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.
Relational: sets $X, Y$ of model states, consistency relation $R \subseteq X \times Y$

restorers $f : X \times Y \rightarrow Y$ and $b : X \times Y \rightarrow X$

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...
Consider model domains \( X, Y \)… of \textit{model states}

Model states \( X, Y \) might be: elements of a set, of an order, objects of a category

\textit{Synchronization data} (various encodings) specifies \textit{consistency} between an \( X \) state and a \( Y \) state

\textbf{Lens} \( L : X \rightarrow Y \) implements \textit{Bidirectional Transformation} and has both:

\begin{itemize}
  \item \textit{synchronization data} and
  \item \textit{consistency restoration} or \textit{re-synchronization} operator(s) responding to state change.
\end{itemize}
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 $R' : X' \leftrightarrow Y'$.

\[
\begin{array}{ccc}
X & \xleftarrow{R} & Y
\end{array}
\]
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 $R' : X' \leftrightarrow Y'$.

\[ X \xleftarrow{R} Y \]
\[ \alpha \downarrow \]
\[ X' \]
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 $R' : X' \leftrightarrow Y'$.

\[
\begin{array}{c}
X \\ \downarrow \alpha \\
X'
\end{array} \quad \xrightarrow{R} \quad \begin{array}{c}
Y \\ \downarrow \beta \\
Y'
\end{array}
\]
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 $R' : X' \leftrightarrow Y'$.

\[
\begin{array}{c}
X \xleftarrow{\alpha} X' \xrightarrow{R'} Y' \\
Y \xrightarrow{\beta} \end{array}
\]

\[
\begin{array}{c}
X \xleftarrow{\alpha} X' \xrightarrow{R} Y \\
Y \xrightarrow{\beta} \end{array}
\]
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 $R'' : X' \leftrightarrow Y'$.

\[
\begin{array}{c}
X \xleftarrow{\delta} \xrightarrow{b} Y \\
\downarrow \delta \quad \downarrow \gamma \\
X' \xleftarrow{\_\_\_\_} \xrightarrow{\_\_\_\_} Y'
\end{array}
\]

- 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 $L = (\delta_X, \delta_Y, f, b)$ from $X$ to $Y$

has a span of sets

$$\delta_X : X_0 \leftarrow R_{XY} \rightarrow Y_0 : \delta_Y$$

where elements of $R_{XY}$ – “cors” – are denoted $R : X \leftrightarrow Y$ and

forward and backward propagations $f, b$ denoted

$$\begin{array}{ccc}
X & \xleftarrow{R} & Y \\
\downarrow \alpha & f & \downarrow \beta \\
X' & \xleftarrow{R'} & Y'
\end{array} \quad \begin{array}{ccc}
X & \xleftarrow{R} & Y \\
\downarrow \delta & b & \downarrow \gamma \\
X' & \xleftarrow{R''} & Y'
\end{array}$$

where $f(\alpha, R) = (\beta, R')$ and $b(\gamma, R) = (\delta, R'')$

and both propagations respect identities and composition.
Suppose $X = Y = \text{set}^2$ 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 \((f_0, f_1) : X \rightarrow X'\) an arrow in \(X\), as in

\[
\begin{array}{c c c c c}
X_0 & f_0 & X' & R : X_1 = Y_0 & Y_0 \\
X & X_0 & X' & & Y \\
X_1 & f_1 & X' & X_1' & Y_1
\end{array}
\]

Forward propagation requires a new arrow \(Y \rightarrow Y'\) say, and a new synchronization \(R'\)
Symmetric Lens: Example

Construct the new arrow \((f_1, g) : Y \rightarrow Y'\) using the pushout, and the new synchronization is \(R' : X'_1 = d_0 Y'\):

\[
\begin{array}{cccccc}
X_0 & \xrightarrow{f_0} & X'_0 & \xrightarrow{R : X_1 = Y_0} & Y_0 & \xrightarrow{f_1} & X'_1 \\
X & \downarrow & X' & \downarrow & Y & \downarrow & Y' \\
X_1 & \xleftarrow{f_1} & X'_1 & \xleftarrow{R' : X'_1 = d_0 Y'} & Y_1 + x_0 X'_1 \\
\end{array}
\]

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 $\delta$’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 \( \sigma \) between corr sets so that:

- \( \sigma \) compatible with the \( \delta \)'s
- \( R\sigma R' \) implies \( Y \) updates of \( f(\alpha, R) \) and \( f'(\alpha, R') \) equal and new corrs are \( \sigma \) related (similarly for b)
- \( \sigma \) 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**: \( f : X \to Y \) and \( b : Y \to X \)

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 \( \tau \):

\[
\begin{array}{ccc}
X_0 & \xleftarrow{\delta_X} & \xrightarrow{\delta_Y} & Y_0 \\
\gamma & \downarrow \tau & & \\
S & \xleftarrow{\delta'_X} & \xrightarrow{\delta'_Y} & R
\end{array}
\]

compatible with the operations
Symmetric lenses and Mealy morphisms

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

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

**Proposition**

*Given a Mealy morphism $f : X \rightarrow Y$ there is a span of functors*

![Diagram](attachment:image.png)

where $\hat{\delta}_X$ is a discrete opfibration and $\hat{f}$ is a functor
Symmetric lenses and Mealy morphisms

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

\[
\begin{array}{ccc}
\hat{R}^+ & \xrightarrow{\hat{f}} & Y \\
\delta_X & \downarrow & \\
X & \downarrow & \\
\hat{R}^- & \xleftarrow{\hat{b}} & X
\end{array}
\]

- will return to this, but for now...

