Computing Generalized Rank Invariant for Persistence Modules via Zigzag Persistence

(SoCG 22)

Tamal K. Dey
computer science
Purdue University

joint work with
Woojin Kim & Facundo Memoli

Persistence Modules

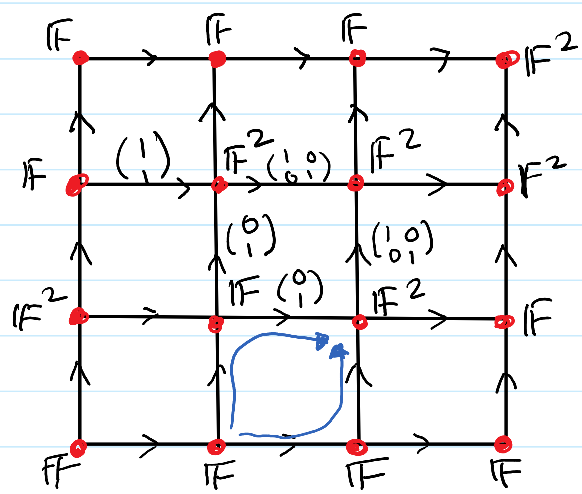
- $M : P \to \mathrm{Vec}$ as a functor (P-module)

  P: finite poset; Vec: finite vector space category

- $\forall\, p, q,\ p \leq q$, $\varphi_M(p,q) : M_p \to M_q$

  $\varphi_M(p,q)$: structure map

$$\cdots \; \mathbb{F} \xrightarrow{1} \mathbb{F} \xrightarrow{\binom{0}{1}} \mathbb{F}^2 \xrightarrow{(1,0)} \mathbb{F} \xrightarrow{1} \mathbb{F} \xrightarrow{\binom{1}{0}} \mathbb{F}^2 \; \cdots \qquad \mathbb{Z}\text{-module}$$

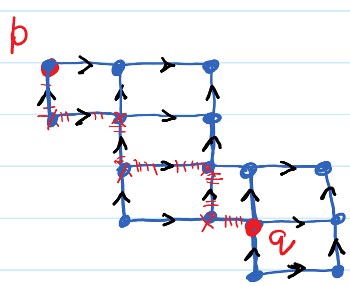


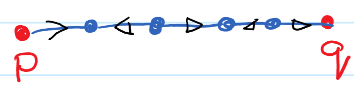
$\mathbb{Z}^2$- module

Intervals and Interval Modules

- $I \subseteq P$ an interval if
  - $p, q \in I$ and $p \le r \le q \Rightarrow r \in I$
  - I is connected

    $\forall p, q \in I$, a path of comparables connecting p, q exists



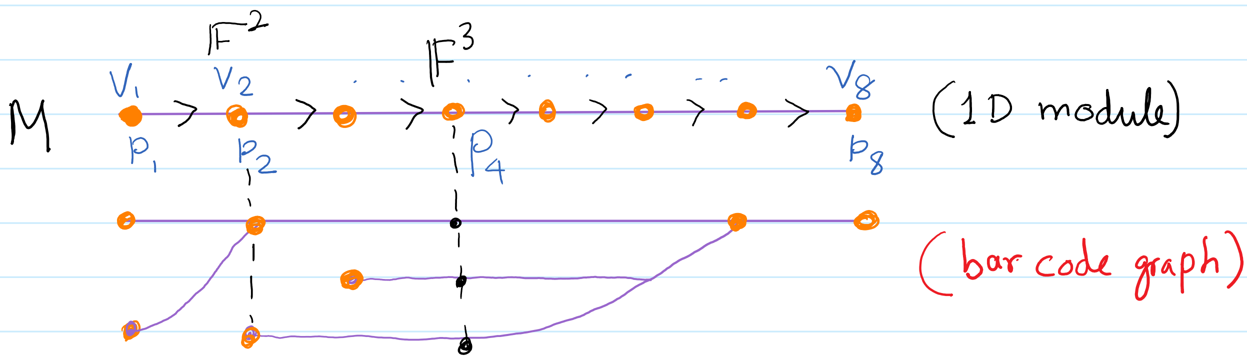


- Interval modules
  - $\mathbb{I}_I : I \longrightarrow \mathbf{vec}$

$$\mathbb{I}_I(p) := \begin{cases} \mathbb{F} & \text{if } p \in I \\ 0 & \text{otherwise} \end{cases}$$

