Numeric Domains meet Algebraic Data Types

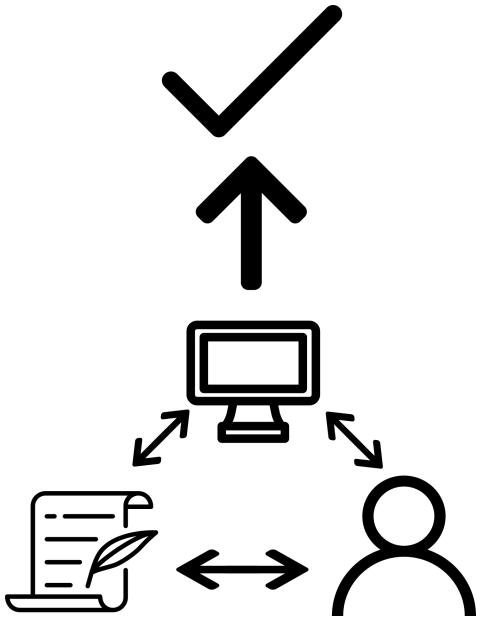
NSAD @ SPLASH 2020

Santiago Bautista Thomas Jensen Benoit Montagu

November 17, 2020

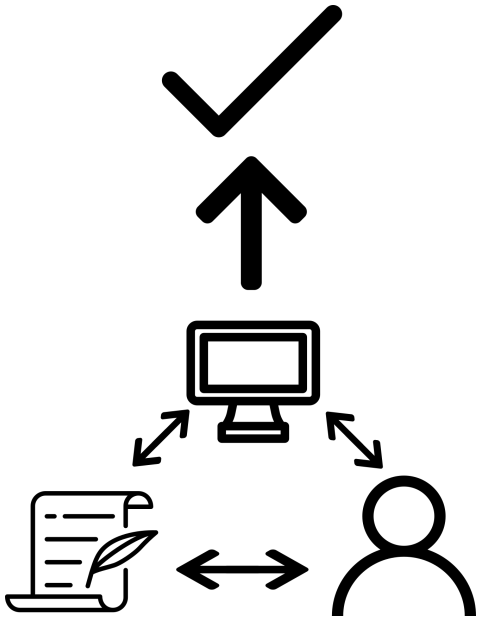
Motivation

Interactive Theorem Proving



Motivation

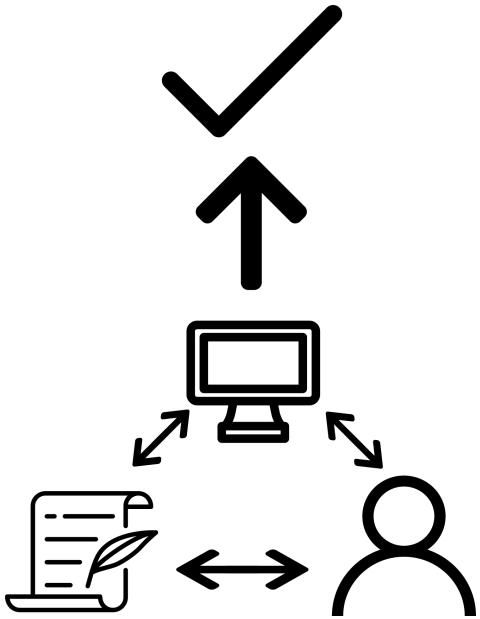
Interactive Theorem Proving



Requires manual effort

Motivation

Interactive Theorem Proving



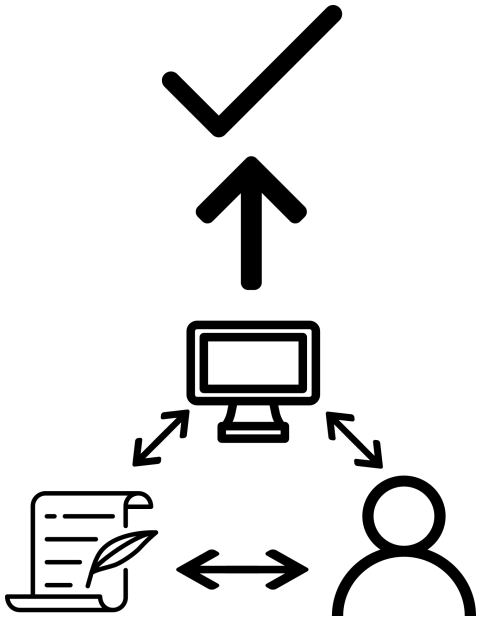
Requires manual effort

Static Analysis

Can alleviate manual effort

Motivation

Interactive Theorem Proving



Requires manual effort

