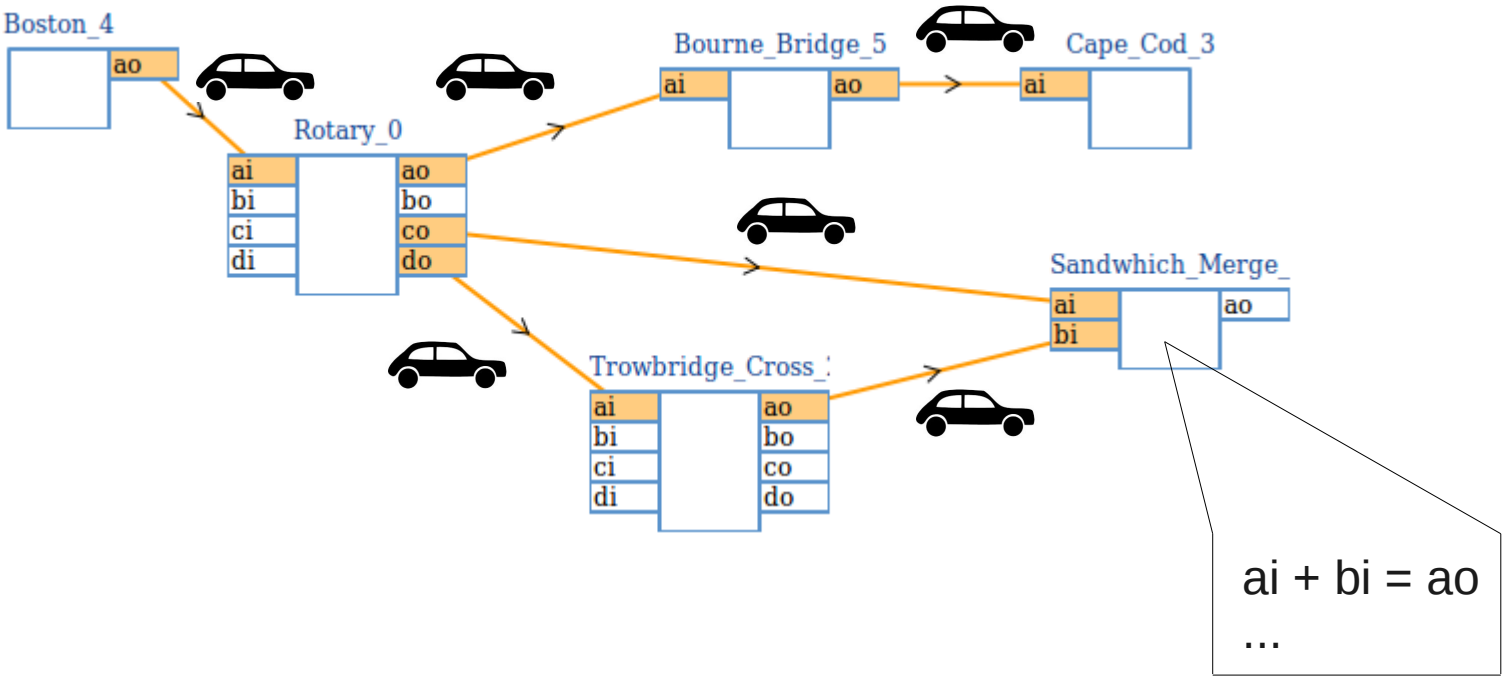
Safe Compositional Equation-based Modeling of Constrained Flow Networks

