Color Coding-Related Techniques

Meirav Zehavi

Tel Aviv University

Outline

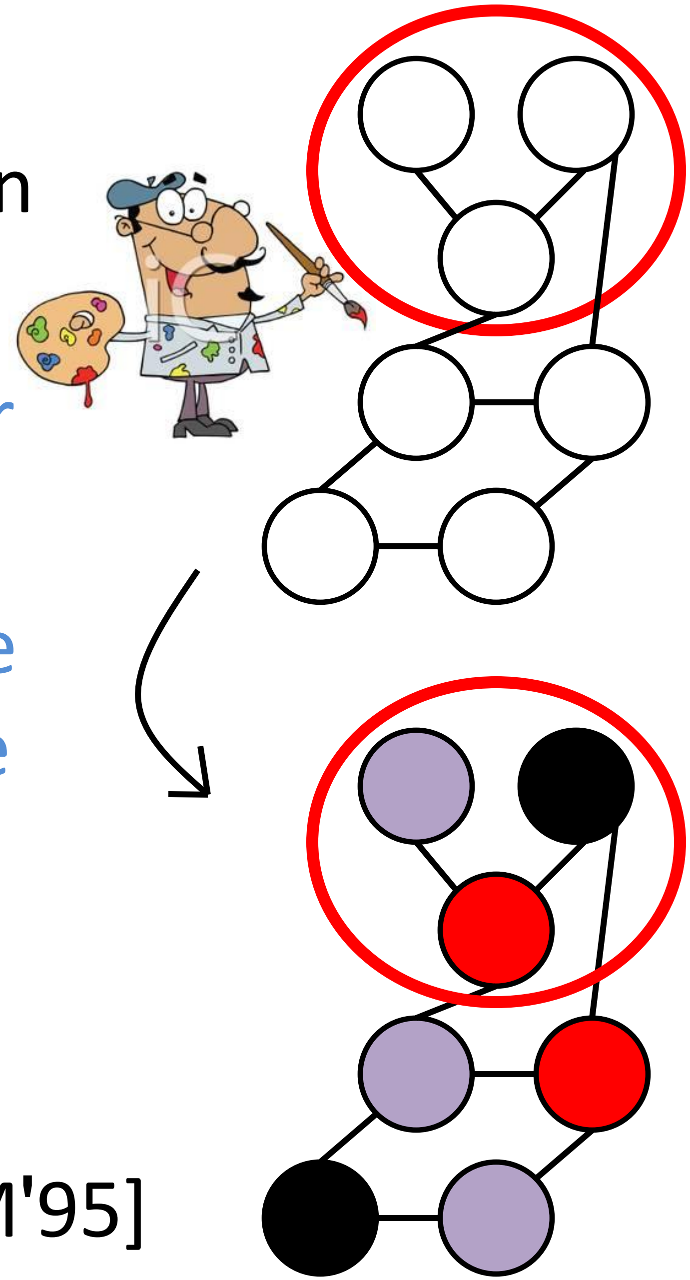
Color Coding-Related Techniques

- Color Coding
- Divide-and-Color
- Narrow Sieves
- Representative Sets

Mixing Color Coding-Related Techniques

Color Coding

Given a graph *G*, seek a solution that is a subgraph on *k* nodes.

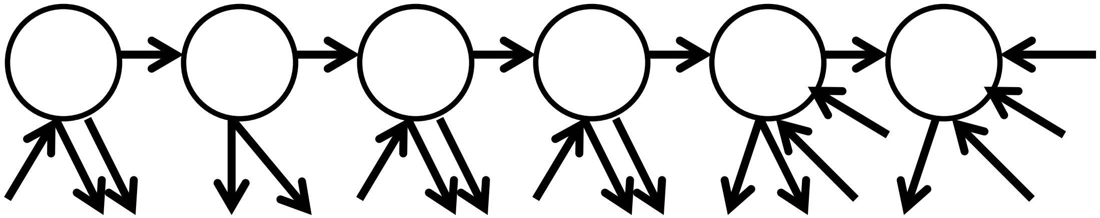


- Given a set of *k* colors, color each node (randomly).
- With high probability, there is a solution where each node has a different color.
- Seek such a solution—easy! (dynamic programming)

[Alon, Yuster and Zwick, J. ACM'95]

Color Coding: Directed *k*-Path

Given a directed graph *G*, seek a path on *k* nodes.



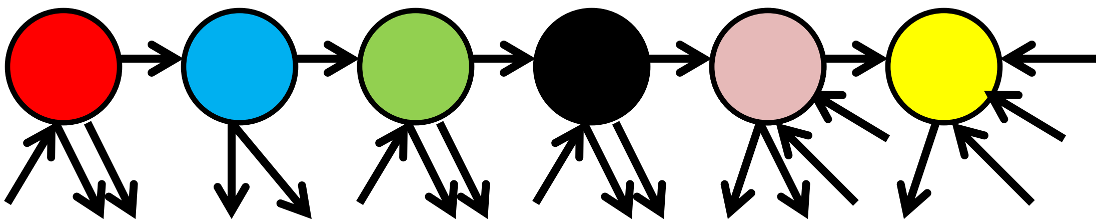
$k = 6$

- Color nodes; use a set of *k* colors.*
- Success (1 iteration): $k!/k^k$.
- Success (*r* iterations): $1 - (1-k!/k^k)^r$.
- $r = O^*(e^k)$ iterations.