Stuctural maps $\varphi_{\mathbb{I}_I}$ are identities

Z - modules



- Int(P) : set of all intervals
- rank invariant

$$rk(M): Int(P) \longrightarrow \mathbb{Z}$$

$$rk(M)([b,d]) = rank(\varphi_{b,d} : V_b \rightarrow V_d)$$

$rk(M)([p_2, p_4]) = 2$ in the figure

- Maximal interval

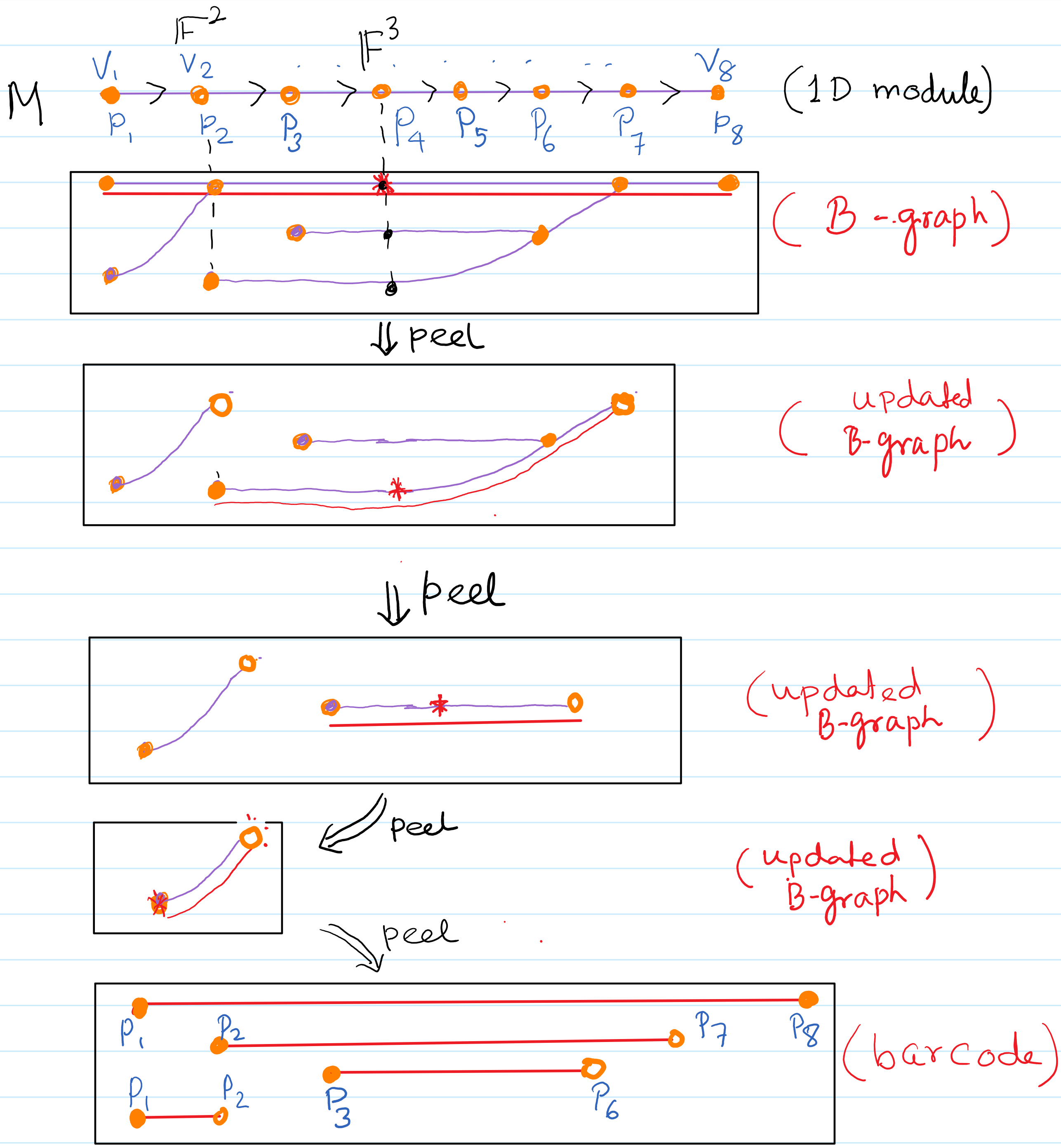
$I = [p_i, p_j]$ is maximal if

$\nexists\, J \supsetneq I$ s.t. $rk(M)(J) > 0$

Algorithm MaxInterval - Peeling

While $M \neq 0$ peel $\mathbb{I}_I$ where I is maximal

Peeling Maximal Intervals



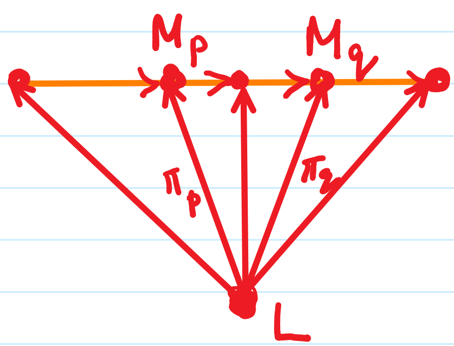
Generalization

Generalization

- Given $I \subseteq P$, define generalized rank
  - $\mathrm{rank}(M_I) = \mathrm{rank}(\lim M_I \to \mathrm{Colim}\, M_I)$

