From State- to Delta-based Bidirectional Model Transformations: the Symmetric Case

Zinovy Diskin
Yingfei Xiong
Krzysztof Czarnecki

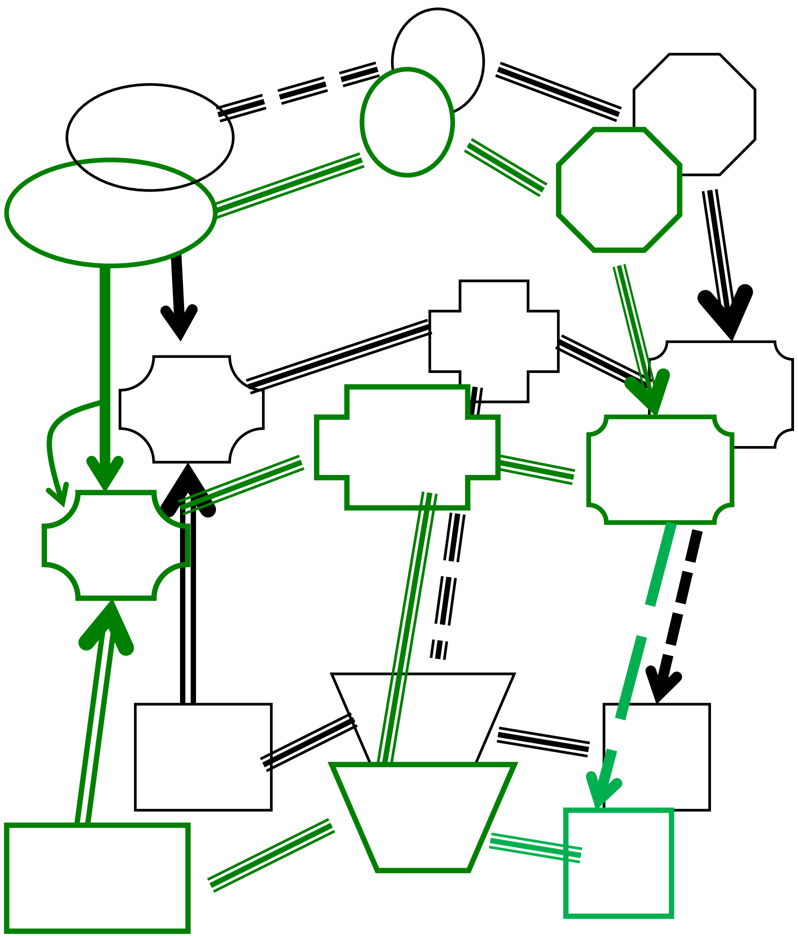
Waterloo

Hartmut Ehrig
Frank Hermann
Fernando Orejas

Berlin & Barcelona

Introduction: Model Synchronization

Problem is hard



Space of models

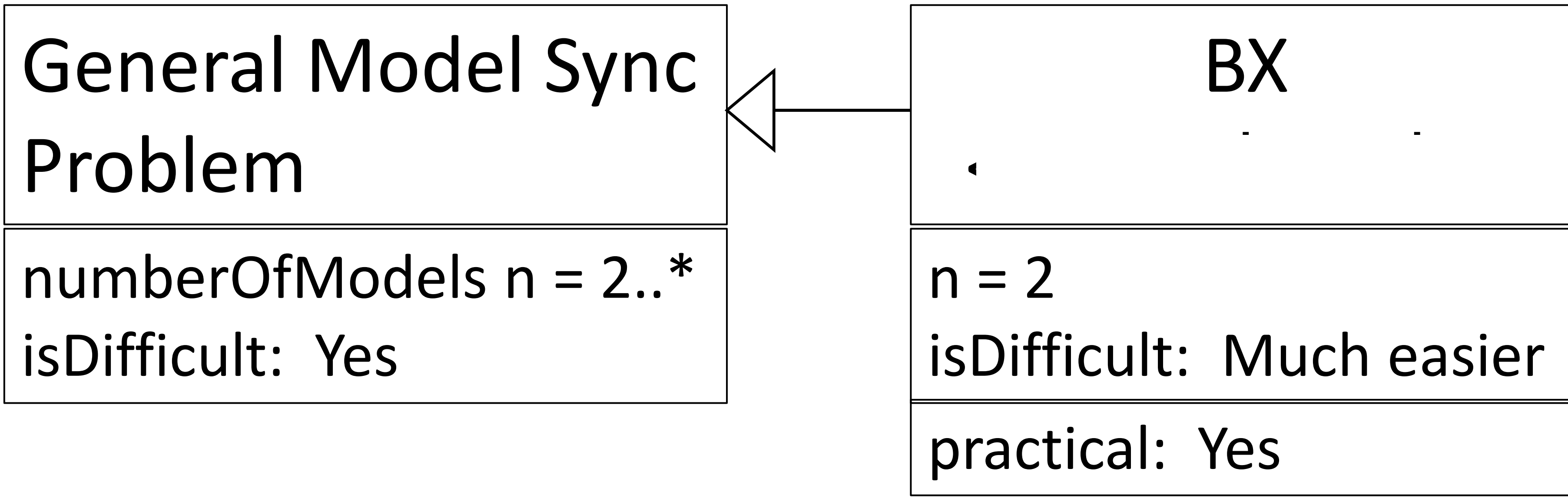
- ❖ Models are complex heterog. struct.
- ❖ Relationships are
  - also complex & heterog.
  - often implicit!
  - form non-trivial networks
- ❖ Changes

Tools are needed!

Introduction: Building model sync tools requires…

- Understanding semantics of sync procedures
- Explaining it to users concisely and clearly (P. Stevens)
- … !
- A theoretical framework as foundation

Introduction: From the **general** problem to **BX (**bidirectional model transformations)



Introduction: State-based BX

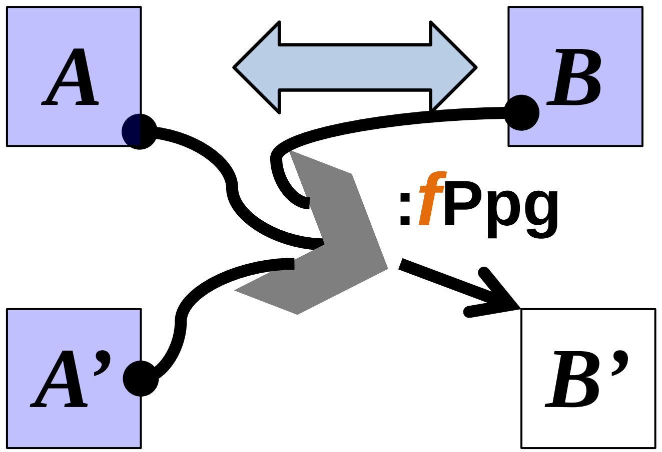
- *Triple Graph Grammars (TGGs)*
- *PL community* (the Harmony Group, B. Pierce et al, POPL'05-10, FP,…)*:*
  – A "product line" of algebraic structures called *lenses*
  – *Boomerang*: a language for string-based data
- *Models community:* Symmetric BX to explain semantics of QVT (P. Stevens)

Introduction: The message of the paper

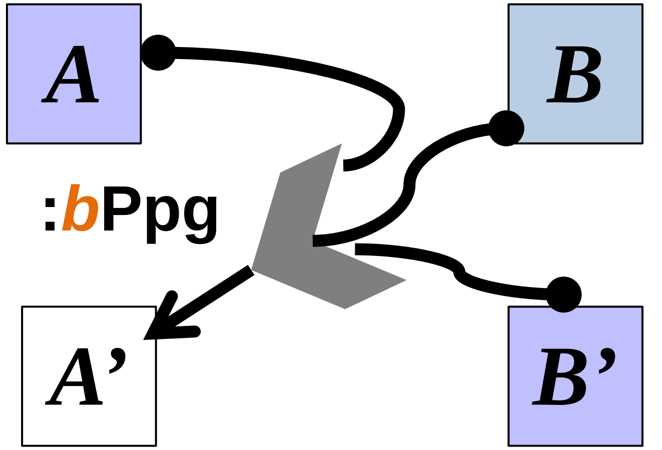
- The **bad** news: state-based BX do not work well for MDE and lead to several essential problems in practice and theory
- The **good** news: the problems can be fixed by using delta-based BX.
- Even a **better** news: theory of delta-based BX is equally simple, and in some aspects is even simpler than state-based

Background: State-based BX

- Two interrelated models
- Relations are implicit
- Propagation is state-based...



