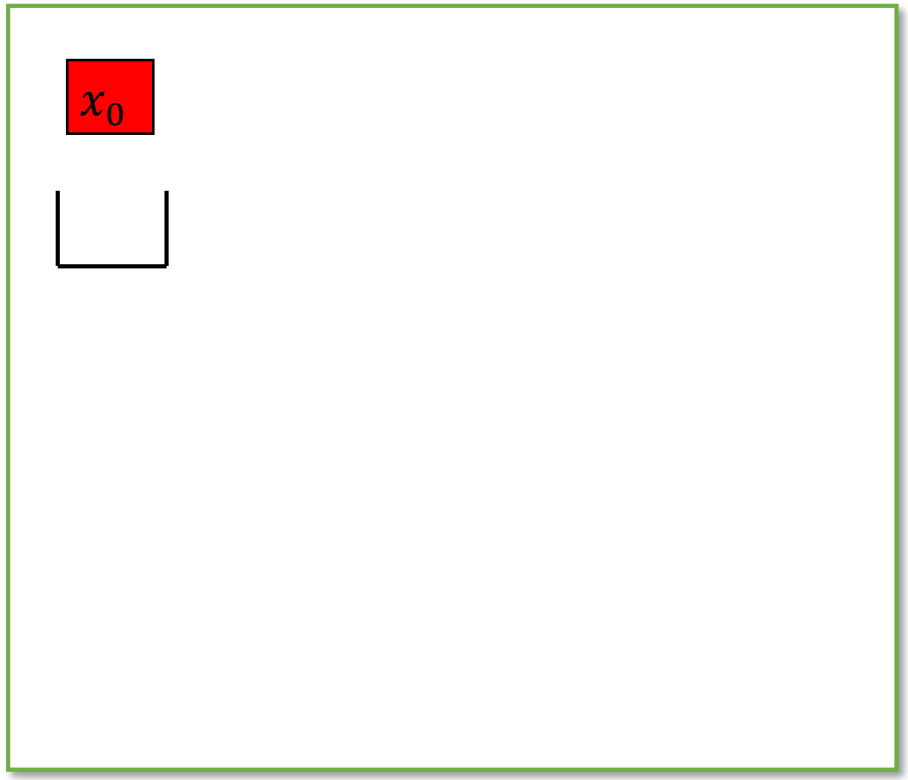
- **Monotone:**
  - Accept an element u only if $f(S)$ strictly increase

Matroid	Monotone Algorithm	Monotone Hardness
k-uniform	**3.38** **3.15** $(k \to \infty)$	**3.15**
Partition	**4**	**4**

Partition: direct sum of uniform matroids
$\Omega = \Omega_1 \cup \cdots \cup \Omega_m$, pick $\leq k_j$ elements from Ω_j

Monotone deterministic cannot beat 4

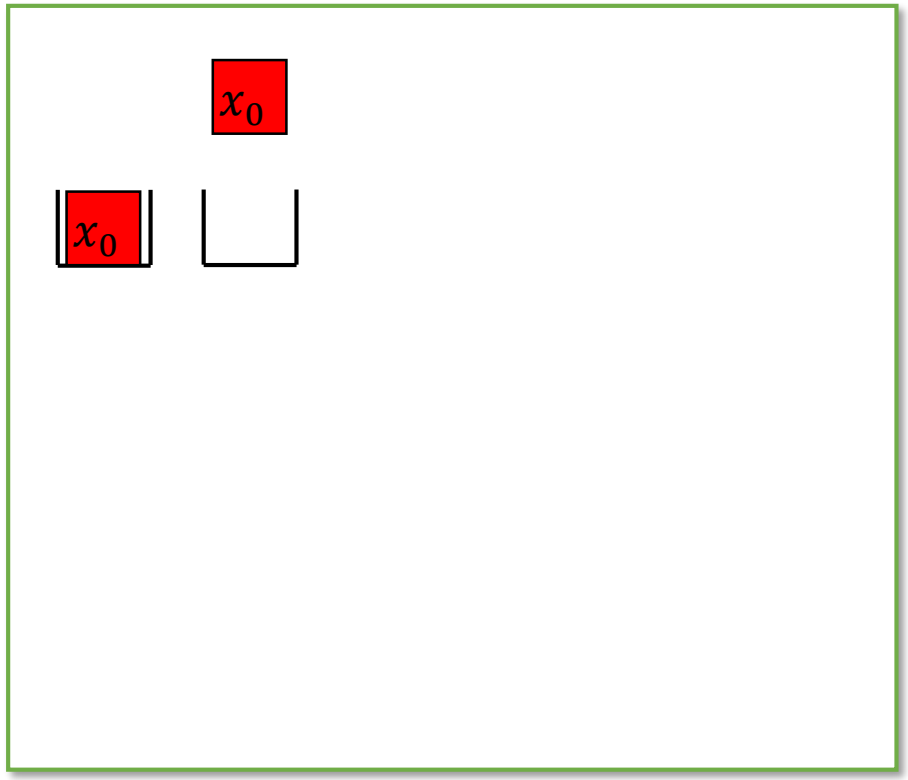
- **Monotone:**
  - Accept an element u only if $f(S)$ strictly increase