Nate Soule - Azer Bestavros - Assaf Kfoury - Andrei Lapets

Boston University

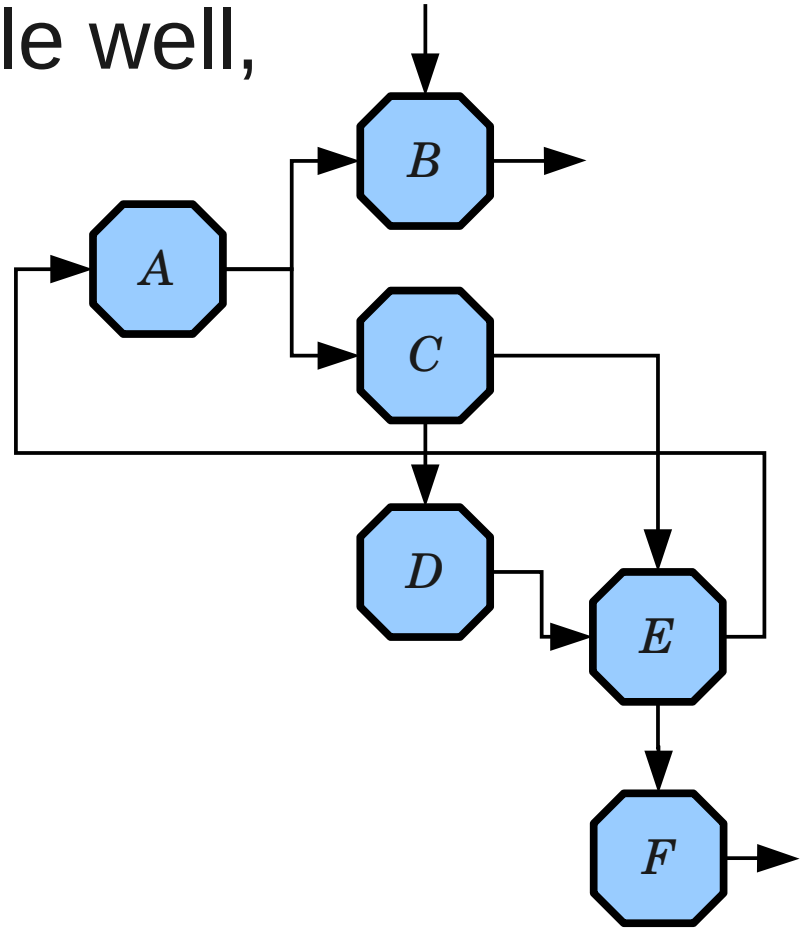
Introduction: Constrained Flow Networks

- Flows between nodes regulated by constraints
- Example domain: Vehicular Road Traffic



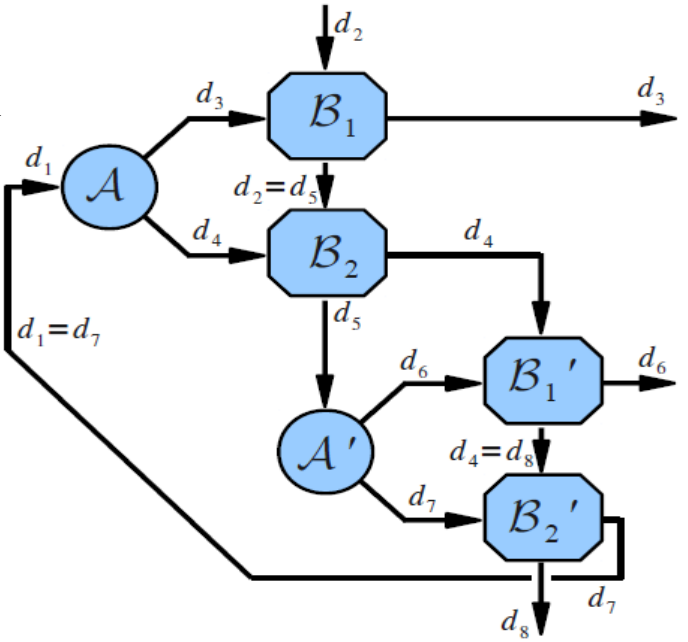
Introduction: NetSketch

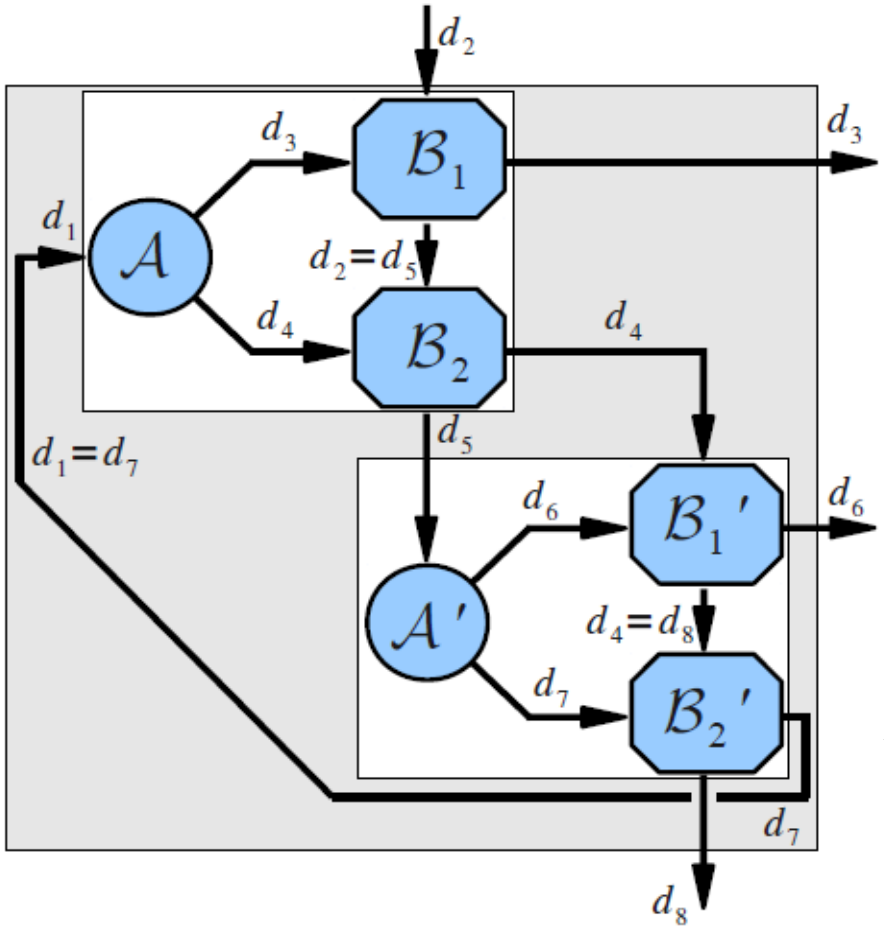
- **Problem:** Traditional analysis of constrained flow networks doesn't scale well, cope with unknowns
- **Solution:** NetSketch
  - Lightweight, efficient, scalable modeling and analysis
  - Formalism/DSL
  - Web-based Tool