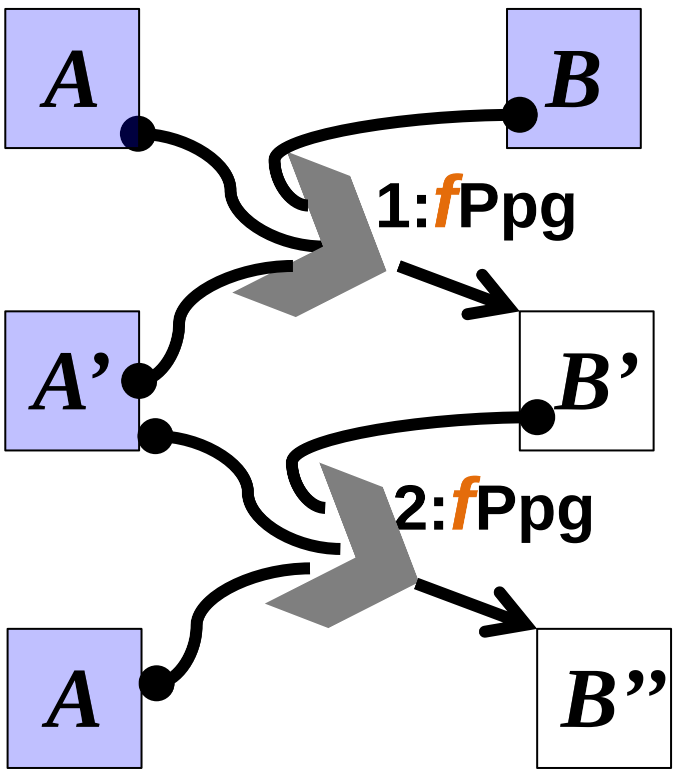
- and bidirectional...



But it's not the end of the story!

Background: fPpg and bPpg are not independent…

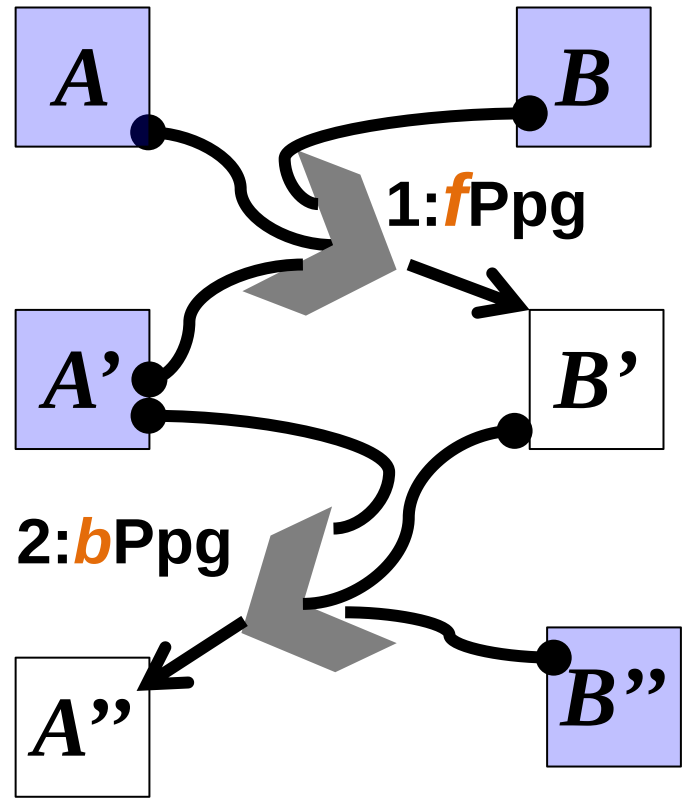
(i) … on undoing updates



B and B'' are to be related by a sort of ***undoability*** law

B''= B [Stevens'07]

(ii) …between themselves



A and A'' are to be related by a sort of ***invertibility*** law

A''= A [Diskin'08]

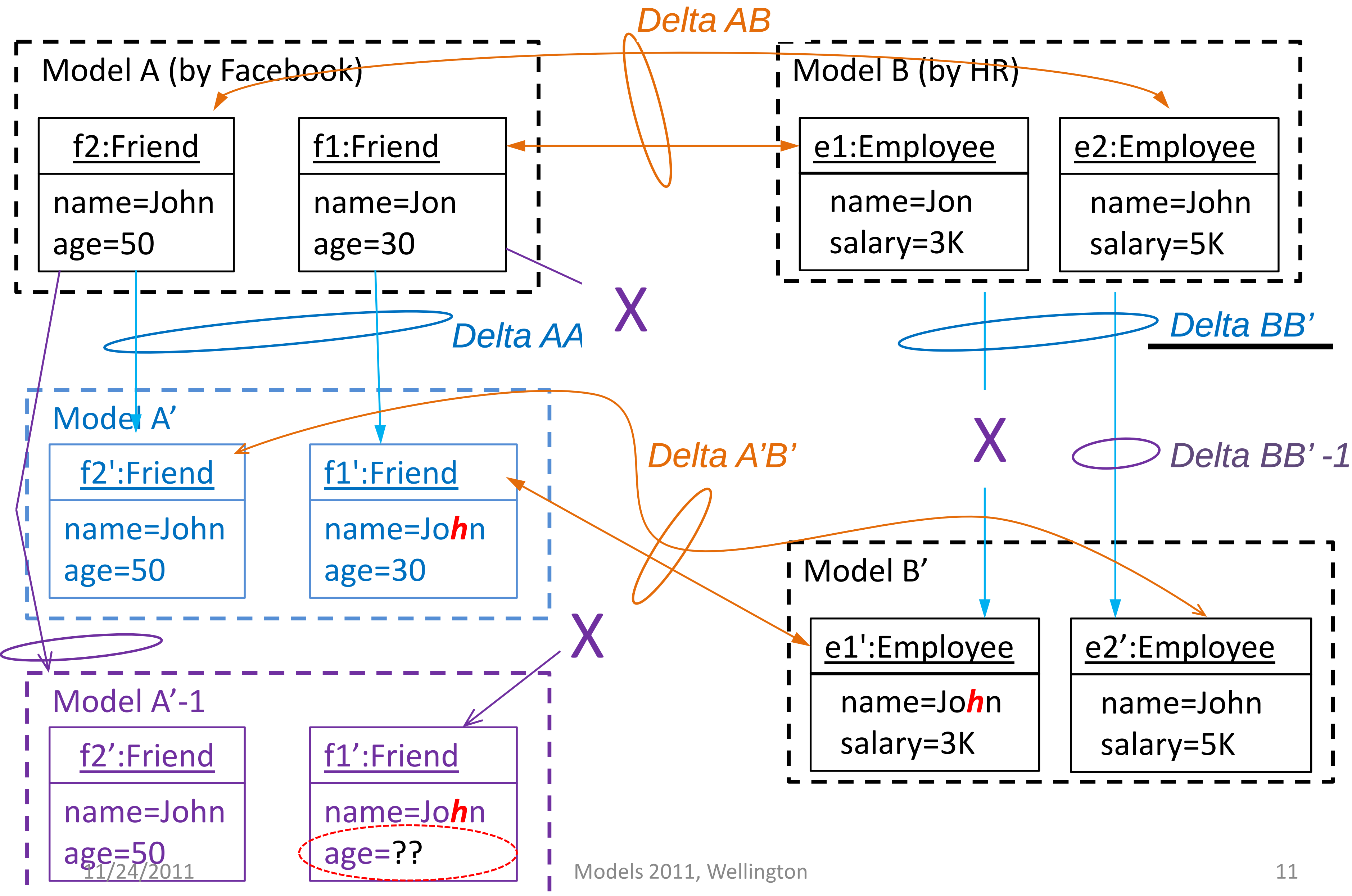
Background: Proper invertibility/undoability laws