- Suppose strictly better than r-competitive,

Monotone deterministic cannot beat 4

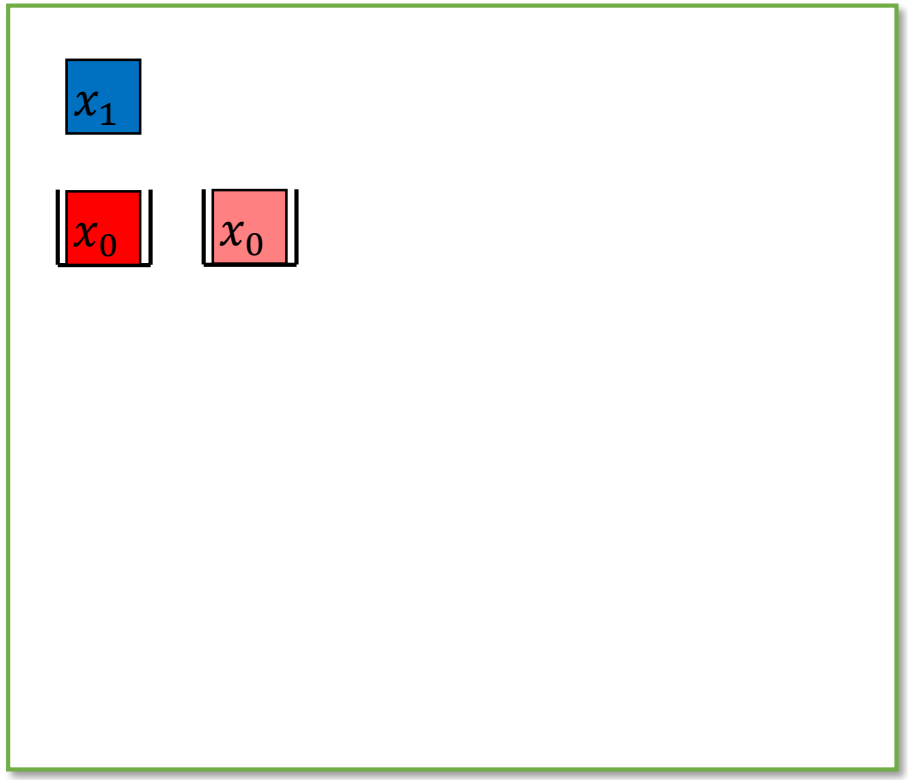
- **Monotone:**
  - Accept an element u only if $f(S)$ strictly increase



