QuB

A Resource Aware Functional Programming Language

Apoorv Ingle

The University of Kansas

Table of Contents

Hard problems in programming

Naming variables

Introduction and Motivation

Hard problems in programming

Resource management

in evolving production code

Resources: Files, database connections, entity with a shared state

Resource Management: File Handling

- Modified File Handling API in Haskell

$$
\begin{aligned}
\texttt{openFile} &:: \texttt{FilePath} &&\to \texttt{IO FileHandle}\\
\texttt{closeFile} &:: \texttt{FileHandle} &&\to \texttt{IO}\ ()\\
\texttt{readLine} &:: \texttt{FileHandle} &&\to \texttt{IO}\ (\texttt{String}, \texttt{FileHandle})\\
\texttt{writeFile} &:: \texttt{String} &&\to \texttt{FileHandle}\\
& &&\to \texttt{IO}\ ((), \texttt{FileHandle})\\
\\
\texttt{upper} &:: \texttt{String} &&\to \texttt{String}
\end{aligned}
$$

Resource Management: File Handling

- File Handling in Haskell

```
do f  ← openFile "sample.txt"
   (s, f)  ← readLine f
   let c = upper s
   ((), f) ← writeLine f c
          .
          .
          .
   ()  ← closeFile f
```

- File Handling in Haskell Gone Wrong (Part I)

```
do f      ← openFile "sample.txt"
   (s, f)  ← readLine f
   let c = upper s
   ((), f) ← writeLine f c
          .
          .
          .
   ()     ← closeFile f
          .
          .
          .
   ()     ← closeFile f
   return c
```

Resource Management: File Handling

- File Handling in Haskell Gone Wrong (Part I)

```
do f ← openFile "sample.txt"
   (s, f) ← readLine f
   let c = upper s
   ((), f) ← writeLine f c
        .
        .
        .
  () ← closeFile f
        .
        .
        .
  () ← closeFile f
   return c
```

- File is closed twice: Run time crash

Resource Management: File Handling

- File Handling in Haskell Gone Wrong (Part II)

```
do f  ← openFile "sample.txt"
   (s, f)  ← readLine f
   let c = upper s
   ((), f) ← writeLine f c
       .
       .
       .
   return c
```

Resource Management: File Handling

- File Handling in Haskell Gone Wrong (Part II)

```
do f      ← openFile "sample.txt"
   (s, f)  ← readLine f
   let c = upper s
   ((), f) ← writeLine f c
       .
       .
       .
   return c /*File not closed!!*/
```

- File not closed: Memory leak

Resource Management: Exception Handling

- `MonadError`[5] in Haskell

```
class Monad m ⇒ MonadError e m | m → e where
    throwError :: e → m a
    catchError :: m a → (e → m a) → m a
```

- `throwError` starts exception processing
- `catchError` exception handler

Resource Management: Exception Handling

- Using `MonadError` in Haskell

```
do f ← openFile "sample.txt"
   ((s, f)  ← readLine f
   let c = upper s
   ()  ← closeFile f
   return $ Right c)
      `catchError` (\_ →
            return $ Left "Error in reading file")
```

- Exception may cause memory leak

Well typed programs do not go wrong.

— R. Milner

Well typed programs do not go wrong.

— R. Milner

~~Lights~~Types will guide you home

— Coldplay

Contributions

- Design and implement QuB type system
  - Resources as first class citizens
  - Program objects are restricted or unrestricted
  - Functions that share resources with their arguments or are separate.
- Formalizing and proving important properties of QuB
- QuB is logic of ***BI*** with steroids
  - Environments as graphs
- Working examples

Background Work: Simply Typed Lambda Calculus (STLC)

$$\lambda x.M \begin{cases} \text{Abstract over computation} \\ \text{Define functions} \end{cases}$$

$$MN \begin{cases} \text{Do the computation} \\ \text{Use functions} \end{cases}$$

Background Work: Simply Typed Lambda Calculus (STLC)

$$\lambda x.M \begin{cases} \text{Abstract over computation} \\ \text{Define functions} \end{cases}$$

$$MN \begin{cases} \text{Do the computation} \\ \text{Use functions} \end{cases}$$

$$\frac{\Gamma_x, x : \tau \vdash M : \tau'}{\Gamma \vdash \lambda x.M : \tau \to \tau'} [\to \mathrm{I}] \qquad \frac{\Gamma \vdash M : \tau \to \tau' \qquad \Gamma \vdash N : \tau}{\Gamma \vdash MN : \tau'} [\to \mathrm{E}]$$

Background Work: Simply Typed Lambda Calculus (STLC)

$$\lambda x.M \begin{cases} \text{Abstract over computation} \\ \text{Define functions} \end{cases}$$

$$MN \begin{cases} \text{Do the computation} \\ \text{Use functions} \end{cases}$$

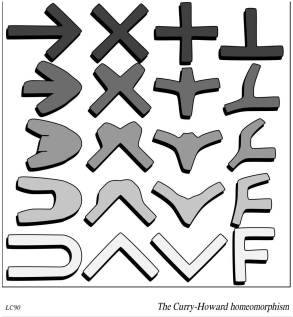
$$\frac{\Gamma_x, x : \tau \vdash M : \tau'}{\Gamma \vdash \lambda x.M : \tau \rightarrow \tau'} [\rightarrow \mathrm{I}] \qquad \frac{\Gamma \vdash M : \tau \rightarrow \tau' \qquad \Gamma \vdash N : \tau}{\Gamma \vdash MN : \tau'} [\rightarrow \mathrm{E}]$$

Hinley-Milner (***HM***) type system ensures sane programs

Background Work: Curry-Howard Correspondence

- Types are Propostions
- Programs are Proofs

HM type system ≡ Second Order Propositional Logic



Source: `http://lucacardelli.name/Artifacts/Drawings/CurryHoward/CurryHoward.pdf`

Background Work: Second Order Propositional Logic

$\boxed{\text{Language}}$

Propostions & connectives $\quad A, B, C ::= x \mid A \supset B \mid \forall x.B \mid \ldots$

Context $\quad \Gamma, \Delta ::= \epsilon \mid \Gamma, A$

$\boxed{\text{Logic Rules}}$

$$\frac{}{A \vdash A}\,[\text{Ax}]$$

$$\frac{\Gamma \vdash B \qquad x \notin \Gamma}{\forall x.B}\,[\forall\text{I}] \qquad \frac{\Gamma \vdash \forall x.B \qquad \Gamma \vdash A}{B[x/A]}\,[\forall\text{E}]$$

$$\frac{\Gamma, A \vdash B}{\Gamma \vdash A \supset B}\,[\supset\text{I}] \qquad \frac{\Gamma \vdash A \supset B \qquad \Gamma \vdash A}{\Gamma \vdash B}\,[\supset\text{E}]$$

