fair termination of binary sessions

Luca Ciccone and <u>**Luca Padovani**</u>, Università di Torino
with contributions by **Francesco Dagnino**, Università di Genova

outline

1. fair termination
2. binary sessions
3. on subtyping and why it matters
4. on fair subtyping and how to use it
5. soundness
6. concluding remarks

outline

termination properties

Definition (reduction system)

A **reduction system** is a pair $(\mathcal{S}, \rightarrow)$ made of a set $\mathcal{S}$ of **states** and a **reduction relation** $\rightarrow \subseteq \mathcal{S} \times \mathcal{S}$.

Definition (run)

A **run** of $C \in \mathcal{S}$ is a **maximal** sequence $C \rightarrow C_1 \rightarrow C_2 \rightarrow \cdots$

- A is **weakly terminating**
- B is **terminated**
- C is **non terminating**

$$\begin{tikzcd} & A \arrow[loop above] \arrow[dl] \arrow[dr] & \\ B & & C \arrow[loop below] \end{tikzcd}$$

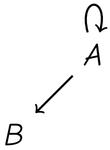
fair termination

Definition (fair termination)

We say that C is **fairly terminating** if every **infinite** run of C is "unfair" (think of unreasonable, unrealistic, extremely unlikely, etc.)

Definition (unfair run)

A run is **unfair** if each of its states is weakly terminating but none of them is terminated (an unfair run is necessarily **infinite**)



- the run $A \to A \to A \to \cdots$ is unfair
- A is **fairly terminating**
- the run $A \to C \to C \to \cdots$ is fair
- A is **not** fairly terminating

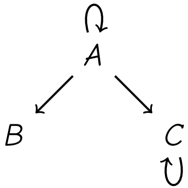
fair termination

Definition (fair termination)

We say that C is **fairly terminating** if every **infinite** run of C is "unfair" (think of unreasonable, unrealistic, extremely unlikely, etc.)

Definition (unfair run)

A run is **unfair** if each of its states is weakly terminating but none of them is terminated (an unfair run is necessarily **infinite**)



- the run $A \to A \to A \to \cdots$ is unfair
- ~~A is **fairly terminating**~~
- the run $A \to C \to C \to \cdots$ is fair
- A is **not** fairly terminating

property of fair termination

Theorem

A is fairly terminating iff $\forall B : A \Rightarrow B$ implies B weakly terminating

Notes

- this property holds for the fairness notion we have adopted
- there are many different fairness notions, the one we use is an instance of **relative fairness** by Queille and Sifakis [1983] also known as **full fairness** [van Glabbeek and Höfner, 2019]
- a type system that ensures **weak termination** is also a type system that ensures **fair termination**!

property of fair termination

Theorem

A is fairly terminating iff $\forall B : A \Rightarrow B$ implies B weakly terminating

Notes

- this property holds for the fairness notion we have adopted
- there are many different fairness notions, the one we use is an instance of **relative fairness** by Queille and Sifakis [1983] also known as **full fairness** [van Glabbeek and Höfner, 2019]
- a type system that ensures **weak termination** is also a type system that ensures **fair termination**!

outline

fair termination of binary sessions

Goal: in a well-typed program all sessions fairly terminate

A fairly terminating binary session

$$Buyer(x) \triangleq x!\{\mathsf{add} : Buyer\langle x\rangle, \mathsf{pay} : \mathsf{done}\}$$
$$Seller(x) \triangleq x?\{\mathsf{add} : Seller\langle x\rangle, \mathsf{pay} : \mathsf{done}\}$$

A non-terminating binary session

$$Buyer_\infty(x) \triangleq x!\mathsf{add}.Buyer_\infty\langle x\rangle$$
$$Seller(x) \triangleq x?\{\mathsf{add} : Seller\langle x\rangle, \mathsf{pay} : \mathsf{done}\}$$

why fair termination?