- Suppose strictly better than r-competitive,

Monotone deterministic cannot beat 4

- **Monotone:**
  - Accept an element u only if $f(S)$ strictly increase

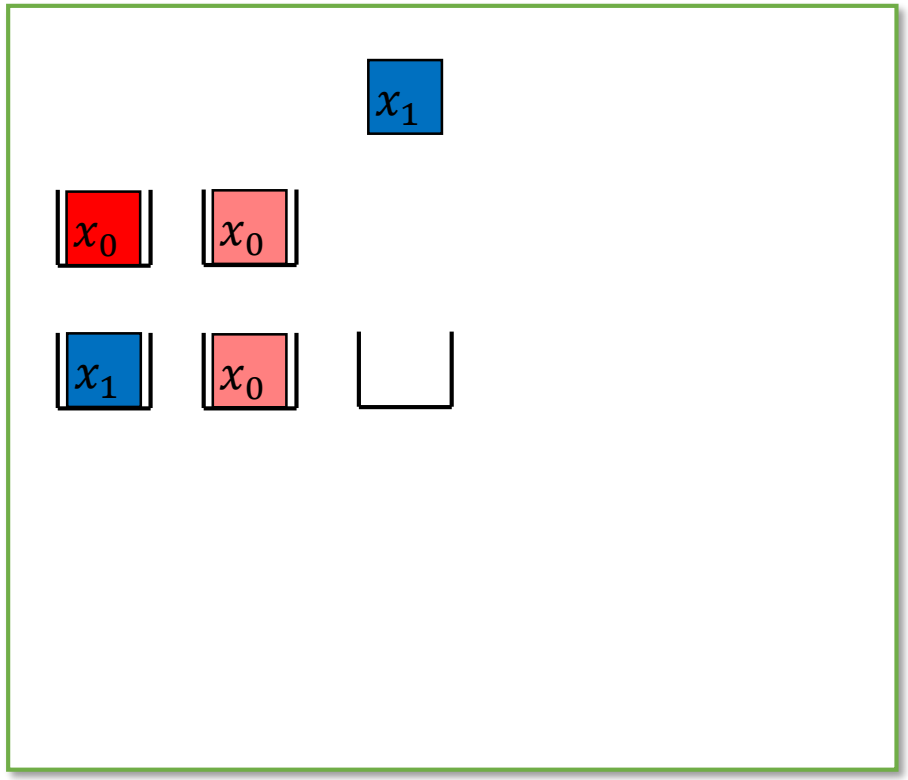


- Suppose strictly better than r-competitive,

$$x_0 = r(x_0 + x_1),$$

Monotone deterministic cannot beat 4

- **Monotone:**
  - Accept an element u only if $f(S)$ strictly increase

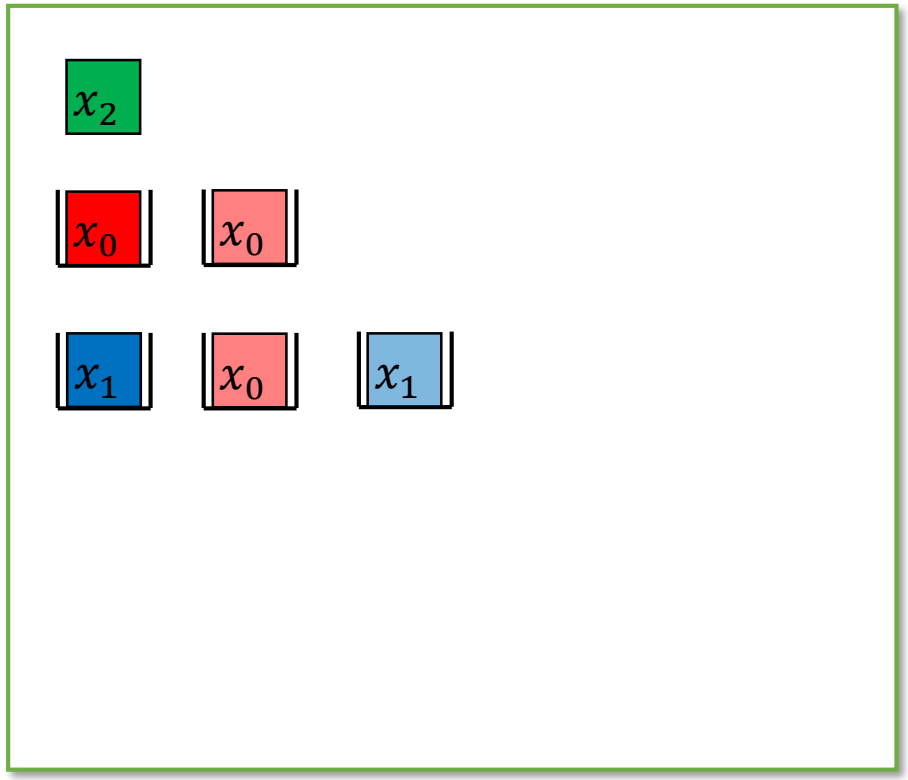


- Suppose strictly better than r-competitive,

$$x_0 = r(x_0 + x_1),$$

Monotone deterministic cannot beat 4

- **Monotone:**
  - Accept an element u only if $f(S)$ strictly increase

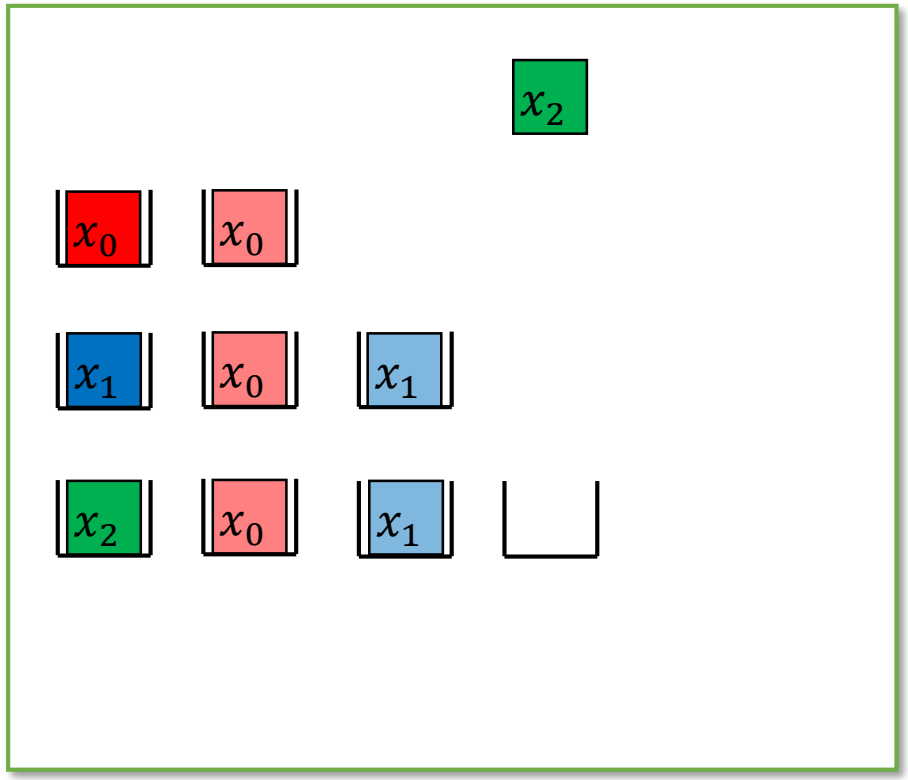