Introduction: NetSketch

Whole System Analysis →





← Compositional Analysis

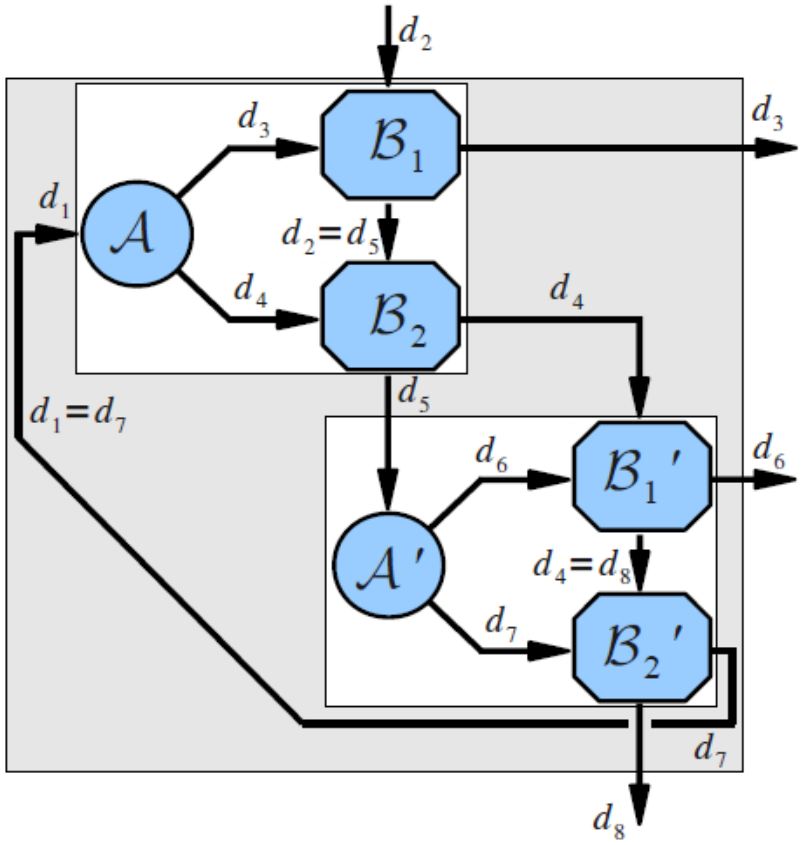
Compositional Analysis via Type Approximations

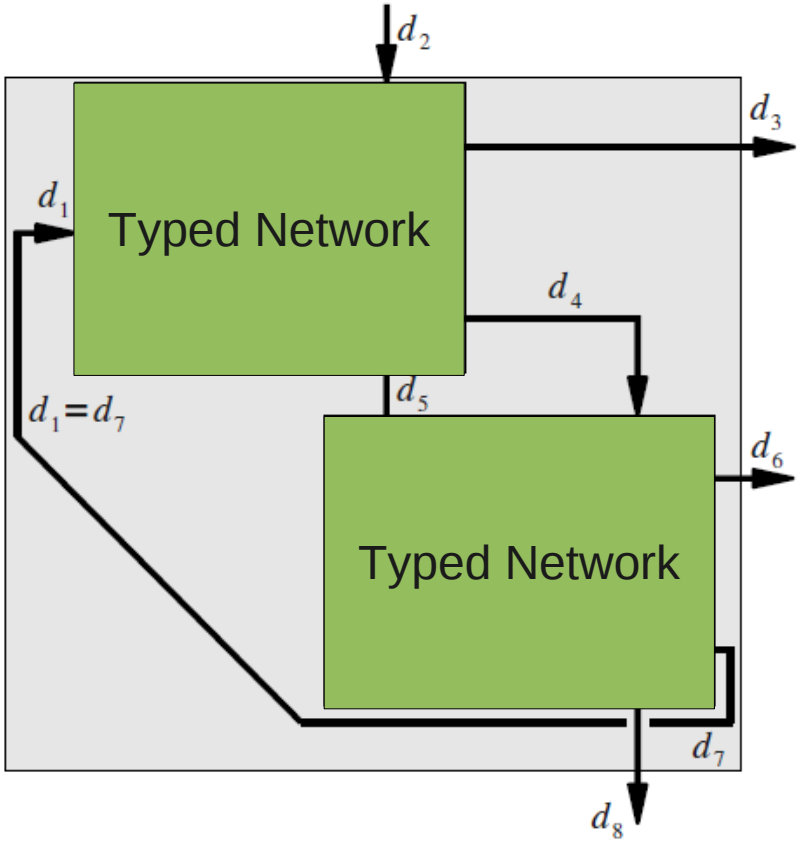
Traditional Types

```
int add(int a, int b) {
        return a + b;
}
```

Type system allows the compiler to verify this is safe without knowledge of the exact values or representation of vars