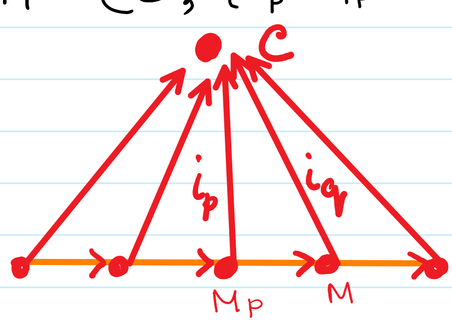
    [Patel, Kim, Mémoli]

  - $\lim M = (L, \{\pi_p : L \to M\}_{p \in P})$, $M : P \to \mathrm{vec}$



$$\pi_q = \varphi_M(p \le q) \circ \pi_p$$

  - $\mathrm{Colim}\, M = (C, \{i_p : M_p \to C\}_{p \in P})$



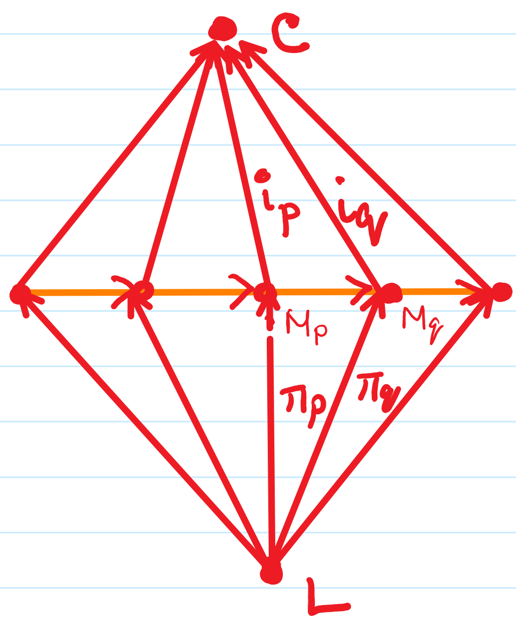
$$i_p = i_q \circ \varphi_M(p \le q)$$

Generalization (cont.)

limit to colimit

- $i_p \circ \pi_p = i_q \circ \pi_q \quad \forall p \le q$

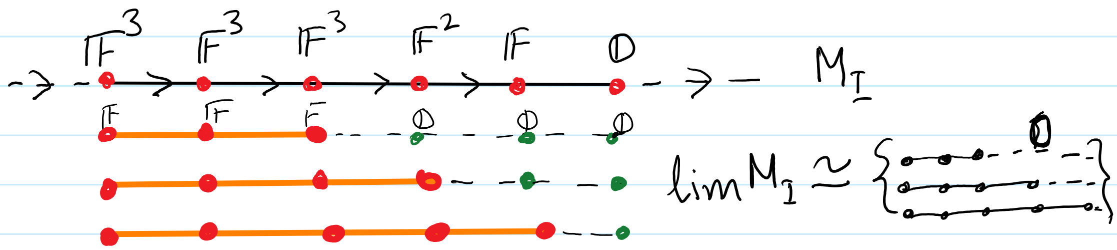
- $\operatorname{rank}(\overset{L}{\lim M} \longrightarrow \overset{C}{\operatorname{colim} M}) = \operatorname{rank}(i_p \circ \pi_p)$

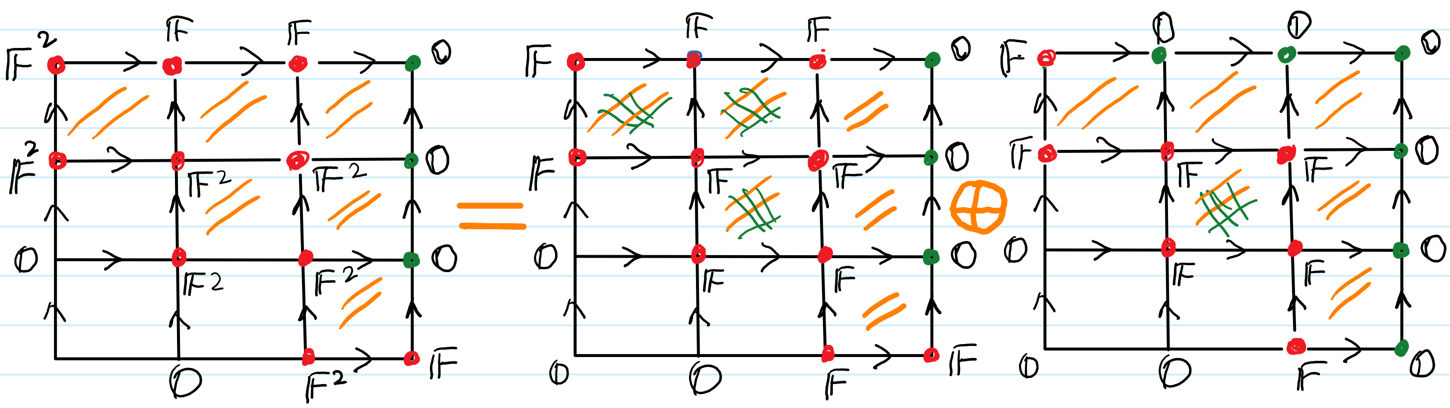


Canonical construction

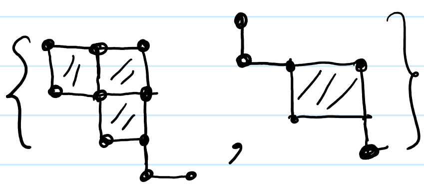
Canonical Construction of Lim & Colim

- $W := \{ (v_p)_{p\in P} \in \bigoplus_{p\in P} M_p \mid \forall p \le q,\ v_p \sim v_q \}$