- Suppose strictly better than r-competitive,

$$x_0 = r(x_0 + x_1),$$
$$x_1 = r(x_0 + x_1 + x_2),$$

Monotone deterministic cannot beat 4

- **Monotone:**
  - Accept an element u only if $f(S)$ strictly increase

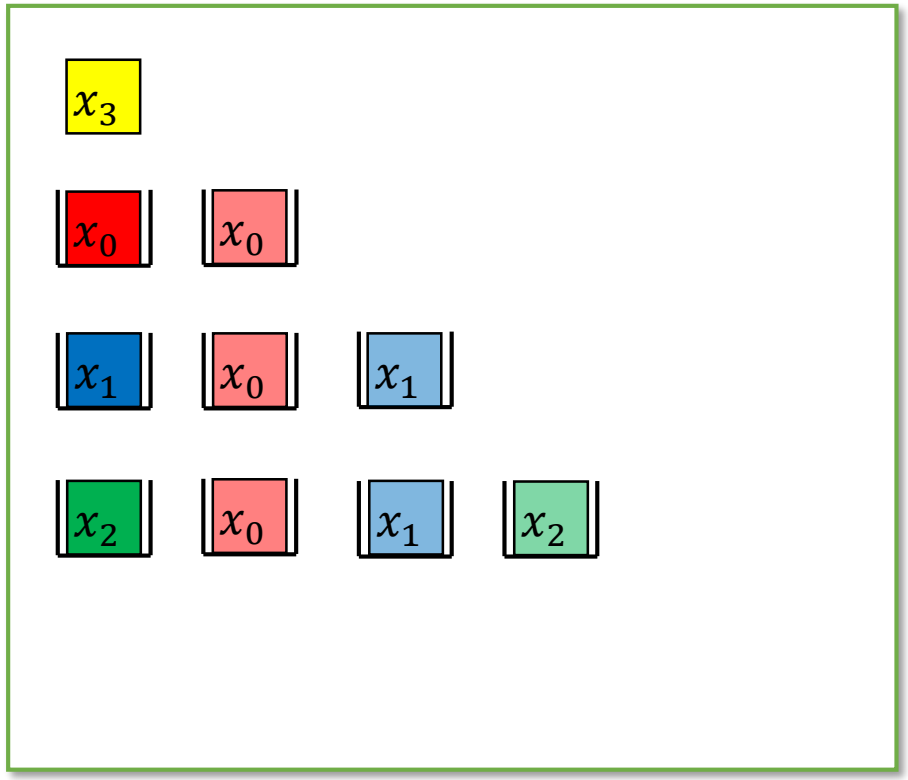


- Suppose strictly better than r-competitive,

$$x_0 = r(x_0 + x_1),$$
$$x_1 = r(x_0 + x_1 + x_2),$$

Monotone deterministic cannot beat 4

- **Monotone:**
  - Accept an element u only if $f(S)$ strictly increase

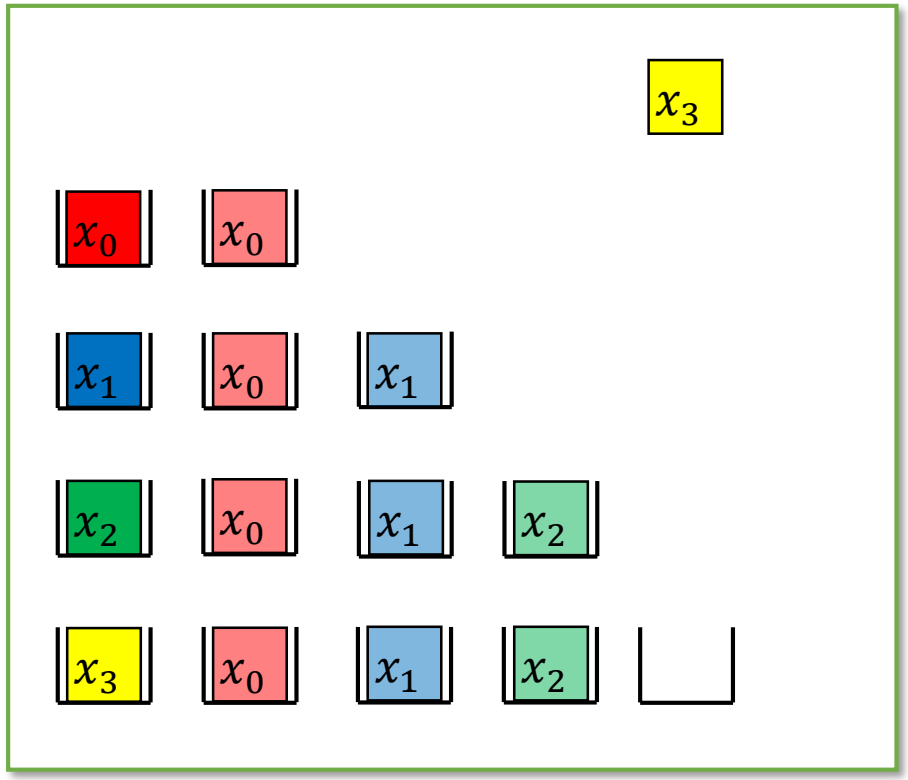


- Suppose strictly better than r-competitive,

$$x_0 = r(x_0 + x_1),$$
$$x_1 = r(x_0 + x_1 + x_2),$$
$$x_2 = r(x_0 + x_1 + x_2 + x_3),$$