- **Serious** problem because
  - Simple equality is **too constraining**
  - But without invert./undo., the behavior of propagating procedures is **unconstrained** at all.
- **Non-trivial** problem,
  - E.g., a failed attempt by Hoffman et al, POPL'10,
- The 2^{nd} goal of the paper is to find proper invertibility and undoability laws

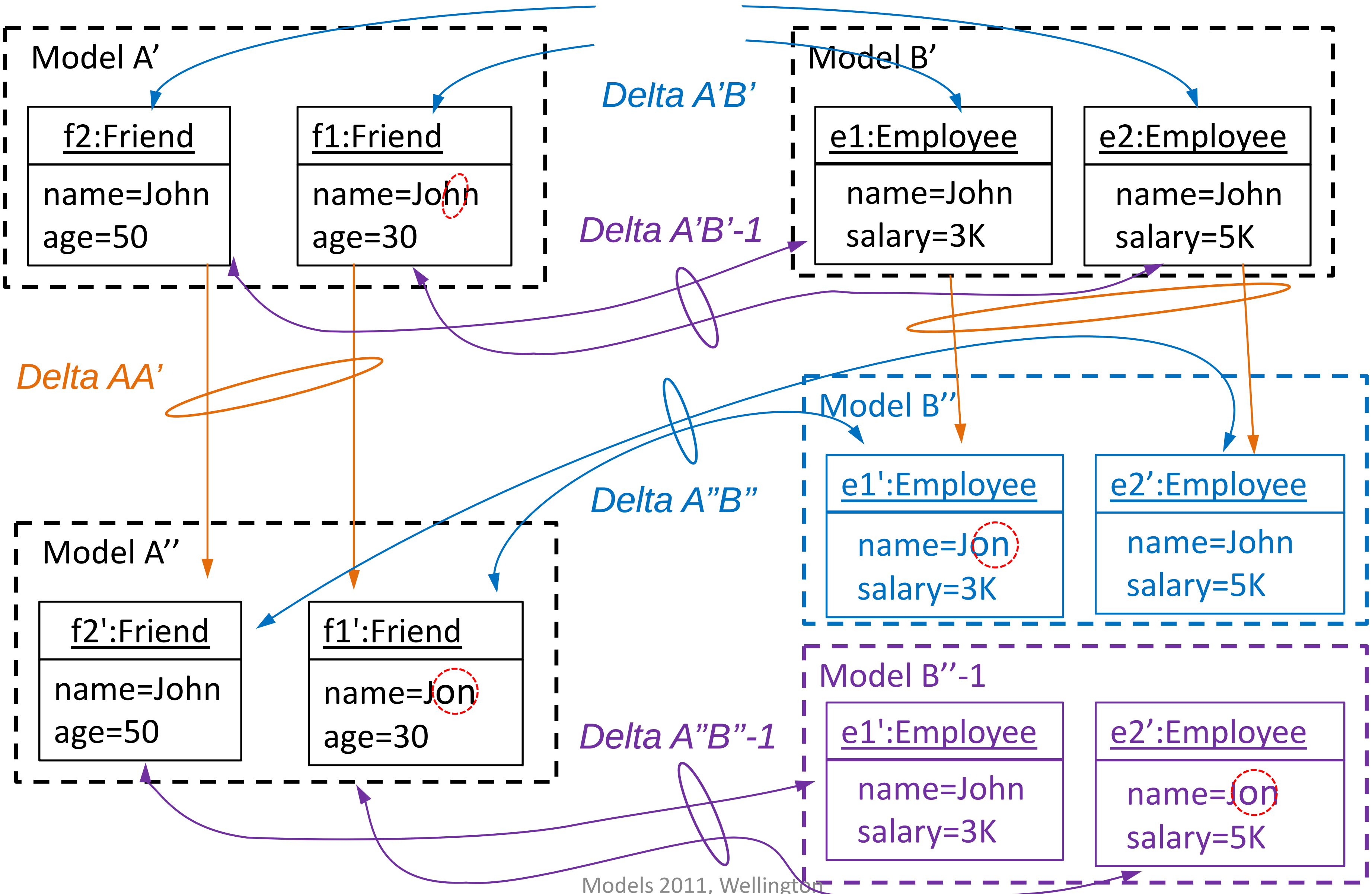
Content

- A simple example to explain the issues of symmetric BX. Deltas do matter!
- Problems of state-based BX
- Algebra for deltas (very sketchy)
- <u>Discussion of tool architectures</u>
- Summary

Example: John vs. Jon



Example cont'd: Jon returns!



Lessons learned, I

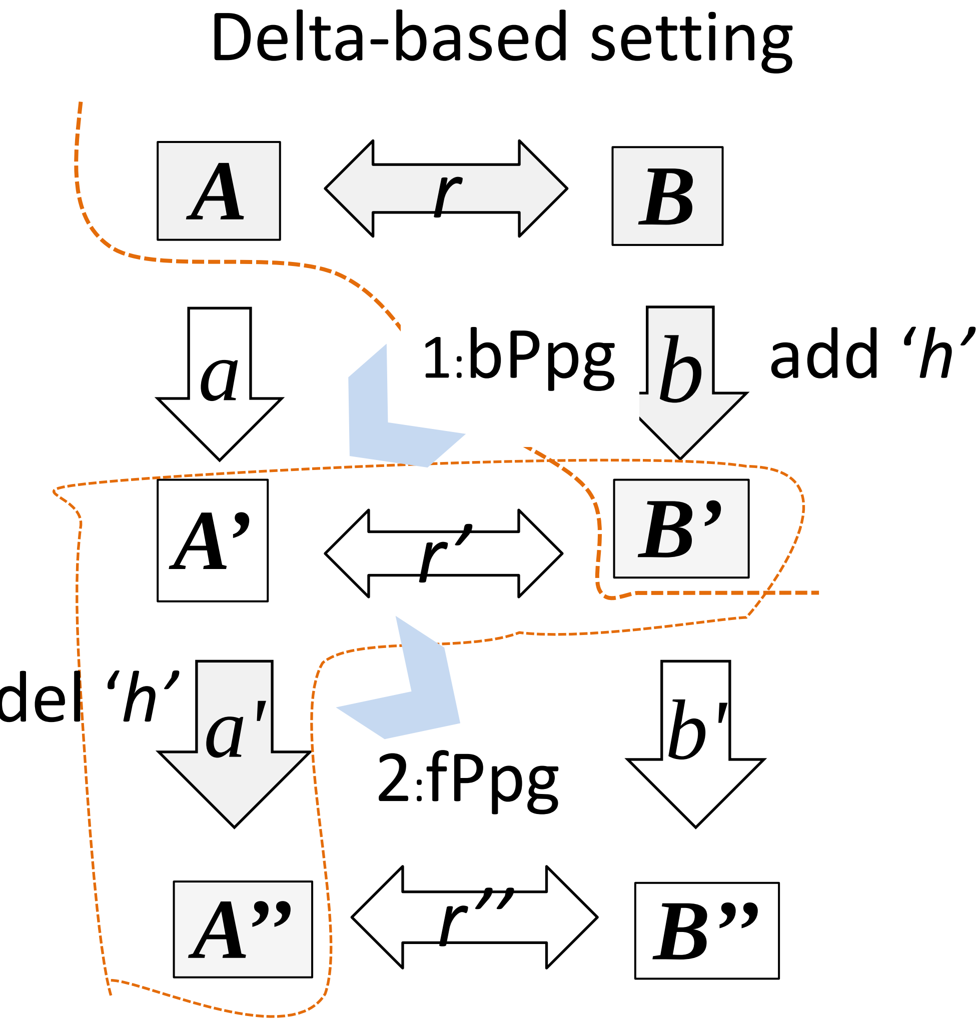
- Deltas do matter
- Delta composition matters too
- Delta reuse is important: ignoring it may lead to erroneous propagation

