Lenses: an introductory view

Bob Rosebrugh

Ottawa Logic Seminar / 2020-11-12

Outline

- Bidirectional transformations
- Lenses: symmetric and asymmetric
- Categories of lenses
- Bicategories of lenses

- Recent developments
  - multiary lenses (and a different composition)
  - learners

Bidirectional transformations (BX)

" some way of specifying algorithmically how consistency should be restored" - P. Stevens 2005

Precedents:

- Database view update
  - Database a set of tables with columns eg Staff, Projects
  - View is query(ies) eg SELECT Name, Role FROM Staff, Projects WHERE ...
  - Propagate a view state update to the database??
  - Can be ill-posed (no/non-unique solution)
- Model driven development
  - Developers work on separate models, focussing on the concerns at hand
  - When one model is edited, others should be updated to restore consistency
  - eg Object–relational mapping: business logic in object-oriented language with data layer stored in a relational database.

Bidirectional transformations: approaches

- Relational: sets X, Y of model states, *consistency* relation
  $R \subseteq X \times Y$
  restorers $f : X \times Y \longrightarrow Y$ and $b : X \times Y \longrightarrow X$
  subject to correctness/Hippocraticness
- Triple-Graph-Grammars: two *graphs* for meta-models,
  with *triples* relating nodes across them,
  and rules *(grammar)* for how they evolve
  multiple implementations and applications (since 1990's)
- Lenses (set based): defined by Pierce et al, 2004...
- Lenses (categorical): studied by J & R, Diskin et al, 2008...

Lens

- Consider model domains X, Y... of *model states*
- Model states X, Y might be:
  elements of a set, of an order, objects of a category
- *Synchronization data* (various encodings) specifies
  consistency between an X state and a Y state
- Lens $L : \mathrm{X} \longrightarrow \mathrm{Y}$ implements
  Bidirectional Transformation and has both:
  - *synchronization data* and
  - *consistency restoration* or *re-synchronization* operator(s)
    responding to state change.

Lens

- *Symmetric* and *asymmetric* cases arise with different, but related, motivation...
- Symmetric: Concurrent models with bidirectional (two-way) re-synchronization: model domains X and Y peers motivating example: database interoperation
- Asymmetric: Only one non-trivial restoration operator returns X (global) state change after Y (local) change: motivating example: database view updates

Symmetric lens

Consistency data (synchronization) for states X in X and Y in Y denoted by $R : X \leftrightarrow Y$.

Suppose X synchronized with Y by $R : X \leftrightarrow Y$,
then given an *update* from state X (with target X', say)

a *symmetric lens* delivers an update to Y (target Y', say)
and, re-synchronization $R' : X' \leftrightarrow Y'$.

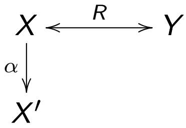
$$X \xleftrightarrow{\ R\ } Y$$

Symmetric lens

Consistency data (synchronization) for states X in X and Y in Y denoted by $R : X \leftrightarrow Y$.

Suppose X synchronized with Y by $R : X \leftrightarrow Y$,
then given an *update* from state X (with target X', say)

a *symmetric lens* delivers an update to Y (target Y', say)
and, re-synchronization $R' : X' \leftrightarrow Y'$.

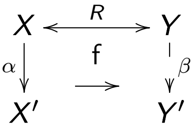


Symmetric lens

Consistency data (synchronization) for states X in X and Y in Y denoted by $R : X \leftrightarrow Y$.

Suppose X synchronized with Y by $R : X \leftrightarrow Y$,
then given an *update* from state X (with target X', say)

a *symmetric lens* delivers an update to Y (target Y', say)
and, re-synchronization $R' : X' \leftrightarrow Y'$.

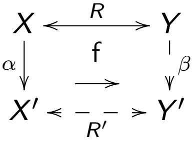


Symmetric lens

Consistency data (synchronization) for states X in X and Y in Y is denoted by $R : X \leftrightarrow Y$.

Suppose X synchronized with Y by $R : X \leftrightarrow Y$,
then given an *update* from state X (with target X', say)

a *symmetric lens* delivers an update to Y (target Y', say)
and, re-synchronization $R' : X' \leftrightarrow Y'$.



