Concurrent monads for shared state

Exequiel Rivas, Tallinn Univ. of Techn.
Tarmo Uustalu, Reykjavik Univ. / Tallinn Univ. of Techn.

MSFP, Tallinn, 8 July 2024

Concurrency and effects

- Is concurrency an effect?
- I'd say no.

- Parallel composition is a high-level control structure, analogous and comparable to sequential composition.

- Monads axiomatize sequential composition of effectful computations
- but they do NOT axiomatize parallel composition.

- Kleene monads axiomatize sequential composition together with nondeterminism and finite iteration.
- Could we do the same for parallel composition?
- Spoiler: Yes, with concurrent monads by Rivas, Jaskelioff 2019; Paquet, Saville 2024

Concurrent monoids

(Gischer 1984/88; Hoare et al. 2009/11)

- A *concurrent monoid* is an ordered set with two ordered monoid structures agreeing in a certain way.

- Explicitly, it is an ordered set $(M, \leq)$ with
  - $\mathrm{id} \in M$, $(;) : M \times M \to M$ with ; monotone, unital, associative,
  - $\mathrm{jd} \in M$, $\| : M \times M \to M$ with $\|$ monotone, unital, associative

  satisfying inequational interchange

$$
\begin{gathered}
\mathrm{id} \leq \mathrm{jd} \\
\mathrm{jd} \,;\, \mathrm{jd} \leq \mathrm{jd} \\
\mathrm{id} \leq \mathrm{id} \parallel \mathrm{id} \\
(k \parallel \ell) \,;\, (m \parallel n) \leq (k \,;\, m) \parallel (\ell \,;\, n)
\end{gathered}
$$

- If the 1st inequation holds as an equality, we say the concurrent monoid is *normal*.

Concurrent monoids ctd

- The 4th inequation implies the 1st.
- The 1st inequation implies the converses of the 2nd and 3rd inequations.

- If $\leq$ is $=$, a concurrent monoid reduces to a *duoid*.
- By Eckmann–Hilton, a normal duoid is the same as a commutative monoid.

- Remark: Classically, one requires commutativity of $\|$ and normality. We don't.

Let's type this: Concurrent categories, take 1

- A *concurrent category* is an ordered "strict monoidal like" category.

- Explicitly, it is a set $|\mathbb{C}|$ and, for all $X, Y \in |\mathbb{C}|$, an ordered set $(\mathbb{C}(X, Y), \leq)$ with
  - $\mathrm{id} \in \mathbb{C}(X, X)$, $(;) : \mathbb{C}(X, Y) \times \mathbb{C}(Y, Z) \to \mathbb{C}(X, Z)$ with $(;)$ monotone, unital, associative,

  (= an ordered category)
  - $\mathrm{I} \in |\mathbb{C}|$, $\otimes : |\mathbb{C}| \times |\mathbb{C}| \to |\mathbb{C}|$ with $\otimes$ strictly unital, associative,
  - $\mathrm{jd} \in \mathbb{C}(\mathrm{I}, \mathrm{I})$, $\| : \mathbb{C}(X, Z) \times \mathbb{C}(Y, W) \to \mathbb{C}(X \otimes Y, Z \otimes W)$ with $\|$ monotone, unital, associative

  satisfying

$$\mathrm{id} \leq \mathrm{jd}$$
$$\mathrm{jd} \,;\, \mathrm{jd} \leq \mathrm{jd}$$
$$\mathrm{id} \leq \mathrm{id} \parallel \mathrm{id}$$
$$(k \parallel \ell) \,;\, (m \parallel n) \leq (k \,;\, m) \parallel (\ell \,;\, n)$$

  (= a strict monoidal like structure, but $\mathrm{I} : 1 \to \mathbb{C}$ and $\otimes : \mathbb{C} \times \mathbb{C} \to \mathbb{C}$ are lax functors, i.e., jd and $\parallel$ preserve identity and composition laxly)

Concurrent categories, properly

- We don't want strict monoidal(-like)ness, but non-strict, however with the unitality and associativity laws "central".
- We assume given an ordered monoidal category $\mathbb{C} = (\mathbb{C}, \leq, \mathrm{id}, (;), \mathsf{I}, \otimes)$.
- A *concurrent category* with base $\mathbb{C}$ is given by an ordered "(non-strict) 'monoidal-like" category $\mathbb{K}$ with the same objects as $\mathbb{C}$ and an identity-on-objects ordered "strict monoidal like" functor J.