* There is a variant of color coding that uses more colors.

Color Coding: Directed *k*-Path

Given a directed graph *G*, seek a path on *k* nodes.



$k = 6$

- Color nodes; use a set of *k* colors.
- Examine an iteration where the solution is colorful.
- Use dynamic programming:

  M[v,p,S] – Is there a path on p nodes that ends at the node v and uses the color-set S?

- Can handle weights.
- Can be derandomized.

Time: $O^*((2e)^k)$

Color Coding-Related Techniques

- **Divide-and-color.** **Fast**

  [Chen, Kneis, Lu, Mölle, Richter, Rossmanith, Sze and Zhang, SICOMP'09]

  - Weighted problems; deterministic; polynomial-space.

- **Multilinear detection & Narrow sieves.** **Fastest**

  [Koutis, ICALP'09], [Williams, IPL'09], [KW, ICALP'10] & [Björklund, FOCS'10], [Björklund, Husfeldt, Kaski and Koivisto, arXiv'10]

  - ~~Weighted problems~~; ~~deterministic~~; polynomial-space.

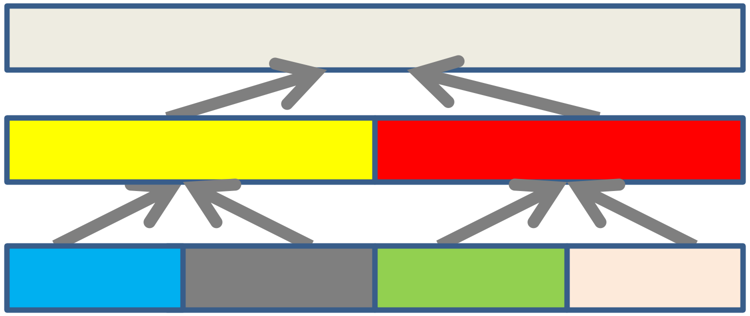
- **Representative sets.** **Faster**

  [Fomin, Lokshtanov and Saurabh, SODA'14]

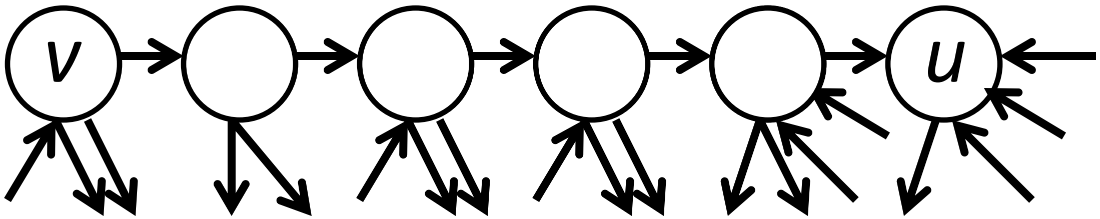
  - Weighted problems; deterministic; ~~polynomial-space~~.

Divide-and-Color

- Based on recursion.
- In each step, we have a set A of n elements, and we seek a certain subset A^* of k elements in A.
- We color each element in A in one of two colors, thus partitioning A into two sets B and C.
- Now, we seek a subset $B^* \subseteq A^*$ in B, and a subset $C^* = A^* \setminus B^*$ in C.



Divide-and-Color: Directed *k*-Path



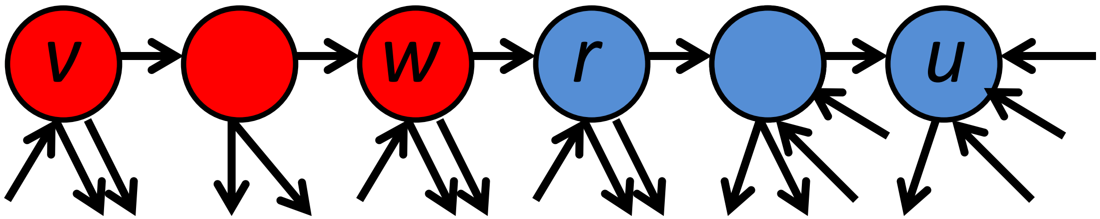
$k = 6$

For each pair of nodes *v* and *u*, we seek a solution that starts at *v* and ends at *u* .

- Color each node in red or blue.

- Examine an option where the first “half” of the solution is red, and the second “half” is blue.

