A GENERIC APPROACH TO FLOW-SENSITIVE POLYMORPHIC EFFECTS

COLIN S. GORDON
ECOOP 2017
BARCELONA, SPAIN
Goal: Give an algebraic characterization of sequential effect systems, sufficient to model prior systems.
TODAY’S TALK

- Goal: Give an algebraic characterization of sequential effect systems, sufficient to model prior systems
  - Guide design, implementation, communication
Goal: Give an algebraic characterization of sequential effect systems, sufficient to model prior systems

- Guide design, implementation, communication

- A new algebraic characterization of sequential effects
TODAY’S TALK

- Goal: Give an algebraic characterization of sequential effect systems, sufficient to model prior systems
  - Guide design, implementation, communication
- A new algebraic characterization of sequential effects
- Derivation of a free effect iteration for most sequential effect systems
TODAY’S TALK

- Goal: Give an algebraic characterization of sequential effect systems, sufficient to model prior systems
  - Guide design, implementation, communication
- A new algebraic characterization of sequential effects
- Derivation of a free effect iteration for most sequential effect systems
- Mention of other results in the paper
REVIEW: EFFECT SYSTEMS
 REVIEW: EFFECT SYSTEMS

- Extend type systems to describe *internals of computations* as well as shape of data:
Extend type systems to describe *internals of computations* as well as shape of data:

\[ \Gamma \vdash e : \tau \implies \Gamma \vdash e : \tau \mid \chi \]
Extend type systems to describe *internals of computations* as well as shape of data:

\[ \Gamma \vdash e : \tau \implies \Gamma \vdash e : \tau | \chi \]

- Locking, memory access, non-termination, Java’s checked exceptions...
Extend type systems to describe *internals of computations* as well as shape of data:

\[ \Gamma \vdash e : \tau \implies \Gamma \vdash e : \tau | \chi \]

- Locking, memory access, non-termination, Java’s checked exceptions...
- For *most* effect systems, we have a concise formulation:
REVIEW: EFFECT SYSTEMS

- Extend type systems to describe *internals of computations* as well as shape of data:
  \[ \Gamma \vdash e : \tau \implies \Gamma \vdash e : \tau \mid \chi \]

- Locking, memory access, non-termination, Java’s checked exceptions...

- For *most* effect systems, we have a concise formulation:
  - A join semilattice of effects (partial order w/ LUB)
Extend type systems to describe *internals of computations* as well as shape of data:

\[ \Gamma \vdash e : \tau \implies \Gamma \vdash e : \tau \mid \chi \]

Locking, memory access, non-termination, Java’s checked exceptions...

For *most* effect systems, we have a concise formulation:

- A join semilattice of effects (partial order w/ LUB)
  - (More needed for effect masking)
A GENERIC APPROACH TO SEQUENTIAL EFFECT SYSTEMS

EFFECT SYSTEMS, GENERICALLY
A GENERIC APPROACH TO SEQUENTIAL EFFECT SYSTEMS

EFFECT SYSTEMS, GENERICALLY

\[
\text{T-Seq} \quad \frac{\Gamma \vdash e : \tau | \chi \quad \Gamma \vdash e' : \tau' | \chi'}{\Gamma \vdash e; e' : \tau' \mid \chi \sqcup \chi'}
\]
A GENERIC APPROACH TO SEQUENTIAL EFFECT SYSTEMS

EFFECT SYSTEMS, GENERICALLY

\[
\begin{align*}
\Gamma \vdash e : \tau \mid \chi & \quad \Gamma \vdash e' : \tau' \mid \chi' \\
\Gamma \vdash e; e' : \tau' \mid \chi \sqcup \chi' \\
\end{align*}
\]

+ plugin for checked exceptions
A GENERIC APPROACH TO SEQUENTIAL EFFECT SYSTEMS

EFFECT SYSTEMS, GENERICALLY

\[
\frac{\Gamma \vdash e : \tau | \chi \quad \Gamma \vdash e' : \tau' | \chi'}{
\Gamma \vdash e ; e' : \tau' | \chi \cup \chi'}
\]

+ plugin for checked exceptions

\[
\Gamma \vdash e : \tau | \{\text{IOException}\} \quad \Gamma \vdash e' : \tau' | \{\text{InvalidArgumentException}\}
\]

\[
\Gamma \vdash e ; e' : \tau' | \{\text{IOException, InvalidArgumentException}\}
\]
“MOST” EFFECT SYSTEMS: COMMUTATIVE EFFECT SYSTEMS
“MOST” EFFECT SYSTEMS: COMMUTATIVE EFFECT SYSTEMS

- Block-structured lock ownership (e.g., for data race freedom)
A GENERIC APPROACH TO SEQUENTIAL EFFECT SYSTEMS