Concurrent categories, properly

- Explicitly, $\mathbb{K}$ consists of a set $|\mathbb{K}| = |\mathbb{C}|$ and, for all $X, Y \in |\mathbb{K}|$, an ordered set $(\mathbb{K}(X,Y), \leq^{\mathsf{K}})$ with
  - $\mathsf{id}^{\mathsf{K}} \in \mathbb{K}(X,X)$, $(;^{\mathsf{K}}) : \mathbb{K}(X,Y) \times \mathbb{K}(Y,Z) \to \mathbb{K}(X,Z)$ with $(;^{\mathsf{K}})$ monotone, unital, associative,

  (= an ordered category)
  - $\mathsf{I}^{\mathsf{K}} = \mathsf{I} \in |\mathbb{K}|$, $\otimes^{\mathsf{K}} = \otimes : |\mathbb{K}| \times |\mathbb{K}| \to |\mathbb{K}|$ with $\otimes^{\mathsf{K}}$ unital, associative (in $\mathbb{K}$!) via $J\lambda$, $J\rho$, $J\alpha$
  - $\mathsf{jd} \in \mathbb{K}(\mathsf{I},\mathsf{I})$, $\parallel : \mathbb{K}(X,Z) \times \mathbb{K}(Y,W) \to \mathbb{K}(X \otimes Y, Z \otimes W)$ with $\parallel$ monotone and unital, associative up to $J\lambda$, $J\rho$, $J\alpha$

  satisfying

$$\begin{gathered}
\mathsf{id}^{\mathsf{K}} \leq^{\mathsf{K}} \mathsf{jd} \\
\mathsf{jd} \mathbin{;} \mathsf{jd} \leq^{\mathsf{K}} \mathsf{jd} \\
\mathsf{id}^{\mathsf{K}} \leq^{\mathsf{K}} \mathsf{id}^{\mathsf{K}} \parallel \mathsf{id}^{\mathsf{K}} \\
(k \parallel \ell) \mathbin{;^{\mathsf{K}}} (m \parallel n) \leq^{\mathsf{K}} (k \mathbin{;^{\mathsf{K}}} m) \parallel (\ell \mathbin{;^{\mathsf{K}}} n)
\end{gathered}$$

  (= a strict monoidal like structure, but jd and $\parallel$ preserve identity and composition laxly)

- J is an identity-on-objects ordered functor that is strict monoidal like, but preserves map operations I and $\otimes$ oplaxly in that $J\mathsf{I} \leq^{\mathsf{K}} \mathsf{jd}$ and $J(f \otimes g) \leq^{\mathsf{K}} Jf \parallel Jg$.

Toward concurrent monads

- For an ordered mon. cat. $\mathbb{C}$, what data and (inequations) we need equip an ordered functor $T : \mathbb{C} \to \mathbb{C}$ with

  to get that $\mathbb{K}$ with
  $|\mathbb{K}| = |\mathbb{C}|$, $\mathbb{K}(X, Y) = \mathbb{C}(X, TY)$, $k \leq^{\mathsf{K}} \ell$ iff $k \leq \ell$
  is a concurrent category?
- We can proceed from these monotone bijections between homposets.

$$\frac{\mathbb{C}(X, TY) \times \mathbb{C}(Y, TZ) \to \mathbb{C}(X, TZ) \text{ nat. in } X \text{ in Poset}}{\mathbb{C}(Y, TZ) \to \mathbb{C}(TY, TZ) \text{ in Poset}}$$

$$\frac{\mathbb{C}(Y, TZ) \to \mathbb{C}(TY, TZ) \text{ nat. in } Y \text{ in Poset}}{T(TZ) \to TZ \text{ in } \mathbb{C}}$$

$$\frac{\mathbb{C}(X, TY) \times \mathbb{C}(U, TV) \to \mathbb{C}(X \otimes U, T(Y \otimes V)) \text{ nat. in } X, U \text{ in Poset}}{TY \otimes TV \to T(Y \otimes V) \text{ in } \mathbb{C}}$$

Concurrent monads

(Rivas, Jaskelioff 2019; Paquet, Saville 2024)

- A *concurrent monad* on an ordered monoidal category $(\mathbb{C}, \leq, \mathrm{I}, \otimes)$ is an ordered functor that carries both an ordered monad and an ordered lax monoidal functor structure, agreeing in a certain way.

- Explicitly, it consists of an ordered functor $T : \mathbb{C} \to \mathbb{C}$ with
  - $\eta_X : X \to TX$, $\mu_X : T(TX) \to TX$ natural in X,
  - $\psi^0 : \mathrm{I} \to T\mathrm{I}$, $\psi_{X,Y} : TX \otimes TY \to T(X \otimes Y)$ natural in X, Y,

  satisfying the equations of a monad and a lax monoidal functor and inequational interchange.

