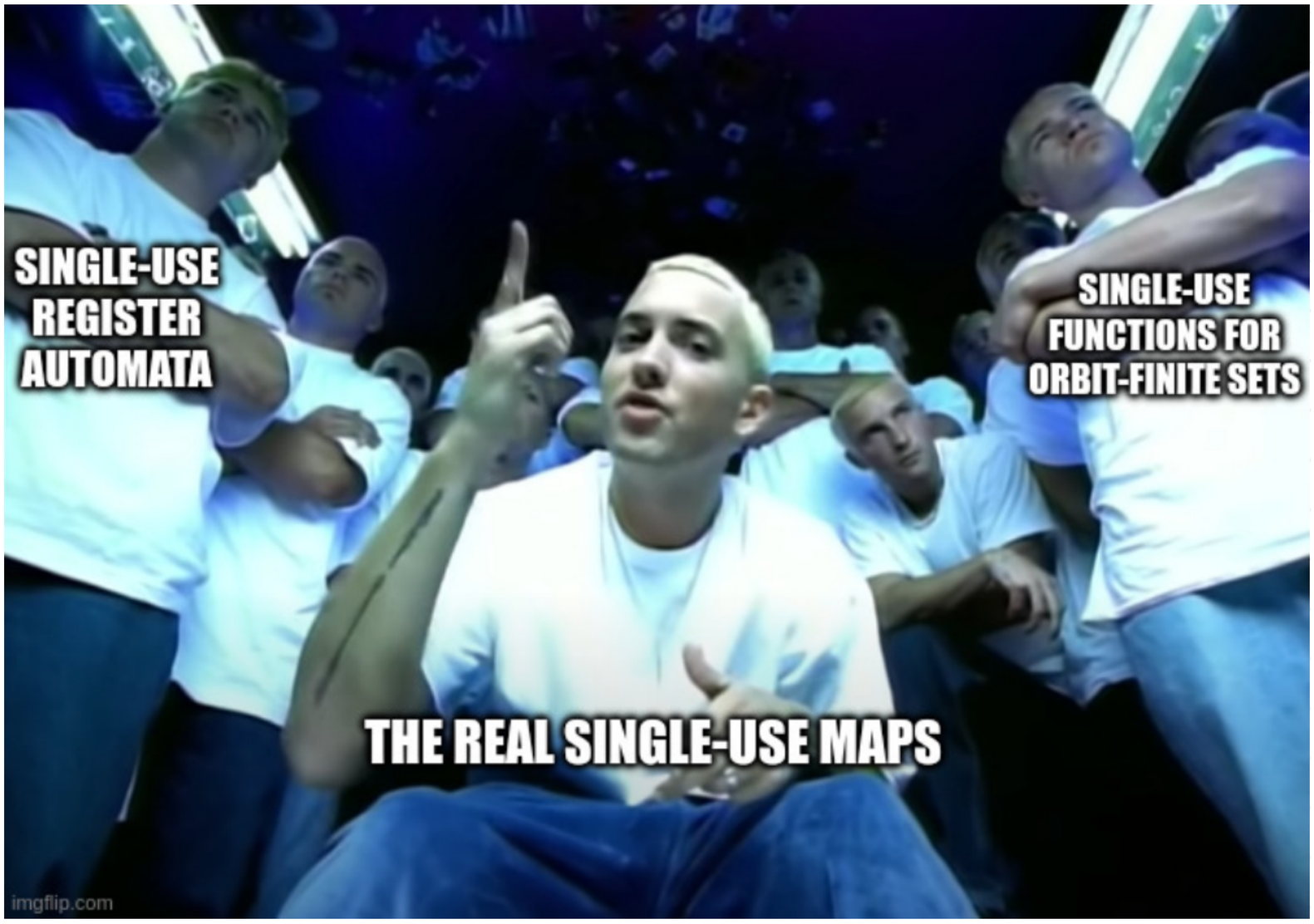
„My Name is" - The Real S...ingle-Use Maps in Nominal (Renaming) Sets

Joint Work with Fabian



F. Lenke und S. Prucker

Agenda:

- Recap: nominal (renaming) sets

- Multiplicities: definition, examples

- Single-use maps: definition, properties, ideas

F. Lenke und S. Prucker

Motivation (extrinsic):