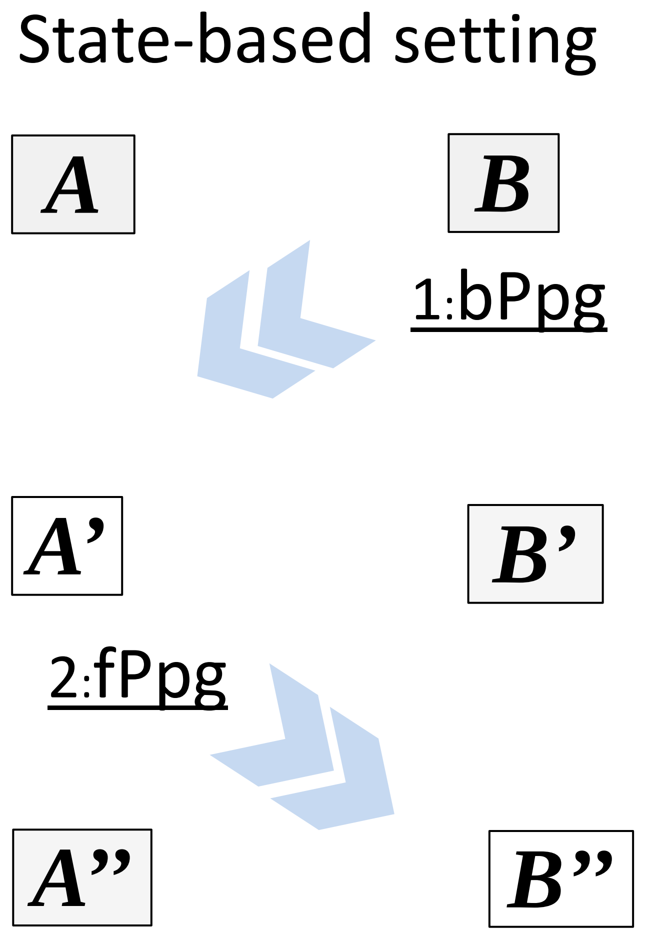
We need a mathematical framework with explicit operations on deltas!

Content

- Problems of state-based BX
- Algebra for deltas (very sketchy)
- Discussion of tool architectures
- <u>Discussion of mathematical modeling</u>
- Summary

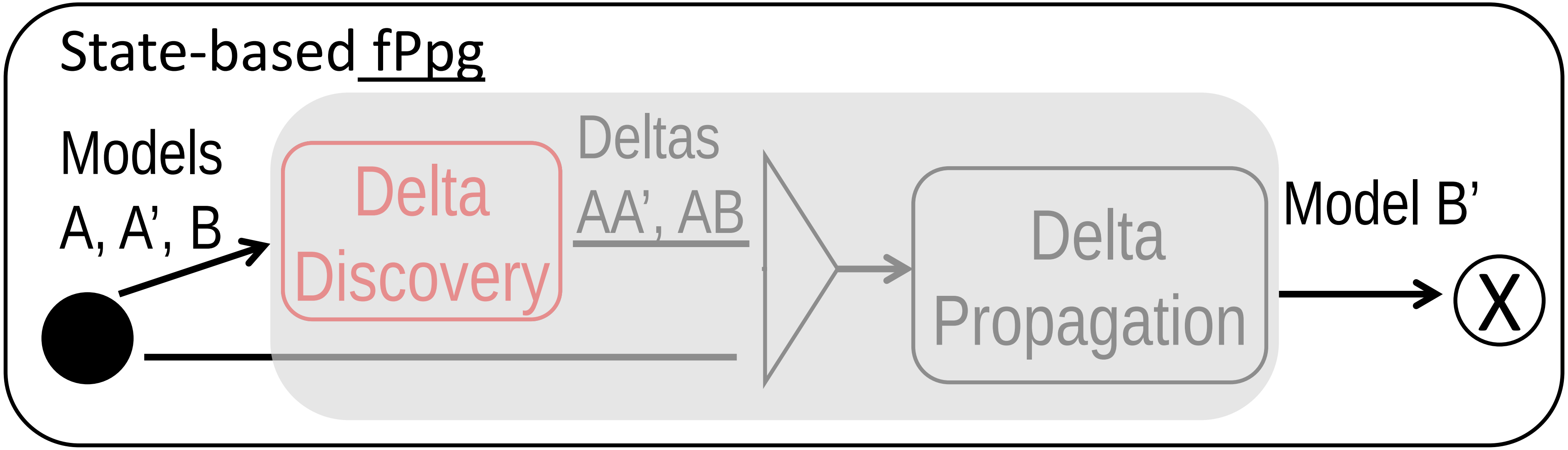
Jon vs. John, abstractly





Looks simpler but ....