Propostions are truth values not resources

$\boxed{\text{Language}}$

Propostions & connectives $\quad A, B, C ::= x \mid A \supset B \mid \forall x.B \mid \ldots$

Context $\quad \Gamma, \Delta ::= \epsilon \mid \Gamma, A$

$\boxed{\text{Logic Rules}}$

$$\frac{}{A \vdash A}\,[\mathrm{Ax}]$$

$$\frac{\Gamma \vdash B \qquad x \notin \Gamma}{\forall x.B}\,[\forall \mathrm{I}] \qquad \frac{\Gamma \vdash \forall x.B \qquad \Gamma \vdash A}{B[x/A]}\,[\forall \mathrm{E}]$$

$$\frac{\Gamma, A \vdash B}{\Gamma \vdash A \supset B}\,[\supset \mathrm{I}] \qquad \frac{\Gamma \vdash A \supset B \qquad \Gamma \vdash A}{\Gamma \vdash B}\,[\supset \mathrm{E}]$$

- Two implicit rules in Propositional Calculus

$$\frac{\Gamma \vdash B}{\Gamma, A \vdash B}\ [\mathrm{WKN}] \qquad\qquad \frac{\Gamma, A, A \vdash B}{\Gamma, A \vdash B}\ [\mathrm{CTR}]$$

Background Work: Substructural Logic

- Two implicit rules in Propositional Calculus

$$\frac{\Gamma \vdash B}{\Gamma, A \vdash B}\,[\mathrm{WKN}] \qquad\qquad \frac{\Gamma, A, A \vdash B}{\Gamma, A \vdash B}\,[\mathrm{CTR}]$$

- Control the use of [WKN] and [CTR]

Certain contexts cannot undergo weakening or contraction[3, 10]

$$\text{Context} \quad \Gamma, \Delta ::= \epsilon \mid \Gamma, \langle A \rangle \mid \Gamma, [A]$$

$$\Gamma, \langle A \rangle \nvdash \Gamma$$
$$\Gamma, \langle A \rangle \nvdash \Gamma, \langle A \rangle, \langle A \rangle$$
$$\Gamma, [A] \vdash \Gamma$$
$$\Gamma, [A] \vdash \Gamma, [A], [A]$$

Language

$$\text{Context}\Gamma, \Delta ::= \epsilon \mid \Gamma, \langle A \rangle \mid \Gamma, [A]$$

$$\text{Propostions} A, B, C ::= X \mid !A \mid A \multimap B \mid A \,\&\, B \mid A \otimes B \mid A \oplus B$$

Structural Rules

$$\frac{}{[A] \vdash A}\,[\mathrm{ID}_{[]}] \qquad \frac{}{\langle A \rangle \vdash A}\,[\mathrm{ID}_{\langle\rangle}]$$

$$\frac{\Gamma, \Delta \vdash A}{\Delta, \Gamma \vdash A}\,[\mathrm{EXCH}] \quad \frac{\Gamma, [A], [A] \vdash B}{\Gamma, [A] \vdash B}\,[\mathrm{CTRN}] \quad \frac{\Gamma \vdash B}{\Gamma, [A] \vdash B}\,[\mathrm{WKN}]$$

$$\frac{[\Gamma] \vdash A}{[\Gamma] \vdash !A}\,[!\mathrm{I}] \qquad \frac{\Gamma \vdash !A \qquad \Delta, [A] \vdash B}{\Gamma, \Delta \vdash B}\,[!\mathrm{E}]$$

$\boxed{\text{Connective Rules}}$

$$\frac{\Gamma, A \vdash B}{\Gamma \vdash A \multimap B}\,[\multimap I] \qquad\qquad \frac{\Gamma \vdash A \multimap B \qquad \Delta \vdash A}{\Gamma, \Delta \vdash B}\,[\multimap E]$$

Connective Rules

$$\frac{\Gamma, A \vdash B}{\Gamma \vdash A \multimap B}\,[\multimap I] \qquad \frac{\Gamma \vdash A \multimap B \qquad \Delta \vdash A}{\Gamma, \Delta \vdash B}\,[\multimap E]$$

$$\frac{\Gamma \vdash A \qquad \Gamma \vdash B}{\Gamma \vdash A \,\&\, B}\,[\& I] \qquad \frac{\Gamma \vdash A \,\&\, B}{\Gamma \vdash A}\,[\& E_1] \qquad \frac{\Gamma \vdash A \,\&\, B}{\Gamma \vdash B}\,[\& E_2]$$

Connective Rules

$$\frac{\Gamma, A \vdash B}{\Gamma \vdash A \multimap B}\,[\multimap I] \qquad \frac{\Gamma \vdash A \multimap B \qquad \Delta \vdash A}{\Gamma, \Delta \vdash B}\,[\multimap E]$$

$$\frac{\Gamma \vdash A \qquad \Gamma \vdash B}{\Gamma \vdash A \,\&\, B}\,[\& I] \qquad \frac{\Gamma \vdash A \,\&\, B}{\Gamma \vdash A}\,[\& E_1] \qquad \frac{\Gamma \vdash A \,\&\, B}{\Gamma \vdash B}\,[\& E_2]$$

$$\frac{\Gamma \vdash A \qquad \Delta \vdash B}{\Gamma, \Delta \vdash A \otimes B}\,[\otimes I] \qquad \frac{\Gamma \vdash A \otimes B \qquad \Delta, A, B \vdash C}{\Gamma, \Delta \vdash C}\,[\otimes E]$$