- Volatility of information
  - Destructive usage of keys and passwords in cryptography
  - Dynamic RAM (generally) has destructive readout
  - Feed-forward-networks internal values
- Task: identify computations on volatile data

Motivation (intrinsic):

- In nominal sets:
  least finite support ≃ occuring names
- But how often? ¯_(ツ)_/¯
- Maybe useful for defining nested allocation

Related Work:

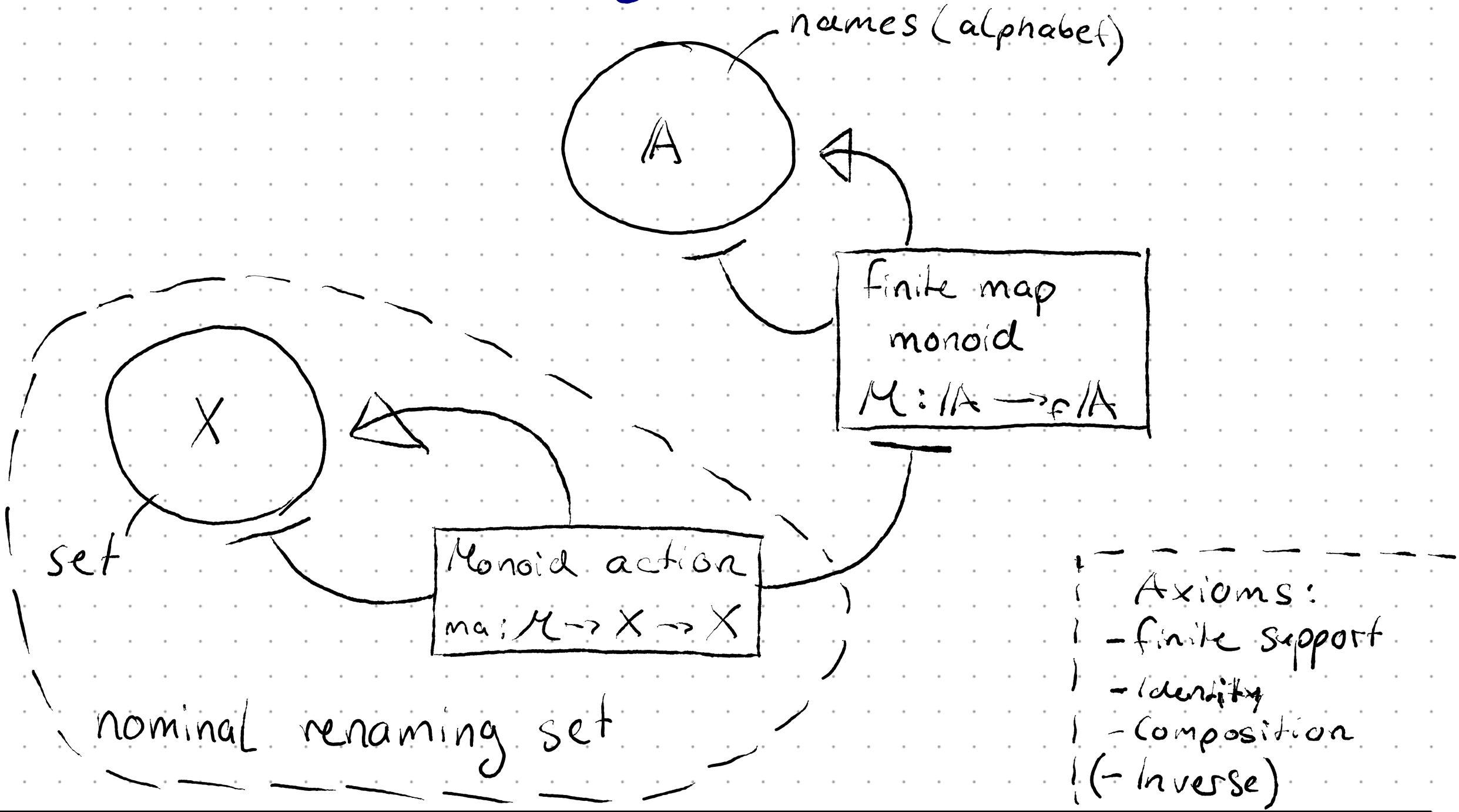
- Equivalence of Single-Use Register Automata & Nominal Monoids[+]
  - Translation via 2-way transducers and forests
  - Nominal Monoid is not the state transition monoid
- Identification of single-use functions on orbit-finite nominal sets[++]
  - Syntactic definition as constructors for functions over special types
  - Translation to maps in Nom, no axioms