Monotone deterministic cannot beat 4

- **Monotone:**
  - Accept an element u only if $f(S)$ strictly increase

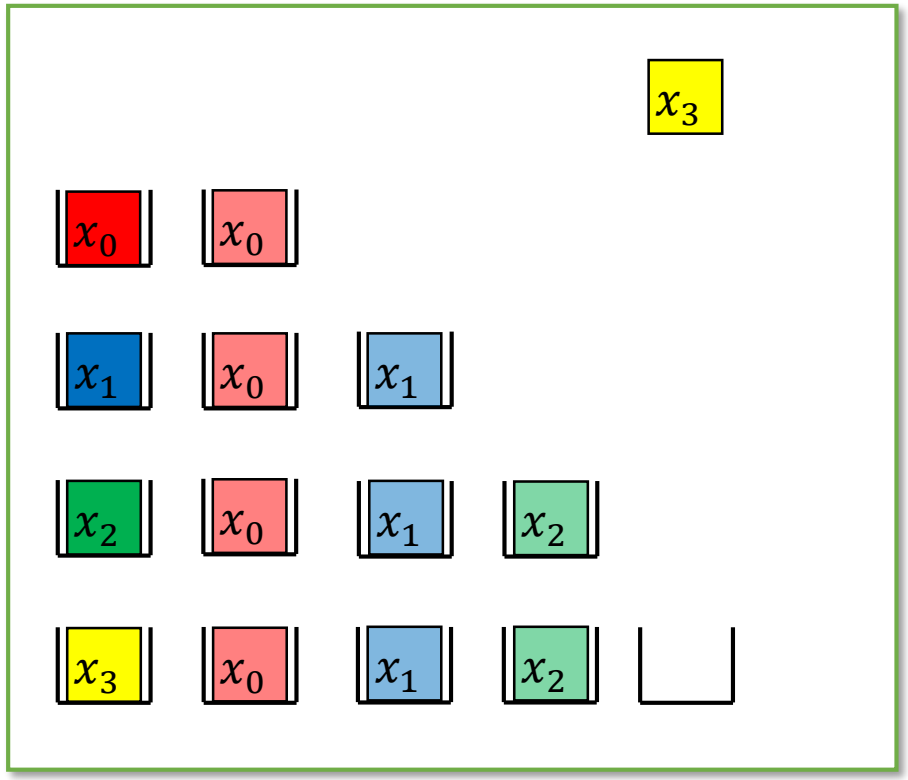


- Suppose strictly better than r-competitive,

$$x_0 = r(x_0 + x_1),$$
$$x_1 = r(x_0 + x_1 + x_2),$$
$$x_2 = r(x_0 + x_1 + x_2 + x_3),$$

Monotone deterministic cannot beat 4

- **Monotone:**
  - Accept an element u only if $f(S)$ strictly increase

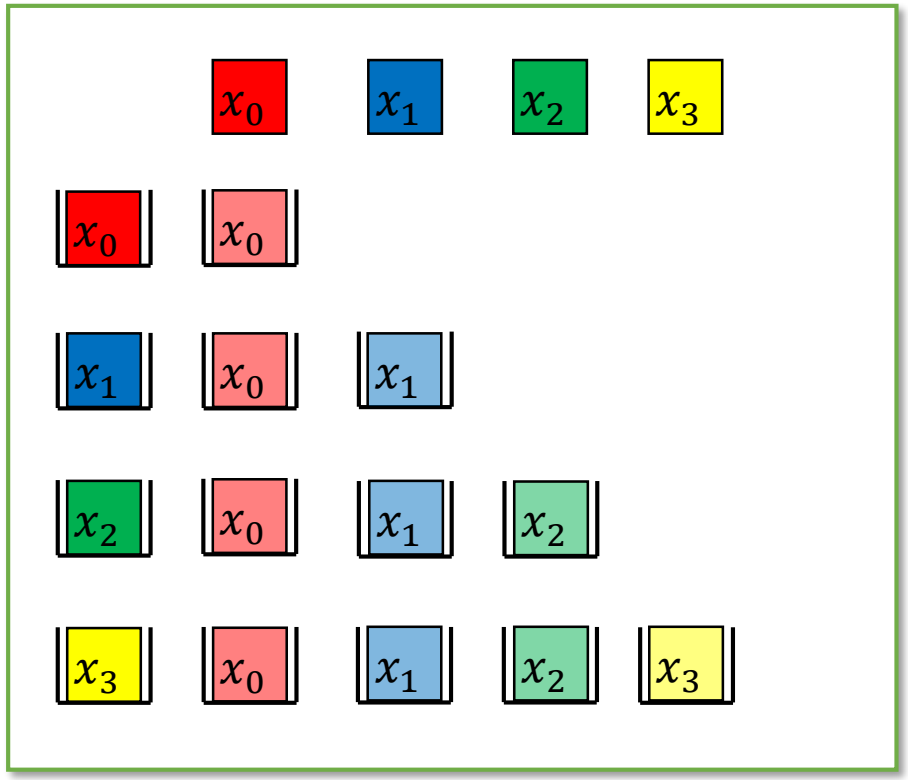


- Suppose strictly better than r-competitive,

$$x_0 = r(x_0 + x_1),$$
$$x_1 = r(x_0 + x_1 + x_2),$$
$$x_2 = r(x_0 + x_1 + x_2 + x_3),$$
$$\ldots\ldots$$