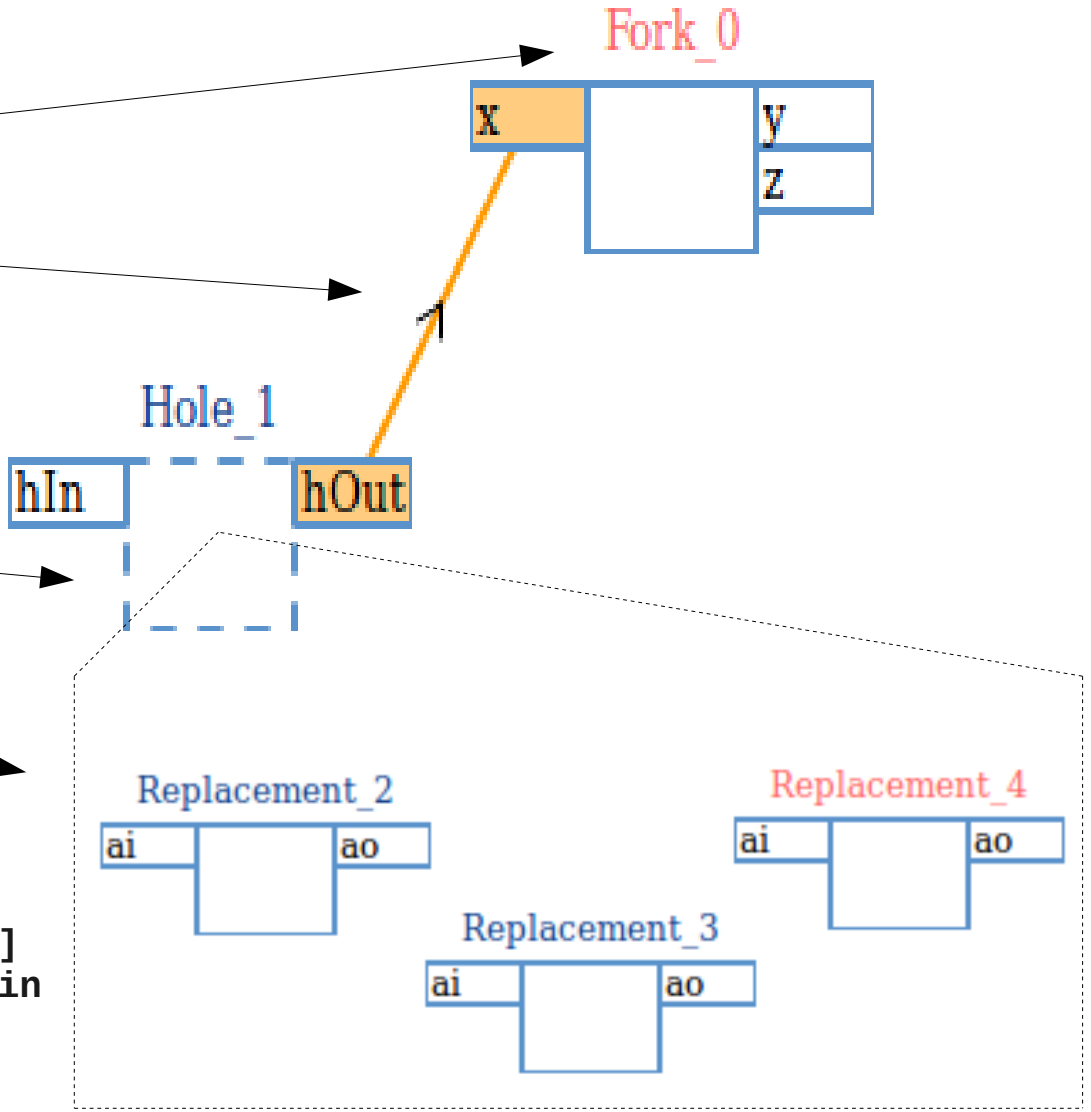
NetSketch Types





NetSketch DSL

- Module
- Connect
- Loop
- Hole
- Let



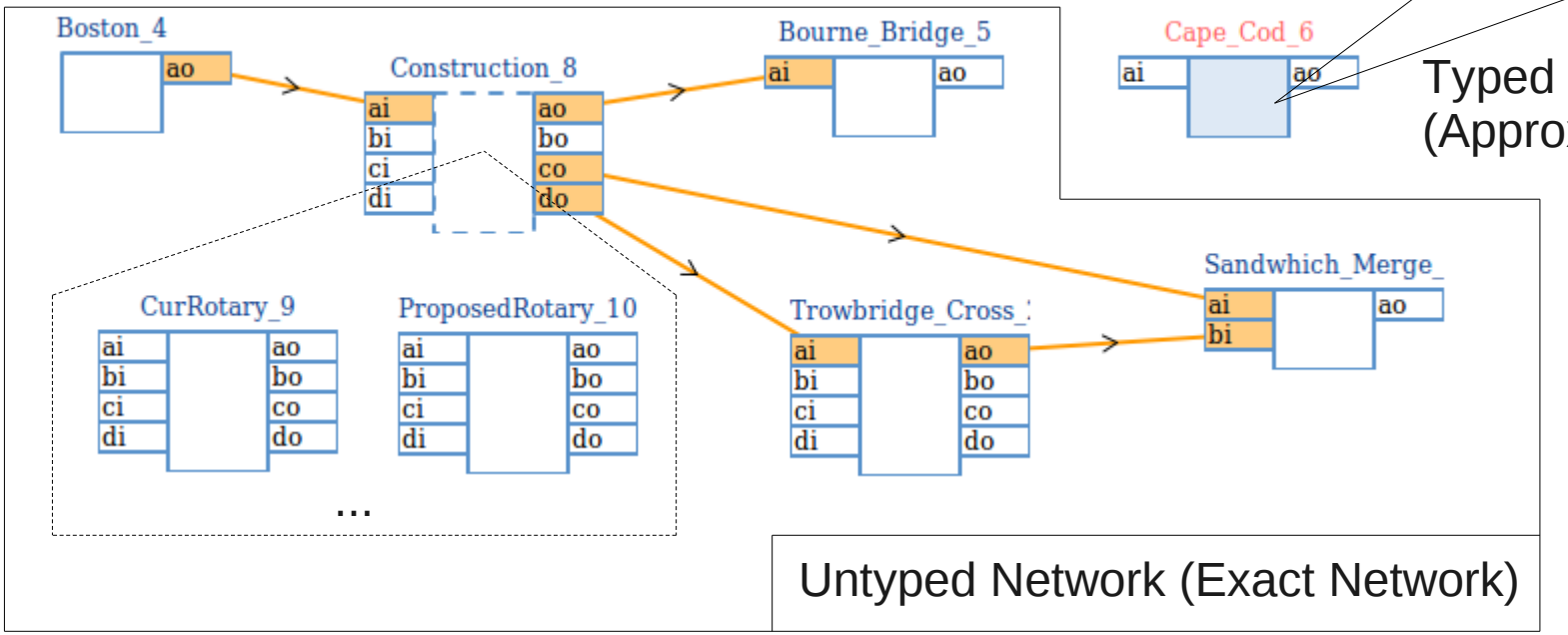

```
Connect [(hOut, x)]
    Module Fork_0 [x] [y,z] [x=y+z]
    Let Hole_1 [Replacement_2...] in
        Hole Hole_1 [hIn] [hOut]
```