++ Boyańczyk, Nguyễn, Stefański

+ PhD Thesis R. Stefanski

F. Lenke und S. Prucker

(Nominal) Renaming Sets [GH'08]



F. Lenke und S. Prucker

Nom vs. Ren:

- Inverses (obviously)
  - $abba \underset{(ab)}{\overset{(ab)}{\rightleftarrows}} baab$ in Nom | $abba \overset{a}{\underset{?}{\overset{\rightarrow}{\not\leftarrow}}} aaaa$ in Ren
- In Ren, actions can decrease supports

$$X = \mathbb{A}^{\#2} + \mathbb{1}^2 \qquad \pi \cdot (a,b) \longmapsto \begin{cases} (\pi(a), \pi(b)), & \text{if } \pi(a) \neq \pi(b) \\ 1, & \text{otherwise} \end{cases}$$

- Each renaming set is a nominal set

$$\mathbf{Ren} \underset{U}{\overset{F}{\underset{\curvearrowright}{\overset{\curvearrowleft}{\perp}}}} \mathbf{Nom}$$

Free Renaming Functor [HR'19]

$$FX = \{(x, \underbrace{\sigma : \mathrm{supp}(x) \to \mathbb{A}}) \mid x \in X\} / \sim$$

nominal element

synthetic renaming

with $\underbrace{(\pi \cdot x, \sigma) \sim (x, \sigma \circ \pi)}$

identifying equal elements (e.g. $\mathcal{P}_f(\mathbb{A})$)