Symmetric lens

Symmetrically, suppose $R : X \leftrightarrow Y$, then given an *update* from Y (with target Y')

symmetric lens delivers update of X in X *and*, *re-synchronization* $R'' : X' \leftrightarrow Y'$.

$$\begin{array}{ccc} X & \overset{R}{\longleftrightarrow} & Y \\ \delta \downarrow & \mathsf{b} \; \longleftarrow & \downarrow \gamma \\ X' & \underset{R''}{\leftarrow - - \rightarrow} & Y' \end{array}$$

- Considered by Hoffman, Pierce, Wagner for X, Y... sets
- More recently Diskin et al. for X, Y... categories
- Also studied by J & R

Symmetric lens

Formally, taking categories X, Y for model domains:

A symmetric lens $L = (\delta_\mathsf{X}, \delta_\mathsf{Y}, \mathsf{f}, \mathsf{b})$ from X to Y
has a span of *sets*

$$\delta_\mathsf{X} : \mathsf{X}_0 \longleftarrow \mathsf{R}_{\mathsf{XY}} \longrightarrow \mathsf{Y}_0 : \delta_\mathsf{Y}$$

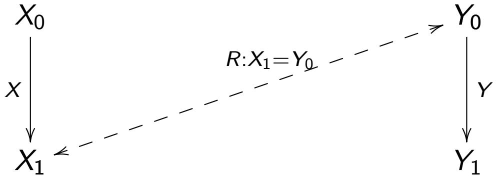
where elements of R_{XY} – *"corrs"* – are denoted $R : X \leftrightarrow Y$ and
forward and *backward propagations* f, b denoted

$$\begin{array}{ccc} X & \overset{R}{\longleftrightarrow} & Y \\ {\scriptstyle\alpha}\downarrow & \overset{\mathsf{f}}{\longrightarrow} & \downarrow{\scriptstyle\beta} \\ X' & \underset{R'}{\leftarrow - - \rightarrow} & Y' \end{array} \qquad \begin{array}{ccc} X & \overset{R}{\longleftrightarrow} & Y \\ {\scriptstyle\delta}\downarrow & \overset{\mathsf{b}}{\longleftarrow} & \downarrow{\scriptstyle\gamma} \\ X' & \underset{R''}{\leftarrow - - \rightarrow} & Y' \end{array}$$

where $\mathsf{f}(\alpha, R) = (\beta, R')$ and $\mathsf{b}(\gamma, R) = (\delta, R'')$
and both propagations respect identities and composition.

Symmetric Lens: Example

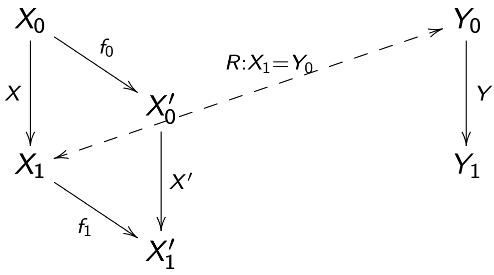
Suppose X = Y = set^2 are model domains (we'll interpret below)



Say X, Y objects of set^2 have synchronization R just when $X_1 = d_1 X = d_0 Y = Y_0$,

Symmetric Lens: Example

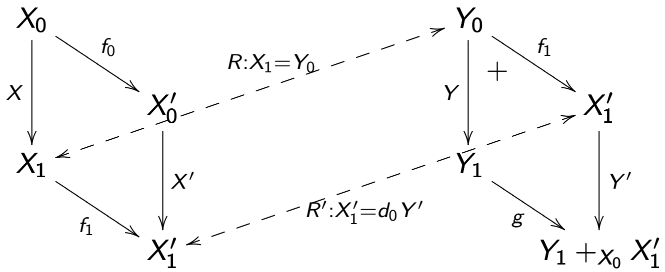
Suppose $(f_0, f_1) : X \longrightarrow X'$ an arrow in X, as in



Forward propagation requires a new arrow $Y \longrightarrow Y'$ say, and a new synchronization R'

Symmetric Lens: Example

Construct the new arrow $(f_1, g) : Y \longrightarrow Y'$ using the pushout, and the new synchronization is $R' : X_1' = d_0 Y'$:

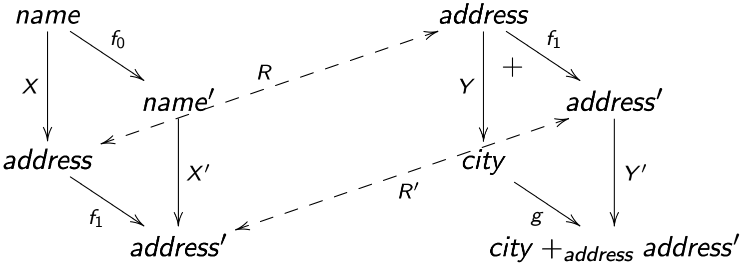


Back propagation uses composition.

Symmetric Lens: Example

For example: a left hand db state assigns *name* to *address;*
a right hand state assigns *address* to *city;*
so a synchronization is an *address* matching

name/address update propagates to a right hand update,
also creating a new *city* set: the pushout



Symmetric Lens: Composition and equivalence

- Symmetric lenses compose by composing propagations; pullbacks of δ's provide corrs for a composite
- *However* two symmetric lenses on the same model domains *may* have the same propagation behaviour i.e. bidirectional transformation implementation

- Should they be distinguished? Depends on preference, and
- J & R defined a congruence relation on lenses $X \longrightarrow Y$

Symmetric Lens: Equivalence

Let L, L' have corrs R_{XY} and R'_{XY}.
Say $L \equiv L'$ if there is relation σ between corr sets so that:

- σ compatible with the δ's
- $R\sigma R'$ implies Y updates of $\mathsf{f}(\alpha, R)$ and $\mathsf{f}'(\alpha, R')$ equal and new corrs are σ related (similarly for b)
- σ total in both directions

Theorem
Equivalence classes of symmetric lenses are arrows of a category, denoted SLens.

Symmetric lenses and Mealy morphisms

Bob Paré observed that

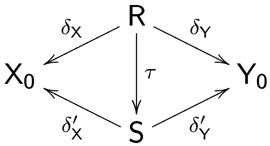
f, b are precisely (cat) Mealy morphisms:

$f : X \longrightarrow Y$ and $b : Y \longrightarrow X$

Bryce Clarke uses this for two important points:

First, composing via span (of sets) composition,
Mealy morphisms are 1-cells of a bicategory
Meal where a 2-cell is:

A map of Mealy morphisms i.e. a span morphism τ:



compatible with the operations

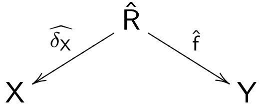
Symmetric lenses and Mealy morphisms

Second, a Mealy morphism $\mathsf{f} : \mathsf{X} \longrightarrow \mathsf{Y}$ has image category $\hat{\mathsf{R}}$ with:
objects: R
morphisms: pairs $(\alpha, R) : R \longrightarrow R'$ where $\mathsf{f}(\alpha, R) = (\beta, R')$

And factors (in Meal) as $\mathsf{X} \longrightarrow \hat{\mathsf{R}} \longrightarrow \mathsf{Y}$ using f, moreover

Proposition

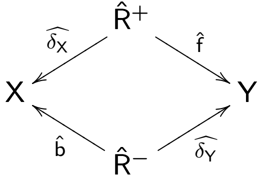
Given a Mealy morphism $\mathsf{f} : \mathsf{X} \longrightarrow \mathsf{Y}$ *there is a span of functors*



where $\widehat{\delta_{\mathsf{X}}}$ *is a discrete opfibration and* $\hat{\mathsf{f}}$ *is a functor*

Symmetric lenses and Mealy morphisms

Symmetric lens $X \longrightarrow Y$ can be represented as a pair of Mealy morphisms:



- will return to this, but for now...
- giving 2-cells by corresponding maps of Mealy morphisms defines a (hom) category $\mathsf{SymLens}(X, Y)$

Asymmetric lens: Background

Arose as strategy for studying the database View Update Problem, indeed long before symmetric lenses.