Problems of state-based Ppg, 1

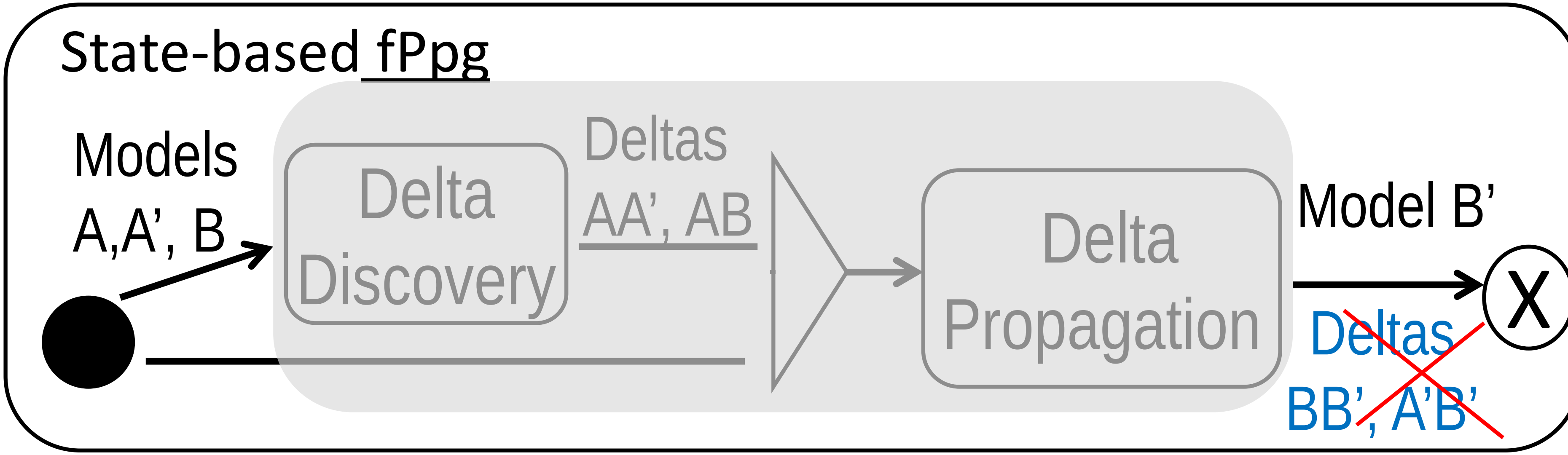


1.1 Semantics of DD is complex, hence semantics of fPpg is complex too

1.2 The user cannot control result of DD

~~Separation of concerns~~ => Bad cohesion

Problems of state-based Ppg, 2

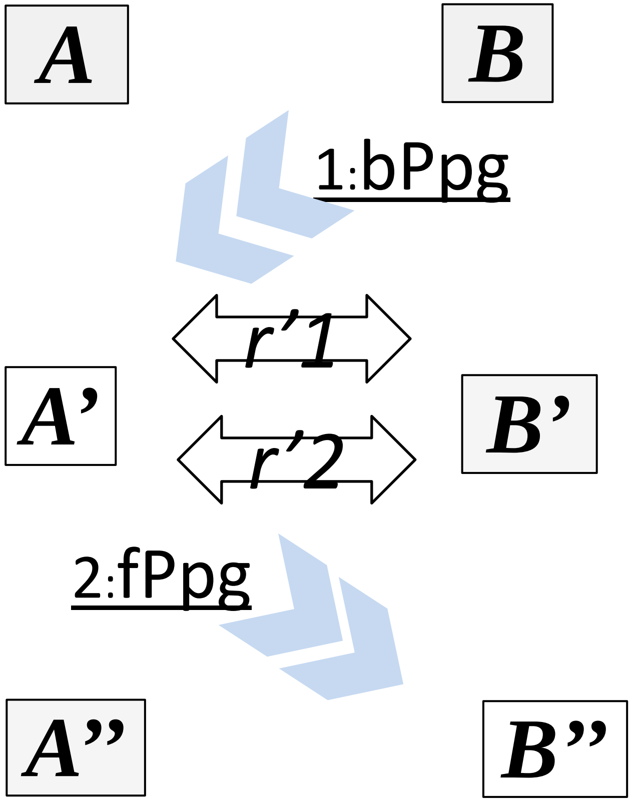


New deltas are not recorded and hence cannot be reused:

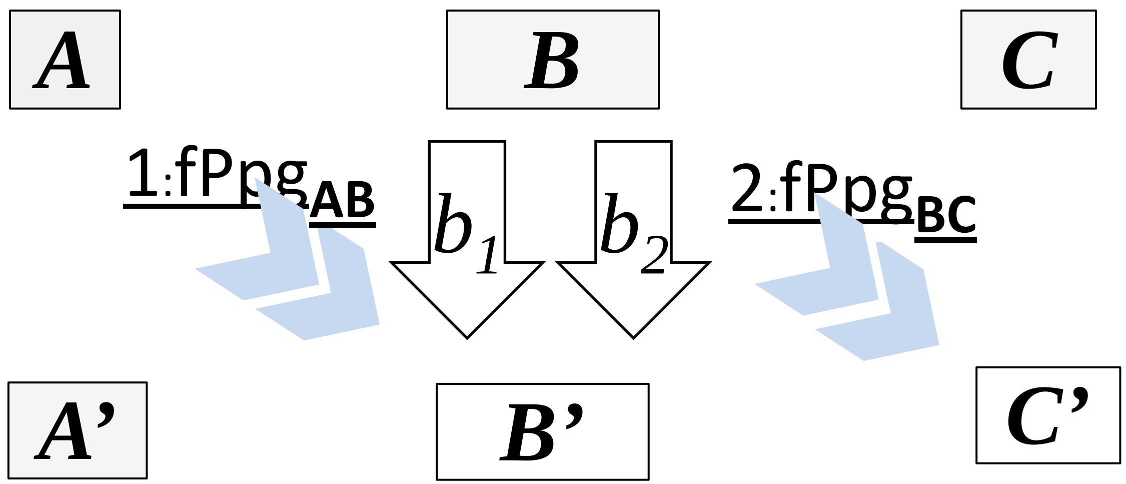
2.1 Low efficiency (DD is an expensive operation)

2.2 Erroneous DP-composition

State-based BX: erroneous vertical composition of Ppgs



State-based BX: erroneous horizontal composition of Ppgs



Lessons learned, II

- State-based frameworks has two major flaws:
  - they merge rather than separate two quite different concerns;
  - they break continuity of delta propagation.
- Simplicity of state-based frameworks is deceiving

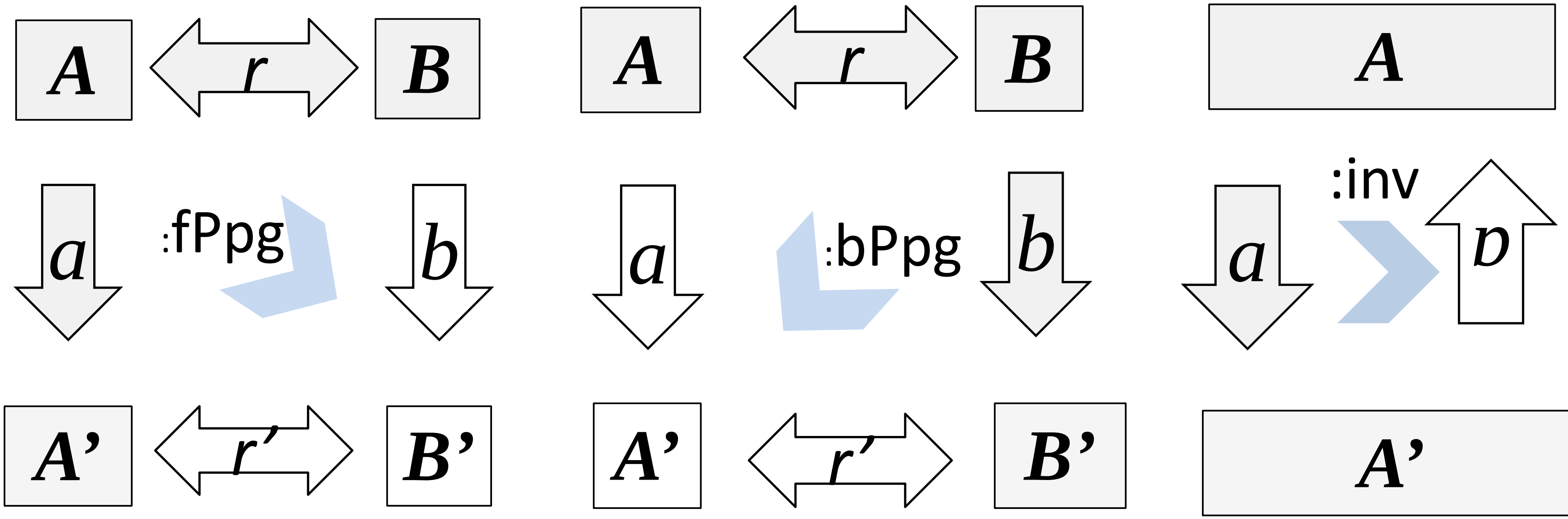
We need an *algebraic* framework operating deltas explicitly!

What is algebra?

An algebra is defined by

- A set of of *carrier* sets (*sorts*)
  – In our case, five sorts: $\mathbb{A}$, $\mathbb{B}$, $\Delta_{\mathbb{A}}$, $\Delta_{\mathbb{B}}$, $\Delta_{\mathbb{AB}}$
- A set of operations over these sets
  -- three ones: fPpg, bPpg, update inversion,
- A set of equational laws:
  – three pairs of laws: Identity propagation, undoability, invertibility (round-tripping)