Monotone deterministic cannot beat 4

- **Monotone:**
  - Accept an element u only if $f(S)$ strictly increase

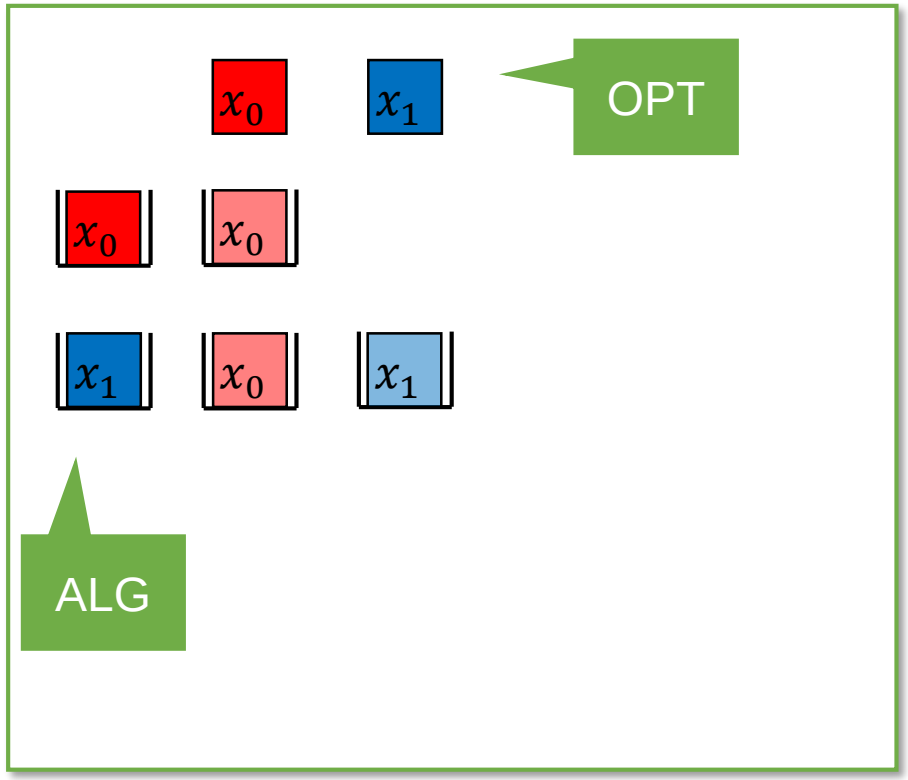


- Suppose strictly better than r-competitive,

$$x_0 = r(x_0 + x_1),$$
$$x_1 = r(x_0 + x_1 + x_2),$$
$$x_2 = r(x_0 + x_1 + x_2 + x_3),$$
$$\ldots\ldots$$

Monotone deterministic cannot beat 4

- **Monotone:**
  - Accept an element u only if $f(S)$ strictly increase



- Suppose strictly better than r-competitive,

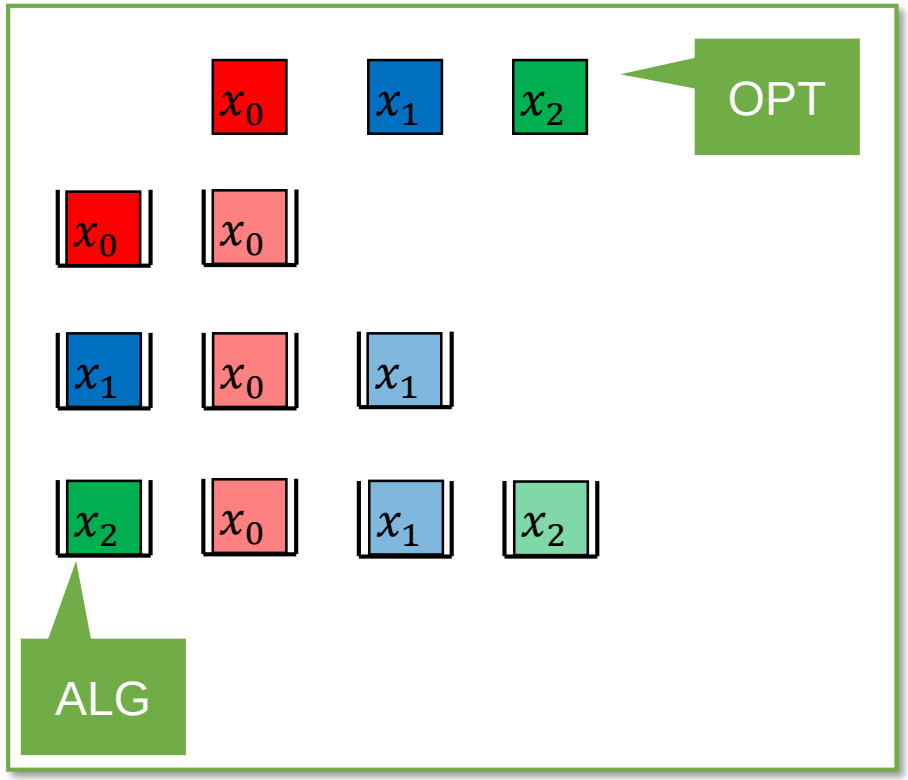
$$x_0 = r(x_0 + x_1),$$
$$x_1 = r(x_0 + x_1 + x_2),$$
$$x_2 = r(x_0 + x_1 + x_2 + x_3),$$
$$\ldots\ldots$$