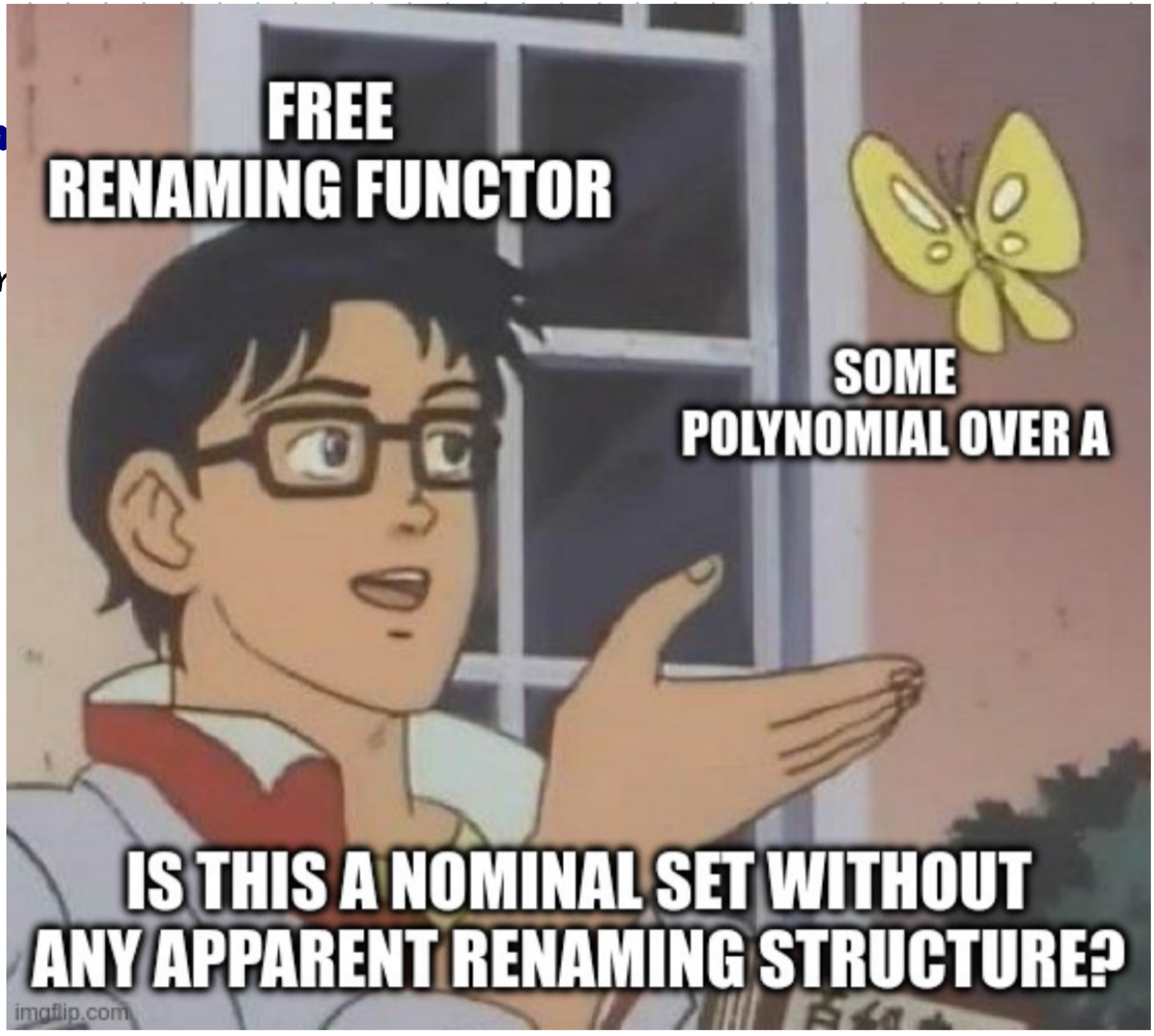
F. Lenke und S. Prucker

Example

- Nominal Set $X = \mathbb{A}^3$ with pointwise action

Exan

– Nom



F. Lenke und S. Pucker

Example

- Nominal Set $X = \mathbb{A}^3$ with pointwise action
- Also has obious renaming structure, but F does not „see" that
- F turns separated products into cortesians ($F\mathbb{A}^{\#3} = \mathbb{A}^3$)

— $X = \mathbb{A}^3 \simeq \mathbb{A}^{\#3} + 3\mathbb{A}^{\#2} + \mathbb{A}$

↳ $FX \simeq \mathbb{A}^3 + 3\mathbb{A}^2 + \mathbb{A}$

Multiplicities

F. Lenke und S. Pucker

Multiplicities:

- Properties of canonical polynomials over $\mathbb{A}$ as expected

- Def.: $\mu_a(x \in X) = \sup \{ |\rho^{-1}(a) \cap \underbrace{\mathrm{supp}(y)}_{\sim \text{counit of } FU}| \; | \; y \in X, \rho \cdot y = x \}$

$\hookrightarrow$ Equivalently: $\mu_a(x \in X) = \sup \{ |\sigma^{-1}(a)| \; | \; [\sigma]_y \in FUX, \varepsilon_x([\sigma]_y) = x \}$

Single-use maps

- Nom-equivariant map $f: X \to Y$ for $X, Y \in \mathbf{Ren}$
- 3 necessary and jointly sufficient conditions:
  - copyless
  - compare-delete
  - single-compare

→ Let's have a Look individually

Copyless

- Def.: $f: X \to Y$ is *copyless* if for all $x \in X$ and $a \in \mathbb{A}$, we have $\mu_a(x \in X) \geq \mu_a(f(x) \in Y)$

- Examples:
  - Copyless: projections, comparators, pseudo-diagonals
  - Copy: diagonals, inclusions
- What about those? 

F. Lenke und S. Prucker

Example

important!

- Map $f : \mathbb{A} \to \mathbb{A}^{2,=}$ given by $a \longmapsto (a,a)$

  - Pseudo-diagonal is single-use: $\mu_a(a \in \mathbb{A}) = 1 = \mu_a((a,a) \in \mathbb{A}^{2,=})$
  - Inclusion $\mathbb{A}^{2,=} \hookrightarrow \mathbb{A}^2$ is <u>not</u> single-use