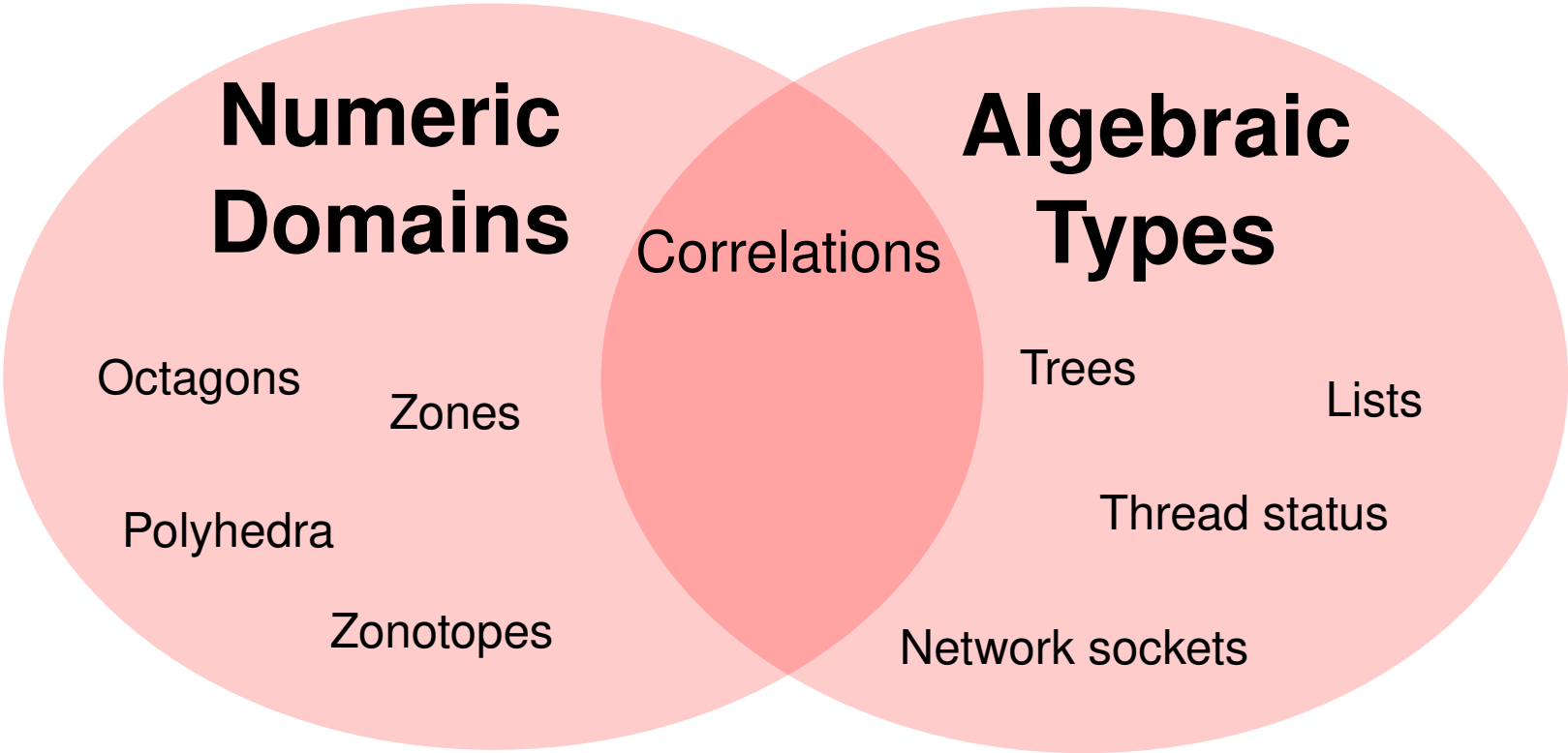
Static Analysis

Can alleviate manual effort

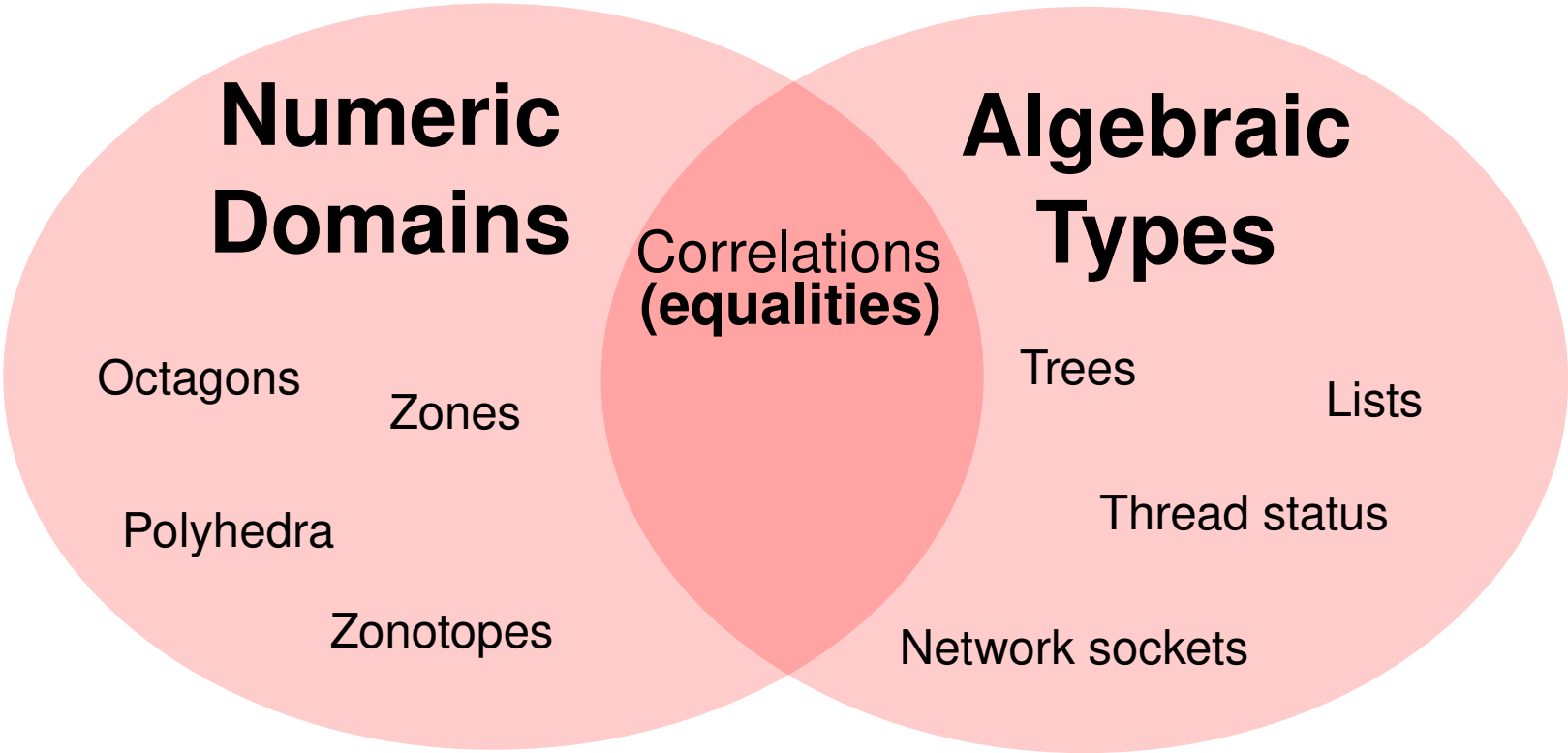
Example

ProvenCore & Correlations : $\frac{2}{3}$ proof obligations discharged

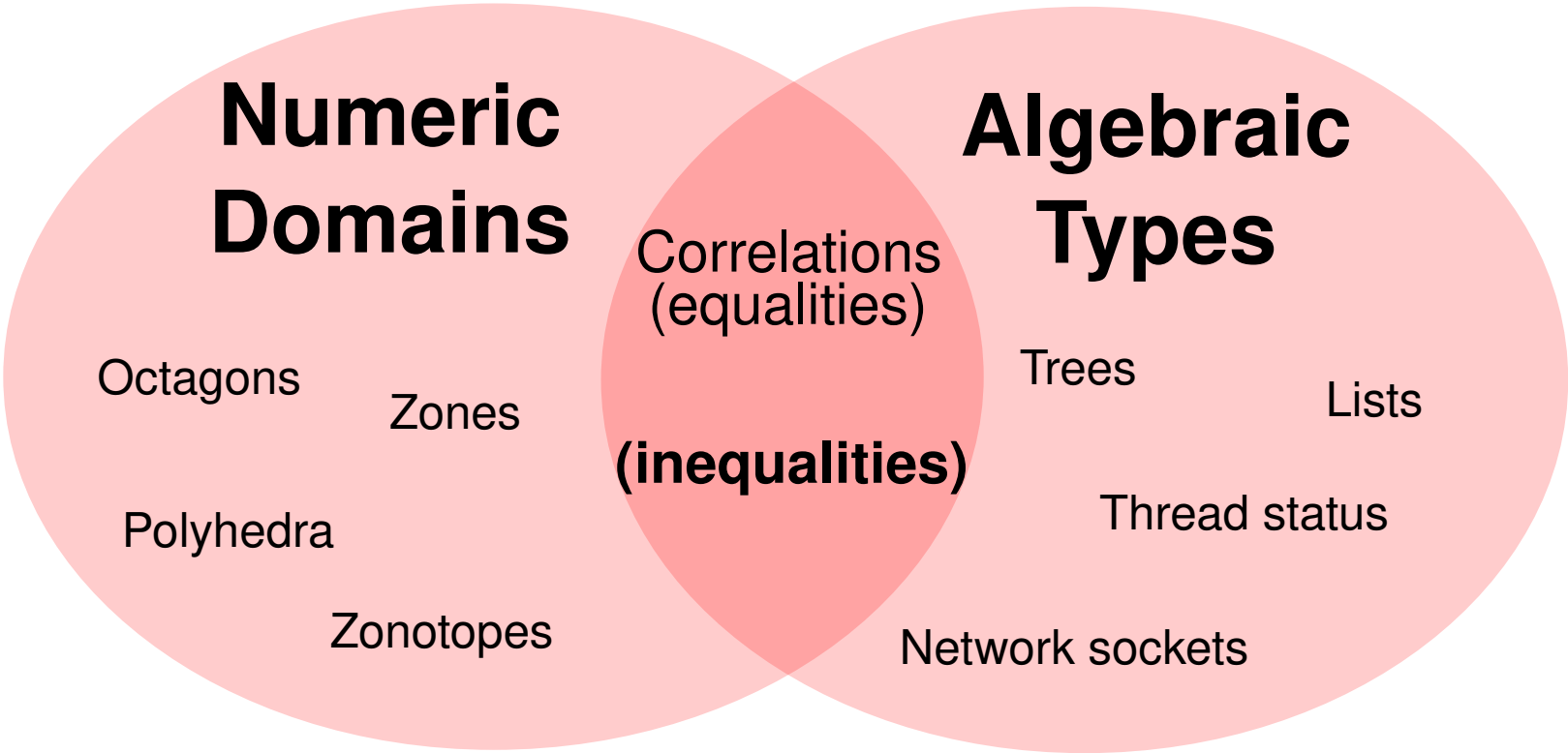
Overview


Numeric Domains
Octagons
Zones
Polyhedra
Zonotopes
Correlations
Algebraic Types
Trees
Lists
Thread status
Network sockets

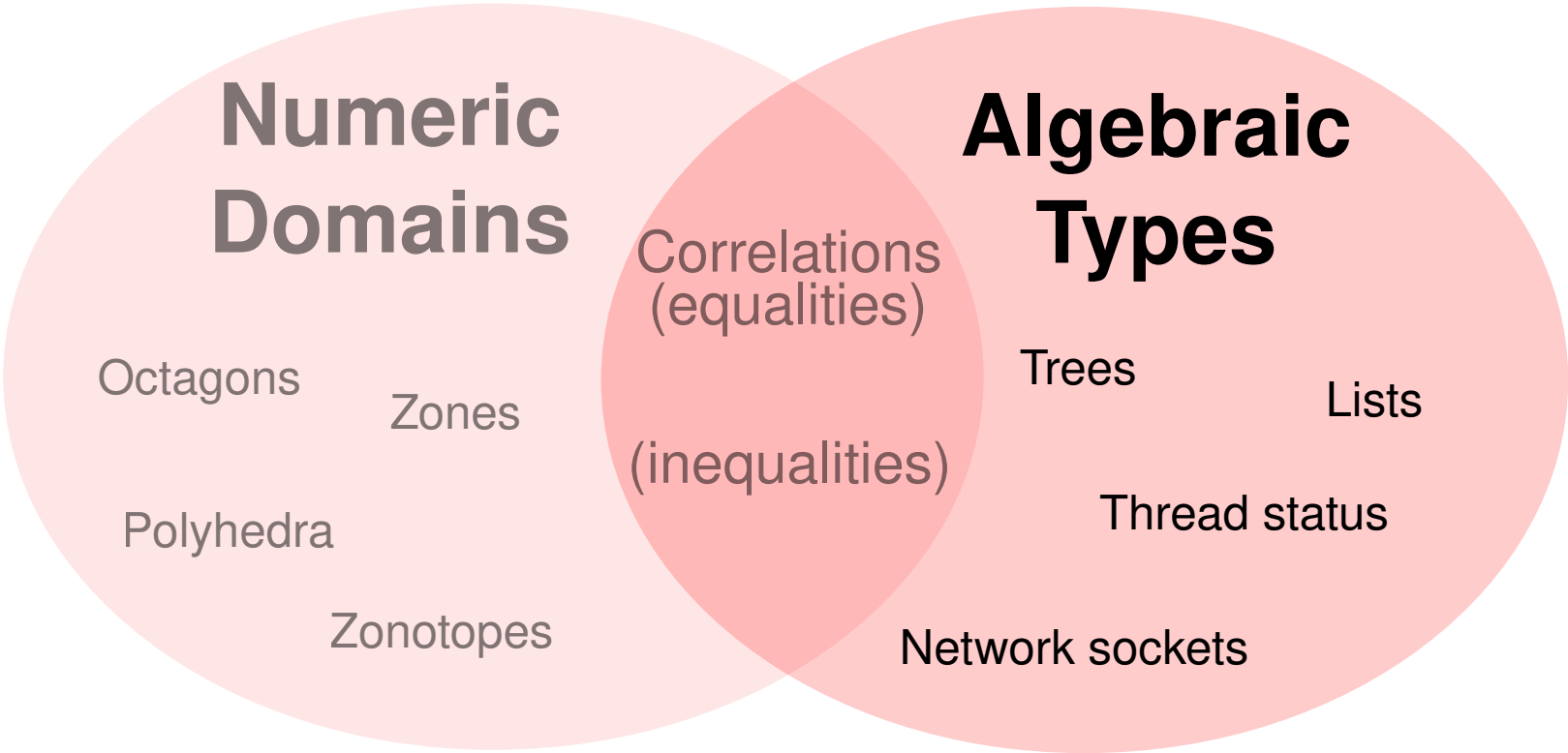
Overview


Numeric Domains
Octagons
Zones
Polyhedra
Zonotopes
Correlations
(equalities)
Algebraic Types
Trees
Lists
Thread status
Network sockets