- On the other hand,

$$x_1 > r(x_0 + x_1),$$

Monotone deterministic cannot beat 4

- **Monotone:**
  - Accept an element u only if $f(S)$ strictly increase



- Suppose strictly better than r-competitive,

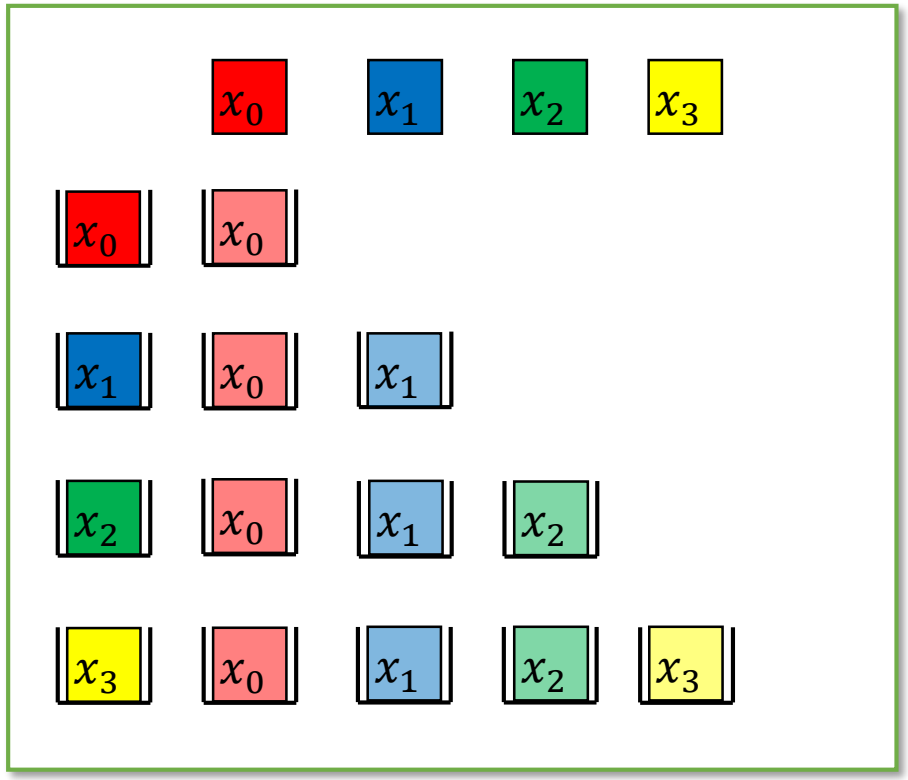
$$x_0 = r(x_0 + x_1),$$
$$x_1 = r(x_0 + x_1 + x_2),$$
$$x_2 = r(x_0 + x_1 + x_2 + x_3),$$
$$\ldots\ldots$$

- On the other hand,

$$x_1 > r(x_0 + x_1),$$
$$x_2 > r(x_0 + x_1 + x_2),$$
$$\ldots\ldots$$

Monotone deterministic cannot beat 4

- **Monotone:**
  - Accept an element u only if $f(S)$ strictly increase



- Suppose strictly better than r-competitive,

$$x_0 = r(x_0 + x_1),$$
$$x_1 = r(x_0 + x_1 + x_2),$$
$$x_2 = r(x_0 + x_1 + x_2 + x_3),$$
$$\ldots\ldots$$

- On the other hand,

$$x_1 > r(x_0 + x_1),$$
$$x_2 > r(x_0 + x_1 + x_2),$$
$$x_3 > r(x_0 + x_1 + x_2 + x_3)$$
$$\ldots\ldots$$

Analyzing the sequence

- With the following equations,

$$
\begin{aligned}
x_0 &= r(x_0 + x_1),\\
x_1 &= r(x_0 + x_1 + x_2),\\
x_2 &= r(x_0 + x_1 + x_2 + x_3),\\
&\cdots\cdots\\
x_1 &> r(x_0 + x_1),\\
x_2 &> r(x_0 + x_1 + x_2),\\
x_3 &> r(x_0 + x_1 + x_2 + x_3)
\end{aligned}
$$

Define:

$$x_0 = 1,\ x_1 = \frac{1-r}{r},$$

$$r x_{n+2} - x_{n+1} + x_n = 0$$

Analyzing the sequence

- With the following equations,

$$x_0 = r(x_0 + x_1),$$
$$x_1 = r(x_0 + x_1 + x_2),$$
$$x_2 = r(x_0 + x_1 + x_2 + x_3),$$
$$\cdots\cdots$$
$$x_1 > r(x_0 + x_1),$$
$$x_2 > r(x_0 + x_1 + x_2),$$
$$x_3 > r(x_0 + x_1 + x_2 + x_3)$$

Define:

$$x_0 = 1,\ x_1 = \frac{1-r}{r},$$
$$rx_{n+2} - x_{n+1} + x_n = 0$$

Need $\{x_n\}$ monotone increasing!

Analyzing the sequence

- With the following equations,

$$x_0 = r(x_0 + x_1),$$
$$x_1 = r(x_0 + x_1 + x_2),$$
$$x_2 = r(x_0 + x_1 + x_2 + x_3),$$
$$\cdots\cdots$$
$$x_1 > r(x_0 + x_1),$$
$$x_2 > r(x_0 + x_1 + x_2),$$
$$x_3 > r(x_0 + x_1 + x_2 + x_3)$$

Define:

$$x_0 = 1,\ x_1 = \frac{1-r}{r},$$
$$rx_{n+2} - x_{n+1} + x_n = 0$$

Need $\{x_n\}$ monotone increasing!

$\Rightarrow \boldsymbol{r \leq 0.25}$!

Randomization helps!

Matroid	Monotone Algorithm	Randomized Algorithm	Monotone Hardness
k-uniform	**3.38** **3.15** $(k \to \infty)$	**3.15**	**3.15**
Partition	**4**	**3.15**	**4**

Partition: direct sum of uniform matroids

$\Omega = \Omega_1 \cup \cdots \cup \Omega_m$, pick $\leq k_j$ elements from Ω_j

3.15 competitive algorithm for k-uniform matroid

Why does randomness help?

- Can pick fractions of an element (effectively, $k = \infty$)

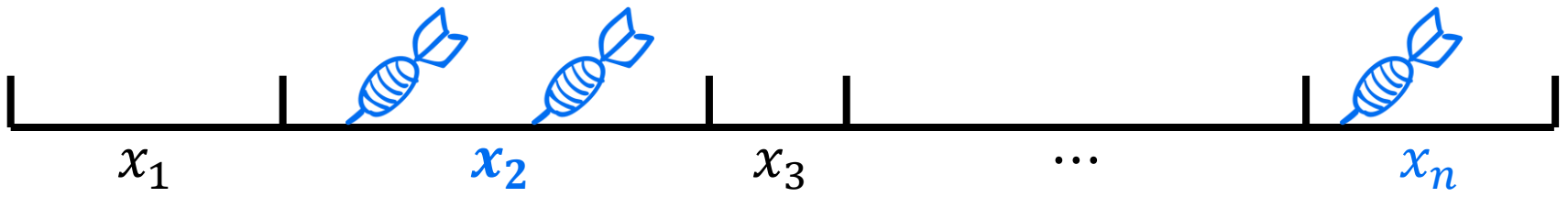
Why does randomness help?