Connective Rules

$$\frac{\Gamma, A \vdash B}{\Gamma \vdash A \multimap B}\ [\multimap I] \qquad \frac{\Gamma \vdash A \multimap B \qquad \Delta \vdash A}{\Gamma, \Delta \vdash B}\ [\multimap E]$$

$$\frac{\Gamma \vdash A \qquad \Gamma \vdash B}{\Gamma \vdash A \,\&\, B}\ [\& I] \qquad \frac{\Gamma \vdash A \,\&\, B}{\Gamma \vdash A}\ [\& E_1] \qquad \frac{\Gamma \vdash A \,\&\, B}{\Gamma \vdash B}\ [\& E_2]$$

$$\frac{\Gamma \vdash A \qquad \Delta \vdash B}{\Gamma, \Delta \vdash A \otimes B}\ [\otimes I] \qquad \frac{\Gamma \vdash A \otimes B \qquad \Delta, A, B \vdash C}{\Gamma, \Delta \vdash C}\ [\otimes E]$$

$$\frac{\Gamma \vdash A}{\Gamma \vdash A \oplus B}\ [\oplus I_1] \qquad \frac{\Delta \vdash B}{\Delta \vdash A \oplus B}\ [\oplus I_2]$$

$$\frac{\Gamma \vdash A \oplus B \qquad \Delta, A \vdash C \qquad \Delta, B \vdash C}{\Gamma, \Delta \vdash C}\ [\oplus E]$$

Background Work: Linear Logic

- Propositions act like Resources

$$
\begin{array}{rl}
\text{“A consumed to give B”} & A, A \multimap B \vdash B \\
\text{“Both A and B”} & A \otimes B \\
\text{“Choice between A and B”} & A \,\&\, B \\
\text{Cannot create new copies} & A \nvdash A \otimes A \\
\text{Fall back to intuistionstic logic} & !A \multimap B \equiv A \supset B
\end{array}
$$

Background Work: Linear Logic

- Propositions act like Resources

$$
\begin{array}{rl}
\text{“A consumed to give B”} & A, A \multimap B \vdash B \\
\text{“Both A and B”} & A \otimes B \\
\text{“Choice between A and B”} & A \mathbin{\&} B \\
\text{Cannot create new copies} & A \nvdash A \otimes A \\
\text{Fall back to intuistionstic logic} & !A \multimap B \equiv A \supset B
\end{array}
$$

- Curry-Howard Corresondence: Type systems based on Linear Logic
- Active Area of research: L^3[1], F°[6], Linear Haskell[2], Quill[7]
- Resources are first class values

Background Work: Linear Logic

Downside: Asymmetry between implication and conjunction
Multiplicative Fragment:

$$A \otimes B \vdash C \text{ iff } A \vdash B \multimap C$$

Additive Fragment:

$$!A \,\&\, !B \vdash C \text{ iff } !A \vdash !B \multimap C$$

Modality is necessary to avoid over-restriction

Background Work: Qualified Types

General framework to incorporate predicates into type language

$$P \mid \Gamma \vdash M : \sigma$$

"Type of M is σ when predicates in P are satisfied and free variables in M are specified in Γ"[4]

HM system with qualified Types:

$$\frac{x : \sigma \in \Gamma}{P \mid \Gamma \vdash x : \sigma}\;[\text{VAR}] \qquad \frac{P \mid \Gamma \vdash M : \sigma \qquad Q \mid \Gamma_x, x : \sigma \vdash N : \tau}{P, Q \mid \Gamma \vdash (\texttt{let}\ x = M\ \texttt{in}\ N) : \tau}\;[\text{LET}]$$

$$\frac{P \mid \Gamma \vdash M : \sigma \qquad t \notin \texttt{fvs}(\Gamma) \cup \texttt{fvs}(P)}{P \mid \Gamma \vdash M : \forall t.\sigma}\;[\forall\ \text{I}] \qquad \frac{P \mid \Gamma \vdash M : \forall t.\sigma}{P \mid \Gamma \vdash M : [\tau/t]\sigma}\;[\forall\ \text{E}]$$

$$\frac{P \mid \Gamma_x, x : \tau \vdash M : \tau'}{P \mid \Gamma \vdash \lambda x.M : \tau \to \tau'}\;[\to \text{I}] \qquad \frac{P \mid \Gamma \vdash M : \tau \to \tau' \qquad P \mid \Gamma \vdash N : \tau}{P \mid \Gamma \vdash MN : \tau'}\;[\to \text{E}]$$

$$\frac{P, \pi \mid \Gamma \vdash M : \rho}{P \mid \Gamma \vdash M : \pi \Rightarrow \rho}\;[\Rightarrow\text{I}] \qquad \frac{P \mid \Gamma \vdash M : \pi \Rightarrow \rho \qquad P \Rightarrow \pi}{P \mid \Gamma \vdash M : \rho}\;[\Rightarrow\text{E}]$$

$\pi \Rightarrow \rho$: π qualifies ρ

$P \Rightarrow \pi$: P entails π

Background Work: Quill

Quill[7]: Language with qualified types for linear types and first class polymorphism

Three Predicates:

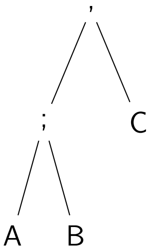
- $\mathtt{Un}\ \tau$ If τ does not have resources or can be copied or dropped easily.
- $\mathtt{Fun}\ \tau$ If τ is a function type
- $\tau \geq \tau'$ If τ less restricting than τ'

Background Work: Quill

- Unrestricted Types: `Un Int`, `Un` Bool
- Restricted or Linear Types: FileHandle
- Function Types: `Fun (Int → Int)`, `Fun (String → String)`

Background Work: Logic of Bunched Implications (***BI***)

- Contexts as trees, called bunches[9]
- Two connective used to combine contexts: $A; B$ or A, B



Background Work: Logic of ***BI***

- Context connective guides choice of implication

$$\frac{\Gamma, A \vdash B}{\Gamma \vdash A \mathrel{-\!\!*} B}\,[\mathrel{-\!\!*}\mathrm{I}] \qquad\qquad \frac{\Gamma; A \vdash B}{\Gamma \vdash A \twoheadrightarrow B}\,[\twoheadrightarrow\mathrm{I}]$$

- Contraction

$$\begin{array}{cc} A; B \vdash A & A; B \vdash B \\ A, B \nvdash A & A, B \nvdash B \end{array}$$

- Weakening

$$\begin{array}{cc} A; B \vdash A; B; B & A; B \vdash A; A; B \\ A, B \nvdash A, B, B & A, B \nvdash A, A, B \end{array}$$

Background Work: Logic of *BI*

Curry-Howard interpretation of logic of ***BI*** [8]

Language

$$
\begin{array}{rl}
\text{Context} & \Gamma, \Delta ::= \{\}_m \mid \{\}_a \mid x : \tau \mid \Gamma, \Delta \mid \Gamma; \Delta \\
\text{Types} & \tau, \upsilon ::= t \mid \iota \mid \tau \twoheadrightarrow \tau \mid \tau \mathrel{-\!\!*} \tau \\
\text{Expressions} & M, N ::= x \mid \lambda x.M \mid \alpha x.M \mid MN
\end{array}
$$

