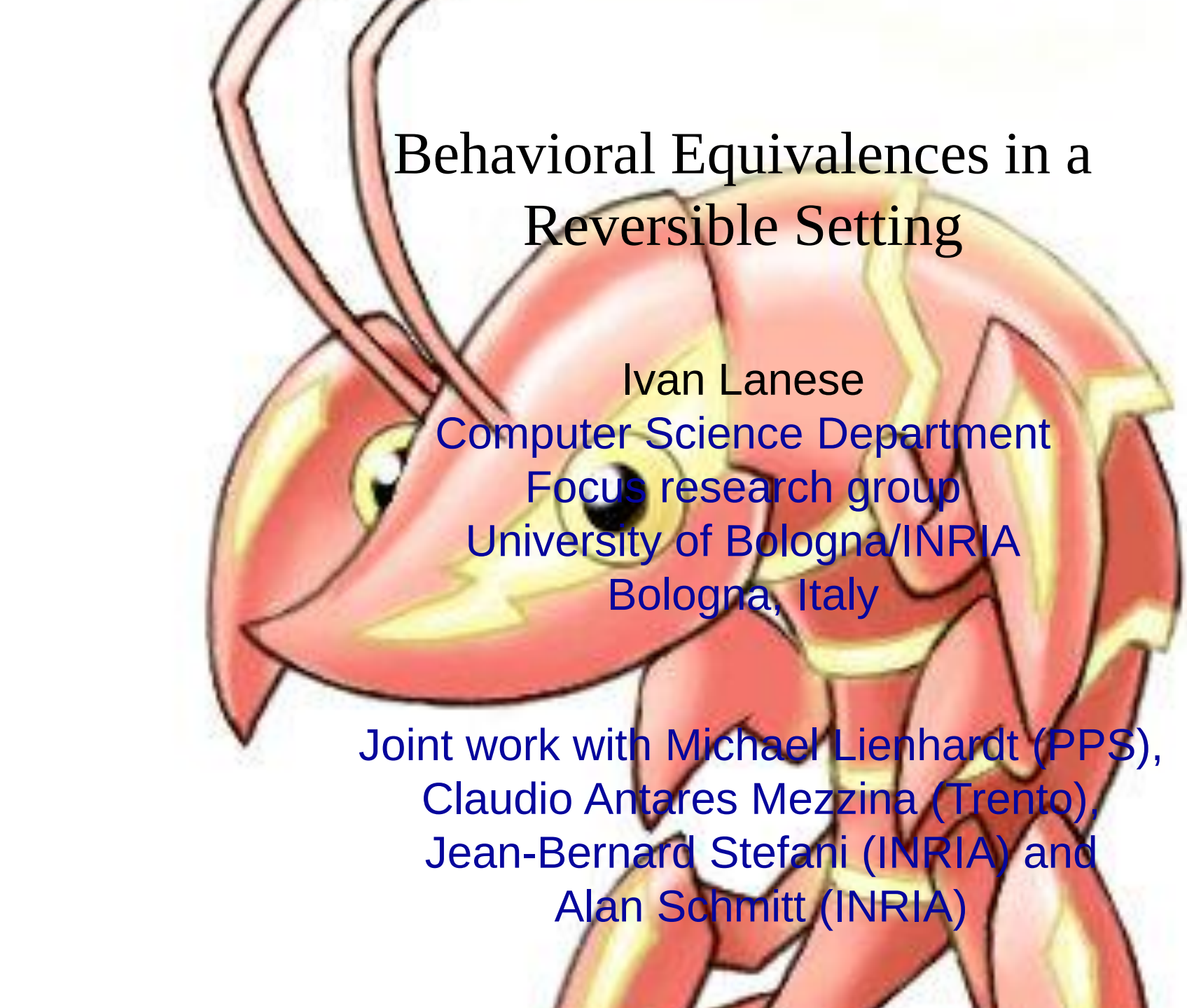


Behavioral Equivalences in a Reversible Setting

Ivan Lanese
Computer Science Department
Focus research group
University of Bologna/INRIA
Bologna, Italy

Joint work with Michael Lienhardt (PPS),
Claudio Antares Mezzina (Trento),
Jean-Bernard Stefani (INRIA) and
Alan Schmitt (INRIA)

Roadmap

- Uncontrolled reversibility
- Barbed congruences in the uncontrolled setting
- Controlled reversibility
- Alternatives
- Conclusions

Roadmap

- Uncontrolled reversibility
- Barbed congruences in the uncontrolled setting
- Controlled reversibility
- Alternatives
- Conclusions

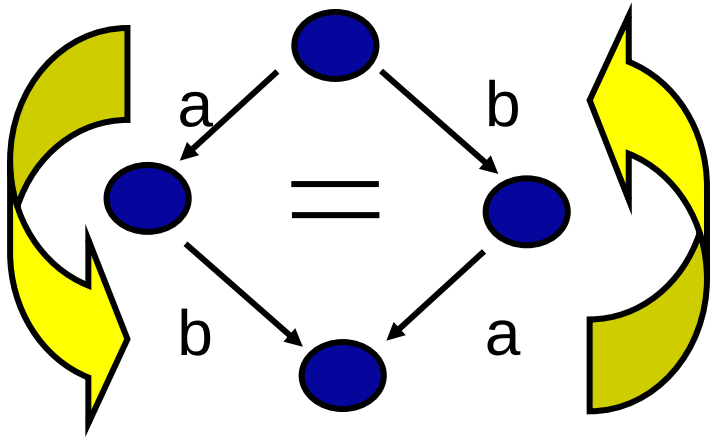
What is reversibility?