“MOST” EFFECT SYSTEMS: COMMUTATIVE EFFECT SYSTEMS

- Block-structured lock ownership (e.g., for data race freedom)
- Checked exceptions
“MOST” EFFECT SYSTEMS: COMMUTATIVE EFFECT SYSTEMS

- Block-structured lock ownership (e.g., for data race freedom)
- Checked exceptions
- Memory access (regions)
“MOST” EFFECT SYSTEMS: COMMUTATIVE EFFECT SYSTEMS

- Block-structured lock ownership (e.g., for data race freedom)
- Checked exceptions
- Memory access (regions)
- Use of capabilities
“MOST” EFFECT SYSTEMS: COMMUTATIVE EFFECT SYSTEMS

- Block-structured lock ownership (e.g., for data race freedom)
- Checked exceptions
- Memory access (regions)
- Use of capabilities
- Access to UI elements
A GENERIC APPROACH TO SEQUENTIAL EFFECT SYSTEMS

“MOST” EFFECT SYSTEMS: COMMUTATIVE EFFECT SYSTEMS

- Block-structured lock ownership (e.g., for data race freedom)
- Checked exceptions
- Memory access (regions)
- Use of capabilities
- Access to UI elements
- Blocking calls
“MOST” EFFECT SYSTEMS: COMMUTATIVE EFFECT SYSTEMS

- Block-structured lock ownership (e.g., for data race freedom)
- Checked exceptions
- Memory access (regions)
- Use of capabilities
- Access to UI elements
- Blocking calls
- ...

A GENERIC APPROACH TO SEQUENTIAL EFFECT SYSTEMS
A GENERIC APPROACH TO SEQUENTIAL EFFECT SYSTEMS

“MOST” EFFECT SYSTEMS: COMMUTATIVE EFFECT SYSTEMS

- Block-structured lock ownership (e.g., for data race freedom)
- Checked exceptions
- Memory access (regions)
- Use of capabilities
- Access to UI elements
- Blocking calls
- …
WHAT ABOUT EFFECT SYSTEMS *WITH* ORDERING?
A GENERIC APPROACH TO SEQUENTIAL EFFECT SYSTEMS

WHAT ABOUT EFFECT SYSTEMS *WITH* ORDERING?

▶ Unstructured locking
WHAT ABOUT EFFECT SYSTEMS *WITH* ORDERING?

- Unstructured locking
- Unstructured memory accesses (regions)
A GENERIC APPROACH TO SEQUENTIAL EFFECT SYSTEMS

WHAT ABOUT EFFECT SYSTEMS *WITH* ORDERING?

- Unstructured locking
- Unstructured memory accesses (regions)
- Heap-shape-dependent locking
WHAT ABOUT EFFECT SYSTEMS *WITH* ORDERING?

- Unstructured locking
- Unstructured memory accesses (regions)
- Heap-shape-dependent locking
- ...

A GENERIC APPROACH TO SEQUENTIAL EFFECT SYSTEMS
A GENERIC APPROACH TO SEQUENTIAL EFFECT SYSTEMS

WHAT ABOUT EFFECT SYSTEMS *WITH* ORDERING?

- Unstructured locking
- Unstructured memory accesses (regions)
- Heap-shape-dependent locking
- ...
- We call such systems “sequential” (following Tate)
A GENERIC APPROACH TO SEQUENTIAL EFFECT SYSTEMS

WHAT ABOUT EFFECT SYSTEMS *WITH* ORDERING?

- Unstructured locking
- Unstructured memory accesses (regions)
- Heap-shape-dependent locking
- ...
- We call such systems “sequential” (following Tate)
- These systems lack a common algebraic characterization
WHAT DO WE NEED TO MODEL PRIOR SEQUENTIAL EFFECT SYSTEMS?
What do we need to model prior sequential effect systems?

- Still need a join semilattice
WHAT DO WE NEED TO MODEL PRIOR SEQUENTIAL EFFECT SYSTEMS?

- Still need a join semilattice
- Need (partial) sequencing of effects
A GENERIC APPROACH TO SEQUENTIAL EFFECT SYSTEMS

WHAT DO WE NEED TO MODEL PRIOR SEQUENTIAL EFFECT SYSTEMS?

- Still need a join semilattice
- Need (partial) sequencing of effects
- Need iteration of effects
WHAT DO WE NEED TO MODEL PRIOR SEQUENTIAL EFFECT SYSTEMS?