Typing rules

$$
\frac{x : \tau \in \Gamma}{\Gamma \vdash x : \tau}\,[\text{VAR}] \qquad \frac{\Gamma; \Gamma \vdash M : \tau}{\Gamma \vdash M : \tau}\,[\text{CTR}] \qquad \frac{\Gamma \vdash M : \tau}{\Gamma; \Delta \vdash M : \tau}\,[\text{WKN}]
$$

$$
\frac{\Gamma_x, x : \tau \vdash M : \tau'}{\Gamma \vdash \lambda x.M : \tau \mathrel{-\!\!*} \tau'}\,[\mathrel{-\!\!*}\text{I}] \qquad \frac{\Gamma \vdash M : \tau \mathrel{-\!\!*} \tau' \qquad \Delta \vdash N : \tau}{\Gamma, \Delta \vdash MN : \tau'}\,[\mathrel{-\!\!*}\text{E}]
$$

$$
\frac{\Gamma_x; x : \tau \vdash M : \tau'}{\Gamma \vdash \alpha x.M : \tau \twoheadrightarrow \tau'}\,[\twoheadrightarrow\text{I}] \qquad \frac{\Gamma \vdash M : \tau \twoheadrightarrow \tau' \qquad \Delta \vdash N : \tau}{\Gamma; \Delta \vdash MN : \tau'}\,[\twoheadrightarrow\text{E}]
$$

QuB

- Quill: Qualified types for type system based on linear logic
- QuB: Qualified types for type system based on logic of ***BI***

QuB: Types and Predicates

$$
\begin{array}{rl}
\text{Types} & \tau, \upsilon, \phi ::= t \mid \iota \mid \tau \to \tau \\
& \quad \text{where} \qquad \to \in \{\overset{!}{\multimap}, \multimap, \overset{!}{\twoheadrightarrow}, \twoheadrightarrow\} \\
\text{Predicates} & \pi, \omega ::= \mathtt{Un}\ \tau \mid \mathtt{SeFun}\ \tau \mid \mathtt{ShFun}\ \tau \mid \tau \geq \tau' \\
\text{Qualified Types} & \rho ::= \tau \mid \pi \Rightarrow \rho \\
\text{Type schemes} & \sigma ::= \rho \mid \forall t.\sigma
\end{array}
$$

- $\mathtt{SeFun}\ \tau$: τ is a function that is separate from its argument
- $\mathtt{ShFun}\ \tau$: τ is a function that is in sharing with its argument
- $\mathtt{Un}\ \tau$: τ does not have resources or they can be copied/dropped easily

QuB: Types and Predicates

$$
\begin{array}{rl}
\text{Types} & \tau, \upsilon, \phi ::= t \mid \iota \mid \tau \to \tau \\
 & \quad \text{where} \quad \to \in \{\overset{!}{-\!\!\ast}, -\!\!\ast, \overset{!}{\twoheadrightarrow}, \twoheadrightarrow\} \\
\text{Predicates} & \pi, \omega ::= \mathtt{Un}\ \tau \mid \mathtt{SeFun}\ \tau \mid \mathtt{ShFun}\ \tau \mid \tau \geq \tau' \\
\text{Qualified Types} & \rho ::= \tau \mid \pi \Rightarrow \rho \\
\text{Type schemes} & \sigma ::= \rho \mid \forall t.\sigma
\end{array}
$$

- $-\!\!\ast$: Function type that is separate with its argument
- $\twoheadrightarrow$: Function type that is in sharing with its argument
- $\overset{!}{-\!\!\ast}$, $\overset{!}{\twoheadrightarrow}$: unrestriced versions of $-\!\!\ast$ and $\twoheadrightarrow$

QuB: Language

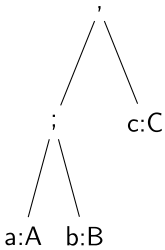
Term Variables $\quad x, y, z \in \mathsf{Var}$

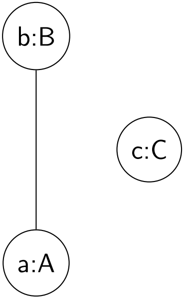
Expressions $\quad M, N ::= x \mid \lambda^{\rightarrow\!\!\!\!\!*} x.M \mid \lambda^{\twoheadrightarrow} x.M \mid MN \mid \texttt{let}\ x = M\ \texttt{in}\ N$

- $\lambda^{\rightarrow\!\!\!\!\!*} x.M$: Introduces $\rightarrow\!\!\!\!\!*$ function
- $\lambda^{\twoheadrightarrow} x.N$: Introduces $\twoheadrightarrow$ function

QuB: Typing Environmnet

- Logic of ***BI***: Contexts are trees
- QuB: Contexts are flattened sharing graphs





- Sharing relation Ψ

$$\forall_{x \in \mathtt{dom}(\Psi)}\; x \in \Psi(x) \qquad \text{(reflexive)}$$

$$\forall_{x,y \in \mathtt{dom}(\Psi)}\; \text{if } y \in \Psi(x) \text{ then } x \in \Psi(y) \qquad \text{(symmetric)}$$

$$\forall_{x,y,z \in \mathtt{dom}(\Psi)}\; \text{if } y \in \Psi(x) \text{ and } z \in \Psi(y) \;\not\Rightarrow\; z \in \Psi(x) \qquad \text{(non-transitive)}$$

QuB: Typing Environment

"x of type σ is in sharing with $\vec{y}$"

$$(x, \sigma, \vec{y}) \in \Gamma$$

$$\text{Typing Context} \quad \Gamma, \Delta ::= \epsilon \mid \Gamma, x^{\vec{y}} : \sigma$$

$$\begin{aligned}
\texttt{Vars}(\Gamma, x^{\vec{y}} : \tau) &= \texttt{Vars}(\Gamma) \cup \{x\} \\
\texttt{Shared}(\Gamma, x^{\vec{y}} : \tau) &= \texttt{Shared}(\Gamma) \cup \{\vec{y}\} \\
\texttt{Used}(\Gamma) &= \texttt{Vars}(\Gamma) \cup \texttt{Shared}(\Gamma)
\end{aligned}$$

$$(\Gamma, x^{\vec{y}} : \tau)^{[a \mapsto \vec{b}]} = \begin{cases} a \notin \vec{y} & (\Gamma^{[a \mapsto \vec{b}]}, x^{\vec{y}} : \tau) \\ a \in \vec{y} & (\Gamma^{[a \mapsto \vec{b}]}, x^{(\vec{y} \setminus a) \cup \vec{b}} : \tau) \end{cases}$$

$$\Gamma^{[\vec{a} \mapsto \vec{b}]} = (\ldots((\Gamma^{[a_1 \mapsto \vec{b}]})^{[a_2 \mapsto \vec{b}]})^{\cdots})^{[a_n \mapsto \vec{b}]}$$

$$\begin{aligned}
\Gamma \circledast \Gamma' &= \Gamma \sqcup \Gamma' \qquad \texttt{if } \texttt{Vars}(\Gamma) \# \texttt{Used}(\Gamma') \wedge \texttt{Vars}(\Gamma') \# \texttt{Used}(\Gamma) \\
\Gamma \oplus \Gamma' &= \Gamma \sqcup \Gamma' \qquad \texttt{if } \texttt{Used}(\Gamma) = \texttt{Used}(\Gamma')
\end{aligned}$$