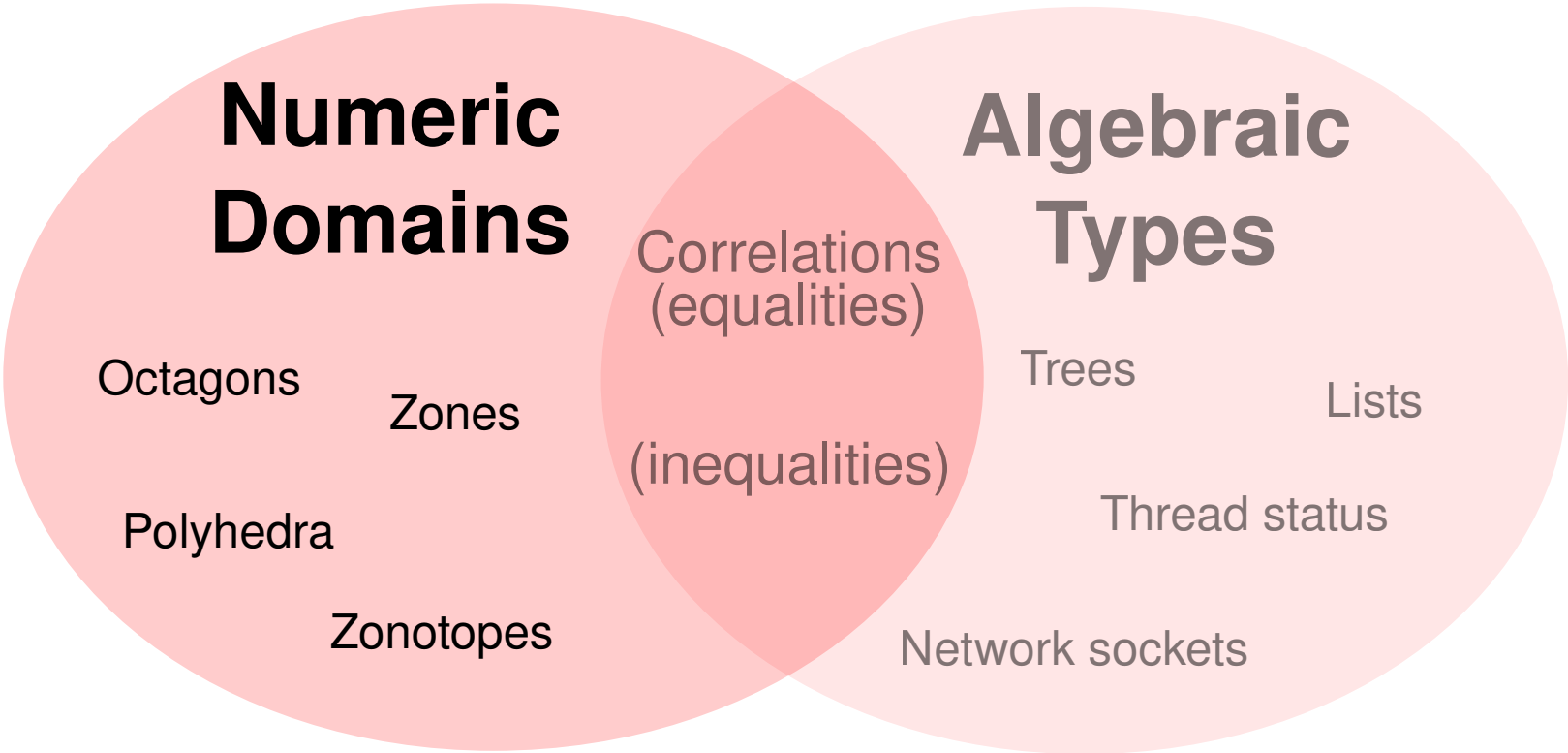
Overview


Numeric Domains
Octagons
Zones
Polyhedra
Zonotopes
Correlations (equalities)
(inequalities)
Algebraic Types
Trees
Lists
Thread status
Network sockets

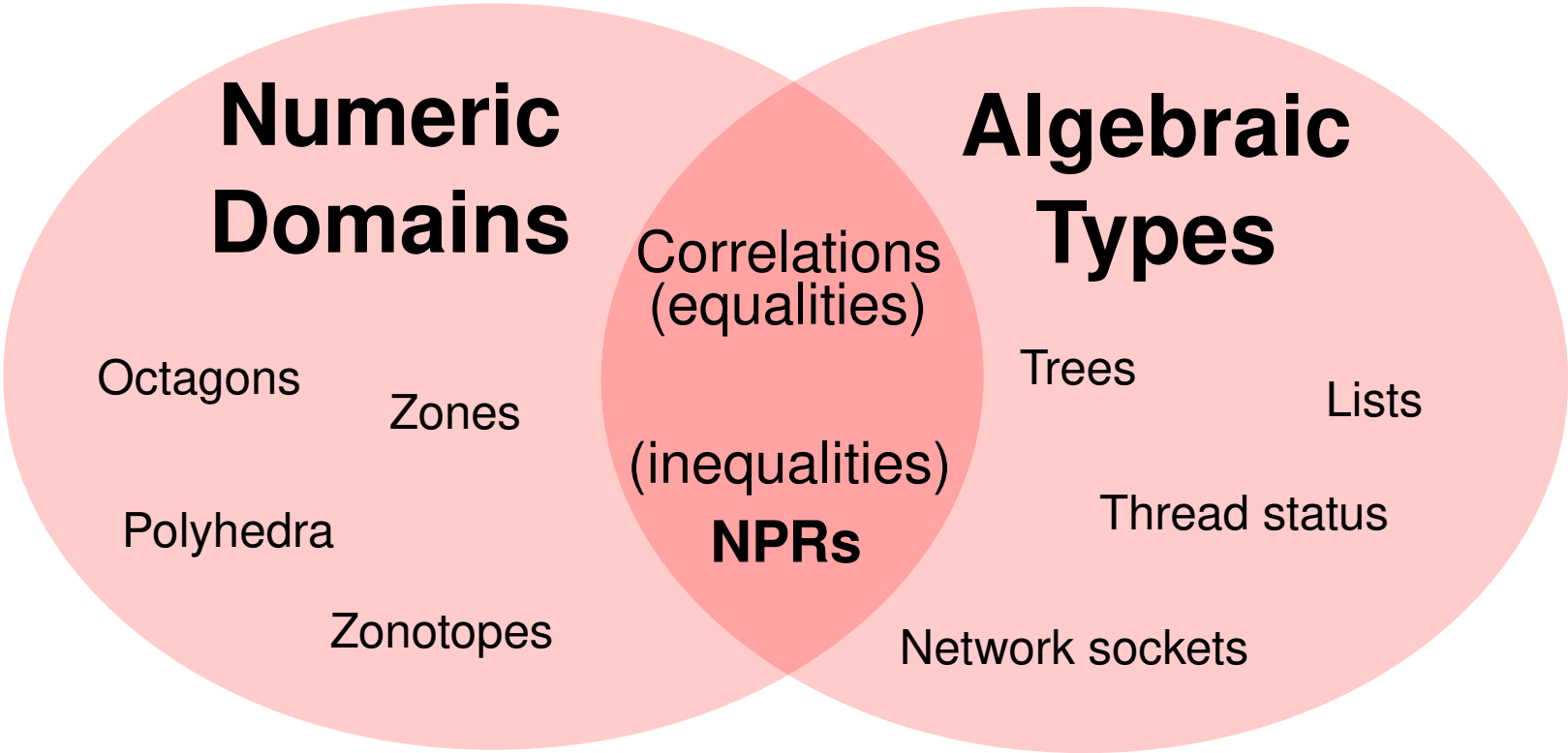
Overview


Numeric Domains
Octagons
Zones
Polyhedra
Zonotopes
Correlations (equalities)
(inequalities)
Algebraic Types
Trees
Lists
Thread status
Network sockets