(Fair) session termination is **expected**

- see meaning of "session"

Fair termination implies **progress**

- suppose $P \Rightarrow Q$ and Q has pending actions in session x
- then $Q \Rightarrow R$ where x is terminated (no pending actions on x)

Fair termination enables **compositional reasoning**

$$\begin{aligned} \mathit{Buyer}(x) &\triangleq \ldots\text{whatever}\ldots \\ \mathit{Seller}(x,y) &\triangleq x?\{\mathsf{add} : \mathit{Seller}\langle x,y\rangle, \mathsf{pay} : y!\mathsf{ship}\} \\ \mathit{Shipper}(y) &\triangleq y?\mathsf{ship} \end{aligned}$$

- x enjoys progress $\Rightarrow$ y …? *Shipper* ☹
- x fairly terminates $\Rightarrow$ y fairly terminates *Shipper* ☺

why fair termination?

(Fair) session termination is **expected**

- see meaning of "session"

Fair termination implies **progress**

- suppose $P \Rightarrow Q$ and Q has pending actions in session x
- then $Q \Rightarrow R$ where x is terminated (no pending actions on x)

Fair termination enables **compositional reasoning**

$$\begin{aligned} Buyer(x) &\triangleq \ldots\text{whatever}\ldots \\ Seller(x, y) &\triangleq x?\{\mathsf{add} : Seller\langle x, y\rangle, \mathsf{pay} : y!\mathsf{ship}\} \\ Shipper(y) &\triangleq y?\mathsf{ship} \end{aligned}$$

- x enjoys progress $\Rightarrow$ y ...? *Shipper* ☹
- x fairly terminates $\Rightarrow$ y fairly terminates *Shipper* ☺

why fair termination?

(Fair) session termination is **expected**

- see meaning of "session"

Fair termination implies **progress**

- suppose $P \Rightarrow Q$ and Q has pending actions in session x
- then $Q \Rightarrow R$ where x is terminated (no pending actions on x)

Fair termination enables **compositional reasoning**

$$
\begin{aligned}
\mathit{Buyer}(x) &\triangleq \ldots\text{whatever}\ldots \\
\mathit{Seller}(x,y) &\triangleq x?\{\mathsf{add} : \mathit{Seller}\langle x,y\rangle, \mathsf{pay} : y!\mathsf{ship}\} \\
\mathit{Shipper}(y) &\triangleq y?\mathsf{ship}
\end{aligned}
$$

- x enjoys progress $\Rightarrow y \ldots ?$ *Shipper* ☹
- x fairly terminates $\Rightarrow y$ fairly terminates *Shipper* ☺

a type system for weak/fair termination

pitfalls

Duality does **not** entail fair termination

- use fairly terminating session types (end always reachable)

Subtyping does **not** preserve fair termination

- use **fair subtyping** [Padovani, 2013, 2016]

Fair subtyping is **unsound** if used "infinitely many times"

- use fair subtyping **carefully**

Some processes simply can**not** terminate

- make sure the effort required for termination is **finite**

outline

one seller, many buyers

The seller complies with **one** protocol

$$S = ?\mathsf{add}.S + ?\mathsf{pay}.\mathsf{end}$$

The buyer may comply with **many** different protocols

$S^\perp = T = !\mathsf{add}.T \oplus !\mathsf{pay}.\mathsf{end}$	**any number** of items
$T_1 = !\mathsf{add}.T$	**at least one** item
$T_{\mathsf{odd}} = !\mathsf{add}.(!\mathsf{add}.T_{\mathsf{odd}} \oplus !\mathsf{pay}.\mathsf{end})$	**odd number** of items
$\dots$	many more possibilities

Only T is the dual of S, but all should be "compatible" with S

subtyping for session types

see Gay and Hole [2005]

$?a \leqslant ?a + ?b$ covariant inputs

$!a \oplus !b \leqslant !a$ contravariant outputs

substitution principle for endpoints

- when $S \leqslant T$ an endpoint of type S can be safely used where an endpoint of type T is expected

substitution principle for endpoint users (aka processes)

- when $S \leqslant T$ a process behaving as T can be safely used where a process behaving as S is expected

expected versus **odd** buyer

$$
\begin{aligned}
T_{\text{odd}} &= !\text{add}.(!\text{add}.T_{\text{odd}} \oplus !\text{pay}.\text{end}) && \text{actual buyer} \\
T &= !\text{add}.T \oplus !\text{pay}.\text{end} && \text{expected buyer} \\
T^{\perp} = S &= ?\text{add}.S + ?\text{pay}.\text{end} && \text{seller}
\end{aligned}
$$

$$
\dfrac{\dfrac{\dfrac{}{x : T_{\text{odd}} \vdash \mathit{Buyer}_{\text{odd}}\langle x \rangle}}{x : T \vdash \mathit{Buyer}_{\text{odd}}\langle x \rangle}\; T \leqslant T_{\text{odd}} \qquad \dfrac{}{x : S \vdash \mathit{Seller}\langle x \rangle}}{\emptyset \vdash (x)(\mathit{Buyer}_{\text{odd}}\langle x \rangle \mid \mathit{Seller}\langle x \rangle)}
$$