Structural Rules

$$\frac{}{P \mid x^{\vec{y}} : \sigma \vdash x : \sigma} \text{[ID]}$$

$$\frac{P \mid \Gamma \circledast \Delta \circledast \Delta \vdash M : \sigma \qquad P \vdash \Delta \ \mathbf{un}}{P \mid \Gamma \circledast \Delta \vdash M : \sigma} \text{[CTR-UN]} \qquad \frac{P \mid \Gamma \oplus \Delta \oplus \Delta \vdash M : \sigma}{P \mid \Gamma \oplus \Delta \vdash M : \sigma} \text{[CTR-SH]}$$

$$\frac{P \mid \Gamma \vdash M : \sigma \qquad P \vdash \Delta \ \mathbf{un}}{P \mid \Gamma \circledast \Delta \vdash M : \sigma} \text{[WKN-UN]} \qquad \frac{P \mid \Gamma \vdash M : \sigma}{P \mid \Gamma \oplus \Delta \vdash M : \sigma} \text{[WKN-SH]}$$

Connective Rules

$$\frac{P \mid \Gamma \vdash M : \sigma \qquad P' \mid \Gamma'_x, x : \sigma \vdash N : \tau}{P \cup P' \mid \Gamma \sqcup \Gamma' \vdash (\mathtt{let}\ x = M\ \mathtt{in}\ N) : \tau}\ [\text{LET}]$$

$$\frac{P \mid \Gamma \vdash M : \sigma \qquad t \notin \mathtt{fvs}(\Gamma) \cup \mathtt{fvs}(P)}{P \mid \Gamma \vdash M : \forall t.\sigma}\ [\forall\ \text{I}] \qquad \frac{P \mid \Gamma \vdash M : \forall t.\sigma}{P \mid \Gamma \vdash M : [\tau \backslash t]\sigma}\ [\forall\ \text{E}]$$

$$\frac{P, \pi \mid \Gamma \vdash M : \rho}{P \mid \Gamma \vdash M : \pi \Rightarrow \rho}\ [\Rightarrow\ \text{I}] \qquad \frac{P \mid \Gamma \vdash M : \pi \Rightarrow \rho \qquad P \vdash \pi}{P \mid \Gamma \vdash M : \rho}\ [\Rightarrow\ \text{E}]$$

$$\frac{\begin{array}{c} P \Rightarrow \mathtt{ShFun}\ \phi \qquad P \vdash \Gamma \geq \phi \\ P \mid \Gamma^{[\varnothing \mapsto \{x\}]}, x^{\mathsf{Vars}(\Gamma)} : \tau \vdash M : \tau' \end{array}}{P \mid \Gamma \vdash \lambda^{\twoheadrightarrow} x.M : \phi \tau \tau'}\ [\twoheadrightarrow\ \text{I}] \qquad \frac{\begin{array}{c} P \Rightarrow \mathtt{ShFun}\ \phi \\ P \mid \Gamma \vdash M : \phi \tau \tau' \qquad P \mid \Gamma' \vdash N : \tau \end{array}}{P \mid \Gamma \oplus \Gamma' \vdash MN : \tau'}\ [\twoheadrightarrow\ \text{E}]$$

$$\frac{\begin{array}{c} P \Rightarrow \mathtt{SeFun}\ \phi \qquad P \vdash \Gamma \geq \phi \\ P \mid \Gamma, x^{\varnothing} : \tau \vdash M : \tau' \end{array}}{P \mid \Gamma \vdash \lambda^{-\!\!*} x.M : \phi \tau \tau'}\ [-\!\!*\ \text{I}] \qquad \frac{\begin{array}{c} P \Rightarrow \mathtt{SeFun}\ \phi \\ P \mid \Gamma \vdash M : \phi \tau \tau' \qquad P \mid \Gamma' \vdash N : \tau \end{array}}{P \mid \Gamma \circledast \Gamma' \vdash MN : \tau'}\ [-\!\!*\ \text{E}]$$

QuB: Typing Terms

$$
\begin{prooftree}
\AxiomC{}
\RightLabel{[ID]}
\UnaryInfC{$\varnothing \mid y^{\varnothing} : \tau' \vdash y : \tau'$}
\AxiomC{}
\RightLabel{[ID]}
\UnaryInfC{$\varnothing \mid x^{\varnothing} : \tau \vdash x : \tau$}
\AxiomC{}
\RightLabel{[ID]}
\UnaryInfC{$\varnothing \mid f^{\varnothing} : \tau \mathrel{-\!\!*} \tau' \mathrel{-\!\!*} \upsilon \vdash f : \tau \mathrel{-\!\!*} \tau' \mathrel{-\!\!*} \upsilon$}
\RightLabel{[$\mathrel{-\!\!*}$E]}
\BinaryInfC{$\varnothing \mid x^{\varnothing} : \tau \circledast f^{\varnothing} : \tau \mathrel{-\!\!*} \tau' \mathrel{-\!\!*} \upsilon \vdash fx : (\tau' \mathrel{-\!\!*} \upsilon)$}
\RightLabel{[$\mathrel{-\!\!*}$E]}
\BinaryInfC{$\varnothing \mid y^{\varnothing} : \tau' \circledast x^{\varnothing} : \tau \circledast f^{\varnothing} : \tau \mathrel{-\!\!*} \tau' \mathrel{-\!\!*} \upsilon \vdash fxy : \upsilon$}
\RightLabel{[$EXCH$]}
\UnaryInfC{$\varnothing \mid x^{\varnothing} : \tau \circledast y^{\varnothing} : \tau' \circledast f^{\varnothing} : \tau \mathrel{-\!\!*} \tau' \mathrel{-\!\!*} \upsilon \vdash fxy : \upsilon$}
\RightLabel{[$\mathrel{-\!\!*}$ I]}
\UnaryInfC{$\varnothing \mid x^{\varnothing} : \tau \circledast y^{\varnothing} : \tau' \vdash \lambda^{-\!*} f.fxy : (\tau \mathrel{-\!\!*} \tau' \mathrel{-\!\!*} \upsilon) \mathrel{-\!\!*} \upsilon$}
\RightLabel{[$\mathrel{-\!\!*}$ I]}
\UnaryInfC{$\varnothing \mid x^{\varnothing} : \tau \vdash \lambda^{-\!*} y.\lambda^{-\!*} f.fxy : \tau' \mathrel{-\!\!*} (\tau \mathrel{-\!\!*} \tau' \mathrel{-\!\!*} \upsilon) \mathrel{-\!\!*} \upsilon$}
\RightLabel{[$\equiv$]}
\UnaryInfC{$\varnothing \mid I \vdash \lambda^{-\!*} x.\lambda^{-\!*} y.\lambda^{-\!*} f.fxy : \tau \mathrel{-\!\!*} \tau' \mathrel{-\!\!*} (\tau \mathrel{-\!\!*} \tau' \mathrel{-\!\!*} \upsilon) \mathrel{-\!\!*} \upsilon$}
\end{prooftree}
$$