Divide-and-Color: Directed *k*-Path

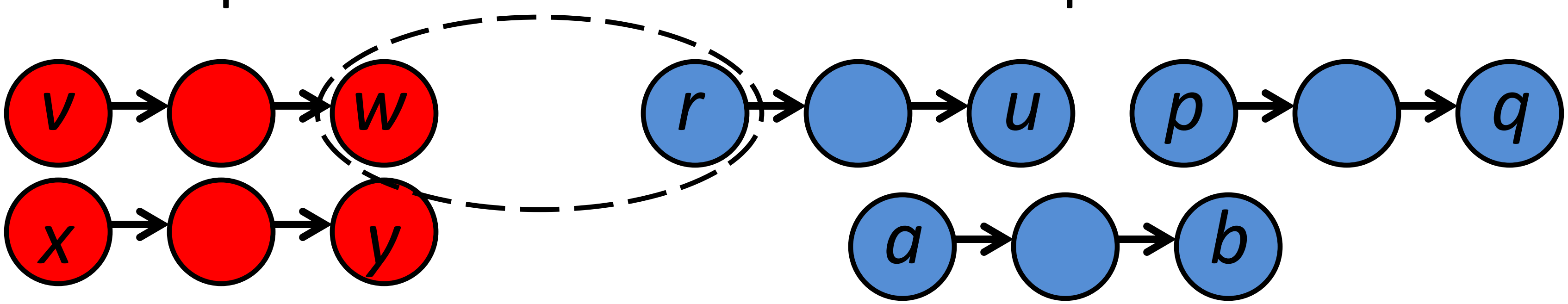


$k = 6$

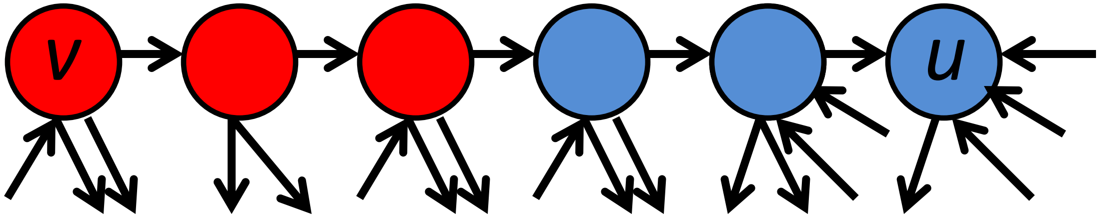
Now, we have two new subproblems: Find paths on $k/2$ nodes in the subgraph induced by the red nodes, and find such paths in the subgraph induced by the blue nodes.

Solutions to the red subproblem:

Solutions to the blue subproblem:



Divide-and-Color: Directed *k*-Path



$k = 6$

Definition: Let $\mathcal{F}$ be a set of functions f: {1,2,…,n} → {0,1}. We say that $\mathcal{F}$ is an (*n*,*k*)-universal set if for every subset I of {1,2,…,*n*} of size *k* and a function *f'*: I→{0,1}, there is a function $f \in \mathcal{F}$ such that for all $i \in I,\ f(i) = f'(i)$.

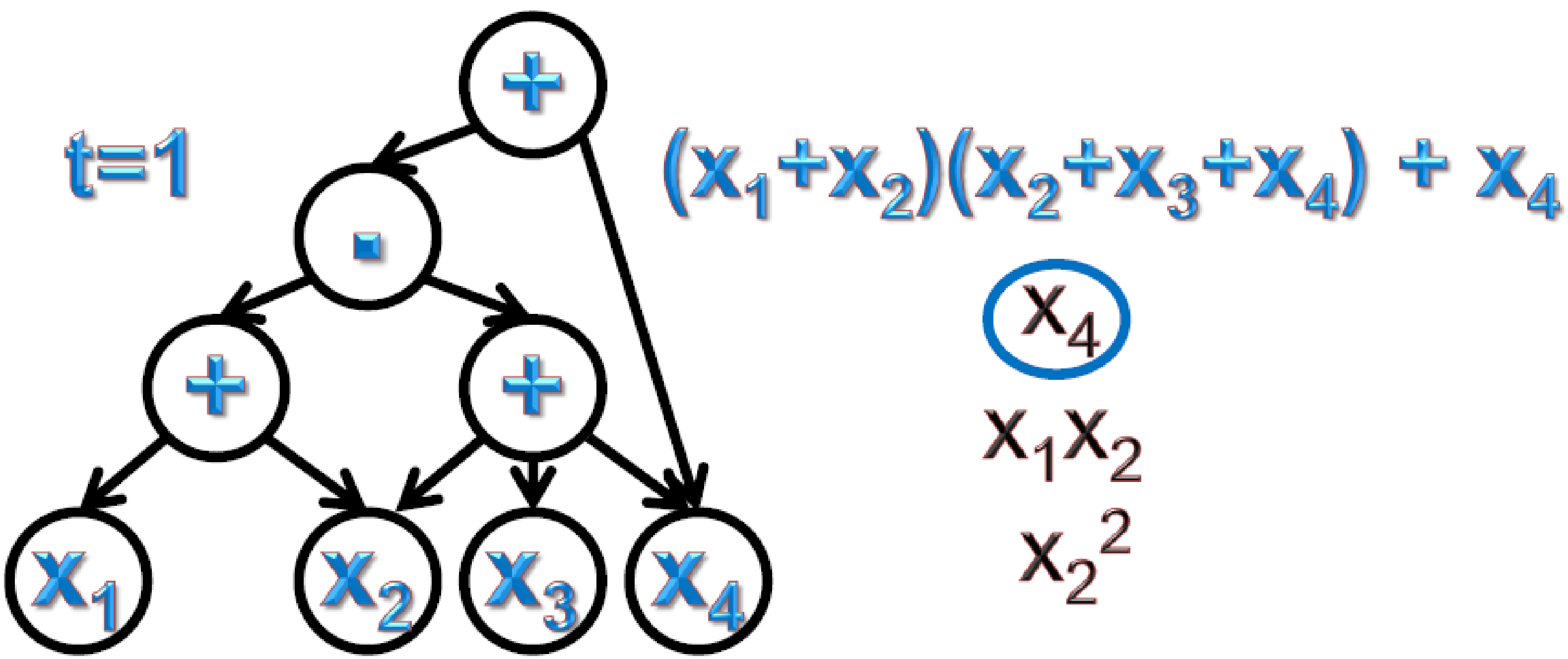
Theorem (Naor, Schulman and Srinivasan, FOCS'95): An (*n*,*k*)-universal family of size $2^{k+o(k)}\log n$ can be computed in time $O(2^{k+o(k)} n \log n)$.