Subtyping is key to reconcile expected and actual behavior

expected versus **compulsive** buyer

$$
\begin{aligned}
T_\infty &= !\mathsf{add}.T_\infty && \text{compulsive buyer} \\
T &= !\mathsf{add}.T \oplus !\mathsf{pay}.\mathsf{end} && \text{expected buyer} \\
T^\perp = S &= ?\mathsf{add}.S + ?\mathsf{pay}.\mathsf{end} && \text{seller}
\end{aligned}
$$

$$
\dfrac{\dfrac{\dfrac{}{x : T_\infty \vdash \mathit{Buyer}_\infty\langle x\rangle}}{x : T \vdash \mathit{Buyer}_\infty\langle x\rangle}\; T \leqslant T_\infty \qquad \dfrac{}{x : S \vdash \mathit{Seller}\langle x\rangle}}{\emptyset \vdash (x)(\mathit{Buyer}_\infty\langle x\rangle \mid \mathit{Seller}\langle x\rangle)}
$$

Subtyping does **not** preserve fair termination

outline

fair subtyping

liveness-preserving refinement of subtyping

$$\frac{}{\mathsf{end} \leqslant \mathsf{end}} \qquad \frac{S_i \leqslant T_i \qquad I \subseteq J}{?\{\mathsf{m}_i : S_i\}_{i \in I} \leqslant ?\{\mathsf{m}_i : T_i\}_{i \in J}}$$

$$\frac{S_i \leqslant T_i \qquad J \subseteq I}{!\{\mathsf{m}_i : S_i\}_{i \in I} \leqslant !\{\mathsf{m}_i : T_i\}_{i \in J}}$$

Intuition for $S \leqslant T$

- $\leqslant$ is **covariant** for inputs
- $\leqslant$ is **contravariant** for outputs
- **at most** n **outputs away** from the region of S and T where $\leqslant$ is **invariant** for outputs (**finite** output contravariance)

fair subtyping

liveness-preserving refinement of subtyping

$$\frac{}{\mathsf{end} \leqslant_n \mathsf{end}} \qquad \frac{S_i \leqslant_n T_i \qquad I \subseteq J}{?\{\mathsf{m}_i : S_i\}_{i\in I} \leqslant_n ?\{\mathsf{m}_i : T_i\}_{i\in J}}$$

$$\frac{S_i \leqslant_n T_i}{!\{\mathsf{m}_i : S_i\}_{i\in I} \leqslant_n !\{\mathsf{m}_i : T_i\}_{i\in I}} \qquad \frac{S_i \leqslant_{n_i} T_i \qquad J \subseteq I}{!\{\mathsf{m}_i : S_i\}_{i\in I} \leqslant_{1+n_k} !\{\mathsf{m}_i : T_i\}_{i\in J}}$$

Intuition for $S \leqslant_n T$

- $\leqslant$ is **covariant** for inputs
- $\leqslant$ is **contravariant** for outputs
- **at most** n **outputs away** from the region of S and T where $\leqslant$ is **invariant** for outputs (**finite** output contravariance)

example of fair subtyping

buyer that purchases **at least one** item

$$T = \,!\mathsf{add}.T \oplus \,!\mathsf{pay}.\mathsf{end} \qquad T_1 = \,!\mathsf{add}.T$$

$$\dfrac{\dfrac{\vdots}{T \leqslant_0 T}}{T \leqslant_1 T_1}$$

- reflexivity $\Rightarrow S \leqslant_0 S$ for every S
- one application of contravariant rule for outputs

example of fair subtyping

buyer that purchases **an odd number** of items

$$T = {!}\mathsf{add}.T \oplus {!}\mathsf{pay}.\mathsf{end} \qquad T_{\mathsf{odd}} = {!}\mathsf{add}.({!}\mathsf{add}.T_{\mathsf{odd}} \oplus {!}\mathsf{pay}.\mathsf{end})$$

$$\dfrac{\dfrac{\dfrac{\vdots}{T \leqslant_2 T_{\mathsf{odd}}} \qquad \dfrac{}{\mathsf{end} \leqslant_0 \mathsf{end}}}{T \leqslant_1 {!}\mathsf{add}.T_{\mathsf{odd}} \oplus {!}\mathsf{pay}.\mathsf{end}}}{T \leqslant_2 T_{\mathsf{odd}}}$$