QuB: Typing Terms

$$
\dfrac{
\dfrac{
\dfrac{
\dfrac{
\dfrac{
\dfrac{}{\varnothing \mid y^{xf} : \tau' \vdash y : \tau'}\,[ID]
\qquad
\dfrac{
\dfrac{}{\varnothing \mid x^{fy} : \tau \vdash x : \tau}\,[ID]
\quad
\dfrac{}{\varnothing \mid f^{xy} : \tau \twoheadrightarrow \tau' \twoheadrightarrow \upsilon \vdash f : \tau \twoheadrightarrow \tau' \twoheadrightarrow \upsilon}\,[ID]
}{\varnothing \mid x^{xy} : \tau \oplus f^{xy} : \tau \twoheadrightarrow \tau' \twoheadrightarrow \upsilon \vdash fx : (\tau' \twoheadrightarrow \upsilon)}\,[\mathrel{-\!\!*}\, E]
}{\varnothing \mid y^{xf} : \tau' \oplus x^{yf} : \tau \oplus f^{xy} : \tau \twoheadrightarrow \tau' \twoheadrightarrow \upsilon \vdash fxy : \upsilon}\,[\twoheadrightarrow E]
}{\varnothing \mid x^{yf} : \tau \oplus y^{xf} : \tau' \oplus f^{xy} : \tau \twoheadrightarrow \tau' \twoheadrightarrow \upsilon \vdash fxy : \upsilon}\,[EXCH]
}{\varnothing \mid x^{y} : \tau \oplus y^{x} : \tau' \vdash \lambda^{\twoheadrightarrow} f.fxy : (\tau \twoheadrightarrow \tau' \twoheadrightarrow \upsilon) \twoheadrightarrow \upsilon}\,[\twoheadrightarrow I]
}{\varnothing \mid x^{\varnothing} : \tau \vdash \lambda^{\twoheadrightarrow} y.\lambda^{\twoheadrightarrow} f.fxy : \tau' \twoheadrightarrow (\tau \twoheadrightarrow \tau' \twoheadrightarrow \upsilon) \twoheadrightarrow \upsilon}\,[\twoheadrightarrow I]
}{\varnothing \mid I \vdash \lambda^{\mathrel{-\!\!*}} x.\lambda^{\twoheadrightarrow} y.\lambda^{\twoheadrightarrow} f.fxy : \tau \mathrel{-\!\!*} \tau' \twoheadrightarrow (\tau \twoheadrightarrow \tau' \twoheadrightarrow \upsilon) \twoheadrightarrow \upsilon}\,[\mathrel{-\!\!*}\, I]
$$

QuB: Syntax Directed Typing Rules

Term structure $\leftrightarrow$ Typing Rule

QuB: Syntax Directed Typing Rules

$$\frac{P \vdash \Gamma_{\vec{y}}\ \mathtt{un} \qquad (P \Rightarrow \tau) \sqsubseteq \sigma}{P \mid \Gamma, x^{\vec{y}} : \sigma \vdash^{s} x : \tau}\ [\mathsf{VAR}^{s}]$$

$$\frac{\begin{array}{c} Q \mid (\Gamma'_x \oplus \Gamma''_x) \circledast \Delta \vdash^{s} M : \upsilon \qquad P \vdash \Delta\ \mathtt{un} \\ P \mid (\Gamma_x, x^{\varnothing} : \sigma) \oplus \Gamma''_x \circledast \Delta \vdash^{s} N : \tau \qquad \sigma = \mathtt{Gen}(\{\Gamma' \oplus \Gamma''_x \circledast \Delta\}, Q \Rightarrow \upsilon) \end{array}}{P, Q \mid (\Gamma \circledast \Gamma') \oplus \Gamma'' \circledast \Delta \vdash^{s} (\mathtt{let}\ x = M\ \mathtt{in}\ N) : \tau}\ [\mathsf{Let}^{s}]$$

$$[\mathsf{VAR}^{s}] \equiv [\mathsf{ID}] + [\forall \mathsf{E}] + [\Rightarrow \mathsf{E}]$$

$$[\mathsf{LET}^{s}] \equiv [\mathsf{LET}] + [\forall \mathsf{I}] + [\Rightarrow \mathsf{I}]$$