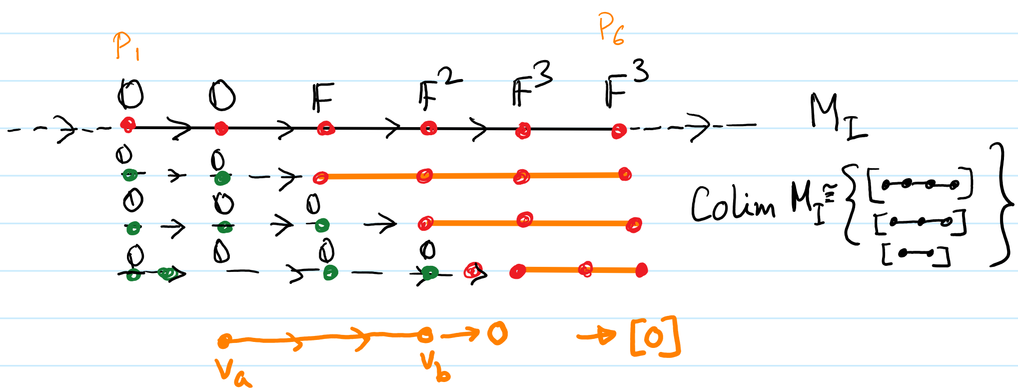


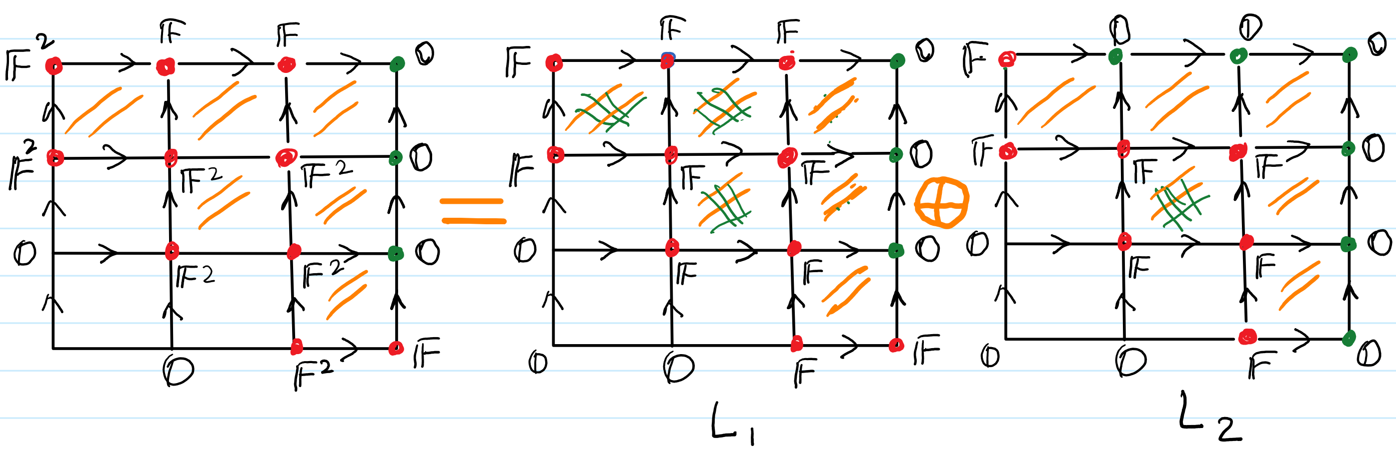
$W :=$



Canonical Construction of Colimit

- $U = \{ [v_p] \mid v_p \approx v_q \text{ if } p \le q,\ v_p \sim v_q \}$

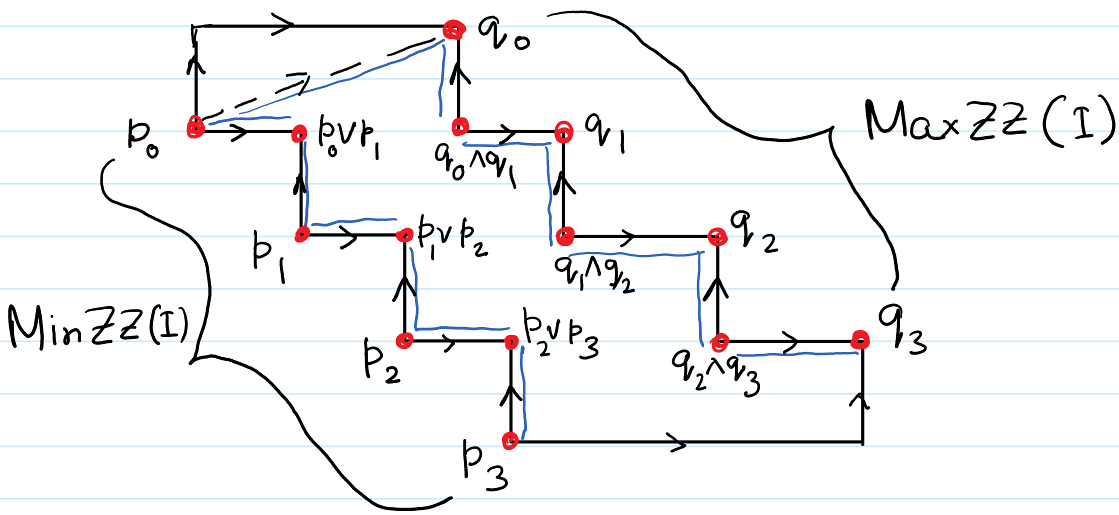




- Let us consider the previous limit construction
- Both sections L_1 and $L_2 \Rightarrow [0]$ in Colim
- $\text{Colim}\, M_I = \{ [0] \}$