Running time: $O^*(4^k)$.

Multilinear Detection & Narrow Sieves

Multilinear Detection:

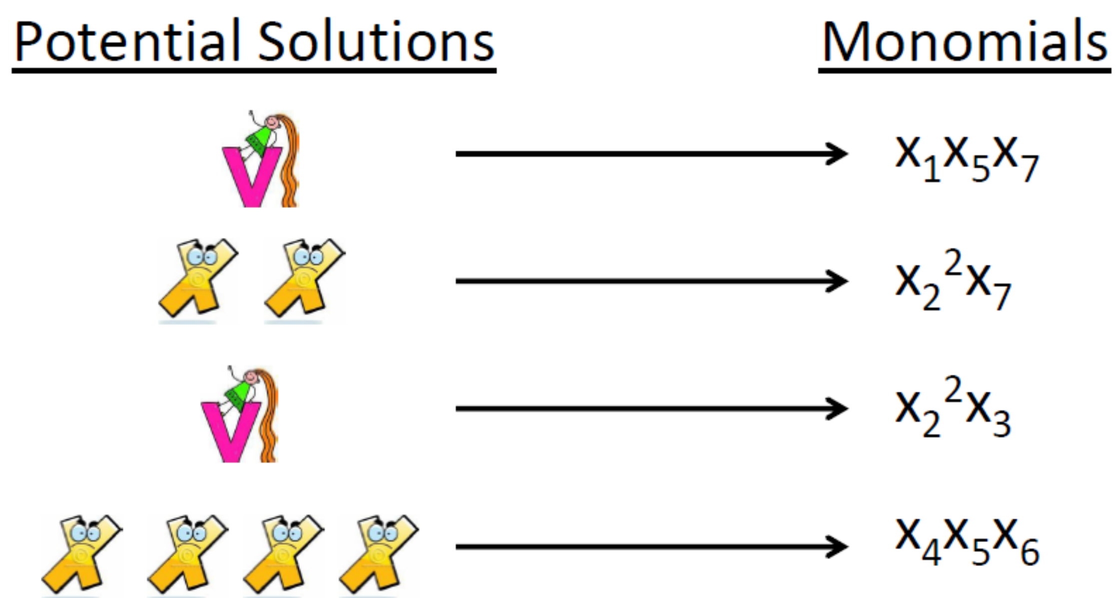
1. Potential solution → Monomial.
   - Correct solution ↔ Multilinear monomial.
2. Use a known $O^*(2^t)$-time **randomized** algorithm for the t-Multilinear Detection Problem.



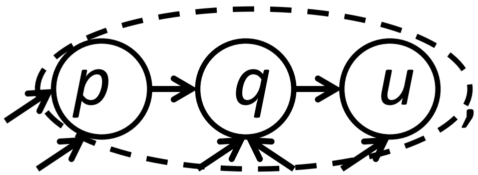
Multilinear Detection & Narrow Sieves

Narrow Sieves:

1. Potential solution → Monomial.
   - Correct solution → Unique monomial.
   - The set of incorrect solutions can be partitioned into pairs, where the elements in each pair are associated with the same monomial.
2. Inclusion-exclusion principle; Schwartz-Zippel lemma; dynamic programming.