QuB: Syntax Directed Typing Rules

$$\frac{P \Rightarrow \texttt{SeFun}\ \phi \qquad P \vdash \Gamma \geq \phi \qquad P \mid \Gamma \circledast x^{\varnothing} : \tau \vdash^{s} M : \upsilon}{P \mid \Gamma \vdash^{s} \lambda^{\multimap} x.M : \phi\tau\upsilon}\ [\multimap \mathsf{I}^{s}] \qquad \frac{P \Rightarrow \texttt{ShFun}\ \phi \qquad P \vdash \Gamma \geq \phi \qquad P \mid \Gamma^{[\varnothing \mapsto \{x\}]} \oplus x^{\texttt{Vars}(\Gamma)} : \tau \vdash^{s} M : \upsilon}{P \mid \Gamma \vdash^{s} \lambda^{\twoheadrightarrow} x.M : \phi\tau\upsilon}\ [\twoheadrightarrow \mathsf{I}^{s}]$$

$$\frac{P \mid \Gamma \circledast \Delta \vdash^{s} M : \phi\upsilon\tau \qquad P \mid \Gamma' \circledast \Delta \vdash^{s} N : \upsilon \qquad P \vdash \Delta\ \texttt{un} \qquad (\Gamma \mathbin{\tilde{\oplus}} \Gamma' \wedge (P \Rightarrow \texttt{ShFun}\ \phi)) \vee (\Gamma \mathbin{\tilde{\circledast}} \Gamma' \wedge (P \Rightarrow \texttt{SeFun}\ \phi))}{P \mid \Gamma \sqcup \Gamma' \circledast \Delta \vdash^{s} MN : \tau}\ [\mathsf{App}^{s}]$$

$\boxed{\text{Context Operations}}$

$$\Gamma \mathbin{\tilde{\circledast}} \Delta = \texttt{Used}(\Gamma) \# \texttt{Vars}(\Delta) \wedge \texttt{Used}(\Delta) \# \texttt{Vars}(\Gamma)$$

$$\Gamma \mathbin{\tilde{\oplus}} \Delta = \texttt{Used}(\Gamma) = \texttt{Used}(\Delta)$$

$$[\twoheadrightarrow \mathsf{I}] \equiv [\twoheadrightarrow \mathsf{I}] \qquad [\multimap \mathsf{I}] \equiv [\multimap \mathsf{I}]$$

$$[\mathsf{APP}^{s}] \equiv [\twoheadrightarrow \mathsf{E}] + [\multimap \mathsf{E}]$$

QuB: Soundess and Completeness of $\vdash^s$

Theorem (Soundness of $\vdash^s$)

If $P \mid \Gamma \vdash^s M : \tau$ *then* $P \mid \Gamma \vdash M : \tau$

Theorem (Completeness of $\vdash^s$)

If $P \mid \Gamma \vdash M : \sigma$ *then* $\exists Q, \tau$ *such that* $Q \mid \Gamma \vdash^s M : \tau$ *and* $(P \mid \sigma) \sqsubseteq \mathit{Gen}(\Gamma, Q \Rightarrow \tau)$

$\boxed{\text{Original Type System}} \equiv \boxed{\text{Syntax Directed Typing Rules}}$

$\boxed{\text{Proof in Original Type System}} \equiv \boxed{\text{Proof in Syntax Directed Typing Rules}}$

QuB: Algorithm $\mathcal{M}$

$$\boxed{\mathcal{M}(S,\Psi,\Gamma \vdash M:\tau) = P, S', \Sigma, \Psi'}$$

$$\begin{aligned}
\mathcal{M}(S,\Psi,\Gamma \vdash x:\tau) &= ([\vec{u}/\vec{t}]P), S'\circ S, \{x\}, \Psi \\
\text{where} \quad (x : \forall\vec{t}.P \Rightarrow \upsilon) &\in S\Gamma \\
S' &= \mathcal{U}([\vec{u}/\vec{t}]\upsilon, S\tau)
\end{aligned}$$

$$\begin{aligned}
&\mathcal{M}(S,\Psi,\Gamma \vdash \lambda^{\twoheadrightarrow}x.M:\tau) = \{P \cup Q\}, S', \Sigma\backslash x, \Psi'' \\
&\quad \text{where} \quad P;S';\Sigma;\Psi' = \mathcal{M}(\mathcal{U}(\tau, u_1 u_2 u_3)\circ S, \Psi, \Gamma, x:u_2 \vdash M:u_3) \\
&\qquad\qquad \Psi'' = \{\forall_{y\in\mathtt{dom}(\Psi')}.\Psi'(y)+x\} \cup \{(x,\{y \mid y \in \mathtt{dom}(\Gamma)\})\} \\
&\qquad\qquad Q = \{\mathtt{ShFun}\ u_1\} \cup \mathtt{Leq}(u_1, \Gamma|_\Sigma) \cup \mathtt{Weaken}(x, u_2, \Sigma, \Psi'')
\end{aligned}$$

$$\begin{aligned}
&\mathcal{M}(S,\Psi,\Gamma \vdash \lambda^{\multimap}x.M:\tau) = \{P \cup Q\}, S', \Sigma\backslash x, \Psi'' \\
&\quad \text{where} \quad P;S';\Sigma;\Psi' = \mathcal{M}(\mathcal{U}(\tau, u_1 u_2 u_3)\circ S, X;\Gamma, x:u_2 \vdash M:u_3) \\
&\qquad\qquad \Psi'' = \Psi' \cup \{(x,\{x\})\} \\
&\qquad\qquad Q = \{\mathtt{SeFun}\ u_1\} \cup \mathtt{Leq}(u_1, \Gamma\,|_\Sigma) \cup \mathtt{Weaken}(x, u_2, \Sigma, \Psi'')
\end{aligned}$$

$$\boxed{\mathcal{M}(S, \Psi, \Gamma \vdash M : \tau) = P, S', \Sigma, \Psi'}$$

$$\begin{aligned}
&\mathcal{M}(S, \Psi, \Gamma \vdash MN : \tau) = \{P \cup P' \cup Q\}, R', \Sigma \cup \Sigma', \Psi'' \\
&\text{where} \qquad P; R; \Sigma; \Psi' = \mathcal{M}(S, \Psi, \Gamma \vdash M : u_1 u_2 \tau) \\
&\qquad P'; R'; \Sigma'; \Psi'' = \mathcal{M}(SR, \Psi', S\Gamma \vdash N : u_2) \\
&\qquad \text{if } \mathcal{C}(\Gamma, \Psi'', \Sigma) = \mathcal{C}(\Gamma, \Psi'', \Sigma') \\
&\qquad\qquad \text{then } Q = \{\texttt{ShFun}\ u_1\} \\
&\qquad\qquad \text{else if } (\Sigma \# \mathcal{C}(R\Gamma, \Psi'', \Sigma') \text{ and } \Sigma' \# \mathcal{C}(R\Gamma, \Psi'', \Sigma)) \\
&\qquad\qquad\qquad \text{then } Q = \{\texttt{SeFun}\ u_1\}
\end{aligned}$$

$$\begin{aligned}
&\mathcal{M}(S, \Psi, \Gamma \vdash \texttt{let}\ x = M\ \texttt{in}\ N : \tau) = (P \cup Q), R', \Sigma \cup \{\Sigma' \backslash x\}, \Psi'' \\
&\text{where} \qquad P; R; \Sigma; \Psi' = \mathcal{M}(S, \Psi, \Gamma \vdash M : u_1) \\
&\qquad \sigma = \texttt{GenI}(R\Gamma; R(P \Rightarrow u_1)) \\
&\qquad P'; R'; \Sigma'; \Psi'' = \mathcal{M}(R, \Psi', \Gamma, x : \sigma \vdash N : \tau) \\
&\qquad Q = \texttt{Un}(\Gamma|_{\Sigma \cap \Sigma'}) \cup \texttt{Weaken}(x, \sigma, \Sigma', \Psi'')
\end{aligned}$$