Generalized rank and boundary zigzag

- Boundary cap ∂I
    - $\partial I := \underbrace{p_k < (p_k \vee p_{k-1}) > p_{k-1} < \cdots > p_0}_{MinZZ(I)} \leq \underbrace{q_0 > (q_0 \wedge q_1) < q_1 \cdots < q_\ell}_{MaxZZ(I)}$

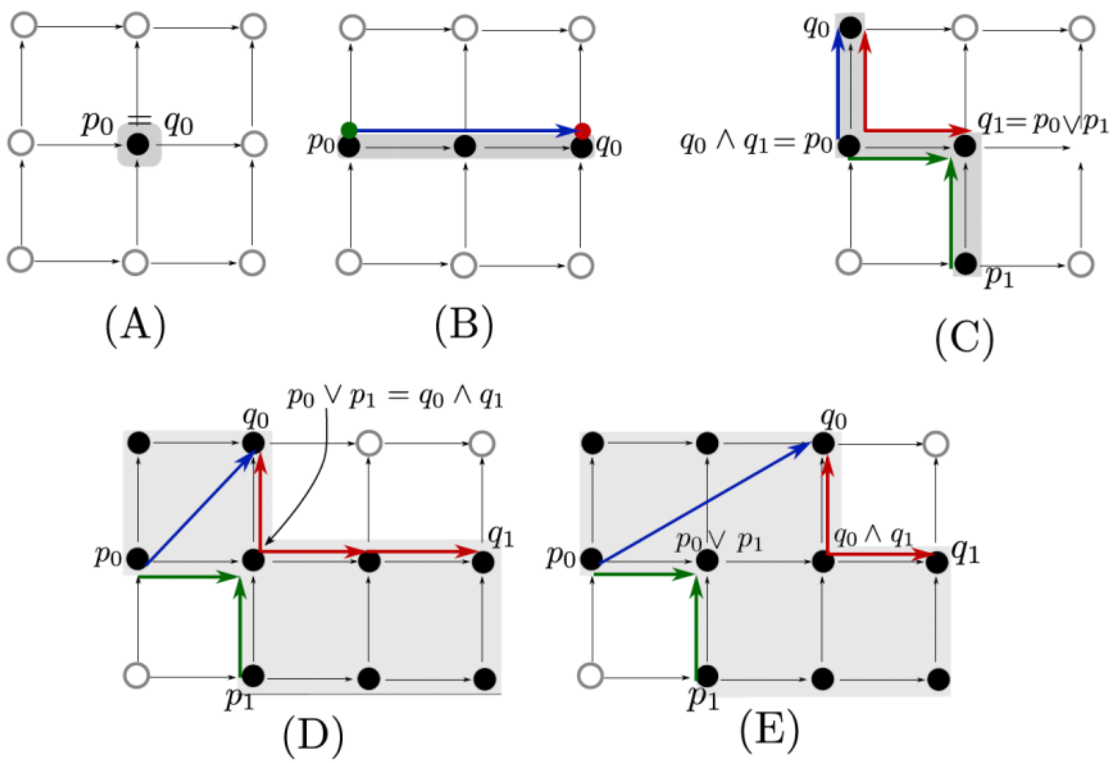


- $ZZ_{\partial I}$: Zigzag Poset corresponding to ∂I
    - $M_{\partial I}$: Zigzag module, $M_{\partial I} : ZZ_{\partial I} \to \mathbf{Vec}$