- two applications of contravariant rule for outputs

example of **unfair** subtyping

buyer that adds **infinitely many** items to the shopping cart but never pays

$$T = !\mathsf{add}.T \oplus !\mathsf{pay}.\mathsf{end} \qquad\qquad T_\infty = !\mathsf{add}.T_\infty$$

$$\cfrac{\cfrac{\cfrac{???}{T \leqslant_0 T_\infty}}{\cfrac{\vdots}{T \leqslant_{n-1} T_\infty}}}{T \leqslant_n T_\infty}$$

- infinitely many applications of contravariant rule for outputs
- T is **not** a fair subtype of T_∞

compulsive shopping is not allowed...

$$\dfrac{\dfrac{\dfrac{}{x : T_\infty \vdash \mathit{Buyer}_\infty\langle x\rangle}}{x : T \vdash \mathit{Buyer}_\infty\langle x\rangle}\ \cancel{T \leqslant T_\infty} \qquad \dfrac{}{x : S \vdash \mathit{Seller}\langle x\rangle}}{\emptyset \vdash (x)(\mathit{Buyer}_\infty\langle x\rangle \mid \mathit{Seller}\langle x\rangle)}$$

compulsive shopping is not allowed…or is it?

there is a **different way** of typing the compulsive buyer ☹

$$\begin{gathered} Buyer_\infty(x) \triangleq x!\mathsf{add}.Buyer_\infty\langle x\rangle \\ T = !\mathsf{add}.T \oplus !\mathsf{pay}.\mathsf{end} \\ T_1 = !\mathsf{add}.T \end{gathered}$$

$$\dfrac{\dfrac{\overline{x : T \vdash Buyer_\infty\langle x\rangle}}{x : T_1 \vdash x!\mathsf{add}.Buyer_\infty\langle x\rangle}}{x : T \vdash x!\mathsf{add}.Buyer_\infty\langle x\rangle}\; T \leqslant_1 T_1$$

Poset of session types ordered by fair subtyping is not ω-complete

- "infinitely many" usages of fair subtyping ($T \leqslant_1 T_1$) may have the same overall effect of unfair subtyping ($T \leqslant T_\infty$)

$$T \leqslant_1 !\mathsf{add}.T \leqslant_2 !\mathsf{add}.!\mathsf{add}.T \leqslant_3 \cdots \not\leqslant T_\infty$$

- well-typed processes should only be allowed to perform a **finite number of casts** before they terminate

cast boundedness

accounting for the aggregate effect of fair subtyping in judgments

$$\frac{\Gamma, x : T \vdash_m P}{\Gamma, x : S \vdash_{m+n} P} \; S \leqslant_n T$$

The compulsive buyer is ill typed

$$\frac{\dfrac{\dfrac{}{x : T \vdash_n \mathit{Buyer}_\infty\langle x\rangle}}{x : T_1 \vdash_n x!\mathsf{add}.\mathit{Buyer}_\infty\langle x\rangle}}{x : R \vdash_{n+1} x!\mathsf{add}.\mathit{Buyer}_\infty\langle x\rangle} \; T \leqslant_1 T_1$$

outline

some cast-bounded processes don't terminate

Diverging processes

- $A \triangleq A \oplus A$
- make sure that every well-typed process has a finite branch **that creates finitely many sessions**

Higher-order sessions

- with **co/contra variance** of higher-order session types, a **single cast** may have the effect of **infinitely many casts**
- fair subtyping for higher-order session types is **invariant**

soundness

If P is **well typed**, then

1. $P \rightarrow Q$ implies Q well typed (subject reduction)
2. P is **weakly terminating**
3. P is **fairly terminating**
   consequence of (1), (2) and property of fair termination