Example Model

Element	Represents
Boston_4	Module (untyped network)
Construction_8	Hole (untyped network)
CurRotary_9	Replacement (untyped network)
ProposedRotary_10	Replacement (untyped network)
Cap_Code_11	Typed Network

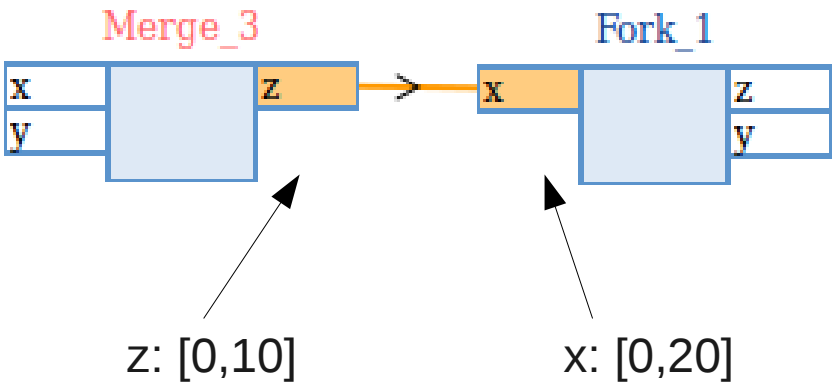
Untyped Network
(Exact Network)

Typed Network
(Approximation)



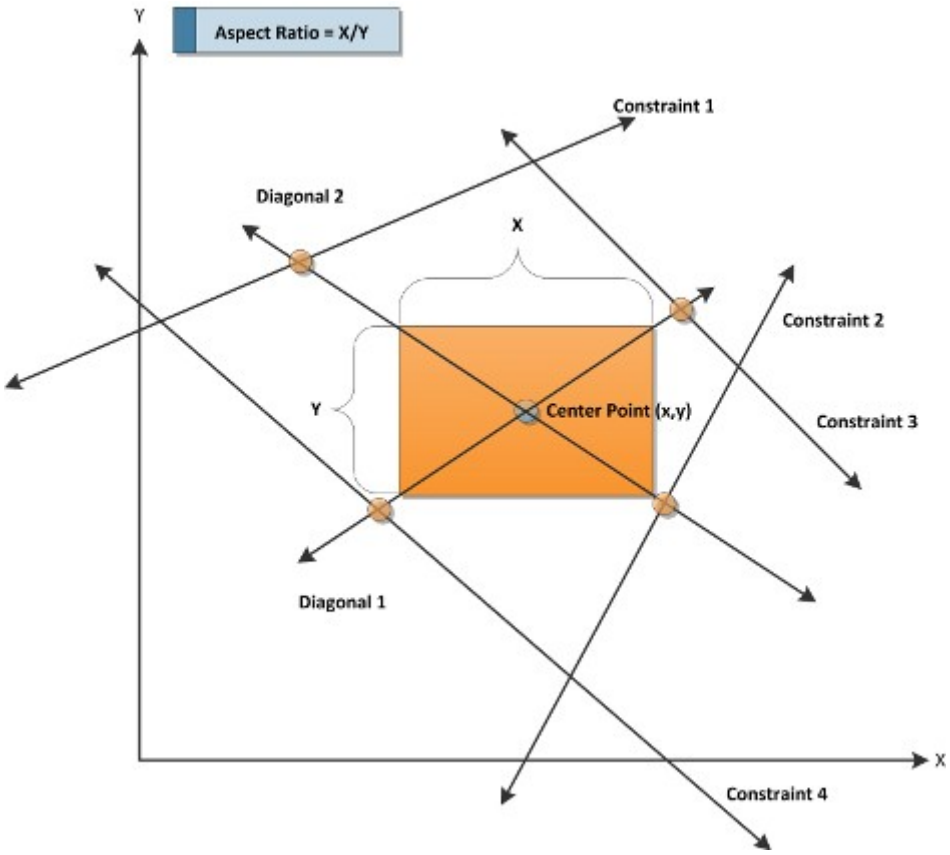
Type System

- Current implementation:
  - Types are open/closed intervals over $\mathbb{R}$
  - Linear constraints are analyzed to determine:
    - Safe intervals for input ports
    - Safe intervals for output ports