Narrow Sieves: Directed *k*-Path

- Use a set C of k colors.
- $\boldsymbol{x_{v,c}}$ for each node $\boldsymbol{v}$ and color $\boldsymbol{c}$; $\boldsymbol{x_e}$ for each edge $\boldsymbol{e}$.
- 

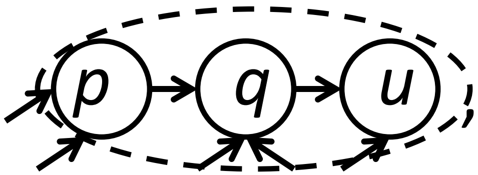
; $k = 3$; $x_{(p,q)}x_{(q,u)}\cdot$ $x_{p,g}x_{q,b}x_{u,g}$, $x_{p,g}x_{q,b}x_{u,r}$, $x_{p,b}x_{q,r}x_{u,r}$, …
- Let $\boldsymbol{W_X}$ be the set of colored walks on k nodes avoiding colors from X.
- Let $\boldsymbol{W_{colorful}}$ be the set of colorful walks on k nodes.
- $\boldsymbol{POL_{colorful}} = \sum_{w \in \boldsymbol{W_{colorful}}} \text{mon}(w) = \sum_{X \subseteq C} (-1)^{|X|} \sum_{w \in \boldsymbol{W_X}} \text{mon}(w)$

→ Evaluate $\boldsymbol{POL_{colorful}}$: $O^*(2^k)$ time, **polynomial-space**.
(Dynamic programming; do not remember color-sets.)

Narrow Sieves: Directed k-Path

- Use a set C of k colors.
- $\boldsymbol{x_{v,c}}$ for each node $\boldsymbol{v}$ and color $\boldsymbol{c}$; $\boldsymbol{x_e}$ for each edge $\boldsymbol{e}$.

$x_{(p,q)}x_{(q,u)}\cdot$

- 

; $k = 3$; $x_{p,g}x_{q,b}x_{u,g}$, $x_{p,g}x_{q,b}x_{u,r}$, $x_{p,b}x_{q,r}x_{u,r}$, …

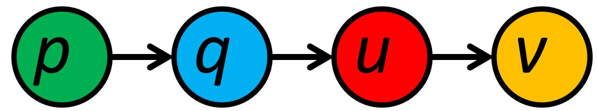
- $\boldsymbol{POL_{colorful}} = \sum_{w \in W_{colorful}} \mathrm{mon}(w) = \sum_{\substack{w \in W_{colorful} \\ correct}} \mathrm{mon}(w) + \sum_{\substack{w \in W_{colorful} \\ incorrect}} \mathrm{mon}(w)$

- *correct*: unique monomials; *incorrect*: partition into pairs having the same monomial.
- Is $\boldsymbol{POL_{colorful}}$ 0?
  (characteristic 2; Schwartz-Zippel lemma; evaluations)

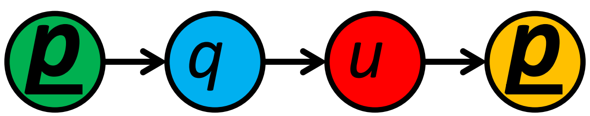
Narrow Sieves: Directed *k*-Path

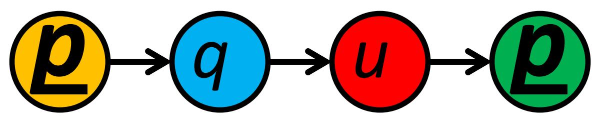
- *correct*: unique monomials; *incorrect*: partition into pairs having the same monomial. $k = 4$

$x_{(p,q)}x_{(q,u)}x_{(u,v)}x_{p,g}x_{q,b}x_{u,r}x_{v,o}$



Swap! (different colors → different potential solutions)





$x_{(p,q)}x_{(q,u)}x_{(u,v)}x_{p,g}x_{q,b}x_{u,r}x_{p,o}$

(same monomial)

Representative Sets: Definition

Let E be a universe of n elements, and let $\mathcal{S}$ be a family of p-subsets of E.

A subfamily $\widehat{\mathcal{S}} \subseteq \mathcal{S}$ ***k*-represents** $\mathcal{S}$ if:

For every pair of sets $X \in \mathcal{S}$ and $Y \subseteq E \setminus X$
such that $|Y| \leq k - p$,
there is a set $\widehat{X} \in \widehat{\mathcal{S}}$ disjoint from Y.

(Weighted problems, matroids: a more general definition.)