  ↳ Diagonal $\mathbb{A} \to \mathbb{A}^2$ is also not single-use

- In syntactic single-use functions solved with & constructor

---

F. Lenke und S. Prucker

Single-Compare

- Def.: $f: X \to Y$ is *single-compare*, if for all $x \in X$ and $a, b, c \in A$, we have

$$\underbrace{\tfrac{a}{b} f(x) = f\left(\tfrac{a}{b} x\right)}_{f \text{ when renaming } b \text{ to } a} \iff \underbrace{\left(\tfrac{c}{a} f\left(\tfrac{c}{b} x\right) = f\left(\tfrac{c}{ab} x\right) \;\vee\; \tfrac{c}{b} f\left(\tfrac{c}{a} x\right) = f\left(\tfrac{c}{ba} x\right)\right)}_{f \text{ when renaming both } a \text{ and } b \text{ to some } c}$$

- Double-compare: $(x,y,z) \longmapsto \begin{cases} 1 & \text{if } x = y \wedge y = z \\ (x,y,z) & \text{otherwise} \end{cases}$

F. Lenke und S. Prucker

Compare-Delete

- Def.: f is compare-delete, if for all maximal x, $a, b \in /A$ and $r \# a, b$ such that $r \cdot x = x'$ we have

$$\frac{a}{b} f(x') \neq f\left(\frac{a}{b} x'\right)$$

path in computation

implies

some comparison

$$a, b \# f(x'), f\left(\frac{a}{b} x'\right)$$

a and b are deleted

Alternative Definition:

- f is compare-delete, if for all x and $a, b \in A$, we have that $\frac{a}{b} f(x) \neq f(\frac{a}{b} x)$ implies

$$\mu_a(\tfrac{a}{b} f(x)) < \mu_a(\tfrac{a}{b} x) > \mu_a(f(\tfrac{a}{b} x))$$

and

$$\mu_a(x) > \mu_b(f(x))$$

Characterisation

- Single-use maps can be curried into a sequence of binary maps
- Prop.: Let $f : \mathbb{A}^n \to Y$ be a Nom-equivariant map. Then, f is single-use if and only if
  1. f is Ren-equivariant and copyless, or
  2. there is $\hat{f} = curry(f) : \mathbb{A}^2 \to \mathbb{A}^{n-2} \to Y$ such that for each $a, b \in \mathbb{A}$, $\hat{f}(a,b)$ is Nom-equivariant and single-use.

F. Lenhe und S. Pucker

Finitely Supported Single-Use Maps

- Nom-finitely-supported map $f : X \to_{fs} Y$
- Parametrise the support
- Def.: f is single-use if there is set S, Nom-equivariant single-use map $f' : \mathbb{A}^S \times X \to Y$ and configuration $\omega \in \mathbb{A}^S$ such that for all $x \in X$ we have $f'(\omega, x) = f(x)$.
- Op.: Can we further qualify S and $\omega \in \mathbb{A}^S$?
  E.g. for renaming? $\pi \cdot f \stackrel{?}{=} f'(\pi \cdot \omega)$

Single-use functions [BNS'24]

- Syntactic definition
- Primitives 1 and !A, constructors +, × and &
- Translation to functions in nominal sets