- For example, the 4th inequation takes the form

$$
\begin{tikzcd}
T(TX) \otimes T(TY) \arrow[r, "\psi_{TX,TY}"] \arrow[d, "\mu_X \otimes \mu_Y"'] & T(TX \otimes TY) \arrow[r, "T\psi_{X,Y}"] \arrow[d, phantom, "\geq"] & T(T(X \otimes Y)) \arrow[d, "\mu_{X \otimes Y}"] \\
TX \otimes TY \arrow[rr, "\psi_{X,Y}"] & & T(X \otimes Y)
\end{tikzcd}
$$

- If $\leq$ is $=$, this reduces to a *lax monoidal* (aka. *commutative*) *monad*.

Kleisli for a concurrent monad

- A concurrent monad T on an ordered monoidal category $\mathbb{C}$ induces a concurrent category $(\mathbb{K}, J)$ via this Kleisli construction:
  - $|\mathsf{Kl}(T)| = |\mathbb{C}|$,
    $\mathsf{Kl}(T)(X, Y) = \mathbb{C}(X, TY)$,
  - $k \leq^{\mathsf{K}} \ell$ in $\mathsf{Kl}(T)(X, Y)$ iff $k \leq \ell$ in $\mathbb{C}(X, TY)$,
  - $\mathrm{id}^{\mathsf{K}}{}_X = \eta_X$,
    $k \,;^{\mathsf{K}} \ell = k \,;\, T\ell \,;\, \mu_Z$ for $k \in \mathsf{Kl}(T)(X, Y)$, $\ell \in \mathsf{Kl}(T)(Y, Z)$,
  - $\mathrm{jd} = \psi^0$,
    $k \parallel \ell = (k \otimes \ell) \,;\, \psi_{Y,V}$ for $k \in \mathsf{Kl}(T)(X, Y)$, $\ell \in \mathsf{Kl}(T)(U, V)$,
  - $JX = X$
    $Jf = f ; \eta_Y$ for $f \in \mathbb{C}(X, Y)$.
- J has a right ordered adjoint K:
  - $KX = TX$,
    $Kk = Tk; \mu_Y$ for $k \in \mathsf{Kl}(T)(X, Y)$
- K is "lax monoidal like", but preserves jd, $\parallel$ oplaxly.

Kleisli for a concurrent monad ctd.

- Conversely,
  if J in a concurrent category $(\mathbb{K}, J)$ has a right adjoint with the above properties,
  then the ordered functor $T = K \cdot J$ on $\mathbb{C}$ carries the structure of a concurrent monad with (K, J) its Kleisli construction.

Example: Writer

- We consider examples on the ordered Cartesian monoidal closed category $(\mathrm{Poset}, \leq, 1, \times, \Rightarrow)$
  which has as objects ordered sets, as maps monotone functions, $f \leq g$ for $f, g : X \to Y$ iff $fx \leq^Y gx$ for all x.

- The writer/reader/state monads from FP are concurrent monads for parameters with suitable structure.

- $TX = M \times X$
  for $(M, o, +, e, \cdot)$ a concurrent monoid

- $\eta_X\, x = (o, x)$
  $\mu_X\, (m, (m', x)) = (m + m', x)$
  $\psi^0 \star = (e, \star)$
  $\psi_{X,Y}\, ((m, x), (m', y)) = (m \cdot m', (x, y))$

- This concurrent monad is normal ($\eta_1 = \psi^0$) if the concurrent monoid is normal ($o = e$).

Reader

- $TX = S \Rightarrow X$
  for S any ordered set (it is ok if $\leq^S$ is $=$)
- NB! $S \Rightarrow X$ is the ordered set of monotone functions from S to X (if $\leq^S$ is $=$, this all functions are monotone).

- $\eta_X\, x = \lambda \_.\, x$
  $\mu_X\, f = \lambda s.\, f\, s\, s$
  $\psi^0\, \star = \lambda \_.\, \star$
  $\psi_{X,Y}\, (f, g) = \lambda s.\, (f\, s, g\, s)$

- The interchange inequations hold as equalities, so this concurrent monad is a commutative ordered monad.