Delta Lenses: Operations



Forward update propagation

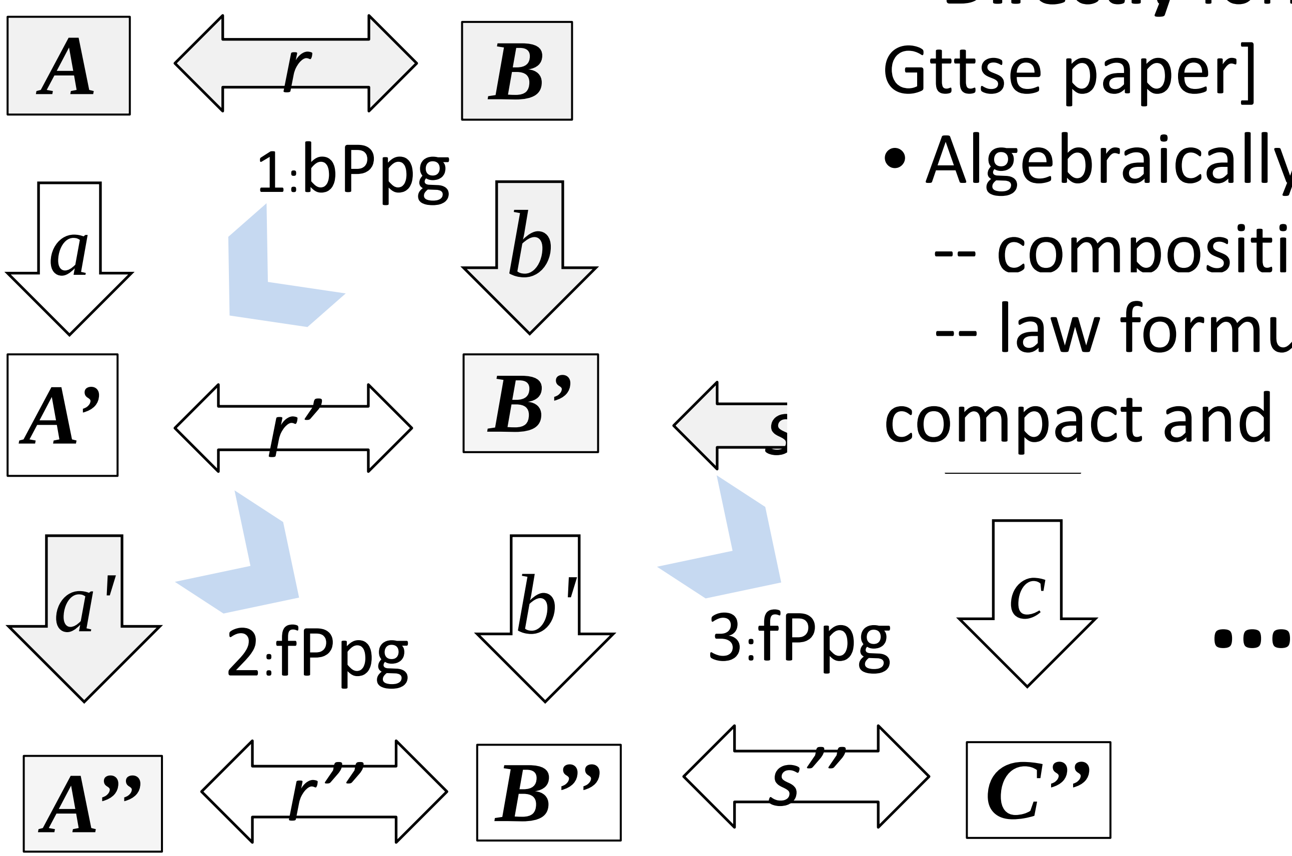
Backward update propagation

Update inversion

Delta lenses: Terms

A string-based term

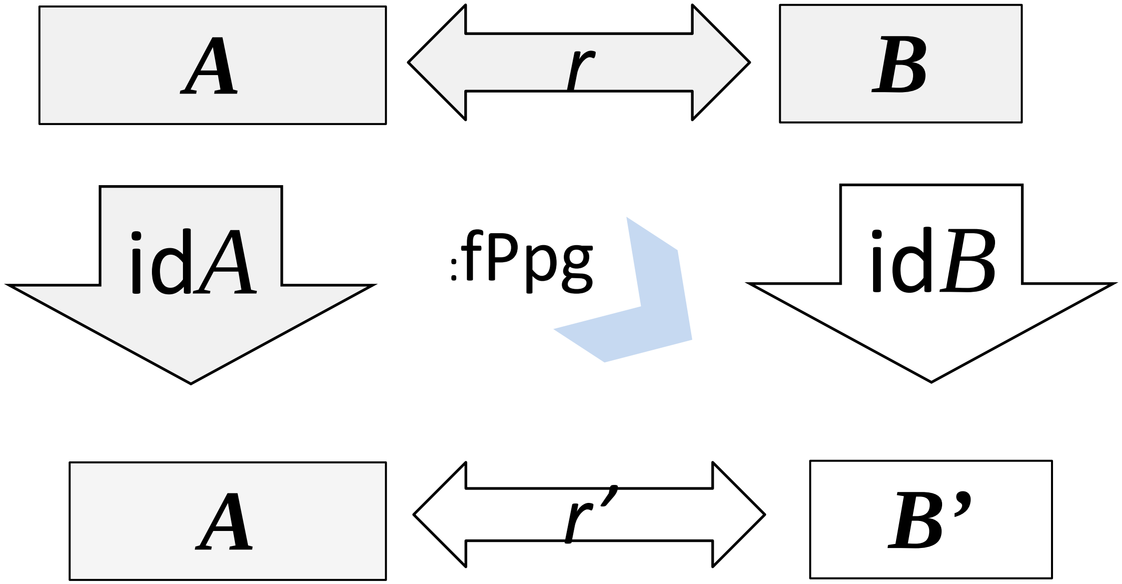
(A + B) * C



Benefits:

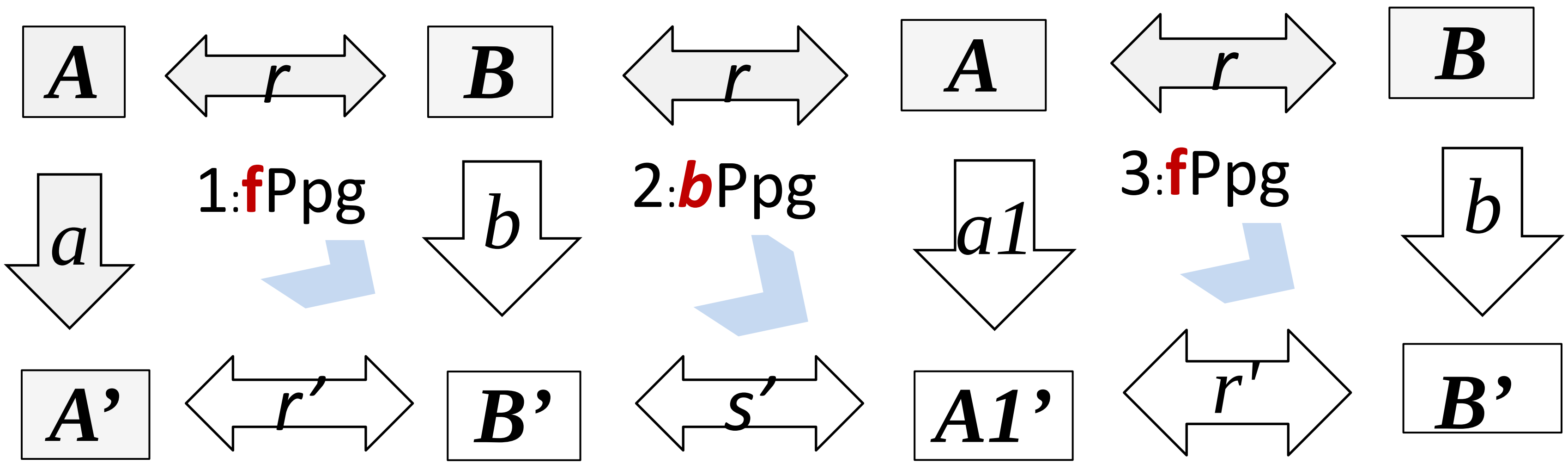
- **Honest** math model
- **Directly** formalizable [my Gttse paper]
- Algebraically manageable:
  - -- composition = **tiling**
  - -- law formulations are compact and graphical

Delta Ppg laws: Identity propagation



Doing nothing is propagated
to doing nothing

Delta Ppg laws: Invertibility (or round-tripping)