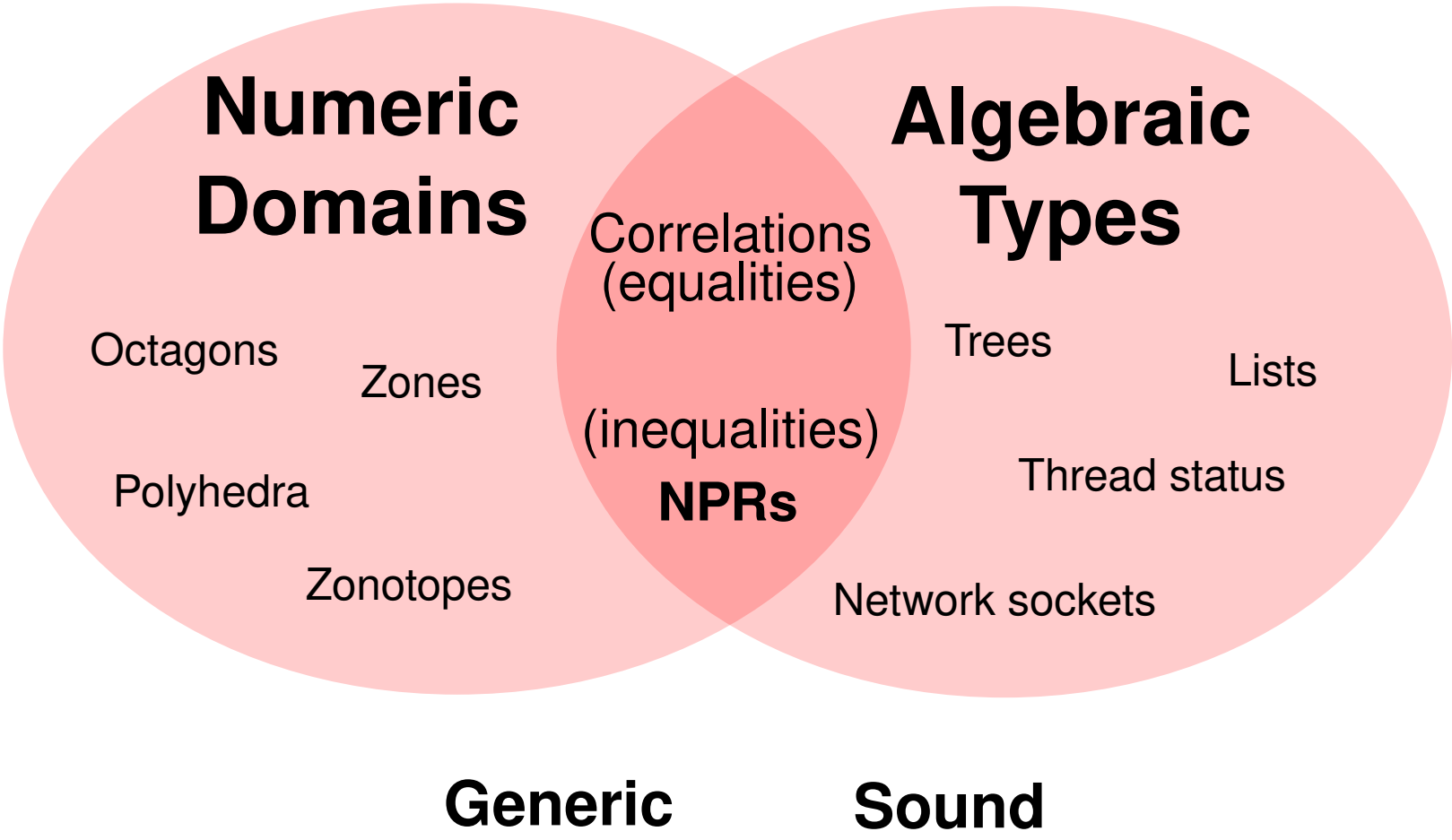
Overview


Numeric Domains
Octagons
Zones
Polyhedra
Zonotopes
Correlations (equalities)
(inequalities)
Algebraic Types
Trees
Lists
Thread status
Network sockets

Overview


Numeric Domains
Octagons
Zones
Polyhedra
Zonotopes
Correlations (equalities)
(inequalities)
NPRs
Algebraic Types
Trees
Lists
Thread status
Network sockets

Overview


Numeric Domains
Octagons
Zones
Polyhedra
Zonotopes
Correlations (equalities)
(inequalities)
NPRs
Algebraic Types
Trees
Lists
Thread status
Network sockets
Generic
Sound

Product types

A product type

type point =
  {absciss: number,
   ordinate: number}

A record

value origin =
  { abciss = 0,
   ordinate = 0}

Product types

$$\left\{\overline{f_i \to \tau_i}^{\,i \in I}\right\}$$

Sum types

A sum type

```
type card =
    | Spades(Number)
    | Hearts(Number)
    | Clubs(Number)
    | Diamonds(Number)
```

A variant

```
value hearts_queen =
    Hearts(12)
```

Sum types

$$\left[\overline{A_i \to \tau_i}^{i \in I}\right]$$

Algebraic Types

Example

type thread_status =
  | Running {thread_id: Number, priority: Number}
  | Asleep {seconds : Number, thread_id: Number, priority: Number}

Definition

Arbitrary nesting of sum types and product types

$$\tau ::= \text{Number} \quad | \quad \left\{ \overline{f_i \to \tau_i}^{i \in I} \right\} \quad | \quad \left[\overline{A_i \to \tau_i}^{i \in I} \right]$$

Algebraic Types

Example

type thread_status =
  | Running {thread_id: Number, priority: Number}
  | Asleep {seconds : Number, thread_id: Number, priority: Number}