- Still need a join semilattice
- Need (partial) sequencing of effects
- Need iteration of effects
- Need equational theory for simplifying complex effects with effect variables
A GENERIC APPROACH TO SEQUENTIAL EFFECT SYSTEMS

EFFECT QUANTALE
EFFECT QUANTALES

- A relaxation of quantales (see paper for references)
EFFECT QUANTALES

- A relaxation of quantales (see paper for references)
- A set $E$ with binary join $\sqcup$, binary sequence $\triangleright$, top $\top$, seq-unit $I$
A GENERIC APPROACH TO SEQUENTIAL EFFECT SYSTEMS

EFFECT QUANTALES

- A relaxation of quantales (see paper for references)
- A set E with binary join $\sqcup$, binary sequence $\triangleright$, top $\top$, seq-unit $I$
- $\triangleright$ distributes over $\sqcup$ on both sides:
  \[ a \triangleright (b \sqcup c) = (a \triangleright b) \sqcup (a \triangleright c) \]
  \[ (b \sqcup c) \triangleright a = (b \triangleright a) \sqcup (c \triangleright a) \]
A GENERIC APPROACH TO SEQUENTIAL EFFECT SYSTEMS

EFFECT QUANTALES

- A relaxation of quantales (see paper for references)
- A set E with binary join $\sqcup$, binary sequence $\triangleright$, top $\mathbb{T}$, seq-unit I
- $\triangleright$ distributes over $\sqcup$ on both sides:
  \[ a \triangleright (b \sqcup c) = (a \triangleright b) \sqcup (a \triangleright c) \]
  \[ (b \sqcup c) \triangleright a = (b \triangleright a) \sqcup (c \triangleright a) \]
- $\mathbb{T}$ is nilpotent for $\triangleright$ (a $\triangleright$ $\mathbb{T}$ = $\mathbb{T}$ = $\mathbb{T}$ $\triangleright$ a)
A GENERIC APPROACH TO SEQUENTIAL EFFECT SYSTEMS

EFFECT QUANTALES

- A relaxation of quantales (see paper for references)
- A set \( E \) with binary join \( \sqcup \), binary sequence \( \triangleright \), top \( \top \), seq-unit \( l \)
- \( \triangleright \) distributes over \( \sqcup \) on both sides:
  \[
  a \triangleright (b \sqcup c) = (a \triangleright b) \sqcup (a \triangleright c)
  \]
  \[
  (b \sqcup c) \triangleright a = (b \triangleright a) \sqcup (c \triangleright a)
  \]
- \( \top \) is nilpotent for \( \triangleright \) (\( a \triangleright \top = \top = \top \triangleright a \))

MANY USEFUL PROPERTIES FOLLOW FROM THIS DEFINITION.

E.G.,
A PARTIAL ORDER \( \sqsubseteq \)
MONOTONICITY OF \( \triangleright \)
A GENERIC APPROACH TO SEQUENTIAL EFFECT SYSTEMS

EFFECT QUANTALE

- A relaxation of quantales (see paper for references)
- A set $E$ with binary join $\sqcup$, binary sequence $\triangleright$, top $\top$, seq-unit $I$
- $\triangleright$ distributes over $\sqcup$ on both sides:
  \[ a \triangleright (b \sqcup c) = (a \triangleright b) \sqcup (a \triangleright c) \]
  \[ (b \sqcup c) \triangleright a = (b \triangleright a) \sqcup (c \triangleright a) \]
- $\top$ is nilpotent for $\triangleright$ (a $\triangleright$ $\top$ = $\top$ = $\top$ $\triangleright$ a)
A GENERIC APPROACH TO SEQUENTIAL EFFECT SYSTEMS

EXAMPLE: AN EFFECT SYSTEM FOR ATOMICITY
Flanagan and Qadeer wrote two atomicity effect systems – let’s model the simpler one (TLDI 2003)
Flanagan and Qadeer wrote two atomicity effect systems – let’s model the simpler one (TLDI 2003)

Movers (Lipton ’75) are a way to reason about atomicity by considering how local actions commute with interference:
Flanagan and Qadeer wrote *two* atomicity effect systems – let’s model the simpler one (TLDI 2003)

Movers (Lipton ’75) are a way to reason about atomicity by considering how local actions *commute* with interference:

The mover types become effects (B, L, R, A, C), with requisite sequencing
EXAMPLE: AN ATOMICITY EFFECT QUANTALE
EXAMPLE: AN ATOMICITY EFFECT QUANTALE

\[
\begin{array}{c}
T \\
| \\
A \\
\downarrow \\
L \\
B \\
\end{array}
\]

<table>
<thead>
<tr>
<th>;</th>
<th>B</th>
<th>L</th>
<th>R</th>
<th>A</th>
<th>T</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>B</td>
<td>L</td>
<td>R</td>
<td>A</td>
<td>T</td>
</tr>
<tr>
<td>R</td>
<td>R</td>
<td>A</td>
<td>R</td>
<td>A</td>
<td>T</td>
</tr>
<tr>
<td>L</td>
<td>L</td>
<td>L</td>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>A</td>
<td>A</td>
<td>A</td>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
</tbody>
</table>