State

- $TX = S \Rightarrow S \times X$
  for $(S, \top, \wedge)$ a lower semilattice
  (so $\leq^S$ cannot be $=$ unless S is a singleton)
- Recall that $S \Rightarrow S \times X$ consists of monotone functions only.
- $\eta_X\, x = \lambda s.\, (s, x)$
  $\mu_X\, f = \lambda s.\, \mathrm{let}\ (s', g) = f\, s\ \mathrm{in}\ g s'$
  $\psi^0 \star = \lambda\_.\, (\top, \star)$
  $\psi_{X,Y}\, (f, g) = \lambda s.\, \mathrm{let}\ ((s_0, x), (s_1, y)) = (f\, s, g\, s)\ \mathrm{in}\ (s_0 \wedge s_1, (x, y))$
- This concurrent monad is not normal ($\eta_1 \neq \psi^0$).
- Every computation is one atomic step.
- In parallel computation, the two atomic steps happen "truly concurrently", the competing writes are reconciled by $\wedge$.

Resumptions for interleaving shared state concurrency

- The idea of resumptions: a computation consists of small steps (organized into a tree).
- $TX = \mu Z.\, \underbrace{X}_{ret} \underbrace{+}_{or} (\underbrace{S \Rightarrow}_{grab} \underbrace{\mathsf{List}}_{branch} (\underbrace{S \times}_{yield} \underbrace{Z}_{repeat}))$

  for S discretely ordered (for simplicity)
- $\leq^{TX}$ induced by $\leq^{\mathsf{List}\,Y}$ induced by order-preserving inclusion between lists

- $\eta\, x = ret\, x$
  $\mu\, (ret\, r) = r$
  $\mu\, (grab\, k) = grab\, (\lambda s.\, [(s', \mu_X\, r \mid (s', r) \leftarrow k\, s])$
- $\psi^0 \star = ret \star$
  $\psi\, (ret\, x, ret\, y) = ret\, (x, y)$
  $\psi\, (ret\, x, grab\, \ell) = grab\, (\lambda s.\, [(s', \psi\, (ret\, x, r)) \mid (s', r) \leftarrow \ell\, s])$
  $\psi\, (grab\, k, ret\, y) = grab\, (\lambda s.\, [(s', \psi\, (r, ret\, y)) \mid (s', r) \leftarrow k\, s])$
  $\psi\, (grab\, k, grab\, \ell) = grab\, (\lambda s.\, [(s', \psi\, (r, grab\, \ell)) \mid (s', r) \leftarrow k\, s]$
  $\qquad +\!\!+ [(s', \psi\, (grab\, k, r)) \mid (s', r) \leftarrow \ell\, s])$

- Although we use List rather than $\mathcal{M}_\mathrm{f}$ or $\mathcal{P}_\mathrm{f}$, asssociativity of ψ and the 4th interchange inequation hold.

Resumptions ctd.

- T supports algebraic operations for reading and writing
- $get : (S \Rightarrow X) \to TX$
  $get\, f = grab\,(\lambda s.\, [(s, f\ s)])$
- $put : S \times X \to TX$
  $put\,(s', x) = grab\,(\lambda \_.\, [(s', x)])$

- and a high-level control operation for atomizing computations (concatenate all small steps into one single one)
- $stitch : TX \to TX$
  $stitch\, r = grab\,(\lambda s.\, stitch'(s, r))$ where
  $stitch' : S \times TX \to \mathsf{List}\,(S \times TX)$
  $stitch'\,(s, ret\, x) = [(s, ret\, x)]$
  $stitch'\,(s, grab\, k) = concat\,(\mathsf{List}\, stitch'\,(k\, s))$

Bags of traces for interleaving shared state concurrency

- $TX = \underbrace{\mathcal{M}_{\mathrm{f}}}_{BRANCH}(T'X)$ where $T'X = (\mu Z.\,\underbrace{X}_{ret}\underbrace{+}_{or}\underbrace{S\times}_{grab}\underbrace{S\times}_{yield}\underbrace{Z}_{repeat})$
- $\leq^{TX}$ induced by $\leq^{\mathcal{M}_{\mathrm{f}}Y}$ induced by multiset inclusion
- $\eta\, x = \{ret\; x\}$
  $\mu\; ts = \bigcup(\mathcal{M}_{\mathrm{f}}\, \mu'\; ts)$ where
  $\mu' : T'(TX) \to TX$
  $\mu'\,(ret\; ts) = ts$
  $\mu'\,(grab\, s\, s'\; t) = T'\,(grab\, s\, s')\,(\mu'\; t)$
- $\psi^0 \star = \{ret \star\}$
  $\psi\,(ts_0, ts_1) = \bigcup\{t_0 \sqcup\!\sqcup\; t_1 \mid t_0 \leftarrow ts_0, t_1 \leftarrow ts_1\}$ where
  $\sqcup\!\sqcup : T'X \times T'Y \to T(X, Y)$
  $ret\; x \sqcup\!\sqcup\; ret\; y = \{ret\,(x, y)\}$
  $ret\; x \sqcup\!\sqcup\; grab\, s\, s'\; t = T'\,(grab\, s\, s')\,(ret\; x \sqcup\!\sqcup\; t)$
  $grab\, s\, s'\; t \sqcup\!\sqcup\; ret\; y = T'\,(grab\, s\, s')\,(t \sqcup\!\sqcup\; ret\; y)$
  $$grab\, s_0\, s_0'\; t_0 \sqcup\!\sqcup\; grab\, s_1\, s_1'\; t_1 = T'\,(grab\, s_0\, s_0')\,(t_0 \sqcup\!\sqcup\; grab\, s_1\, s_1'\; t_1) \cup T'\,(grab\, s_1\, s_1')\,(grab\, s_0\, s_0'\; t_0 \sqcup\!\sqcup\; t_1)$$
- *get*, *put*, *stitch* also definable, e.g.,
  $put\,(s', x) = \{grab\, s\, s'\,(ret\; x) \mid s \in S\}$