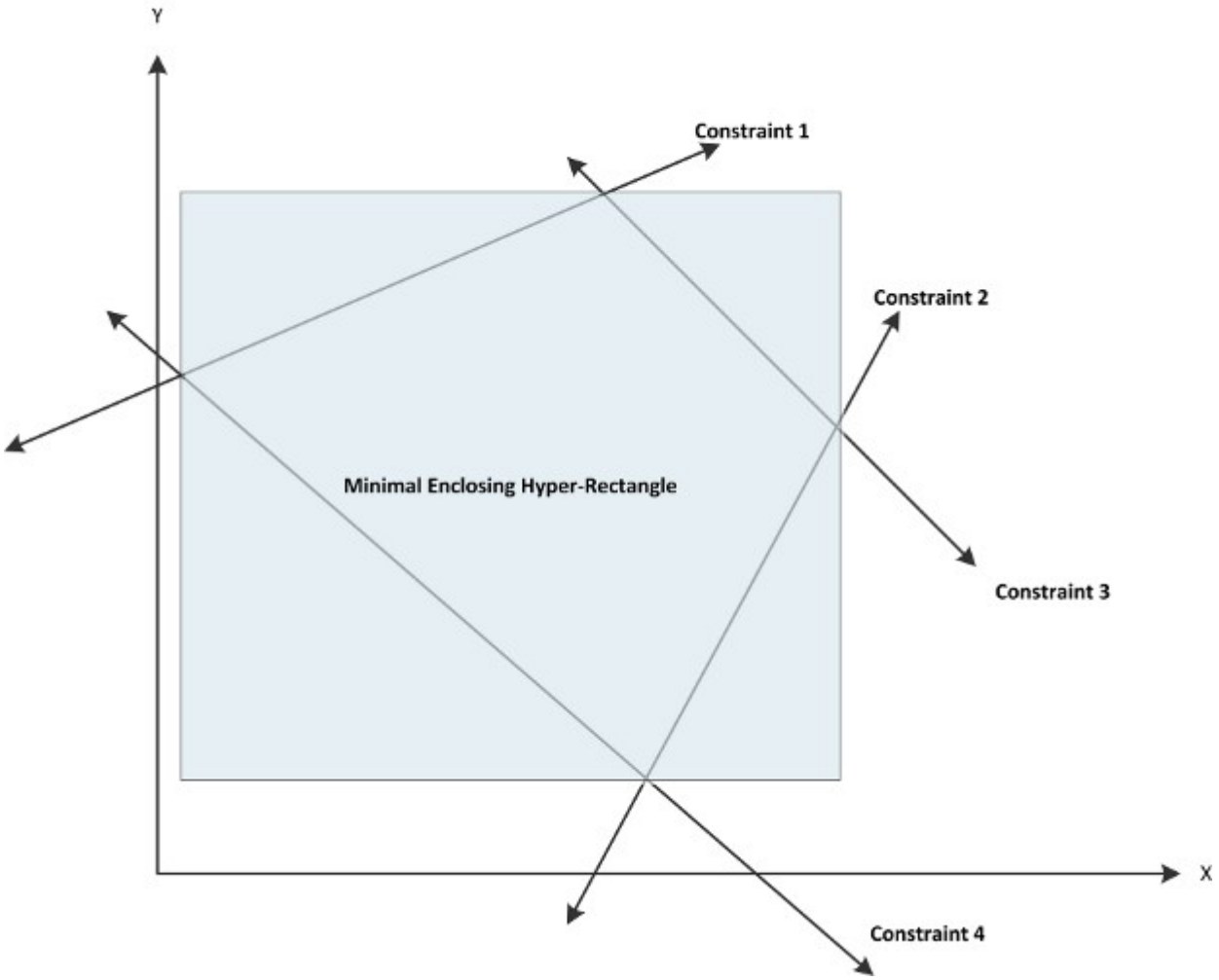
Type Inference: Input Types

- Linear constraints form a convex hull
- Input types approximate the feasible region
- Types derived from maximally enclosed axis-aligned hyperrectangle
- Made unique via:
  - Center point
  - Aspect ratio