The set is the mover effects + ERR
EXAMPLE: AN ATOMICITY EFFECT QUANTALE

- The set is the mover effects + ERR
- Join follows Flanagan and Qadeer (plus ERR)
EXAMPLE: AN ATOMICITY EFFECT QUANTALE

- The set is the mover effects + ERR
- Join follows Flanagan and Qadeer (plus ERR)
- Sequencing follows Flanagan and Qadeer (plus ERR)
A GENERIC APPROACH TO SEQUENTIAL EFFECT SYSTEMS

EXAMPLE: AN ATOMICITY EFFECT QUANTALE

- The set is the mover effects + ERR
- Join follows Flanagan and Qadeer (plus ERR)  
- Sequencing follows Flanagan and Qadeer (plus ERR)
- Flanagan and Qadeer already proved the EQ laws
A GENERIC APPROACH TO SEQUENTIAL EFFECT SYSTEMS

HOW GENERAL ARE EFFECT QUANTALES?
HOW GENERAL ARE EFFECT QUANTALES?

- EQs cover more than just Flanagan and Qadeer’s atomicity
HOW GENERAL ARE EFFECT QUANTALES?

- EQs cover more than just Flanagan and Qadeer’s atomicity
- Derived from prior systems’ type judgments (see paper)
HOW GENERAL ARE EFFECT QUANTALES?

- EQs cover more than just Flanagan and Qadeer’s atomicity
- Derived from prior systems’ type judgments (see paper)
- Trickier examples: unstructured locking with recursive acquisition, product of effect quantales
HOW GENERAL ARE EFFECT QUANTALES?

- EQs cover more than just Flanagan and Qadeer’s atomicity
- Derived from prior systems’ type judgments (see paper)
- Trickier examples: unstructured locking with recursive acquisition, product of effect quantales
- Clear relationship to more “foundational” work
HOW GENERAL ARE EFFECT QUANTALES?

- EQs cover more than just Flanagan and Qadeer’s atomicity
- Derived from prior systems’ type judgments (see paper)
- Trickier examples: unstructured locking with recursive acquisition, product of effect quantales
- Clear relationship to more “foundational” work
  - Short version: similar algebras, EQs are slightly more restrictive, EQs induce the other algebras
HOW GENERAL ARE EFFECT QUANTALES?

- EQs cover more than just Flanagan and Qadeer’s atomicity
- Derived from prior systems’ type judgments (see paper)
- Trickier examples: unstructured locking with recursive acquisition, product of effect quantales
- Clear relationship to more “foundational” work
  - Short version: similar algebras, EQs are slightly more restrictive, EQs induce the other algebras
- Free iteration construct for most EQs!
ITERATING SEQUENTIAL EFFECTS: HARDER THAN IT LOOKS

\[
\frac{\Gamma \vdash e : \text{bool} \mid \chi}{\Gamma \vdash \text{while } (e) e' : \tau \mid \chi'} \\
\frac{\Gamma \vdash e' : \tau \mid \chi'}{
\Gamma \vdash \text{while } (e) e' : \tau \mid \chi \triangleright (\chi' \triangleright \chi)^*}
\]
ITERATING SEQUENTIAL EFFECTS: HARDER THAN IT LOOKS

\[ \begin{align*}
\Gamma \vdash e : \text{bool} \mid \chi & \quad \Gamma \vdash e' : \tau \mid \chi' \\
\Gamma \vdash \text{while } (e) e' : \tau \mid \chi \rhd (\chi' \rhd \chi)^* 
\end{align*} \]

- Prior abstract work on sequential effects defers iteration
Prior abstract work on sequential effects defers iteration

Mycroft et al. note that a naive fixed point operator makes every effect idempotent \((\forall X, X \triangleright X = X)\), which is too strong
ITERATING SEQUENTIAL EFFECTS: HARDER THAN IT LOOKS

\[
\Gamma \vdash e : \text{bool} \mid \chi \quad \Gamma \vdash e' : \tau \mid \chi'
\]
\[
\Gamma \vdash \text{while} \ (e) \ e' : \tau \mid \chi \triangleright (\chi' \triangleright \chi)^* \]

- Prior abstract work on sequential effects defers iteration.
- Mycroft et al. note that a naive fixed point operator makes every effect idempotent (\(\forall X, X \triangleright X = X\)), which is too strong.
- Many prior sequential effect systems with iteration are incompatible with that: e.g., Flanagan and Qadeer’s work: 
  - \(B \triangleright B = B\)
  - \(L \triangleright L = L\)
  - \(R \triangleright R = R\)
  - \(A \triangleright A = C\)
  - \(C \triangleright C = C\)
ITERATING SEQUENTIAL EFFECTS: HARDER THAN IT LOOKS