      $(M_{\partial I})_p := M_p$, $\varphi_{M_{\partial I}}(p,q) := \varphi_M(p \leq q)$

More Examples

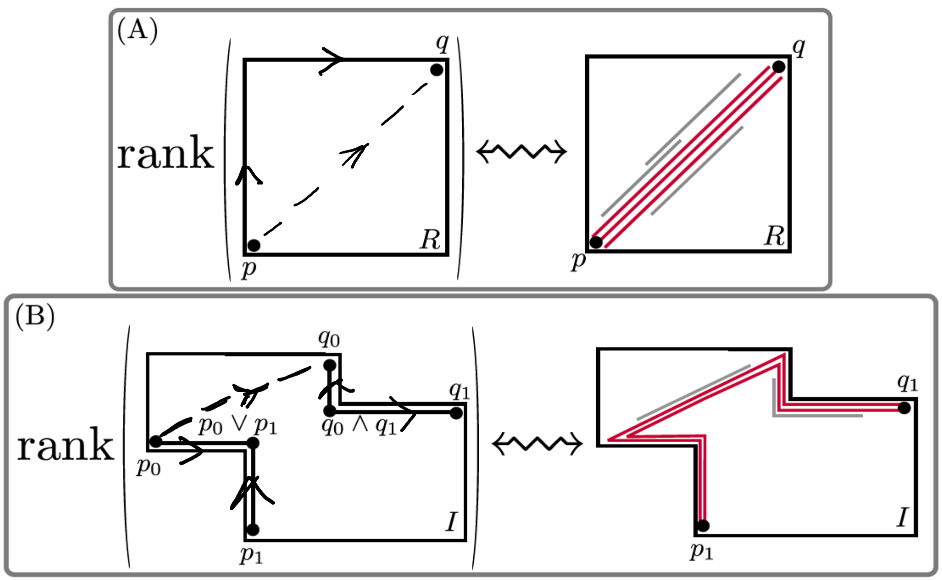
More Examples of boundary cap



Generalized rank via zigzag persistence

Thm: $\mathrm{rank}(M_I) = \mathrm{rank}(M_{\partial I})$

- $\mathrm{rank}(M_{\partial I})$ = # full bars in the decomposition of $M_{\partial I}$

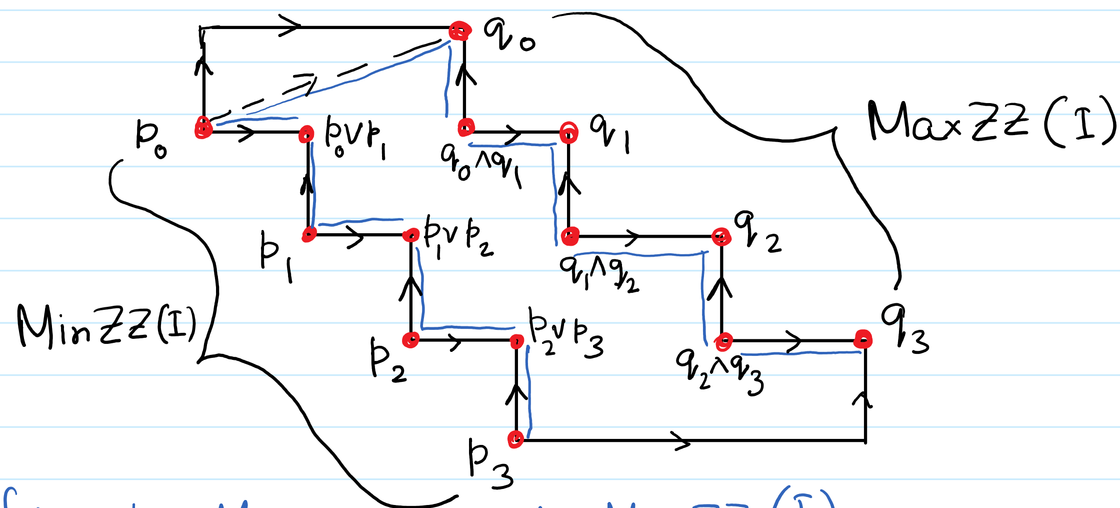


Algorithm GRank(M_I)

$O(t^{\omega})$

- Compute ∂I and $F_{\partial I}$ (Filtration F induces M)
- Compute persistence from $F_{\partial I}$
- Output # full bars.