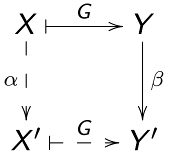
- Defined equationally by B. Pierce et al for sets X, Y
- S. Hegner had axiomatics for orders X, Y, a special case of...
- Lenses for X, Y categories (defined by J & R) and:
  - defined lens *in* category $\mathcal{C}$ with finite products
  - characterized lens as algebra for a monad on $\mathcal{C}/Y$
  - generalized to a categorical version (c-lenses).
- Diskin et al. defined (related) categorical version that we will call *asymmetric lenses*

Set based lenses also arose (1980's) in considering "store shapes" (F. Oles thesis) where there is a similar update problem

Asymmetric lens: Motivation

Database *views* consider a *Get* process $G : \mathsf{X} \longrightarrow \mathsf{Y}$ from global database states X *to* view states Y.

For global state X *synched* with view state $Y = GX$:
when can update to Y, e.g. formal insertion β
lift through G to global update α, *and*
compatibly – meaning $\beta = G(\alpha)$?
This is (an instance of) the View Update Problem.



Asymmetric lens

Given an *update* from state $Y = GX$ in Y (with target Y') the asymmetric lens delivers (by a "Putback" process P) an update to X in X (with target X', say) *along with compatible* re-synchronization data, that is $Y' = GX'$.

$$X \xmapsto{G} Y$$

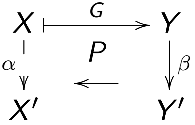
Asymmetric lens

Given an *update* from state $Y = GX$ in Y (with target Y') the asymmetric lens delivers (by a "putback" process P) an update to X in X (with target X', say) *along with compatible* re-synchronization data, namely $Y' = GX'$.

$$\begin{array}{ccc} X & \stackrel{G}{\longmapsto} & Y \\ & & \downarrow{\scriptstyle \beta} \\ & & Y' \end{array}$$

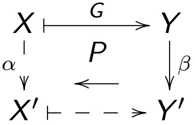
Asymmetric lens

Given an *update* from state $Y = GX$ in Y (with target Y') the asymmetric lens delivers (by a "putback" process P) an update to X in X (with target X', say) *along with compatible* re-synchronization data, namely $Y' = GX'$.



Asymmetric lens

Given an *update* from state $Y = GX$ in Y (with target Y') the asymmetric lens delivers (by a "putback" process P) an update to X in X (with target X', say) *along with compatible* re-synchronization data, namely $Y' = GX'$.



Asymmetric lens

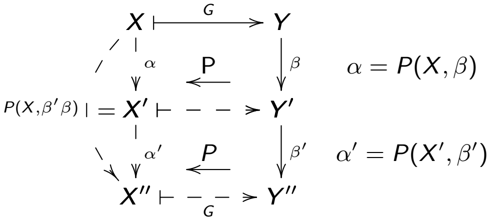
The formal axioms (Diskin et al) are:

An asymmetric lens is $L = (G, P)$
where $G : \mathrm{X} \longrightarrow \mathrm{Y}$ is the "Get" functor and
P is the "Put(back)" function and the data G, P satisfy:

(i) PutGet: $GP(X, \beta) = \beta$

(ii) PutId: $P(X, 1_{GX}) = 1_X$

(iii) PutPut:



or

$$P(X, \beta'\beta : GX \longrightarrow Y' \longrightarrow Y'') = P(X', \beta' : GX' \longrightarrow Y'')P(X, \beta : GX \longrightarrow Y')$$

Asymmetric lens: examples

- Given a split op-fibration $G : \mathsf{X} \longrightarrow \mathsf{Y}$:
  Just define $P(X, \beta)$ to be the op-Cartesian arrow.

- For example, $d_0 : \mathsf{set}^2 \longrightarrow \mathsf{set}$ or $d_1 : \mathsf{set}^2 \longrightarrow \mathsf{set}$

- Or indeed for C, D small categories a functor
  $V : \mathsf{C} \longrightarrow \mathsf{D}$ is fully-faithful
  iff (R L-W) $V^* : \mathsf{set}^{\mathsf{D}} \longrightarrow \mathsf{set}^{\mathsf{C}}$ is an opfibration
- Similar op-fibration characterization holds for
  small, lex C, D and lex functors

Asymmetric lens: examples

Split op-fibs called “c-lenses” by J & R and studied earlier (in the context of View Update Problem)