Prior abstract work on sequential effects defers iteration

Mycroft et al. note that a naive fixed point operator makes every effect idempotent ($\forall X, X \triangleright X = X$), which is too strong

Many prior sequential effect systems with iteration are incompatible with that: e.g., Flanagan and Qadeer’s work:

\[
\begin{align*}
B \triangleright B &= B \\
L \triangleright L &= L \\
R \triangleright R &= R \\
A \triangleright A &= C \\
C \triangleright C &= C
\end{align*}
\]

EFFECT QUANTALES INDUCE AN ITERATION OPERATOR COMPATIBLE WITH PRIOR WORK!
A GENERIC APPROACH TO SEQUENTIAL EFFECT SYSTEMS

A LITTLE BIT OF LATTICE THEORY: CLOSURE OPERATORS
A closure operator on a poset $P$ is a function $f:P \rightarrow P$ that is

- Extensive: $\forall e, e \sqsubseteq f(e)$
- Idempotent: $\forall e, f(f(e)) \sqsubseteq f(e)$
- Monotone: $\forall e, e', e \sqsubseteq e' \Rightarrow f(e) \sqsubseteq f(e')$
A closure operator on a poset $P$ is a function $f:P \rightarrow P$ that is

Extensive: $\forall e, e \sqsubseteq f(e)$

Idempotent: $\forall e, f(f(e)) \sqsubseteq f(e)$

Monotone: $\forall e, e', e \sqsubseteq e' \Rightarrow f(e) \sqsubseteq f(e')$

Codomain($f$) is also the set of fixed points of $f$
A closure operator on a poset $P$ is a function $f:P \rightarrow P$ that is

- **Extensive**: $\forall e, e \sqsubseteq f(e)$
- **Idempotent**: $\forall e, f(f(e)) \sqsubseteq f(e)$
- **Monotone**: $\forall e, e', e \sqsubseteq e' \Rightarrow f(e) \sqsubseteq f(e')$

- Codomain($f$) is also the set of fixed points of $f$
- A closure operator (if it exists) is uniquely defined by its range
  - Simple check, constructive proof
A closure operator on a poset $P$ is a function $f: P \to P$ that is
- **Extensive**: $\forall e, e \sqsubseteq f(e)$
- **Idempotent**: $\forall e, f(f(e)) \sqsubseteq f(e)$
- **Monotone**: $\forall e, e', e \sqsubseteq e' \Rightarrow f(e) \sqsubseteq f(e')$

Codomain($f$) is also the set of fixed points of $f$

A closure operator (if it exists) is uniquely defined by its range

Simple check, constructive proof

2/5 laws required for iteration!
A GENERIC APPROACH TO SEQUENTIAL EFFECT SYSTEMS

ITERATION VIA CLOSURE OPERATORS
Picking the results of iteration is easier to think about, constrained by properties.
Picking the results of iteration is easier to think about, constrained by properties.

Other 3/5 iteration laws require the range elements are idempotent, closed under joins, and above 1.
Picking the results of iteration is easier to think about, constrained by properties.

Other 3/5 iteration laws require the range elements are idempotent, closed under joins, and above $I$.

Taking $X$ to the least idempotent element above $X \sqcup I$ is a valid closure operator satisfying all 5 iteration laws.
Picking the results of iteration is easier to think about, constrained by properties

- Other 3/5 iteration laws require the range elements are idempotent, closed under joins, and above $I$

- Taking $X$ to the least idempotent element above $X \sqcup I$ is a valid closure operator satisfying all 5 iteration laws

- Under some mild conditions
ITERATION VIA CLOSURE OPERATORS

- Picking the results of iteration is easier to think about, constrained by properties
  - Other 3/5 iteration laws require the range elements are idempotent, closed under joins, and above I
  - Taking $X$ to the least idempotent element above $X \sqcup I$ is a valid closure operator satisfying all 5 iteration laws
- Under some mild conditions

CLOSURE OPERATORS ALSO APPLY TO SEMANTIC APPROACHES
A GENERIC APPROACH TO SEQUENTIAL EFFECT SYSTEMS

DOES ITERATION DO WHAT WE WANT? YES!
DOES ITERATION DO WHAT WE WANT? YES!

- For the EQ induced by a commutative system (i.e., reuse join as sequencing), iteration is the identity function, as expected.
DOES ITERATION DO WHAT WE WANT? YES!

- For the EQ induced by a commutative system (i.e., reuse join as sequencing), iteration is the identity function, as expected.
- For the atomicity EQ, the derived operator coincides with Flanagan and Qadeer’s hand-constructed version.
A GENERIC APPROACH TO SEQUENTIAL EFFECT SYSTEMS

DOES ITERATION DO WHAT WE WANT? YES!