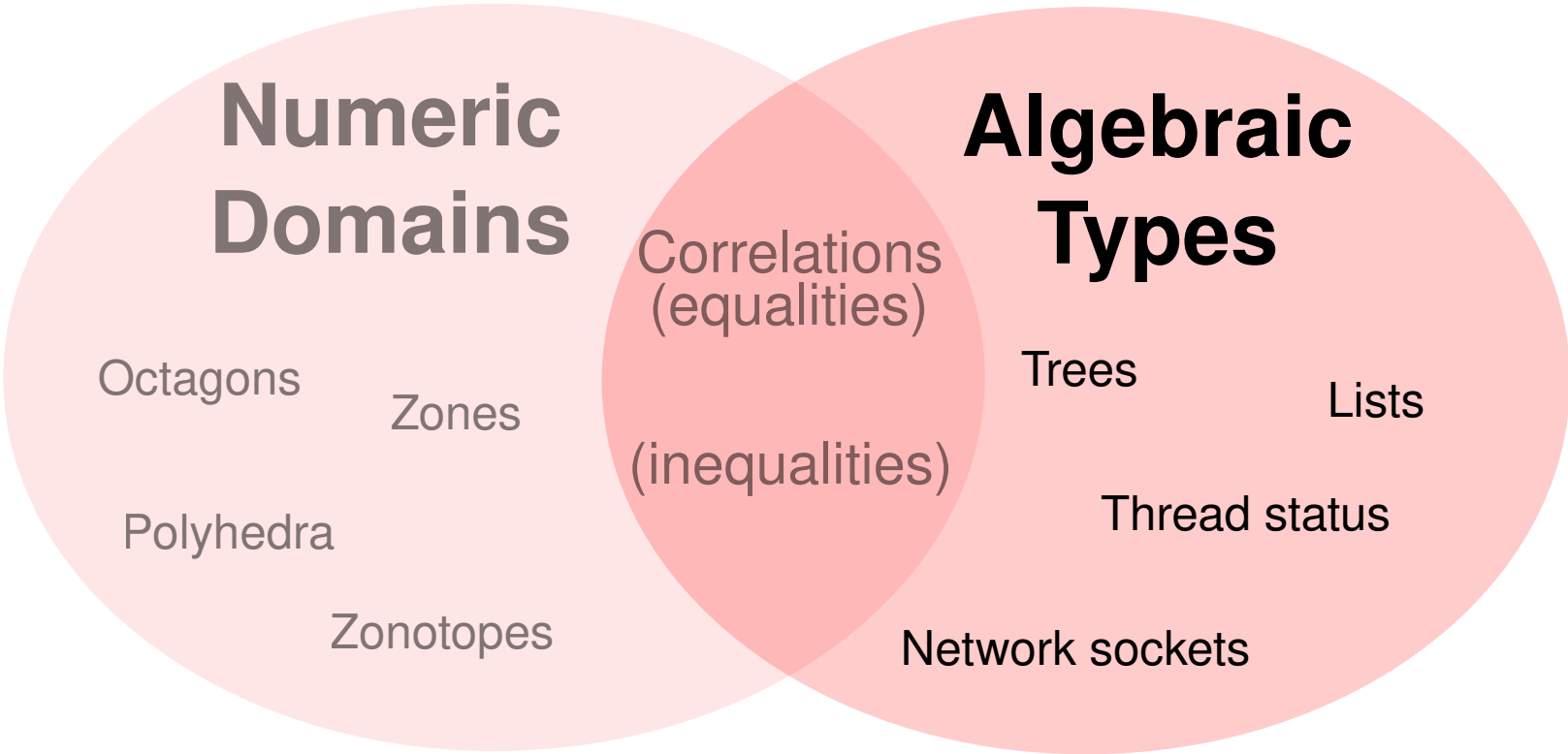
Definition

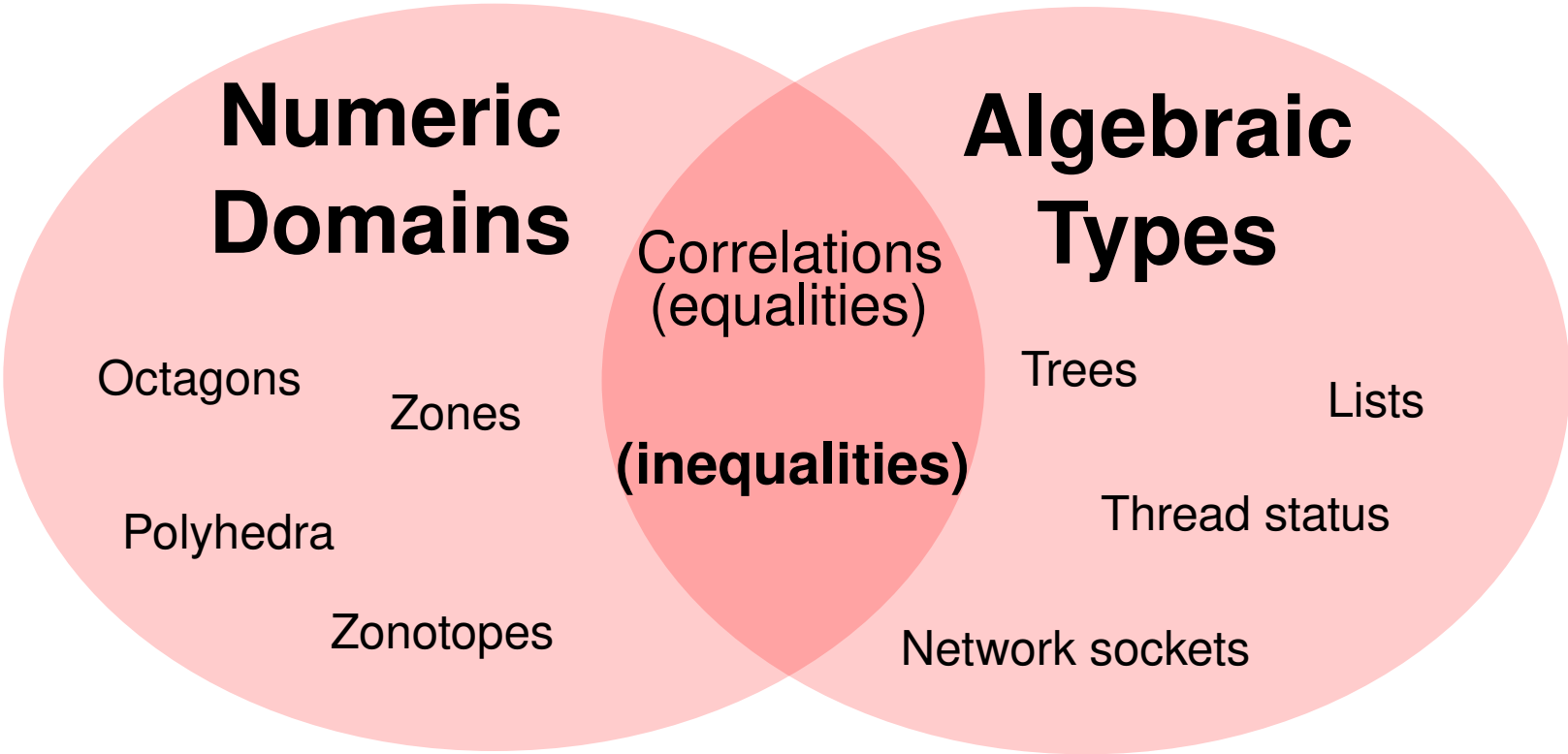
Arbitrary nesting of sum types and product types

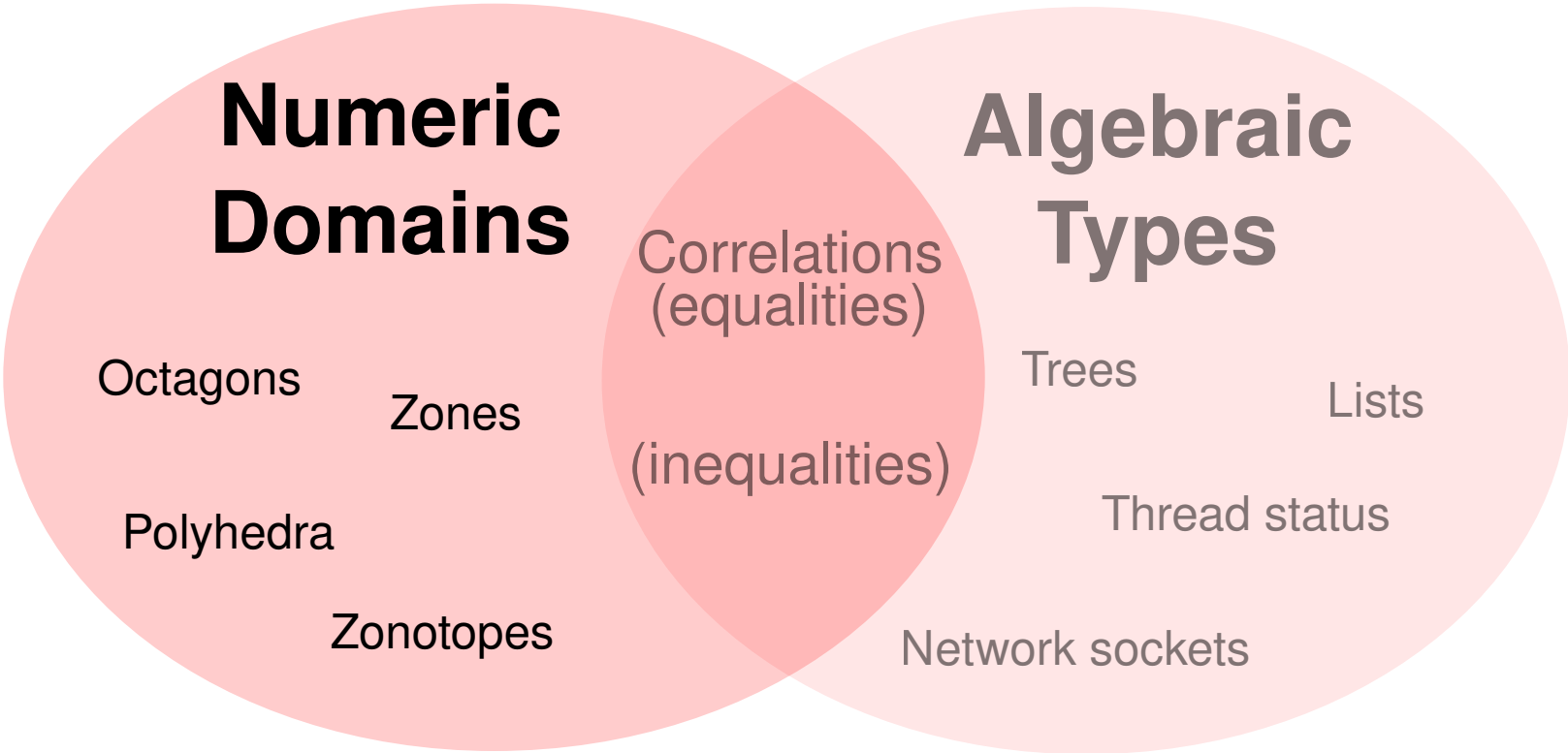
$$\tau ::= \text{Number} \quad | \quad \left\{ \overline{f_i \to \tau_i}^{i \in I} \right\} \quad | \quad \left[\overline{A_i \to \tau_i}^{i \in I} \right]$$

Limitation

We do not handle recursive types.


Numeric Domains
Octagons
Zones
Polyhedra
Zonotopes
Correlations (equalities)
(inequalities)
Algebraic Types
Trees
Lists
Thread status
Network sockets


Numeric Domains
Octagons
Zones
Polyhedra
Zonotopes
Correlations (equalities)
(inequalities)
Algebraic Types
Trees
Lists
Thread status
Network sockets


Numeric Domains
Octagons
Zones
Polyhedra
Zonotopes
Correlations (equalities)
(inequalities)
Algebraic Types
Trees
Lists
Thread status
Network sockets

Numeric domains

Abstract domains

- Polyhedra
  - Octagons
- Zonotopes

Numeric domains

Abstract domains

Polyhedra

Octagons

Zonotopes

Abstract values

Constraints (Variables)

Numeric domains

Abstract domains

Polyhedra

Octagons

Zonotopes

Abstract values

Constraints (Variables)

Program

```
r := 1 ;
while (n > 1) {
   r := n * r ;
   n := n – 1 ;
}
```

Numeric domains

Abstract domains

Polyhedra

Octagons

Zonotopes

Abstract values

Constraints (Variables)

Program

```
r := 1 ;
while (n > 1) {
   r := n * r ;
   n := n – 1 ;
}
```

Variables

n r

Numeric domains

Abstract domains

Polyhedra

Octagons

Zonotopes

Abstract values

Constraints (Variables)

Program

```
r := 1 ;
while (n > 1) {
   r := n * r ;
   n := n – 1 ;
}
```

◆ Program point

Variables

n r

Numeric domains

Abstract domains

Polyhedra

Octagons

Zonotopes

Abstract values

Constraints (Variables)

Program

```
r := 1 ;
while (n > 1) {
   r := n * r ;
   n := n – 1 ;
}
```

Program point

Variables

$n \quad r$

Constraint

$r \geq n + 1$

Numeric domains

Abstract domains

Polyhedra

Octagons

Zonotopes

Abstract values

Constraints (Variables)

$\{r \geq n + 1\}$

Program

```
r := 1 ;
while (n > 1) {
   r := n * r ;
   n := n – 1 ;
}
```

Program point

Variables

$n \quad r$

Constraint

$r \geq n + 1$

The Apron interface

Numeric domain D

Abstract values $\quad$ Constraints(Variables)