- defined by equations analogous to asymmetric set-lens
- algebras for a monad on cat/Y
- the Put satisfies a “least change” property

Indeed, *any* asymmetric lens is an algebra for a related *semi*-monad on cat/Y
(Clarke recently showed them to be algebras for a monad)

However: *not every* asymmetric lens is an op-fibration - there are small counterexamples

Asymmetric lens: composition and equivalence

- As for symmetric lenses, there is an obvious composition of asymmetric lenses and category called ALens

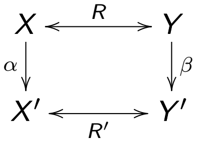
- A *span* of asymmetrics

$$X \xleftarrow{(G_L,P_L)} S \xrightarrow{(G_R,P_R)} Y$$

determines a symmetric lens $X \longrightarrow Y$ via:
corrs are objects of S, δ's from Gets
f is the left leg Put P_L, then the right leg Get G_L
b is the right leg Put, then the left leg Get

Asymmetric lens: composition and equivalence

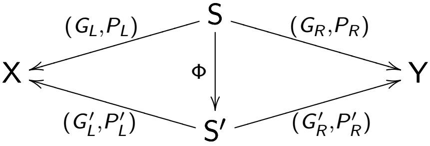
- Conversely, symmetric lens $X \longrightarrow Y$ determines
  a *span* of asymmetrics with:
  head of span (the category) has objects the corrs
  arrows are formal squares

  Gets by projection; Puts use f, b
- J & R sought equivalence of the category SLens of symmetrics
  and a category of spans of asymmetrics

Asymmetric lens: composition and equivalence

- Define span equivalence (again motivated by behaviour):
- Equivalence is generated by functors Φ as in



with Φ surj-on-obj and semi-monad homom (both sides)

Theorem

Equivalence classes of spans define a category SpALens*;* SpALens *is isomorphic to* SLens.

Asymmetric lens and cofunctors

- Ahman and Uustalu observed that for a lens (G,P):
  Object function of G_0 of G together with P determines what Aguiar called a cofunctor
  from Y to X (Note direction!!)

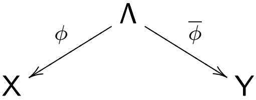
- Cofunctors compose via their functions (axioms are ok)

- Cofunctors generalize both boo functors and discrete opfibrations.

Asymmetric lens and cofunctors

- For cofunctor $(G_0, P) : \mathsf{Y} \longrightarrow \mathsf{X}$ let Λ the category with:
  objects X_0
  morphisms $(X, \beta) : X \longrightarrow P(X, \beta)$ for $\beta : G_0(X) \longrightarrow Y'$

- A cofunctor $(G_0, P) : \mathsf{Y} \longrightarrow \mathsf{X}$ defines a span of functors:

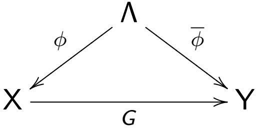


  with ϕ identity on objects and $\overline{\phi}$ a discrete opfibration

Asymmetric lens and cofunctors

Clarke then points out:

- An asymmetric lens $(G, P) : \mathsf{X} \longrightarrow \mathsf{Y}$ defines a commutative diagram of functors:

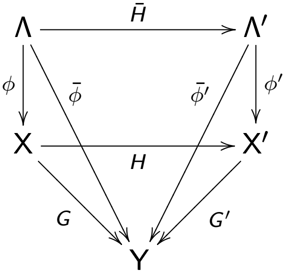


  with ϕ identity on objects and $\overline{\phi}$ a discrete opfibration

- Compose asymmetric lenses (seen thus) by composing the functor/cofunctor parts (giving ALens again)
- but more important from this perspective...