If P is **also closed**, then

4. $P \nrightarrow$ implies $P = \mathsf{done}$ (deadlock freedom)
5. $P \Rightarrow Q$ implies $Q \Rightarrow \mathsf{done}$ (progress/liveness)
   consequence of (1), (2) and (4)

example: buyer-seller-shipper

$$\emptyset \vdash (x)(\mathit{Buyer}\langle x\rangle \mid (y)(\mathit{Seller}\langle x,y\rangle \mid \mathit{Shipper}\langle y\rangle))$$

$$\begin{aligned}
\mathit{Buyer}(x) &\triangleq \ldots\text{any well-typed buyer}\ldots \\
\mathit{Seller}(x,y) &\triangleq x?\{\mathsf{add} : \mathit{Seller}\langle x,y\rangle, \mathsf{pay} : y!\mathsf{ship}\} \\
\mathit{Shipper}(y) &\triangleq y?\mathsf{ship}
\end{aligned}$$

Note

- *Shipper* makes progress only after *Buyer* has paid
- the infinite run in which *Buyer* keeps adding items is unfair

example: parallel divide-and-conquer

Fibonacci, quick sort, merge sort, fast Fourier transform, etc.

$$\emptyset \vdash (x)(x!\mathsf{req}.x?\mathsf{res} \mid \mathit{Worker}\langle x\rangle)$$

$$\mathit{Master}(x,y,z) \triangleq y!\mathsf{req}.z!\mathsf{req}.y?\mathsf{res}.z?\mathsf{res}.x!\mathsf{res}$$
$$\mathit{Worker}(x) \triangleq x!\mathsf{res} \oplus (y,z)(\mathit{Master}\langle x,y,z\rangle \mid \mathit{Worker}\langle y\rangle \mid \mathit{Worker}\langle z\rangle)$$

Note

- creates an **unbounded** number of **nested** sessions
- the run in which *Worker* keeps creating new sessions is unfair

outline

summary

A compositional static analysis ensuring fair termination

- well-typed sessions (fairly) terminate
- a well-typed composition of fairly-terminating sessions results in a fairly-terminating program

Want more?

- many simplifications (and some novelties) in this talk
- see Ciccone and Padovani [2022] for details
  (higher-order sessions, proofs, type checking algorithm, ...)
- **ranked characterization** of fair subtyping is new (unpublished)

further work

FairCheck

- Haskell implementation of the type checker
- available on GitHub (link also on my home page)

Application to multiparty sessions (unpublished)

- type system scales easily to the multiparty setting
- see the **live** predicate of Scalas and Yoshida [2019]

further work

FairCheck

- Haskell implementation of the type checker
- available on GitHub (link also on my home page)

Application to multiparty sessions (unpublished)

- type system scales easily to the multiparty setting
- see the **live** predicate of Scalas and Yoshida [2019]

thank you!

references

Luca Ciccone and Luca Padovani. Fair Termination of Binary Sessions. *Proceedings of the ACM on Programming Languages*, 6:5:1–5:30, 2022.

Simon J. Gay and Malcolm Hole. Subtyping for session types in the pi calculus. *Acta Informatica*, 42(2-3):191–225, 2005.

Luca Padovani. Fair subtyping for open session types. In Fedor V. Fomin, Rusins Freivalds, Marta Z. Kwiatkowska, and David Peleg, editors, *Automata, Languages, and Programming - 40th International Colloquium, ICALP 2013, Riga, Latvia, July 8-12, 2013, Proceedings, Part II*, volume 7966 of *Lecture Notes in Computer Science*, pages 373–384. Springer, 2013.

Luca Padovani. Fair subtyping for multi-party session types. *Math. Struct. Comput. Sci.*, 26(3):424–464, 2016.

references (cont.)

Jean-Pierre Queille and Joseph Sifakis. Fairness and related properties in transition systems - A temporal logic to deal with fairness. *Acta Informatica*, 19:195–220, 1983. URL `https://doi.org/10.1007/BF00265555`.

Alceste Scalas and Nobuko Yoshida. Less is more: multiparty session types revisited. *Proc. ACM Program. Lang.*, 3(POPL):30:1–30:29, 2019.

Rob van Glabbeek and Peter Höfner. Progress, justness, and fairness. *ACM Comput. Surv.*, 52(4):69:1–69:38, 2019.