The Apron interface

Numeric domain D

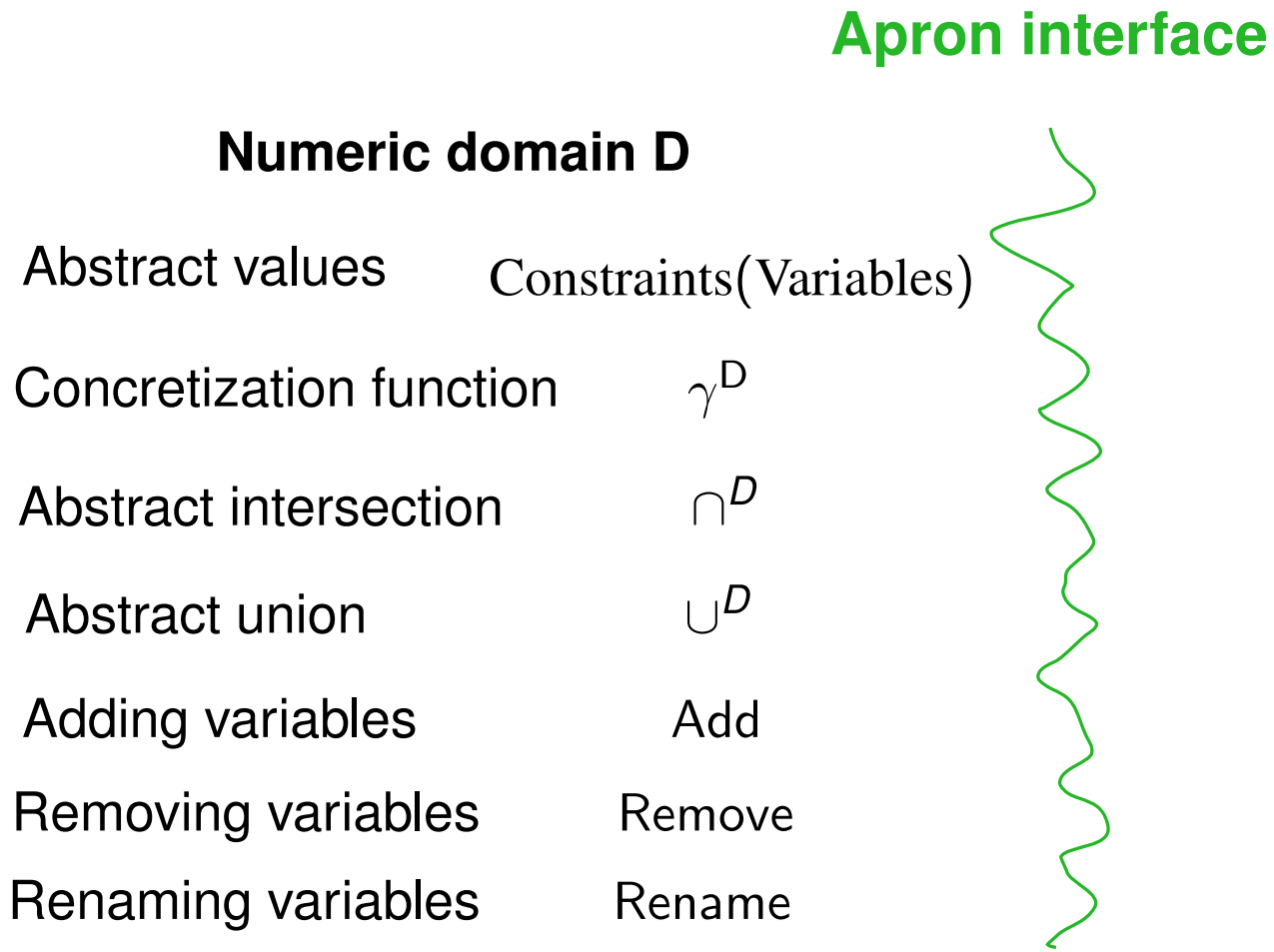
Abstract values	$\mathrm{Constraints}(\mathrm{Variables})$
Concretization function	γ^{D}
Abstract intersection	$\cap^{D}$
Abstract union	$\cup^{D}$

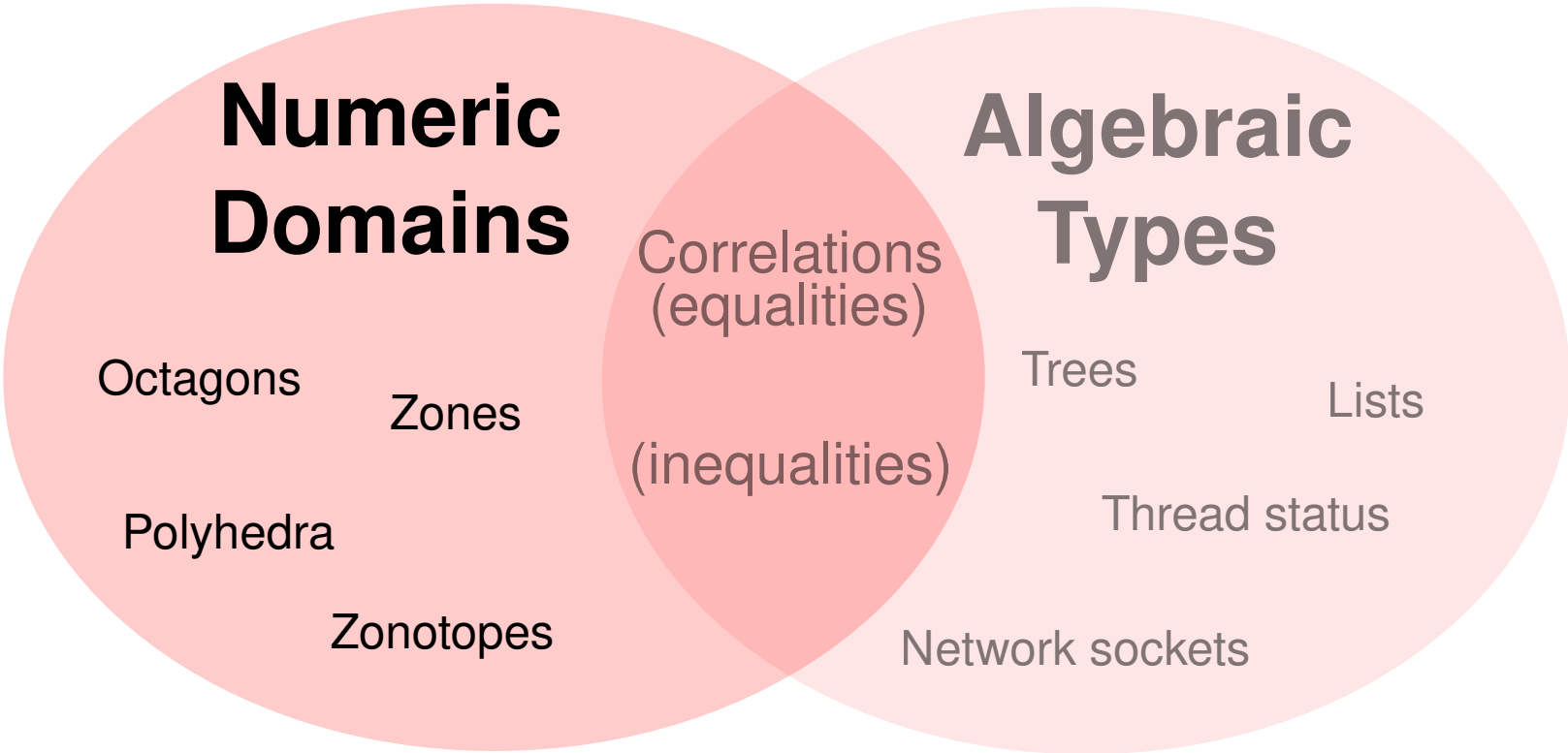
The Apron interface

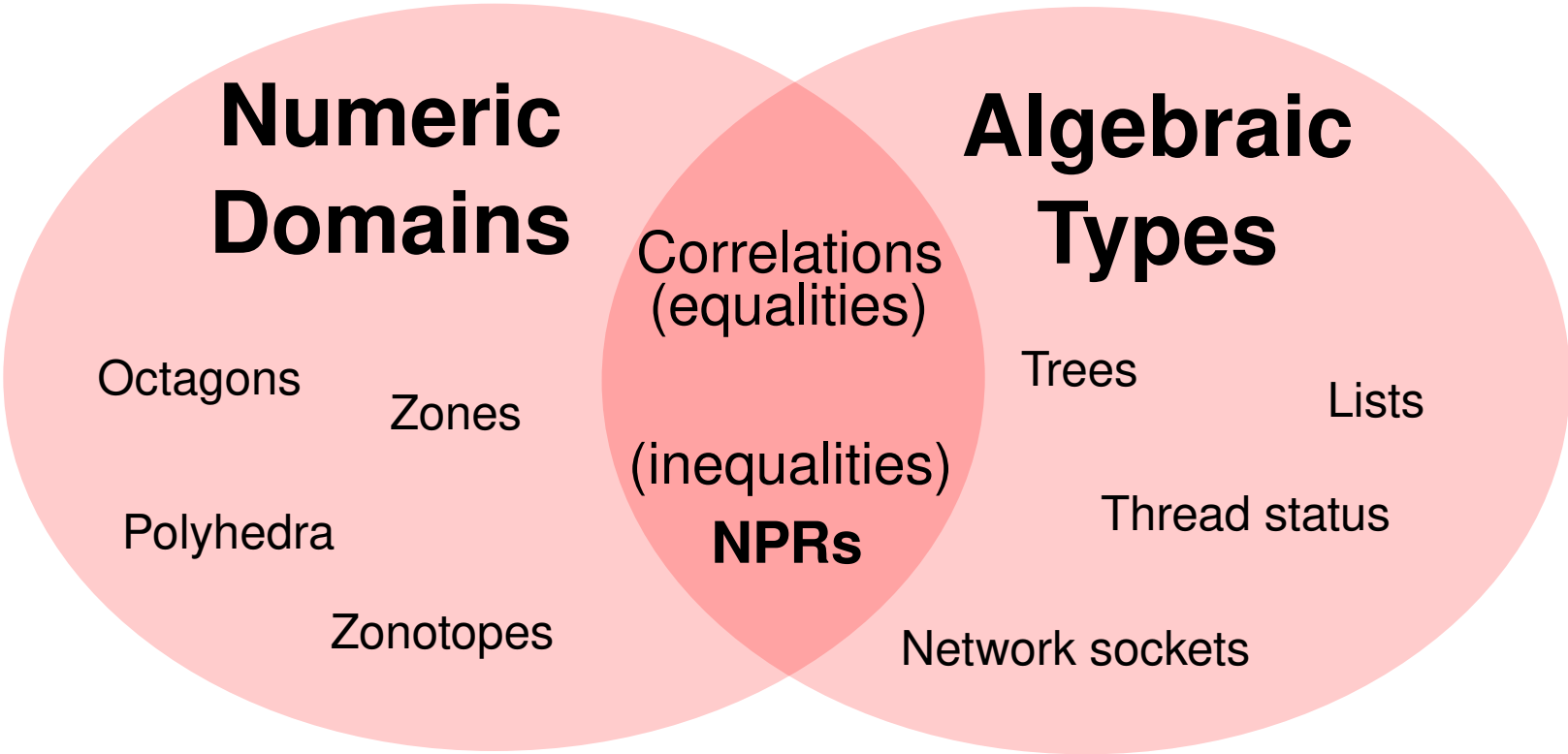
Numeric domain D

Abstract values	$\mathrm{Constraints}(\mathrm{Variables})$
Concretization function	γ^{D}
Abstract intersection	$\cap^{D}$
Abstract union	$\cup^{D}$
Adding variables	Add
Removing variables	Remove
Renaming variables	Rename