Spans of asymmetric lens

- For category Y, the category Lens(Y) has:
  objects are asymmetric lenses to Y;
  arrows (using the representation above) are comm diagrams:



- Lens(Y) has products

Spans of asymmetric lens

- There is a forgetful functor $\mathsf{Lens}(\mathsf{Y}) \longrightarrow \mathsf{cat}$ sending an object to domain of G.
- The head of the pullback diagram

$$\begin{array}{ccc} & \mathsf{SpnLens}(\mathsf{X},\mathsf{Y}) & \\ \swarrow & & \searrow \\ \mathsf{Lens}(\mathsf{X}) & & \mathsf{Lens}(\mathsf{Y}) \\ \searrow & & \swarrow \\ & \mathsf{cat} & \end{array}$$

  defines the hom categories for a bicategory SpnLens
- morphisms of $\mathsf{SpnLens}(\mathsf{X},\mathsf{Y})$ (2-cells of SpnLens) are span morphisms of Gets compatible with cofunctor parts

Symmetric lens adjunctions

- There is forgetful functor $\mathrm{Meal}(X, Y) \longrightarrow \mathit{Span}(\mathrm{cat})(X, Y)$ from the span representation above
- Further, there is $\mathrm{Meal}(Y, X) \longrightarrow \mathit{Span}(\mathrm{cat})(X, Y)$ by first reversing the span representation
- The head of the pullback diagram

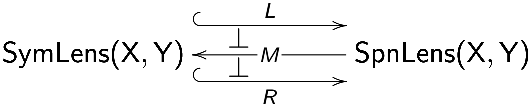
$$\begin{array}{ccc} & \mathrm{SymLens}(X, Y) & \\ \swarrow & & \searrow \\ \mathrm{Meal}(X, Y) & & \mathrm{Meal}(Y, X) \\ \searrow & & \swarrow \\ & \mathit{Span}(\mathrm{cat})(X, Y) & \end{array}$$

defines the hom categories for a bicategory SymLens

Symmetric lens adjunctions

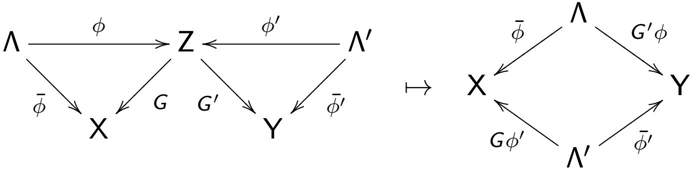
Theorem (Clarke, ACT20 paper)

There is an adjoint triple



with R reflective and (hence) L coreflective

Using functor/cofunctor representations, define M on objects by



Symmetric lens adjunctions

- The definition of R is related to the J & R construction
- The definition of L is a bit more complicated
- Using that everything is identity on objects and that the constructions are compatible with composition, he obtains:

Corollary (Clarke)

There are identity on objects pseudofunctors

$$\mathsf{SymLens} \underset{R}{\overset{L}{\rightrightarrows}} \mathsf{SpnLens}, \qquad \mathsf{SymLens} \xleftarrow{M} \mathsf{SpnLens}$$

with L and R locally fully faithful and locally adjoint to M.

Summary (so far)

- Lenses (either flavour) model BX well
- Symmetric lenses and asymmetrics closely related via spans
- J & R: Isomorphism of categories from (classes of) symmetric lenses to spans of asymmetrics
- Using Mealy morphism and functor/cofunctor representations
- Clarke describes the bicategories and adjoint triple above

Multiary lenses

- Multidirectional transformations modelled as *n*-ary lenses proposed by Diskin and Konig
  - first generalize (binary) symmetric lenses – more propagations
  - also generalize spans of asymmetric lenses – to wide spans
- J & R found equivalences similar to the binary case
- Subject to some mild conditions the resulting *multiary lenses* compose via wide spans
- A multicategory of multiary lenses arises

Lenses and learners

Fong and Johnson (BX 2019) relate supervised learning algorithms to (set-based) symmetric lenses

- Goal: approximate $f : A \longrightarrow B$ by $(a, f(a))$ pairs (training data) parameterized by P, then allow updates
- Learner is $(P, I, U, , R) : A \longrightarrow B$ with
  $I : P \times A \longrightarrow B$ (implementation),
  $U : B \times P \times A \longrightarrow P$ (update),
  $R : B \times P \times A \longrightarrow A$ (request)
  (for details see their paper)
- They find faithful, symmetric monoidal functor from a category with learner arrows to a category with symmetric lens arrows

Conclusion

- Lenses implement BX with categorical precision
- Categories, bicategories (even double cats) clarify structure
- Some urls:
- `www.mta.ca/~rrosebru`
- `www.comp.mq.edu.au/~mike/`
- Bryce is on Twitter...