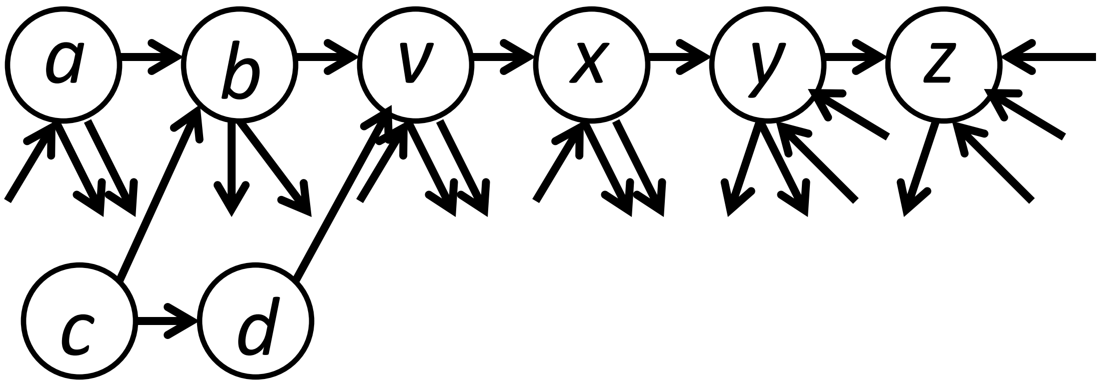
Representative Sets: Technique

- Consider a parameterized algorithm, **A**, based on dynamic programming.
- At each stage, **A** computes a family $\mathcal{S}$ of sets that are partial solutions.
- We compute a subfamily $\widehat{\mathcal{S}} \subseteq \mathcal{S}$ that represent $\mathcal{S}$.
- Each reference to $\mathcal{S}$ is replaced by a reference to $\widehat{\mathcal{S}}$.
- Can we efficiently compute representative families that are small enough?
  - Fomin, Lokshtanov and Saurabh, SODA'14:

  $[\binom{k}{p}2^{o(k)}\log n$, $O(|\mathcal{S}|(k/(k-p))^{k-p}2^{o(k)}\log n)$]

  Size Time

Representative Sets: Directed k-Path



$k = 6$

$p = 3$

Let $S_{v,p}$ be the family of node-sets of directed paths on p nodes that end at v. $|S_{v,p}|$ can be very large! $\binom{n}{p}$

Use dynamic programming + representative sets:

M[v,p] stores a family that (k-p)-represents $S_{v,p}$.

$\binom{k}{p}2^{o(k)}\log n$

→ Running time: $O^*(2.851^k)$

Mixing Color Coding-Related Techniques

Sometimes **mixtures** of color coding-related techniques result in faster algorithms.

<u>Directed *k*-Path (for example)</u>:

1. Divide-and-Color: $O^*(4^k)$
   (weighted; det.; pol.-space)
2. Narrow Sieves: $O^*(2^k)$
   (~~weighted~~; ~~det.~~; pol.-space)
3. Rep. Sets: $O^*(2.851^k)$ (weighted; det.; ~~pol.-space~~)

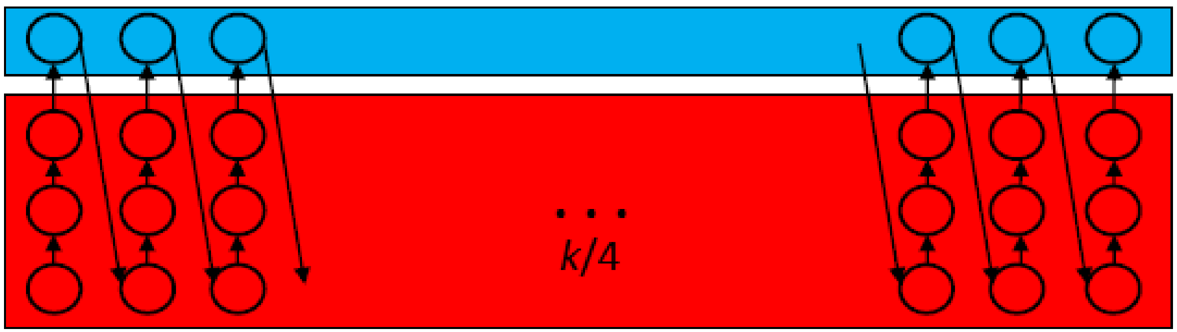
- Rep. Sets + Tradeoff + Div-and-Col: $O^*(2.597^k)$ [ESA'15]

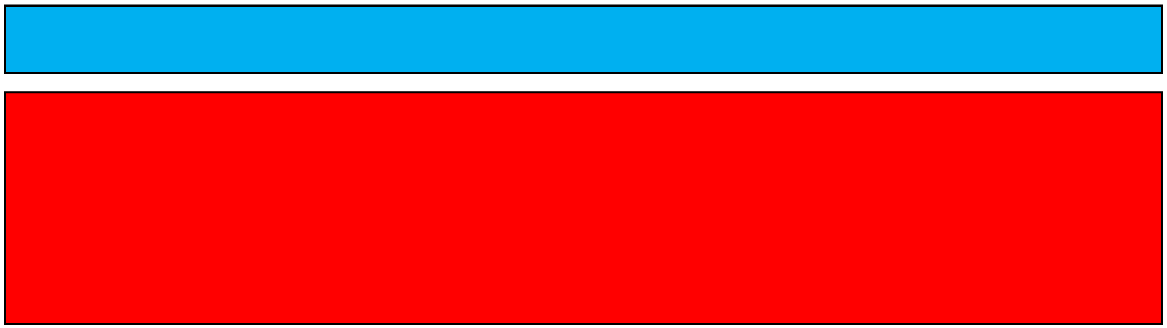
Tradeoff: Fomin, Lokshtanov, Panolan and Saurabh, ESA'14;
with Shachnai, ESA'14.