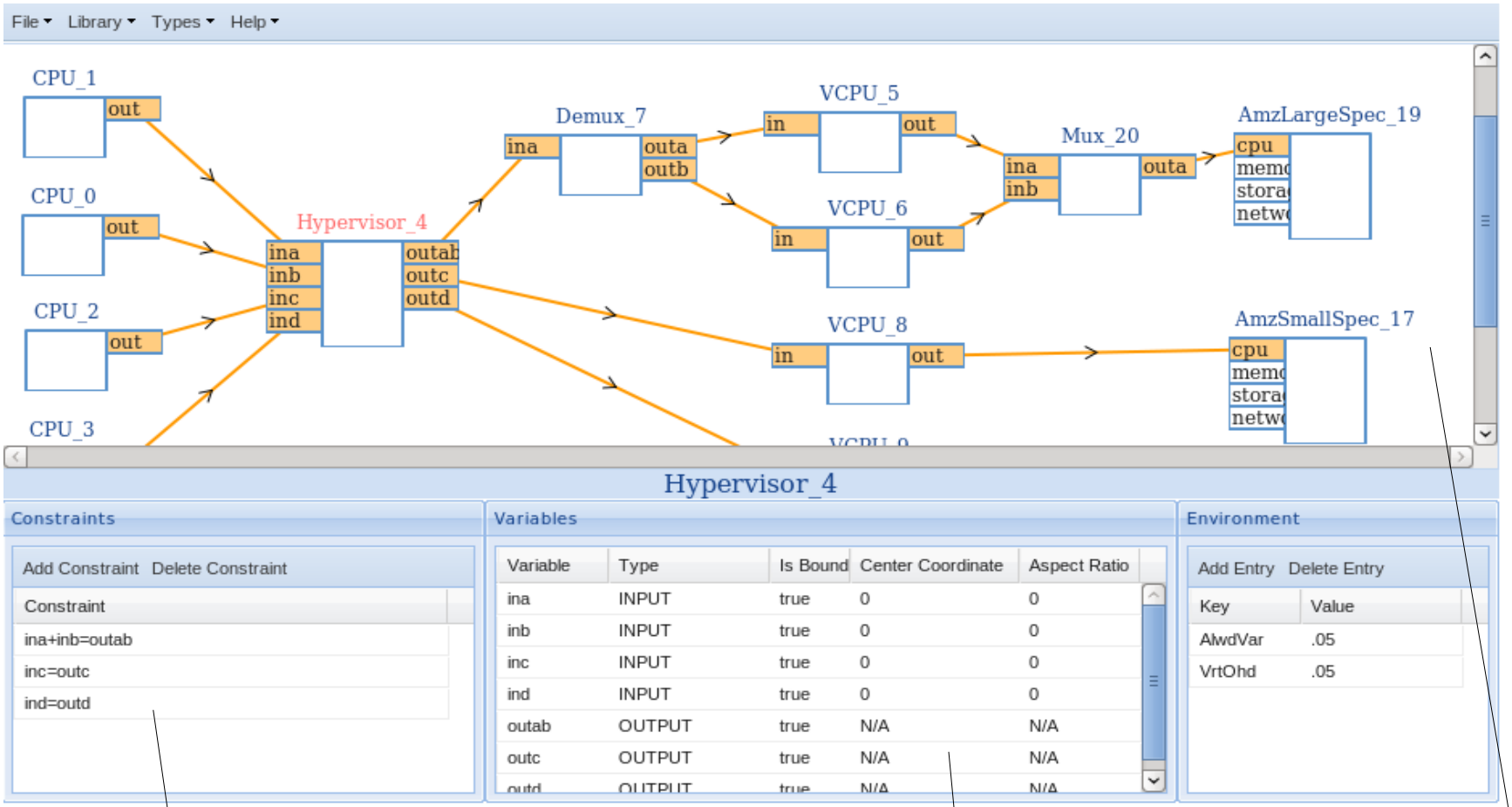


Type Inference: Output Types

- Again constraints form convex hull
- Must use maximally *enclosing* hyperrectangle
- Unique without further user input