- Can pick fractions of an element (effectively, $k = \infty$)
- **Challenge:** Online rounding algorithm
  - Rounding algorithms that preserve value (e.g., pipage rounding) do not work in online setting
  - Rounding algorithms that lose a factor of β gives only $\alpha_\infty \beta$ using naïve analysis

Dart-throw rounding

Fractional Solution

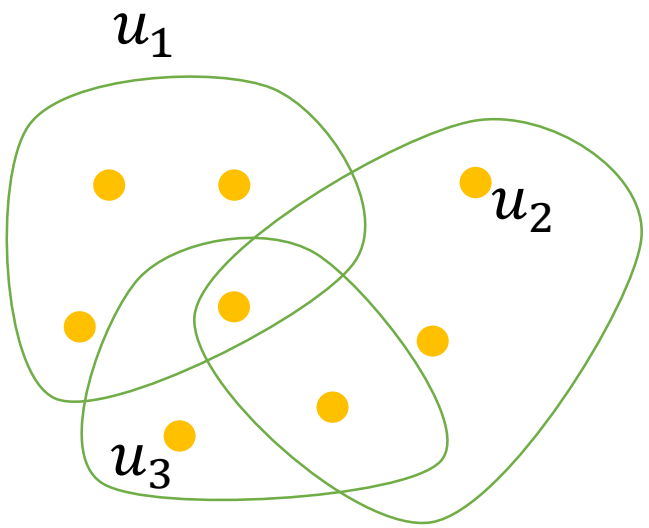
Throw k darts randomly



Pick elements with at least one dart

- It is $1 - \frac{1}{e}$ approximate: (for coverage functions)
  - Consider any ground item i covered by $S \subseteq \Omega$
  - Rounded solution covers item i with probability

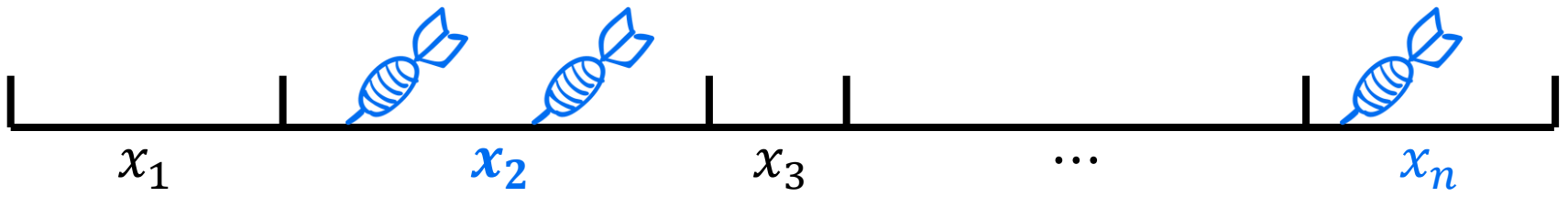
$$1 - \left(1 - \frac{1}{k}\sum_{S \ni i} x_S\right)^k \geq 1 - e^{-\sum_{S \ni i} x_S}$$



Dart-throw rounding

Fractional Solution

Throw k darts randomly



Pick elements with at least one dart

- $S = OPT, x_u = 1$ for $u \in S$ and $= 0$ otherwise
- Each item can only be chosen with probability

$$1 - \left(1 - \frac{1}{k}\right)^k \approx 1 - \frac{1}{e}$$

- Only get factor $\frac{3.15}{1-\frac{1}{e}} > 4$?

New framework

- $g(x) = E[f(x)]$ w.r.t. dart-throw rounding for fractional x
- Coverage case: expected number of grounded items covered, a ground item i is covered with probability $1 - e^{-\sum_{S \ni i} x_S}$
- New inequality:

$$f(OPT) \leq g(A) + \sum_{u \in OPT} g(u|A)$$

  - Run algorithm for kp-uniform matroid, w.r.t. $\frac{1}{p}$ fractions of elements for large p, with objective g
    REMARK: allow pick >1 fraction of an element
  - Use the above inequality in place of the old one in the analysis

Summary

Matroid	Objective f	Algorithm	Hardness
k-uniform	Monotone	**3.15**	2 [Buchbinder et al.]
	General	**8.63**	2.28 [Buchbinder et al.]
Partition	Monotone	**3.15**	**2.61**
	General	16 [Chakrabarti, Kale] [Chekuri et al.]	**2.61**

Partition: direct sum of uniform matroids

$\Omega = \Omega_1 \cup \cdots \cup \Omega_m$, pick $\leq k_j$ elements from Ω_j

Future directions

- Beating factor 4 for general matroid:
  - Need a good online rounding algorithm

Future directions

- Beating factor 4 for general matroid:
  - Need a good online rounding algorithm
- Hardness results:
  - E.g., is the monotonicity assumption necessary?
  - Is $\alpha_\infty \approx 3.15$ the best possible for k-uniform matroid?
- Can deterministic non-monotone or randomized monotone algorithms beat factor 4?