Mixing: Directed k-Path

Nodes: red and blue.

There is a solution → there is a solution that looks like this:

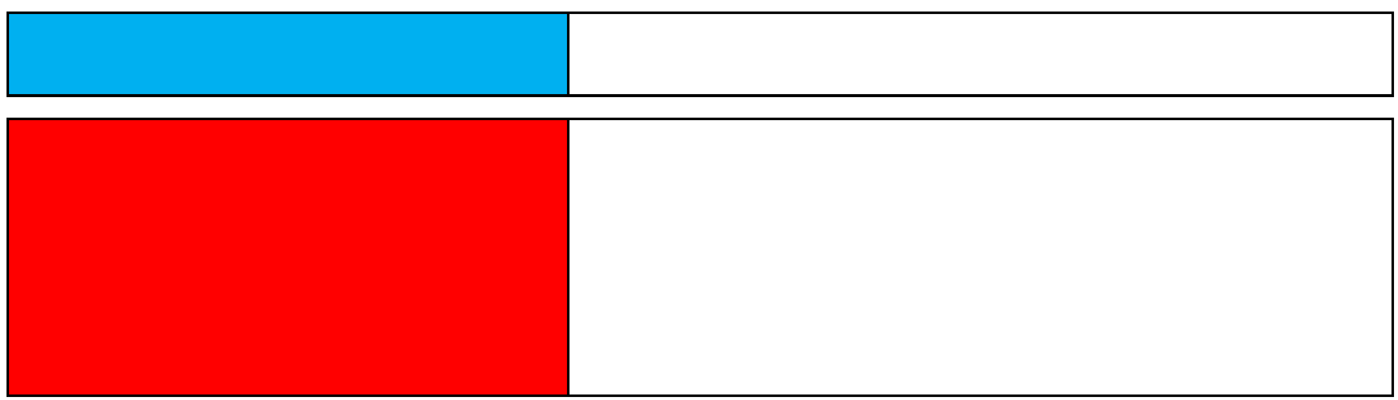




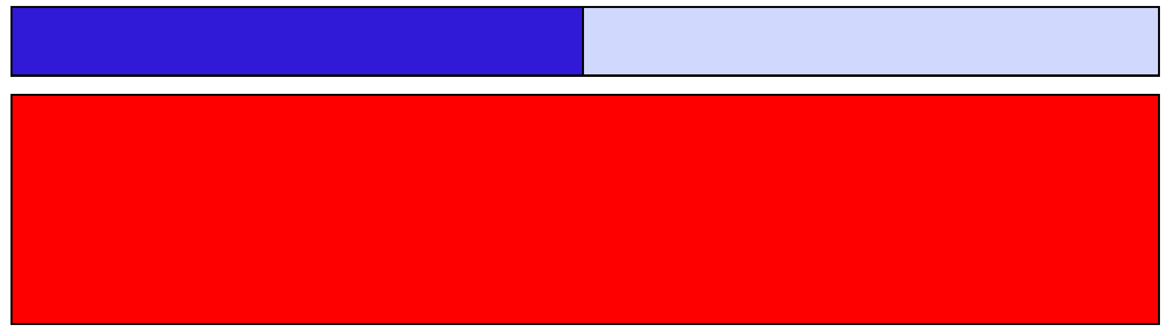
Mixing: Directed *k*-Path

Standard dynamic programming + representative sets:

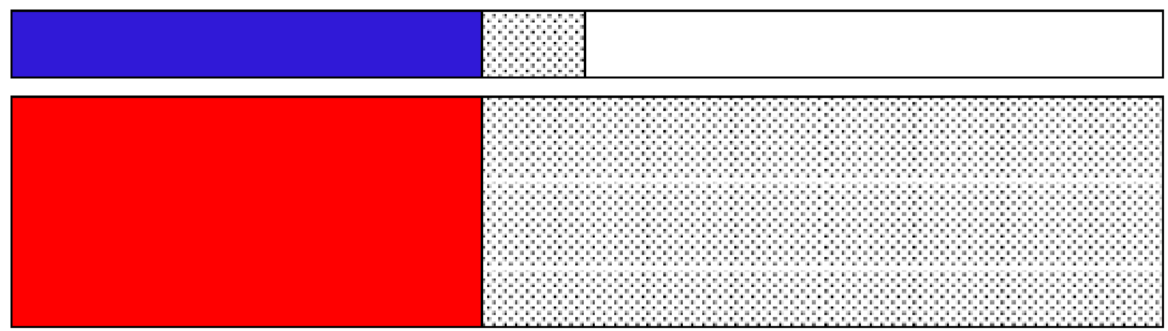
At each stage, for each node *v* and integer *p*, we have a family of partial solutions; each partial solution is the node-set of a path on *p* nodes that ends at *v*.