NetSketch Tool



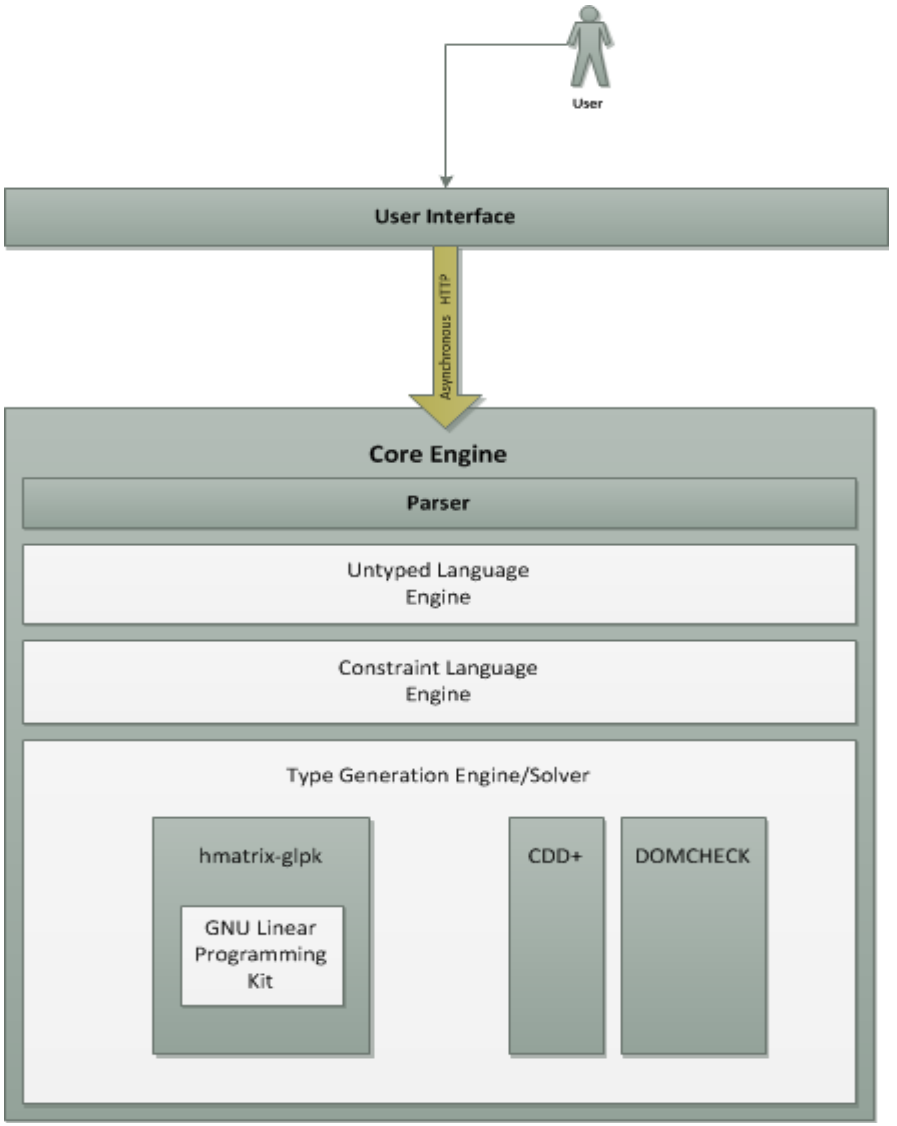
Canvas

Constraints

Variable Metadata

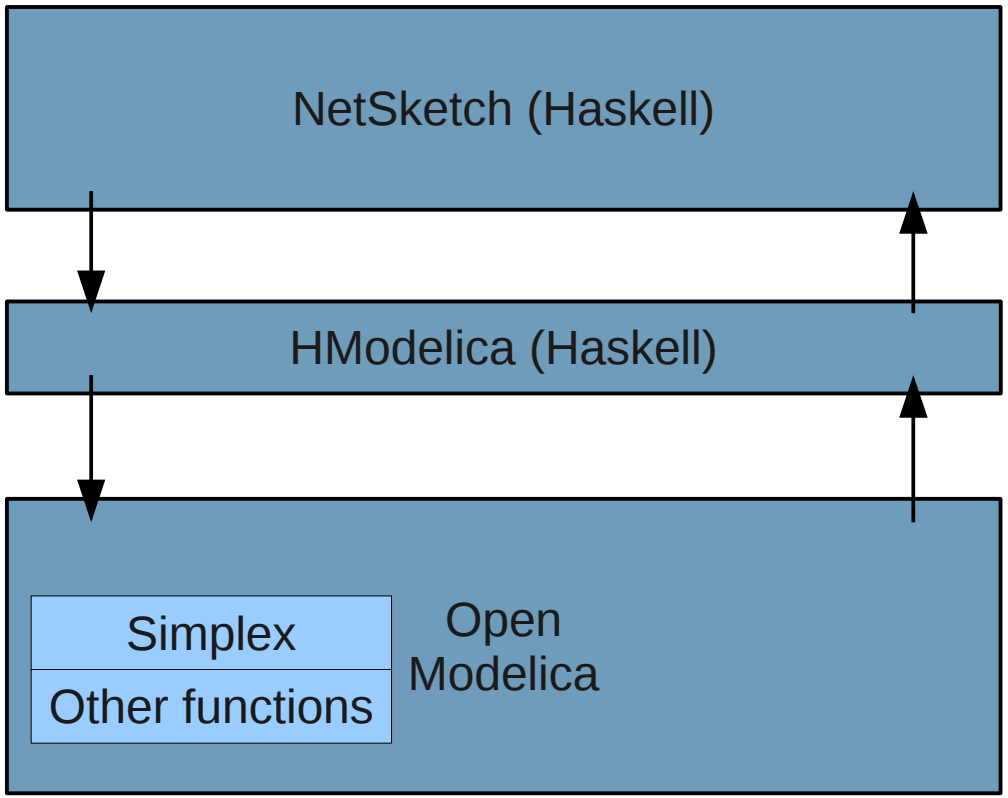
NetSketch Architecture

- Web Based Front End
  - JavaScript
  - HTML
- Server Back End
  - Haskell
  - C
  - C++
- Asynchronous JavaScript and XML (AJAX) based communication



Harnessing Modelica

Computation Platform



Simulation Platform

```
Conn [(c,x)]
    Module Mod1 [a,b] [c] [a+b=c]
    Module Mod2 [x] [y,z] [x=y+z]
```

```
Package TranslatedNetSketch

Connector OutPort = output Real;
Connector InPort = input Real;

Class Mod1
    InPort a;
    InPort b;
    OutPort c;
Equation
    a + b = c;
End Mod1
...
```

Current and Future Work

- Constraint and type extensions
  - Adaptive Dynamic Types
  - Variations of constraints
- Tool extensions
  - Bi-directional flow, extended *Let* functionality, internal variables, export to Modelica
- Modelica integration
  - Extend/refine HModelica
  - Modelica -> NetSketch translation

Thank You

More info at: **http://www.cs.bu.edu/groups/ibench**