The Apron interface




Numeric Domains
Octagons
Zones
Polyhedra
Zonotopes
Correlations (equalities)
(inequalities)
Algebraic Types
Trees
Lists
Thread status
Network sockets

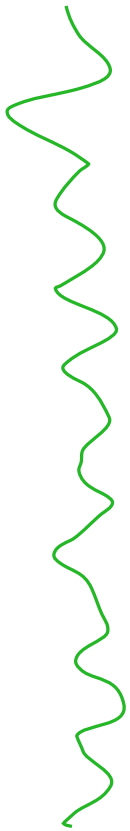

Numeric Domains
Octagons
Zones
Polyhedra
Zonotopes
Correlations (equalities)
(inequalities)
NPRs
Algebraic Types
Trees
Lists
Thread status
Network sockets

Numeric path relations

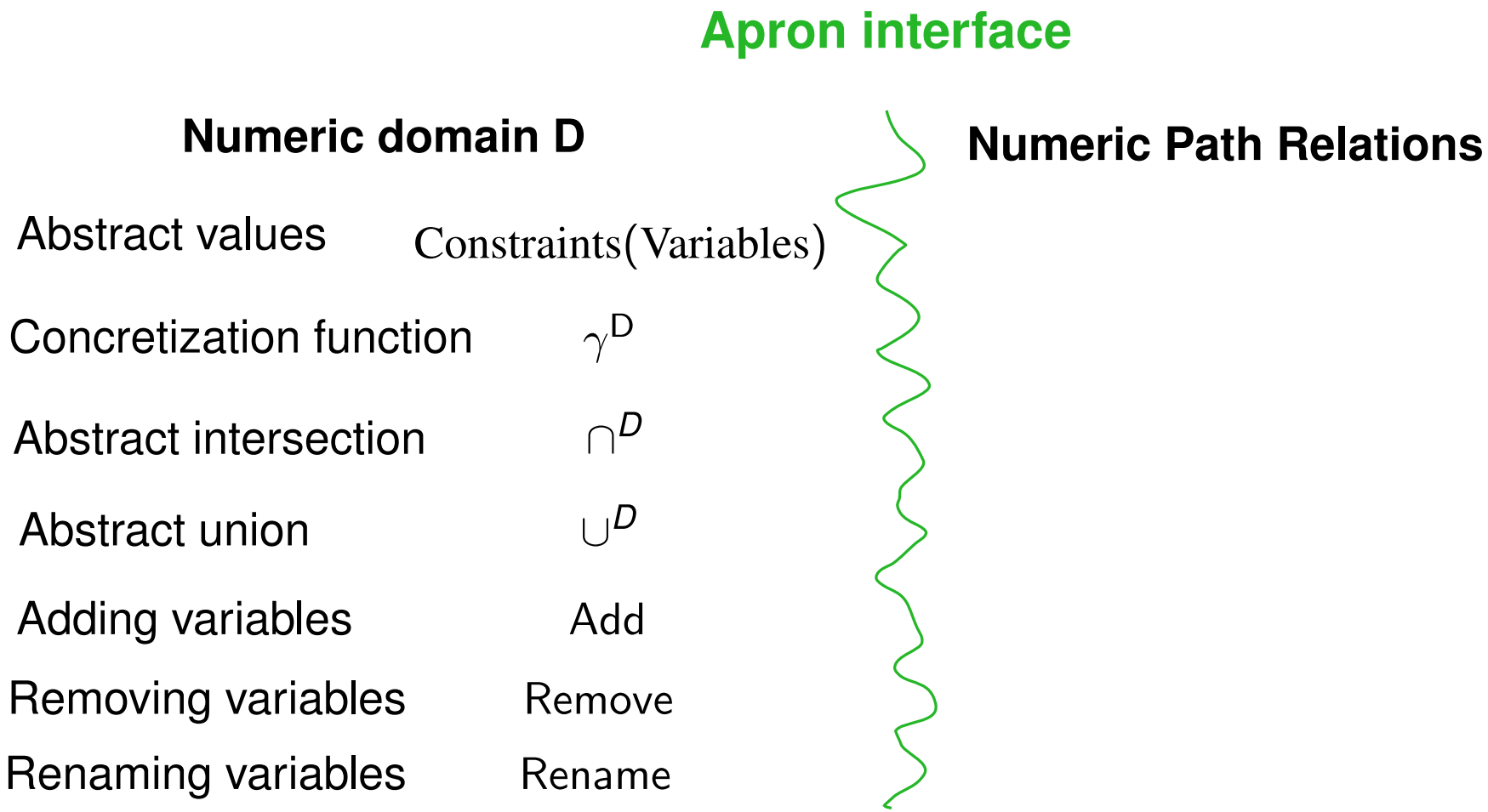
Apron interface

Numeric domain D

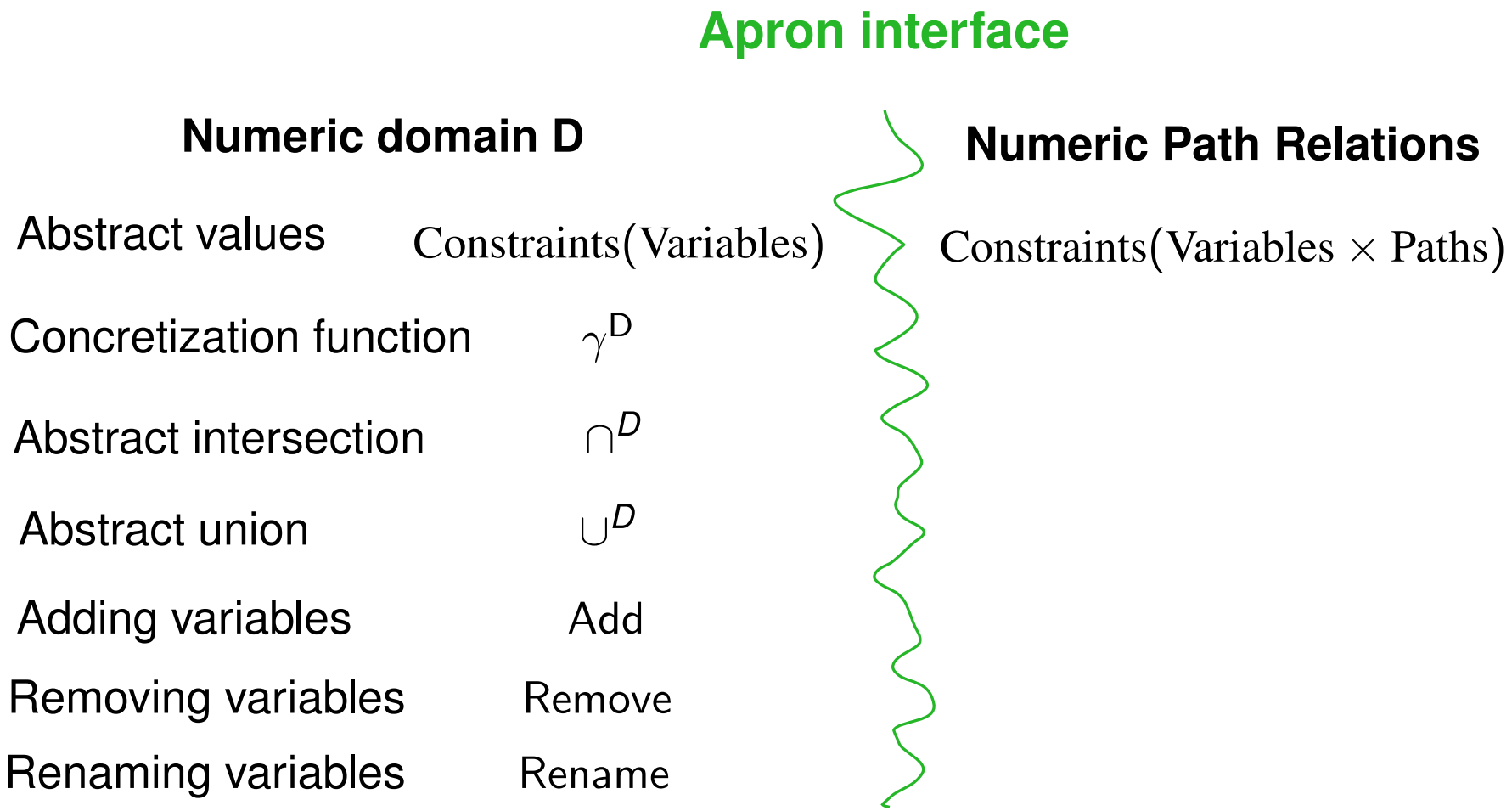
Abstract values	$\text{Constraints}(\text{Variables})$
Concretization function	γ^{D}
Abstract intersection	$\cap^{D}$
Abstract union	$\cup^{D}$
Adding variables	Add
Removing variables	Remove
Renaming variables	Rename