- For the EQ induced by a commutative system (i.e., reuse join as sequencing), iteration is the identity function, as expected.
- For the atomicity EQ, the derived operator coincides with Flanagan and Qadeer’s hand-constructed version.
- For lock ownership:
  - Iterating acquire/release is an error.
  - Iterating something that preserves lock ownership is the identity.
  - i.e., iteration is valid only for loop-invariant lock ownership.
A GENERIC APPROACH TO SEQUENTIAL EFFECT SYSTEMS

ALSO IN THE PAPER
An abstract core language with singleton effects and effect polymorphism, parameterized by effect quantale and primitives
A GENERIC APPROACH TO SEQUENTIAL EFFECT SYSTEMS

ALSO IN THE PAPER

- An abstract core language with singleton effects and effect polymorphism, parameterized by effect quantale and primitives
- Effect-preserving translation between Flanagan-Qadeer calculus and (instantiation of) our abstract core language
A GENERIC APPROACH TO SEQUENTIAL EFFECT SYSTEMS

ALSO IN THE PAPER

- An abstract core language with singleton effects and effect polymorphism, parameterized by effect quantale and primitives
- Effect-preserving translation between Flanagan-Qadeer calculus and (instantiation of) our abstract core language
- Precise (formal) relationship to prior semantic work
Also in the paper

- An *abstract* core language with singleton effects and effect polymorphism, parameterized by effect quantale and primitives
- Effect-preserving translation between Flanagan-Qadeer calculus and (instantiation of) our abstract core language
- Precise (formal) relationship to prior semantic work
- Subtleties related to substitution with singleton effects
A GENERIC APPROACH TO SEQUENTIAL EFFECT SYSTEMS

ALSO IN THE PAPER

- An abstract core language with singleton effects and effect polymorphism, parameterized by effect quantale and primitives
- Effect-preserving translation between Flanagan-Qadeer calculus and (instantiation of) our abstract core language
- Precise (formal) relationship to prior semantic work
- Subtleties related to substitution with singleton effects

THANKS! QUESTIONS?
BACKUP SLIDES
A GENERIC APPROACH TO SEQUENTIAL EFFECT SYSTEMS

SEQUENTIAL EFFECT SYSTEMS

\[
\Gamma \vdash e : \tau | \chi \quad \Gamma \vdash e' : \tau' | \chi'
\]

\[
\frac{}{\Gamma \vdash e ; e' : \tau' | \chi \triangleright \chi'}
\]
A GENERIC APPROACH TO SEQUENTIAL EFFECT SYSTEMS

SEQUENTIAL EFFECT SYSTEMS

\[ \Gamma \vdash e : \tau \mid \chi \quad \Gamma \vdash e' : \tau' \mid \chi' \]
\[ \Gamma \vdash e ; e' : \tau' \mid \chi \triangleright \chi' \]

\[ \Gamma \vdash e : \text{bool} \mid \chi \quad \Gamma \vdash e' : \tau \mid \chi' \]
\[ \Gamma \vdash \text{while} \ (e) \ e' : \tau \mid \chi \triangleright (\chi' \triangleright \chi)^* \]
A GENERIC APPROACH TO SEQUENTIAL EFFECT SYSTEMS

OTHER SEQUENTIAL EFFECT SYSTEMS
Some effect systems have “pre” and “post” states $\Delta$, like lock sets, or heap shapes.
Some effect systems have “pre” and “post” states $\Delta$, like lock sets, or heap shapes.

\[
\frac{
\Gamma; \Delta \vdash e : \tau \rightarrow \Delta' \mid \chi 
\quad \Gamma; \Delta' \vdash e' : \tau' \rightarrow \Delta'' \mid \chi'
}{
\Gamma; \Delta \vdash e; e' : \tau' \rightarrow \Delta'' \mid \chi \triangleright \chi'
}
\]
Some effect systems have “pre” and “post” states $\Delta$, like lock sets, or heap shapes

$$\Gamma;\Delta \vdash e : \tau \rightarrow \Delta' | \chi \quad \Gamma;\Delta' \vdash e' : \tau' \rightarrow \Delta'' | \chi'$$

$$\Gamma;\Delta \vdash e; e' : \tau' \rightarrow \Delta'' | \chi \triangleright \chi'$$

$$\Gamma;\Delta \vdash e : \text{bool} \rightarrow \Delta' | \chi \quad \Gamma;\Delta' \vdash e' : \tau \rightarrow \Delta | \chi'$$

$$\Gamma;\Delta \vdash \text{while (e) } e' : \tau \rightarrow \Delta' | \chi \triangleright (\chi' \triangleright \chi)^*$$
OTHER SEQUENTIAL EFFECT SYSTEMS