Thm: $\mathrm{rank}(M_I) = \mathrm{rank}(M_{\partial I})$



proof: $L = \mathrm{MinZZ}(I)$, $U = \mathrm{MaxZZ}(I)$

Claim: $\lim M_L \xrightarrow[\simeq]{e} \lim M_I$, $\mathrm{colim}\, M_U \xrightarrow[\simeq]{i} \mathrm{colim}\, M_I$

$$\begin{tikzcd}
\lim M_L \arrow[r, "\xi"] \arrow[d, "e"', "\simeq"] & \mathrm{colim}\, M_U \arrow[d, "i", "\simeq"'] \\
\lim M_I \arrow[r, "\psi_{M_I}"] & \mathrm{colim}\, M_I
\end{tikzcd}$$

$\boxed{\xi = i^{-1} \circ \psi_{M_I} \circ e}$

- $\mathrm{rank}(M_I) := \mathrm{rank}(\psi_{M_I}) = \mathrm{rank}(\xi)$
- Sufficient to show $\mathrm{rank}(\psi_{M_{\partial I}}) = \mathrm{rank}(\xi)$
- $\exists\; f : \lim M_{\partial I} \to \lim M_L$, $g : \mathrm{colim}\, M_U \to \mathrm{colim}\, M_{\partial I}$

$$\psi_{M_{\partial I}} = g \circ \xi \circ f$$

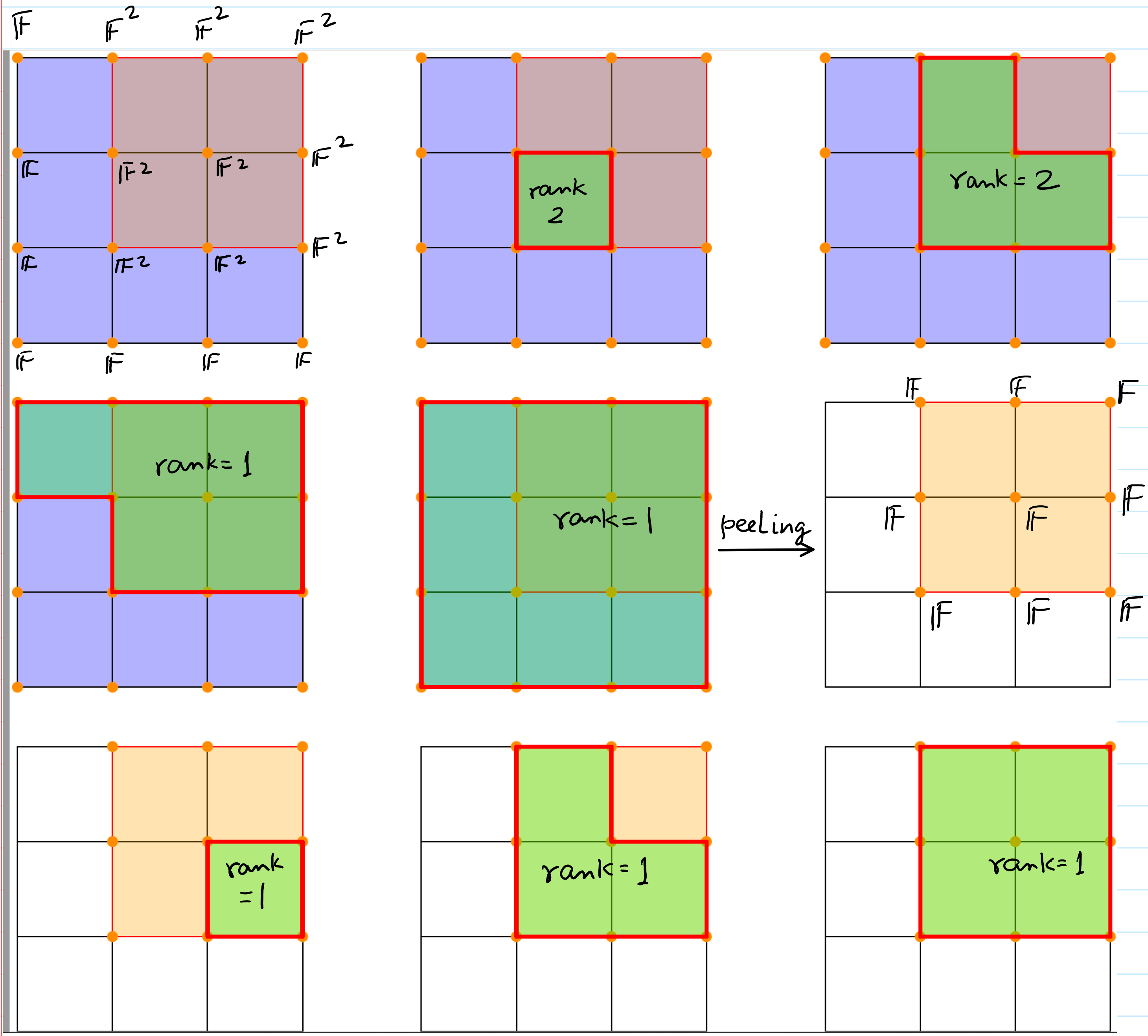
- f is surjective, g is injective

$$\Downarrow$$

$$\mathrm{rank}(\psi_{M_{\partial I}}) = \mathrm{rank}(\xi) = \mathrm{rank}(\psi_{M_I}) \quad \square$$

Algorithm Interval

Illustration Algorithm Interval



Algorithm Interval

Interval (F, P): F a bifiltration on P, M_F induced

While $\exists p \in P$ s.t. $\dim(M_F)_p > 0$

$I := p$;

while $\mathrm{rank}(M_{\partial I}) > 0$

expand I

endwhile

Output I with multiplicity $\mathrm{rank}(M_{\partial I})$

→ Peel $\mathbb{I}_I$ from M_F

endwhile

- → peeling is only simulated
- Actual peeling requires "quotienting"
- Avoid costly 'quotienting' with a "fake peeling"

Algorithm Interval

- $t = \max\{\#\text{ simplices in } F,\ \#\text{ points in } P\}$

Thm: Interval (F,P) returns all interval summands of M_F if M_F is interval decomposable.