Dark blue and light blue:



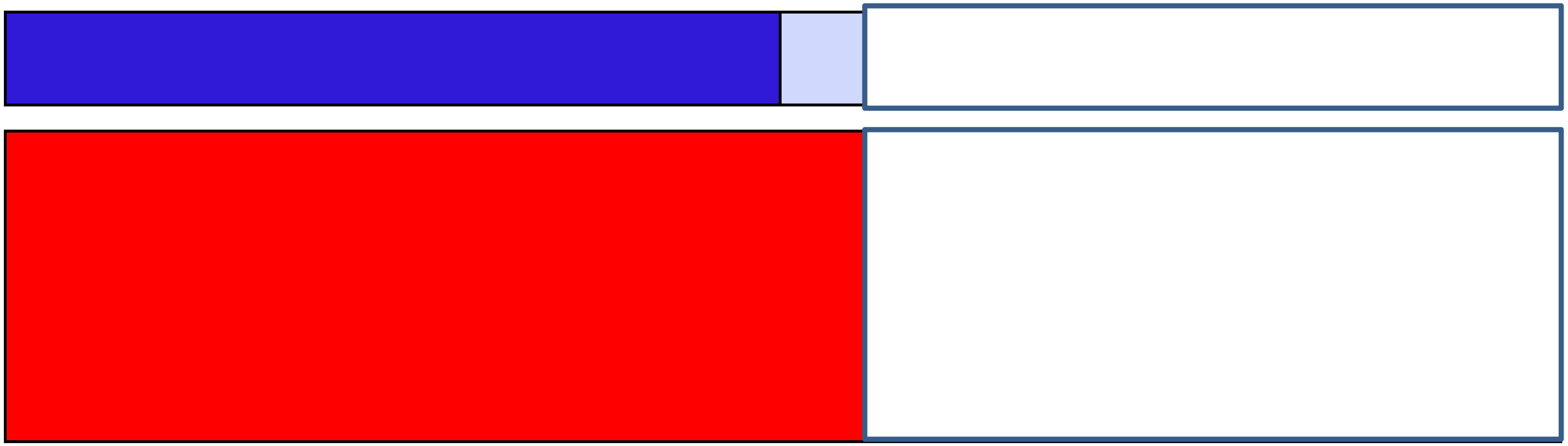
First half of the computation:



Second half of the computation:

Mixing: Directed *k*-Path

The worst time to compute representative sets:

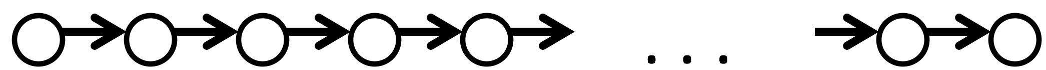


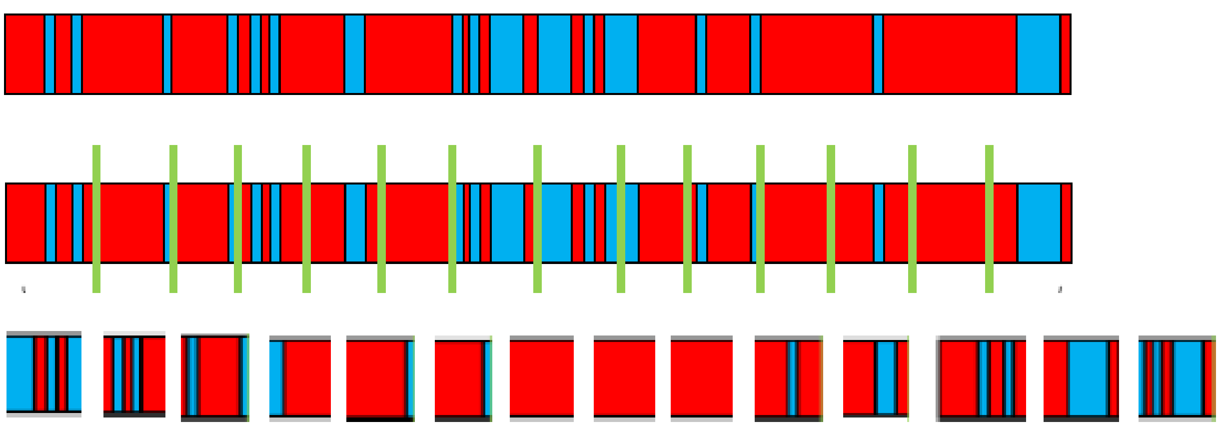
Mixing: Directed *k*-Path

- A more general definition of representative sets (+ the necessary computation).

- Given the blue set, to find the dark and light blue sets, we use one step of divide-and-color.

- Balanced cutting:






thanks