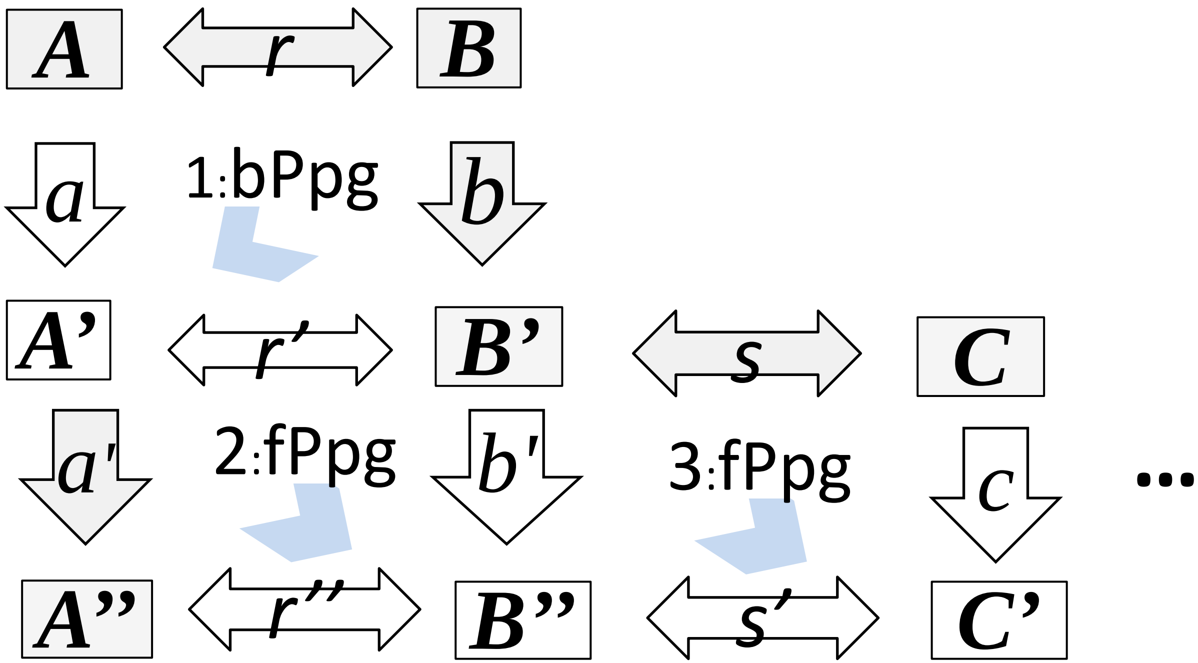


Weak invertibility: $a1 \neq a$

but $a1 \approx a$ in the following sense:

Weak undoability – see the paper

Multi-propagation scenarios and laws



- Complex propagation scenarios are algebraic terms
- Terms + Laws provide:
  - ✓ Compositionality (Combinators)
  - ✓ Possibilities for Optimization