- Some effect systems have “pre” and “post” states \( \Delta \), like lock sets, or heap shapes

\[
\Gamma;\Delta \vdash e : \tau \rightarrow \Delta' \mid \chi \quad \Gamma;\Delta' \vdash e' : \tau' \rightarrow \Delta'' \mid \chi' \\
\Gamma;\Delta \vdash e; e' : \tau' \rightarrow \Delta'' \mid \chi \triangleright \chi'
\]

\[
\Gamma;\Delta \vdash e : \text{bool} \rightarrow \Delta' \mid \chi \quad \Gamma;\Delta' \vdash e' : \tau \rightarrow \Delta \mid \chi' \\
\Gamma;\Delta \vdash \text{while} \ (e) \ e' : \tau \rightarrow \Delta' \mid \chi \triangleright (\chi' \triangleright \chi)^*
\]

- This obscures the fact that \( \Delta \) and \( \chi \) are managed the same way!
A GENERIC APPROACH TO SEQUENTIAL EFFECT SYSTEMS

OTHER SEQUENTIAL EFFECT SYSTEMS — REWRITTEN
\[ \Gamma \vdash e : \tau \mid (\Delta \leadsto \Delta') \otimes \chi \quad \Gamma \vdash e' : \tau' \mid (\Delta' \leadsto \Delta'') \otimes \chi' \]

\[ \Gamma \vdash e; e' : \tau' \mid ((\Delta \leadsto \Delta') \triangleright (\Delta' \leadsto \Delta'')) \otimes (\chi \triangleright \chi') \]
OTHER SEQUENTIAL EFFECT SYSTEMS — REWRITTEN

\[
\Gamma \vdash e : \tau | (\Delta \sim \Delta') \otimes \chi \quad \Gamma \vdash e' : \tau' | (\Delta' \sim \Delta'') \otimes \chi'
\]
\[
\Gamma \vdash e; e' : \tau' | ((\Delta \sim \Delta') \triangleright (\Delta' \sim \Delta'')) \otimes (\chi \triangleright \chi')
\]

\[
\Gamma \vdash e : \text{bool} | (\Delta \sim \Delta') \otimes \chi \quad \Gamma \vdash e' : \tau | (\Delta' \sim \Delta) \otimes \chi'
\]
\[
\Gamma \vdash \text{while } (e) e' : \tau | ((\Delta \sim \Delta') \triangleright ((\Delta' \sim \Delta) \triangleright (\Delta \sim \Delta'))^*) \otimes (\chi \triangleright (\chi' \triangleright \chi)^*)
\]
OTHER SEQUENTIAL EFFECT SYSTEMS — REWRITTEN

\[
\Gamma \vdash e : \tau | (\Delta \leadsto \Delta') \otimes \chi \quad \Gamma \vdash e' : \tau' | (\Delta' \leadsto \Delta'') \otimes \chi'
\]

\[
\Gamma \vdash e; e' : \tau' | ((\Delta \leadsto \Delta') \triangleright (\Delta' \leadsto \Delta'')) \otimes (\chi \triangleright \chi')
\]

\[
\Gamma \vdash e : \text{bool} | (\Delta \leadsto \Delta') \otimes \chi \quad \Gamma \vdash e' : \tau | (\Delta' \leadsto \Delta) \otimes \chi'
\]

\[
\Gamma \vdash \text{while (e) e'} : \tau | ((\Delta \leadsto \Delta') \triangleright ((\Delta' \leadsto \Delta) \triangleright (\Delta \leadsto \Delta')))\star \otimes (\chi \triangleright \chi' \triangleright \chi)\star
\]

- We can run two effect systems at once!

- Look at the \((\Delta \leadsto \Delta')\) effects – there is no natural bottom for their lattice!
A GENERIC APPROACH TO SEQUENTIAL EFFECT SYSTEMS

DESIDERATA FOR ITERATED EFFECTS: X*
A GENERIC APPROACH TO SEQUENTIAL EFFECT SYSTEMS

DESIDERATA FOR ITERATED EFFECTS: $X^*$

$\forall e, e \sqsubseteq e^*$
A GENERIC APPROACH TO SEQUENTIAL EFFECT SYSTEMS

DESIDERATA FOR ITERATED EFFECTS: X*

- **P1:** \( \forall e, e \sqsubseteq e^* \)**
  
  **EXTENSIVE**

- **P2:** \( \forall e, e \triangleright e^* \sqsubseteq e^* \) and \( e^* \triangleright e \sqsubseteq e^* \)
  