Duoidal categories

- Concurrent monoids are objects with structure of the ordered category Poset.

- To define what makes a concurrent monoid object in a general ordered category $\mathbb{D}$, this category has to be ordered duoidal.
- An *ordered duoidal category* is an ordered category $\mathbb{D}$ with two ordered monoidal structures $(I, \odot)$, $(J, \circledast)$ and maps

$$\begin{gathered}
J \to I \\
J \to J \odot J \\
I \circledast I \to I \\
(F \odot G) \circledast (H \odot K) \to (F \circledast H) \odot (G \circledast K) \text{ nat. in } F,\, G,\, H,\, K
\end{gathered}$$

satisfying certain equations.

Concurrent monoid objects

- A *concurrent monoid object* in an ordered duoidal cataegory is an object M with two monoid structures (o, a) wrt. $(I, \odot)$ and (e, m) wrt. $(J, \circledast)$ satisfying inequational interchange.

- A (classical) concurrent monoid is a concurrent monoid object in the ordered duoidal category $(\mathrm{Poset}, 1, \times, 1, \times)$.

Concurrent monads as concurrent monoids

- If $(\mathbb{C}, \leq, \mathsf{I}, \otimes)$ is an ordered monoidal category, then $([\mathbb{C}, \mathbb{C}]_{\mathsf{a}}, \leq, \mathsf{Id}, \cdot, \mathsf{Jd}, \star)$ is an ordered duoidal category.
- Here $(\mathsf{Jd}, \star)$ is the Day convolution monoidal structure

$$\mathsf{Jd}Z = \mathbb{C}(\mathsf{I}, Z) \bullet \mathsf{I}$$
$$(F \star G)Z = \int^{X,Y} \mathbb{C}(X \otimes Y, Z) \bullet (FX \otimes GY)$$

- A (accessible) concurrent monad is the same as a concurrent monoid object in this ordered duoidal category.

  The carrier is $T \in |[\mathbb{C}, \mathbb{C}]_{\mathsf{a}}|$ and the structure maps are
  - $\eta : \mathsf{Id} \to T$, $\mu : T \cdot T \to T$,
  - $\mathsf{e} : \mathsf{Jd} \to T$, $\mathsf{m} : T \star T \to T$.

Conclusions and future work

- Parallel composition is axiomatizable relatively smoothly for typed effectful computation.
- Ordered category theory streamlines the development. Ordered duoidal categories are particularly important.
- The resumption model can be captured easily and is flexible for variations.

- How to extend an ordered monad canonically to a concurrent monad?
- Concurrent monads for transactional memories, relaxed memories?
- Axiomatization of atomization, of operations of cooperative concurrency?

References

- J. L. Gischer. The equational theory of posets. *Theor. Comput. Sci.* 1988.
- C. A. R. Hoare, B. Möller, G. Struth, I. Wehrman. Concurrent Kleene algebra and its foundations. *J. Log. Algebraic Program.* 2011.
- E. Rivas, M. Jaskelioff. Merging monads. HAL 2019.
- H. Paquet, P. Saville. Effectful semantics in bicategories: strong, commutative and concurrent pseudomonads. LICS 2024.
- C. Heunen, J. Sigal. Duoidally enriched Freyd categories. RELMICS 2023.
- P. Cenciarelli, E. Moggi. A syntactic approach to modularity in denotational semantics. CTCS '93.
- S. Goncharov, L. Schröder. A coinductive calculus for asynchronous side-effecting. *Inf. Comput.* 2013.
- Y. Dvir, O. Kammar, O. Lahav. An algebraic theory for shared-state concurrency. APLAS 2022.
- Y. Dvir, O. Kammar, O. Lahav. A denotational approach to release/acquire concurrency. ESOP 2024.