$\Downarrow$

[Azumaya-Krull-Remak-Schmidt theorem]

$M_F = \bigoplus I_i$ (Each indeomposable is interval)

- Algorithm Interval runs in $O(t^{\omega+2})$ time where $\omega \in [2, 2.373]$ is the exponent of matrix multiplication.

Interval Decomposability

- Given a module M_F, determine if M_F is interval decomposable
  - Use Decomposition algorithm [D.-Xin 19] to compute $M_F = \bigoplus M_i$
  - Test if each M_i is interval
  - takes $O(t^{2\omega+1})$ time
  - Caveat: M_F needs to distinctly graded
  - Use [Asashiba18] et al. algorithm
  - Caveat: takes time exponential in t
  - Our algorithm solves the problem in $O(t^{3\omega+2})$ time without any assumption about distinct grading

Interval Decomposability

- [Asashiba et al.] enumerated exponentially many intervals to test
- We need to test only $O(t^2)$ intervals giving a polynomial time algorithm.

IsInterval (F, P)

for each $I \leftarrow$ Interval(F, P) do
test if $\mathbb{I}_I$ is an interval summand of M_F
- If 'no' output 'false'; quit

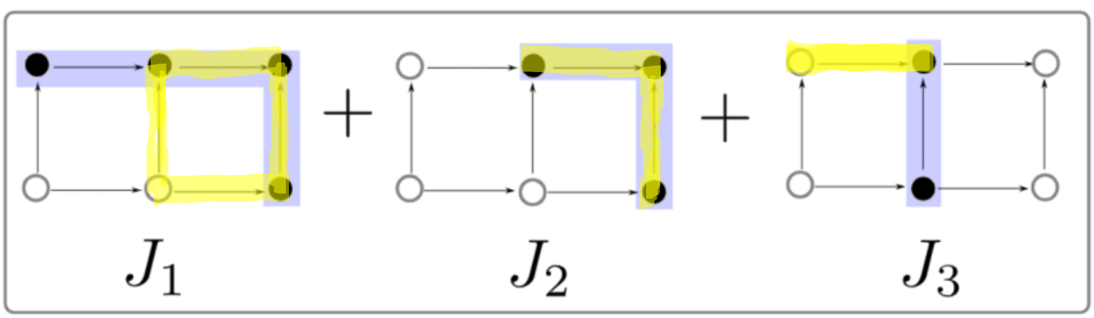
endfor
Output Interval (F, P)

Thm: IsInterval (F, P) runs in time $O(t^{3\omega+2})$

Interval Revisited

- What happens when M_F in **Interval**(M_F, P) is not interval decomposable

$$
N \cong
\underbrace{\begin{tikzcd}
\mathbb{F} \arrow[r, "\binom{1}{1}"] & \mathbb{F}^2 \arrow[r, "(1\ 0)"] & \mathbb{F} \\
0 \arrow[r] \arrow[u] & \mathbb{F} \arrow[r, "1"] \arrow[u, "\binom{1}{0}"] & \mathbb{F} \arrow[u, "1"]
\end{tikzcd}}_{N'}
\oplus
\underbrace{\begin{tikzcd}
0 \arrow[r] & \mathbb{F} \arrow[r, "1"] & \mathbb{F} \\
0 \arrow[r] \arrow[u] & 0 \arrow[r] \arrow[u] & \mathbb{F} \arrow[u, "1"]
\end{tikzcd}}_{\mathbb{I}^{I_2}}
$$



Two possible intervals (,) returned by **Interval**

Computed Intervals

Thm: Intervals $\mathcal{I}$ computed by Interval(F, P):

- I: subset of support of a submodule of an indecomposable M_j when $M_F = M_j \oplus M_F'$
- $\sum_{I \in \mathcal{I}} \dim(\mathbb{I}_I)_p = \dim(M_F)_p$

Conclusion

- Efficient algorithm for rank(M_I)

  What about d-parameter modules, $d>2$

- Efficient algorithm for intervals

  Can this be improved from $O(t^{\omega+2})$?

- Efficient algorithm for interval decomposability

  D.-Xin algorithm more efficient than $O(t^{3\omega+2})$

  bottleneck is testing with Asashiba et al. algo

  Can the testing be improved?

- Application of the generalized rank computation

  Is there any other use?