QuB: Soundess of Algorithm $\mathcal{M}$

Theorem (Soundness of $\mathcal{M}$)

if $\mathcal{M}(S, \Psi, \Gamma \vdash M : \tau) = P, S', \Sigma, \Psi'$ *then* $S'P \mid S'(\Gamma|_{\Sigma}) \vdash M : S'\tau$

$$\boxed{\text{Algorithm M}} \rightarrow \boxed{\text{Type system}}$$

Examples: Basic Structures

- Multiplicative Product

$$\tau \otimes \tau' = \tau \twoheadrightarrow^{*} \tau' \twoheadrightarrow^{*} (\tau \twoheadrightarrow^{*} \tau' \twoheadrightarrow^{*} \upsilon) \twoheadrightarrow^{*} \upsilon$$

$$(,) = \lambda^{\twoheadrightarrow^{*}} x.\lambda^{\twoheadrightarrow^{*}} y.\lambda^{\twoheadrightarrow^{*}} f.fxy$$

- Additive Product

$$\tau \,\&\, \tau' = \tau \twoheadrightarrow^{*} \tau' \twoheadrightarrow (\tau \twoheadrightarrow^{*} \tau' \twoheadrightarrow \upsilon) \twoheadrightarrow \upsilon$$

$$(;) = \lambda^{\twoheadrightarrow^{*}} x.\lambda^{\twoheadrightarrow} y.\lambda^{\twoheadrightarrow} f.fxy$$

- Sums

$$\tau \oplus \tau' = (\tau \to \upsilon) \to (\tau' \to \upsilon) \to \upsilon$$

$$\texttt{case}\ c\ \texttt{of}\ \{f; g\} = \lambda^{\twoheadrightarrow^{*}} c.\lambda^{\twoheadrightarrow} f.\lambda^{\twoheadrightarrow} g.cfg$$

$$\texttt{inl} : \tau \twoheadrightarrow^{*} (\tau \oplus \tau') \qquad \texttt{inr} : \tau' \twoheadrightarrow^{*} (\tau \oplus \tau')$$

$$\texttt{inl} = \lambda^{\twoheadrightarrow^{*}} x.\lambda^{\twoheadrightarrow} f.\lambda^{\twoheadrightarrow} g.fx \qquad \texttt{inr} = \lambda^{\twoheadrightarrow^{*}} y.\lambda^{\twoheadrightarrow} f.\lambda^{\twoheadrightarrow} g.gy$$

QuB: Extension

- User defined types and type classes

- User defined types and type classes
- Kind System with type constructors

$$
\begin{aligned}
\text{Type Variables} &\quad t, u \in \text{Type Variables} \\
\text{Kinds} &\quad \kappa ::= \star \mid \kappa' \to \kappa \\
\text{Types} &\quad \tau^{\kappa} ::= t^{\kappa} \mid T^{\kappa} \mid \tau^{\kappa' \to \kappa} \tau^{\kappa'} \\
\text{Type Constructors} &\quad T^{\kappa} \in \mathcal{T}^{\kappa} \text{ where } \quad \{\otimes, \&, \oplus, \overset{!}{\multimap}, \multimap, \overset{!}{\twoheadrightarrow}, \twoheadrightarrow\} \subseteq \mathcal{T}^{\star \to \star \to} \\
\text{Predicates} &\quad \pi, \omega ::= \mathtt{Un}\ \tau \mid \mathtt{SeFun}\ \tau \mid \mathtt{ShFun}\ \tau \mid \tau \geq \tau' \\
\text{Qualified Types} &\quad \rho ::= \tau^{\star} \mid \pi \Rightarrow \rho \\
\text{Type schemes} &\quad \sigma ::= \rho \mid \forall t.\sigma
\end{aligned}
$$

What next?

Thank You!

Q & A

References I

[1] Amal Ahmed, Matthew Fluet, and Greg Morrisett. L^3: A linear language with locations. *Fundamenta Informaticae*, 77(4):397–449, 2007.

[2] Jean-Philippe Bernardy, Mathieu Boespflug, Ryan R. Newton, Simon Peyton Jones, and Arnaud Spiwack. Linear haskell: Practical linearity in a higher-order polymorphic language. *Proc. ACM Program. Lang.*, 2:5:1–5:29, 2017.

[3] Jean-Yves Girard. Linear logic. *Theoretical Computer Science*, 50(1):1–101, 1987.

[4] Mark P. Jones. A theory of qualified types. *Science of Computer Programming*, 22(3):231 – 256, 1994.

[5] Sheng Liang, Paul Hudak, and Mark Jones. Monad transformers and modular interpreters. In *Proceedings of the 22Nd ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, POPL '95, pages 333–343. ACM, 1995.

References II

[6] Karl Mazurak and Steve Zdancewic. Lolliproc: To concurrency from classical linear logic via curry-howard and control. In *Proceedings of the 15th ACM SIGPLAN International Conference on Functional Programming*, ICFP '10, pages 39–50. ACM, 2010.

[7] J. Garrett Morris. The best of both worlds: Linear functional programming without compromise. In *Proceedings of the 21st ACM SIGPLAN International Conference on Functional Programming*, ICFP 2016, pages 448–461. ACM, 2016.

[8] Peter W. O'Hearn. On bunched typing. *Journal of functional Programming*, 13(4):747–796, 2003.

[9] Peter W. O'Hearn and David J. Pym. The logic of bunched implications. *The Bulletin of Symbolic Logic*, 5(2):215–244, 1999.

[10] Philip Wadler. A taste of linear logic. In Andrzej M. Borzyszkowski and Stefan Sokolowski, editors, *Mathematical Foundations of Computer Science 1993: 18th International Symposium, MFCS'93 Gdańsk, Poland, August 30–September 3, 1993 Proceedings*, pages 185–210. Springer Berlin Heidelberg, 1993.