F. Lemke und S. Prucker

Function	**Type**	**Definition**
Functions about $\mathbb{A}$		
equality test	$\mathbb{A} \times \mathbb{A} \to 1 + 1$	$(a, b) \mapsto a == b$
constant a	$1 \to \mathbb{A}$	$x \mapsto a$
identity	$\mathbb{A} \to \mathbb{A}$	$x \mapsto x$
Functions about $\otimes$		
left projection	$X \otimes Y \to X$	$(x, y) \mapsto x$
right projection	$X \otimes Y \to Y$	$(x, y) \mapsto y$
append 1	$X \to X \otimes 1$	$x \mapsto (x, \heartsuit)$
associativity	$X \otimes (Y \otimes Z)$ $\to (X \otimes Y) \otimes Z$	$(x, (y, z))$ $\mapsto ((x, y), z)$
commutativity	$X \otimes Y \to Y \otimes X$	$(x, y) \mapsto (y, x)$
Functions about $+$		
left injection	$X \to X + Y$	$x \mapsto l\, x$
right injection	$Y \to X + Y$	$y \mapsto r\, y$
co-diagonal	$X + X \to X$	$l\, x \mapsto x,$ $r\, x \mapsto x$
associativity	$X + (Y + Z)$ $\to (X + Y) + Z$	$l\,(l\, x) \mapsto l\, x,$ $r\,(r\, x) \mapsto r\, x,$ $r\,(l\, x) \mapsto l\,(r\, x)$
commutativity	$X + Y \to Y + X$	$l\, x \mapsto r\, x,$ $r\, x \mapsto l\, x$
Functions about $\&$		
diagonal	$X \to X \& X$	$x \mapsto x \& x$
left projection	$X \& Y \to X$	$x \& y \mapsto x$
right projection	$X \& Y \to Y$	$x \& y \mapsto y$
Distributivity		
$+$ over $\otimes$	$X \otimes (Y + Z) \to$ $(X \otimes Y) + (X \otimes Z)$	$(x, l\, y) \mapsto l\,(x, y),$ $(x, r\, z) \mapsto r\,(x, z)$
$\&$ over $\otimes$	$X \otimes (Y \& Z) \to$ $(X \otimes Y) \& (X \otimes Z)$	$(x, y \& z) \mapsto$ $(x \otimes y) \& (x \otimes z)$
$\&$ over $+$	$X + (Y \& Z) \to$ $(X \& Y) + (X \& Z)$	$x \& (l\, y) \mapsto l\, x \& y,$ $x \& (r\, z) \mapsto r\, x \& z$

F. Lenke und S. Prucker

Single-Use Register Automata [BS20]

▶ **Definition 3.** *The syntax of a* two-way single-use transducer[2] *consists of*

- *input and output alphabets* Σ *and* Γ*, both polynomial orbit-finite sets;*
- *a finite set of states* Q*, with a distinguished initial state* $q_0 \in Q$*;*
- *a finite set* R *of register names;*
- *a* transition function *which maps each state* $q \in Q$ *to an element of:*

$$\underbrace{\textit{questions}}_{\textit{question that is asked}} \times \underbrace{(Q \times \textit{actions})}_{\substack{\textit{what to do if the} \\ \textit{question has a yes answer}}} \times \underbrace{(Q \times \textit{actions})}_{\substack{\textit{what to do if the} \\ \textit{question has a no answer}}}$$

where the allowed questions and actions are taken from the following toolkit:

1. Questions.
   - **a.** *Apply an equivariant function* $f : \Sigma + \{\vdash, \dashv\} \to \{\textit{yes}, \textit{no}\}$ *to the letter under the head, and return the answer.*
   - **b.** *Are the atoms stored in registers* r_1, r_2 *equal and defined? If any of these registers is undefined, then the run immediately stops and rejects*[3]*.* ***This question has the side effect of setting the values of*** r_1 ***and*** r_2 ***to*** $\bot$.
2. Actions.
   - **a.** *Apply an equivariant function* $f : \Sigma + \{\vdash, \dashv\} \to \mathbb{A} + \bot$ *to the letter under the head, and store the result in register* $r \in R$.
   - **b.** *Apply an equivariant function* $f : \mathbb{A}^k \to \Gamma$ *to the contents of distinct registers* $r_1, \ldots, r_k \in R$*, and append the result to the output string. If any of the registers is undefined, stop and reject.* ***This action has the side effect of setting the values of*** $r_1, r_2, \ldots, r_k$ ***to*** $\bot$.
   - **c.** *Move the head to the previous/next input position.*
   - **d.** *Accept/reject and finish the run.*

F. Lenke und S. Pucker

Nominal Single-Use Automata

- Maybe equivalent to nominal automata
- Transition function $\delta : Q \to_{\mathrm{Su}} \underbrace{(\mathbb{A} \to_{\mathrm{fs}} Q)}_{\in \mathrm{Ren}}$
- Translation to nominal monoid $m : \mathbb{A} \to_{\mathrm{eq}} \underbrace{(Q \to_{\substack{\mathrm{su} \\ \mathrm{fs}}} Q)}_{\text{Renaming Set?}}$

F. Lenke und S. Prucker

References

- Function Spaces for Orbit-Finite Sets, Bojańczyk et al., 2024
- Nominal Renaming Sets, Gabbay & Hofmann, 2008
- Separation and Renaming in Nominal Sets, Moerman & Rot, 2019