The possibility of executing a (concurrent) computation both in the standard, forward direction, and in the backward direction, going back to a past state

- In some areas systems are naturally reversible: biology, quantum computing, …
- In concurrent systems reversibility allows for recoverability
  - In case of error I go back to a past state which is safe
  - We want to use reversibility as a general framework for programming reliable applications

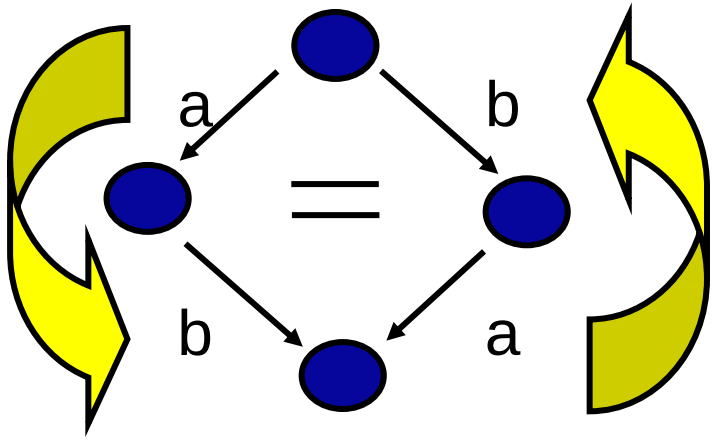
Concurrent reversibility

- Reversibility in a sequential setting: “recursively undo the last action”
- In a concurrent setting it is not clear which is the last action
- Independent threads are reversed independently
- Causal dependencies should be respected
  - First reverse the consequences, then the causes



Causal consistent reversibility

- Reversibility in a sequential setting: “recursively undo the last action”
- In a concurrent setting it is not clear which is the last action
- Independent threads are reversed independently
- Causal dependencies should be respected
  - First reverse the consequences, then the causes



How do I define a causal consistent calculus?

- A few approaches in the literature
  - RCCS by Danos and Krivine [CONCUR 2004]
  - CCSk by Phillips and Ulidovski [FoSSaCS 2006]
  - Rhopi by Lanese et al. [CONCUR 2010]
  - Reversible structures by Laneve and Cardelli [CMSB 2011]
  - Reversible µOz by Lanese et al. [FMOODS/FORTE 2012]
- Different technical solutions, same idea
- The term contains information on past actions and on causal dependencies
  - Computation should cause no loss of information
  - Substitutions normally causes loss of information

This is uncontrolled reversibility

- Two kinds of transitions, forward and backward
  - No hint on when to use one and when to use the other
  - Nondeterministic choice on the direction
- Useful to understand the possible behaviors
- More useful as model than as programming language

Roadmap

- Uncontrolled reversibility
- Barbed congruences in the uncontrolled setting
- Controlled reversibility
- Alternatives
- Conclusions

Behavioral equivalences

- Not all the reversible calculi above have an LTS – some have just a reduction semantics
- Barbed congruence seems a reasonable candidate
- We adapt the standard definitions

Barbed congruence and equivalence

- Barbs: a configuration M has a barb at a ($M \downarrow a$) if it contains a message on a
  - We do not observe history and causality information
- A symmetric relation R is a barbed equivalence if $(M, N) \in R$ implies
  - $M \rightarrow M'$ implies $N \rightarrow N'$ and $(M', N') \in R$
    - » No distinction between backward and forward reductions
  - $M \downarrow a$ implies $N \downarrow a$
- Classic extension to the weak case
- A barbed congruence is the largest congruence included in barbed equivalence

Barbed congruence and barbed equivalence

- Weak barbed congruence is not very discriminating
- Each configuration is weak barbed congruent to all its descendants and predecessors
  - In some sense an observational characterization of reversibility
- Weak barbed equivalence is even less discriminating
- Each configuration is weak barbed equivalent to one with all the barbs visible and no reductions