  **FOLDING**
A GENERIC APPROACH TO SEQUENTIAL EFFECT SYSTEMS

DESIDERATA FOR ITERATED EFFECTS: $X^*$

- **P1:** $\forall e, e \sqsubseteq e^*$ [EXTENSIVE]

- **P2:** $\forall e, e \triangleright e^* \sqsubseteq e^*$ and $e^* \triangleright e \sqsubseteq e^*$ [FOLDING]

- **P3:** $\forall e, (e^*)^* = e^*$ [IDEMPOTENT]
A GENERIC APPROACH TO SEQUENTIAL EFFECT SYSTEMS

DESIDERATA FOR ITERATED EFFECTS: X*

P1: \( \forall e, e \sqsubseteq e^* \)  
EXTENSIVE

P2: \( \forall e, e \triangleright e^* \sqsubseteq e^* \)  
FOLDING

P3: \( \forall e, (e^*)^* = e^* \)  
IDEMPOTENT

P4: \( \forall e,f, (e \sqcup f)^* = e^* \sqcup f^* \)  
DISTRIBUTIVE
A GENERIC APPROACH TO SEQUENTIAL EFFECT SYSTEMS

DESIDERATA FOR ITERATED EFFECTS: $X^*$

- **P1:** $\forall e, e \subseteq e^*$
  - **EXTENSIVE**

- **P2:** $\forall e, e \triangleright e^* \subseteq e^*$ and $e^* \triangleright e \subseteq e^*$
  - **FOLDING**

- **P3:** $\forall e, (e^*)^* = e^*$
  - **IDEMPOTENT**

- **P4:** $\forall e, f, (e \sqcup f)^* = e^* \sqcup f^*$
  - **DISTRIBUTIVE**

- **P5:** $\forall e, l \subseteq e^*$
  - **“SIMPLE”**
A GENERIC APPROACH TO SEQUENTIAL EFFECT SYSTEMS

DESIDERATA FOR ITERATED EFFECTS: X*

P1: ∀e, e ⊑ e*  
    ➔ EXTENSIVE

P2: ∀e, e ⊓ e* ⊑ e* and e* ⊓ e ⊑ e*  
    ➔ FOLDING

P3: ∀e, (e*)* = e*  
    ➔ IDEMPOTENT

P4: ∀e, f, (e ⊔ f)* = e* ⊔ f*  
    ➔ DISTRIBUTIVE

P5: ∀e, I ⊑ e*  
    ➔ “SIMPLE”
A GENERIC APPROACH TO SEQUENTIAL EFFECT SYSTEMS

DESIDERATA FOR ITERATED EFFECTS: X*

P1: ∀e, e ⊑ e*  
- **EXTENSIVE**

P2: ∀e, e ⊳ e* ⊑ e* and e* ⊳ e ⊑ e*  
- **FOLDING**

P3: ∀e, (e*)* = e*  
- **IDEMPOTENT**

P4: ∀e,f, (e △ f)* = e* △ f*  
- **DISTRIBUTIVE**

P5: ∀e, I ⊑ e*  
- **“SIMPLE”**

Hand-IDed by Flanagan & Qadeer  
Byproduct of I=⊥ in Flanagan and Qadeer
BRING ON THE MONADS!
A GENERIC APPROACH TO SEQUENTIAL EFFECT SYSTEMS

THE SEQUENTIAL SEMANTICS OF PRODUCER EFFECT SYSTEMS

- Ross Tate, POPL 2013

- Derived effectoids: algebraic structure with sequencing, "subeffecting"
  - Non-deterministic sequencing operation
  - Coherence condition ~ "non-determinism respects subeffects"

- Every effect quantale induces an effectoid
  - Effectoids lack an explicit join

- Many (most reasonable) effectoids induce an effect quantale
A GENERIC APPROACH TO SEQUENTIAL EFFECT SYSTEMS

PARAMETRIC EFFECT MONADS AND SEMANTICS OF EFFECT SYSTEMS

- Shin-ya Katsumata, POPL 2014
- Index a monad by an algebra for sequencing: a partially-ordered monoid
- Now called “graded monads”
- “Most of the time” equivalent to effectoids
- Every effect quantale induces a graded monad
- Most partially-ordered monoids induce an effect quantale
A GENERIC APPROACH TO SEQUENTIAL EFFECT SYSTEMS

EFFECT SYSTEMS REVISITED — CONTROL-FLOW ALGEBRA AND SEMANTICS


- Extend graded monads to graded joinads: index by a joinoid rather than a po-monoid
  - monoid + parallel composition + ordered-conditional ?(-,-,-)
  - ?(I,-,-) induces a form of join

- Similar, but weaker equations to effect quantales (only right distributive laws for ?(-,-,-)

- Every total effect quantale induces a joinoid (w/ degenerate parallelism)

- Joinoids can model control effects (effect quantales can’t)