Numeric path relations



Numeric path relations

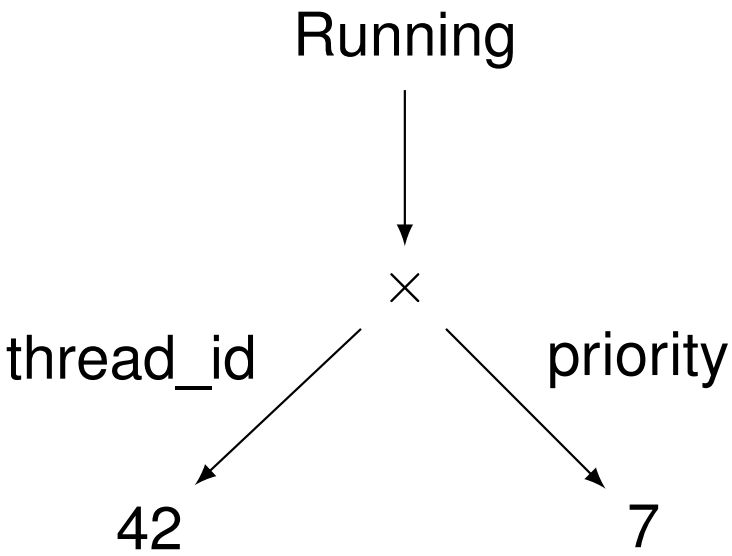


Numeric path relations

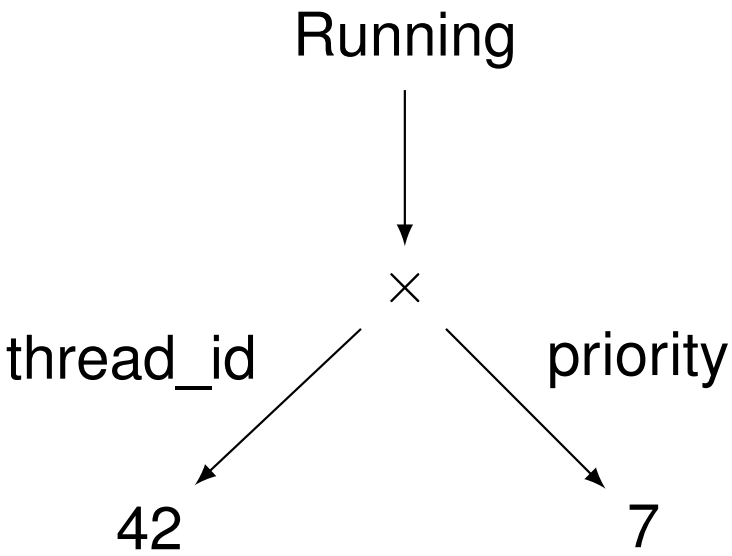
Apron interface

	Numeric domain D	**Numeric Path Relations**
Abstract values	$\text{Constraints}(\text{Variables})$	$\text{Constraints}(\text{Variables} \times \text{Paths})$
Concretization function	γ^{D}	γ^{NPR}
Abstract intersection	$\cap^{D}$	$\cap^{\text{NPR}}$
Abstract union	$\cup^{D}$	$\cup^{\text{NPR}}$
Adding variables	Add	
Removing variables	Remove	
Renaming variables	Rename	

Projection over a path



Projection over a path



$$\mathrm{th} \Downarrow^{\mathsf{val}} @\mathrm{Running.thread\_id} = 7$$

Projection over extended variables

Extended variables

Pair of a variable name and a path

Examples

$(x, \text{@Running.thread\_id})$ and $(x, \text{@Running.priority})$

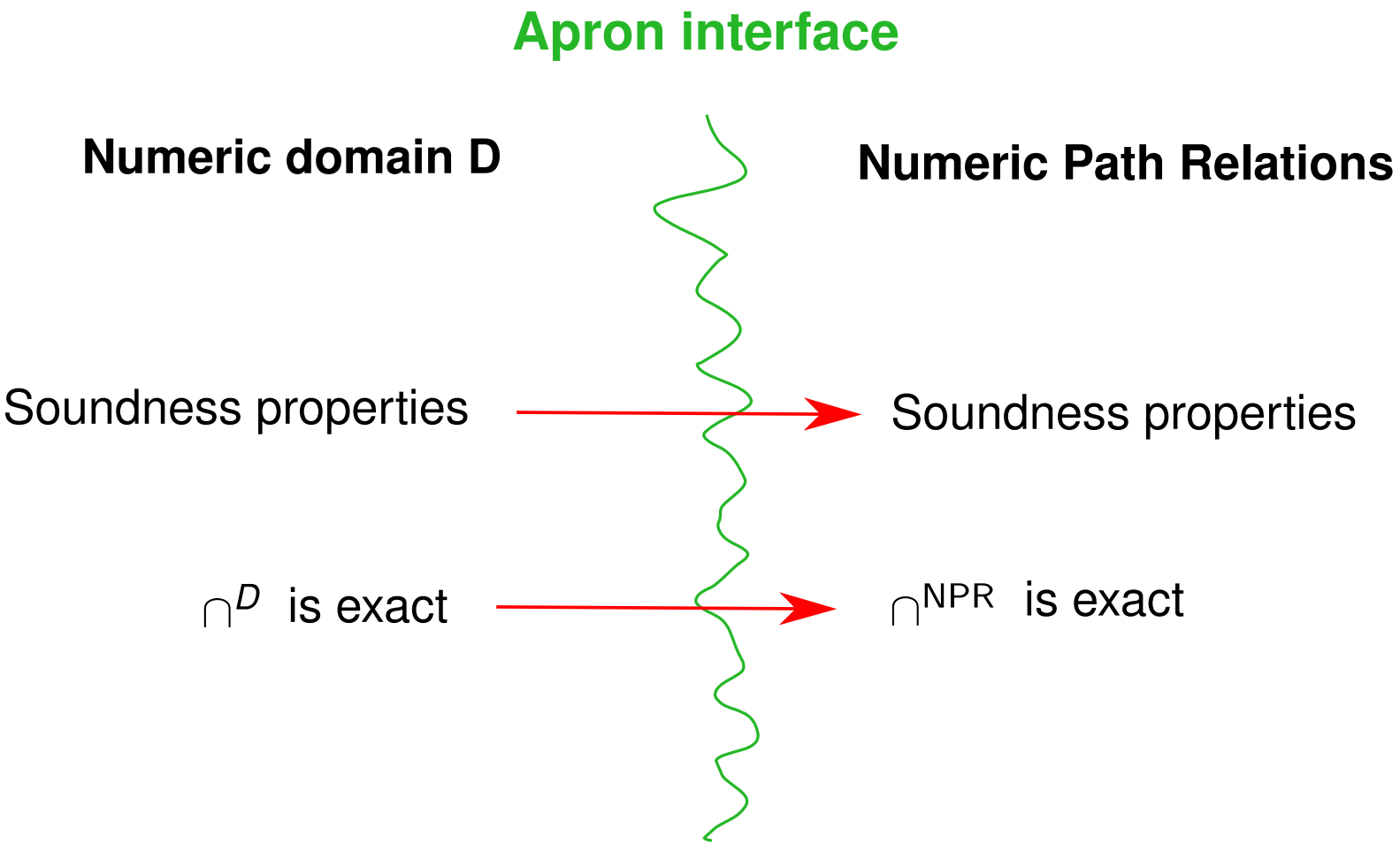
$$[x \mapsto \text{th}] \Downarrow^{\text{env}} \{(x, \text{@Running.id}), (x, \text{@Running.priority})\} = \begin{bmatrix} (x, \text{@Running.thread\_id}) & \mapsto & 42 \\ (x, \text{@Running.priority}) & \mapsto & 7 \end{bmatrix}$$

Numeric Path Relations (NPRs)

```
function tick (x) =
  match x with
    | Asleep {seconds = s, thread_id = id, priority = p} →
      if (seconds == 1)
      then r := Running {thread_id = id, priority = p}
      else r := Asleep {seconds = s – 1, thread_id = id, priority = p}
   | _ →r := x ;
 return r
```

$$\left\{ \begin{array}{l} \left\{ \begin{array}{lcl} (r, \text{@Asleep.seconds}) & = & (x, \text{@Asleep.seconds}) - 1, \\ (r, \text{@Asleep.thread\_id}) & = & (x, \text{@Asleep.thread\_id}), \\ (r, \text{@Asleep.priority}) & = & (x, \text{@Asleep.priority}) \end{array} \right\}, \\ \left\{ \begin{array}{lcl} (r, \text{@Running.thread\_id}) & = & (x, \text{@Asleep.thread\_id}), \\ (r, \text{@Running.priority}) & = & (x, \text{@Asleep.priority}) \end{array} \right\} \end{array} \right\}$$

Transferred properties



Take away and future work

- Numeric Path Relations :
  - Extend any numeric domain (given the Apron Interface)
  - Preserve soundness properties
  - Preserve exactness of abstract intersection
- Future work
  - Recursive types
  - Implementation

Image credits

Images from https://www.flaticon.com

- TO BE COMPLETED

[0] Go to https://www.flaticon.com/authors/<author_name> for more of their icons