Back and forth barbed equivalence

- We distinguish forward reductions $\twoheadrightarrow$ from backward reductions $\rightsquigarrow\!\!\!\!\twoheadrightarrow$
- We can define back and forth barbed equivalence
  [De Nicola et al. CONCUR 1990]
  [Phillips and Ulidowski SOS 2007]
- A symmetric relation R is a back and forth barbed equivalence if $(M, N) \in R$ implies
  - $M \twoheadrightarrow M'$ implies $N \twoheadrightarrow N'$ and $(M', N') \in R$
  - $M \rightsquigarrow\!\!\!\!\twoheadrightarrow M'$ implies $N \rightsquigarrow\!\!\!\!\twoheadrightarrow N'$ and $(M', N') \in R$
  - $M \downarrow a$ implies $N \downarrow a$

Back and forth barbed congruence

- More expressive than standard equivalences
- Distinguishes $a|b$ from $a.b + b.a$
- The former can do a, then b, then undo a, the latter cannot
- Back and forth bisimulation corresponds to hereditary history-preserving bisimulation (with no auto-concurrency and no auto-causation) [Phillips and Ulidowski SOS 2007]

Weak back and forth barbed congruence?

- Not yet studied as far as I know
- A few possible design choices
  - Which kind of τ steps do I allow in reductions?
  - And to reach weak barbs?
- Which choices give an equivalence that matches the intuition?

Roadmap

- Uncontrolled reversibility
- Barbed congruences in the uncontrolled setting
- Controlled reversibility
- Alternatives
- Conclusions

Power is nothing without control

- Programs based on uncontrolled reversibility are not very useful
  - They always diverge
  - No way to make a result persistent
- We want to go back only when needed
  - In particular, in case of errors
- We want to specify how far back to go

Reversibility control

- Different approaches in the literature
  - Irreversible actions by Danos and Krivine [CONCUR 2005]
  - Energy parameters by Bacci et al [CALCO 2011]
  - Rollback operator by Lanese et al [CONCUR 2011]
  - Monitors by Phillips et al [RC 2012]

Rollback operator idea

- Normal computation goes forward
- There is an explicit primitive, roll γ, to trigger a rollback
- γ refers to a specific action done in the past
  - We specify which action to undo
  - As a result we undo all the actions depending on it
  - Independent actions are not undone

Is rollback enough?

- Rollback allows to control reversibility
- In case of rollback
  - We go back to a past consistent state
  - And we execute forward again from it
  - We may take the same path, obtaining the same error again
  - Good for transient errors, not for permanent ones
- Each program with a (reachable) rollback is divergent
- We want to remember the past tries and learn from them

Roadmap

- Uncontrolled reversibility
- Barbed congruences in the uncontrolled setting
- Controlled reversibility
- Alternatives
- Conclusions

Actions with alternatives

- Instead of actions A we use actions with alternatives
  - A%0 : try A then stop trying
  - A%B%0 : try A then B then stop trying
- If the action with alternative is the target of the rollback, it is replaced by its alternative
- Using alternatives the programmer may now avoid looping

Alternatives in the literature?

- We proposed asynchronous HOπ with rollback and alternatives [ESOP 2013]
  - Only messages have alternatives
  - Only messages and 0 may be used as alternatives
- No other calculi with alternatives in the literature
- Some forms of control of reversibility allow anyway to avoid divergence

Are alternatives useful?

- Can we programme interesting applications exploiting rollback and alternatives?
- Can we recover/improve recoverability patterns from the literature?
- And invent new ones?

Messages with alternatives are robust

- We can encode different idioms:
  - General alternatives: not only messages
  - Finite retry: try n times
  - Endless retry: try forever
  - Triggers with alternatives: we attach alternatives to triggers instead of to messages

What can we model?

- Interesting applications:
  - State space exploration with backtracking: 8 queens problem
  - Error handling scenario: Automotive case study from Sensoria project
- Can we recover/improve existing techniques?
  - Software transactional memory model from Acciai et al. [ESOP 2007]
  - Interacting transactions from Hennessy et al. [CONCUR 2010]

Which equivalence?

- Behavioral equivalences useful for proving correctness of our encodings
- We used weak barbed congruence
- More discriminating with control and alternatives
  - Not all actions can be undone
  - Alternatives change the barbs
- Allows for a context lemma
  - Only parallel contexts and substitutions need to be considered
- More discriminating equivalences should be meaningful
  - The same as for uncontrolled reversibility?

Roadmap

- Uncontrolled reversibility
- Barbed congruences in the uncontrolled setting
- Controlled reversibility
- Alternatives
- Conclusions

Summary

- Causal consistent reversible calculi
- Mechanisms for controlling reversibility
- Alternatives for programming what to do after rollback

- Strong back and forth barbed congruence for the uncontrolled setting
- Weak barbed congruence in a setting with control and alternatives

Future work

- Many possible research directions
- Which LTS for reversible calculi?
  - LTS for reversible π [Krivine et al. LICS 2013]
  - A complex LTS for controlled reversibility
- Many open issues for behavioral equivalences
  - Which definition can be used for weak back and forth barbed congruence in the uncontrolled setting?
  - Is the same equivalence needed/reasonable with control and alternatives?
- Consider other languages/constructs
  - Klaim, object-oriented languages,...
- Reversibility seems useful for debugging

Finally