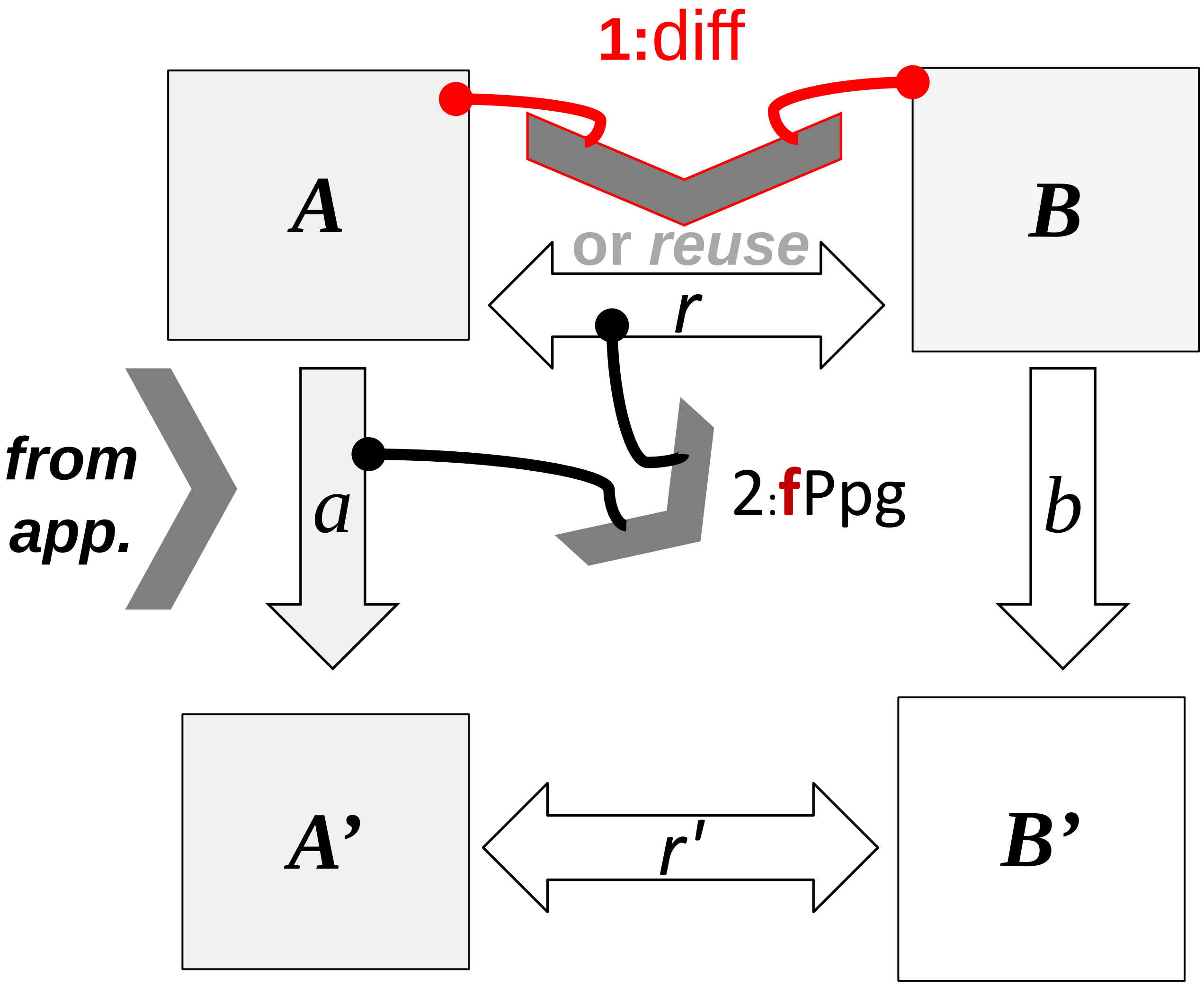
Some summary

The key question: How to get deltas

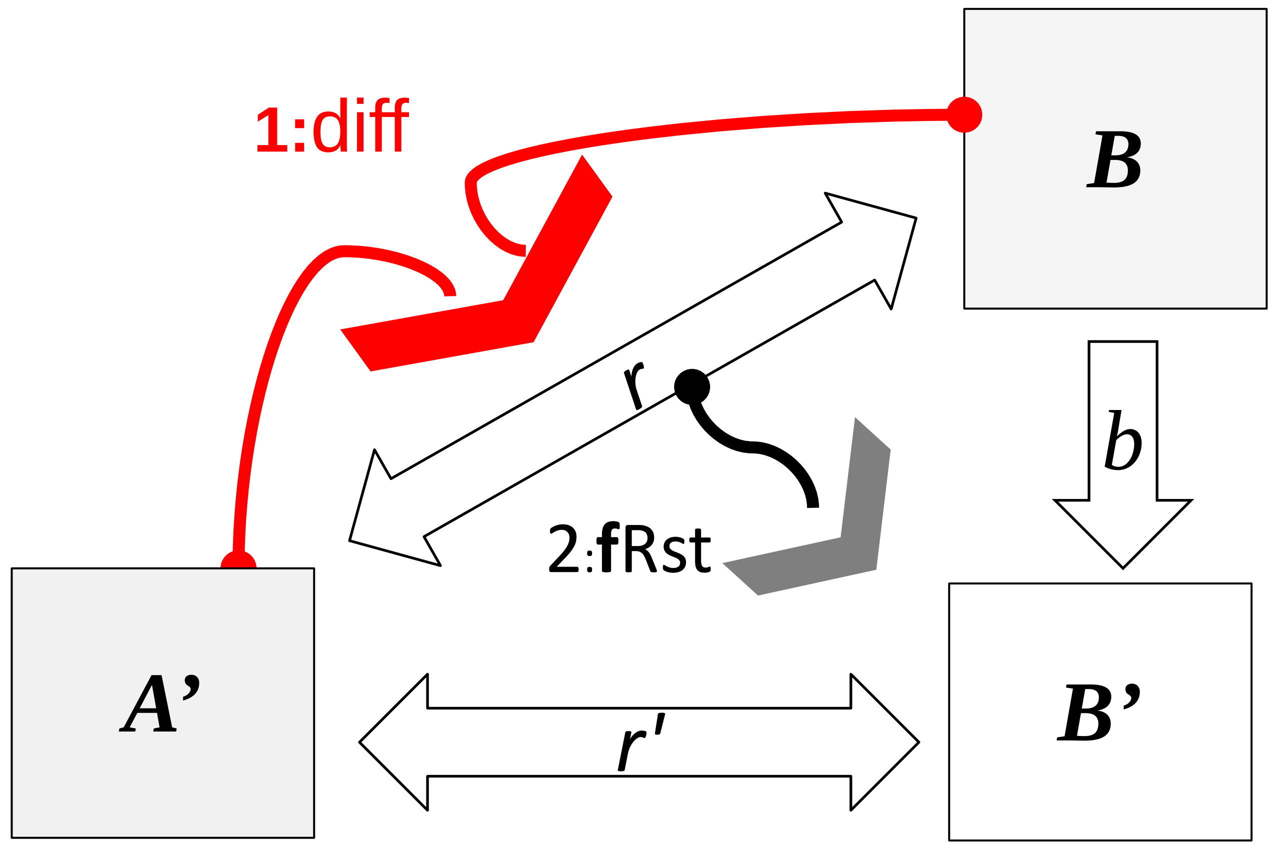
Deltas can be

- computed internally by the sync tool (but outside the propagation module!),
  - Particularly, reused
- provided by the outside applications
- both: say, update deltas are provided externally while corrs are computed internally

How to get deltas.
Architecture 1: Delta Lenses



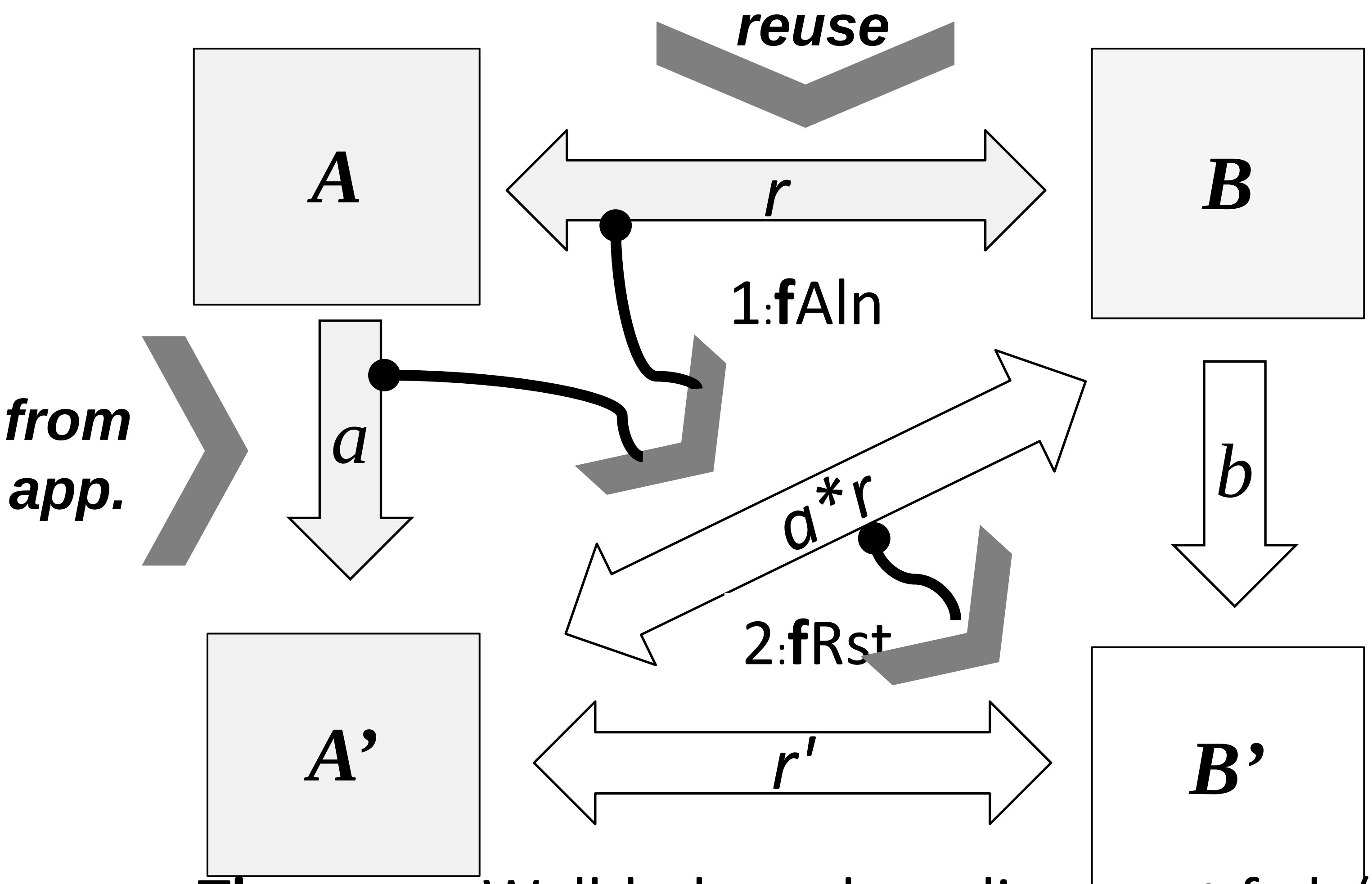
How to get deltas. Architecture 2: Delta Maintainers



Pros: No need for update deltas at the input (loose coupling)

Cons: No reuse of corrs

Architecture 1*: Lenses= Re-alignment + Maintainers



Theorem. Well-behaved re-alignment fwk (RF) and constraint maintainer (CM) give rise to a well-behaved delta-lens, **"DL = RF + CM".**

Summary of architecture discussion

- We have three delta-based operations:
  1) model diff (delta discovery),
  2) re-alignment (delta composition),
  3) consistency restoration (delta maintainer),
- Amongst the three operations, 2) and 3) are algebraic, and subject to simple laws. Operation 1) -- diff -- is not algebraic!
- Separation of concerns: Having each operation implemented by a separate module, we can assemble a series of sync architectures (entirely state-based, with external update deltas, …) .

What is in the companion paper

- A concrete implementation of delta lenses with TGGs (Triple Graph Grammars);
- And more….

Overall summary:

- **BX** is an important special case of the big problem of **model sync**.
- The **state-based** BX framework does **not** work well for models. Its simplicity is **deceiving**.
- We need a **delta-based** BX framework (operations + laws) as introduced in the paper

Overall summary cont'd

- The delta-based framework is
  - Much **more flexible** (and delta-based architecture subsumes the state-based one),
  - **Less error** prone,
  - More **manageable** algebraically.
- **Tile algebra:** a happy marriage of formal rigor and graphical handiness

References

Our papers on delta-based asymm. sync -- ICMT’10 JOT’11

General algebraic framework for both asymm. and symm. Cases in GTTSE’09 paper by Z. Diskin

Algebraic models of delta-based ppg (problematic slide)

- Why algebra?

-- semantics

-- algebraic manipulations