- giving 2-cells by corresponding maps of Mealy morphisms defines a (hom) category $\text{SymLens}(X, Y)$
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 C with finite products
  ▶ characterized lens as algebra for a monad on 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 $G : X \rightarrow Y$ 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 $\beta$ *lift through* $G$ to global update $\alpha$, and compatibly – meaning $\beta = G(\alpha)$? This is (an instance of) the *View Update Problem*.

\[
\begin{align*}
    X & \overset{G}{\longrightarrow} Y \\
    \downarrow\alpha & \quad \quad \quad \quad \quad \downarrow\beta \\
    X' & \overset{G}{\longrightarrow} Y'
\end{align*}
\]
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 $Y' = GX'$. 

$$X \xrightarrow{G} Y$$
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 $Y' = GX'$.

\[
\begin{array}{c}
X \xrightarrow{G} Y \\
\downarrow^\beta \\
Y'
\end{array}
\]
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 $Y' = GX'$. 

\[
\begin{array}{ccc}
X & \xrightarrow{G} & Y \\
\downarrow{\alpha} & & \downarrow{\beta} \\
X' & \leftarrow{P} & Y'
\end{array}
\]
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 \( Y' = GX' \).

\[
\begin{array}{ccc}
X & \xrightarrow{G} & Y \\
\downarrow{}_{\alpha} & & \downarrow{}_{\beta} \\
X' & \xleftarrow{P} & Y'
\end{array}
\]
Asymmetric lens

The formal axioms (Diskin et al) are:

An asymmetric lens is \( L = (G, P) \)
where \( G : X \to Y \) 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:

\[
\begin{align*}
X & \xrightarrow{G} Y \\
/ & \quad \downarrow \alpha \quad \uparrow P \quad \downarrow \beta \\
P(X, \beta') & = X' \quad -- \rightarrow Y' \\
/ & \quad \downarrow \alpha' \quad \uparrow P \quad \downarrow \beta' \\
X'' & \xrightarrow{G} Y''
\end{align*}
\]

or

\[
P(X, \beta' \beta : GX \to Y' \to Y'') = P(X', \beta' : GX' \to Y'') P(X, \beta : GX \to Y')
\]
Asymmetric lens: examples

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

- For example, $d_0 : \text{set}^2 \rightarrow \text{set}$ or $d_1 : \text{set}^2 \rightarrow \text{set}$

- Or indeed for $C, D$ small categories a functor $V : C \rightarrow D$ is fully-faithful
  iff $(R \text{ L-W}) V^* : \text{set}^D \rightarrow \text{set}^C$ 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

\[
\begin{array}{c}
X & \xrightarrow{(G_L,P_L)} & S & \xrightarrow{(G_R,P_R)} & Y \\
\end{array}
\]

determines a symmetric lens \( X \rightarrow Y \) via:
corrs are objects of \( S \), \( \delta \)'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 $X \rightarrow Y$ determines a span of asymmetrics with:
  - head of span (the category) has objects the corrs
  - arrows are formal squares

\[
\begin{array}{ccc}
X & \xrightarrow{R} & Y \\
\downarrow{} & {} & \downarrow{} \\
X' & \xleftarrow{R'} & Y'
\end{array}
\]

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 $\Phi$ as in

\[
\begin{array}{ccc}
X & S & Y \\
\downarrow & \Phi & \downarrow \\
S' & (G_R,P_R) & (G'_R,P'_R) \\
\end{array}
\]

with $\Phi$ surj-on-obj and semi-monad homom (both sides)

Theorem

*Equivalence classes of spans define a category $\text{SpALens}$; $\text{SpALens}$ is isomorphic to $\text{SLens}$.*
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 \((G_0, P) : Y \rightarrow X\) let \(\Lambda\) the category with:
  objects \(X_0\)
  morphisms \((X, \beta) : X \rightarrow P(X, \beta)\) for \(\beta : G_0(X) \rightarrow Y'\)

- A cofunctor \((G_0, P) : Y \rightarrow X\) defines a span of functors:

\[
\begin{array}{ccc}
X & \xleftarrow{\Lambda} & Y \\
\phi & \xlongrightarrow{} & \bar{\phi} \\
\end{array}
\]

with \(\phi\) identity on objects and \(\bar{\phi}\) a discrete opfibration
Clarke then points out:

- An asymmetric lens \((G, P) : X \to Y\) defines a commutative diagram of functors:

\[
\begin{array}{ccc}
X & \xrightarrow{G} & Y \\
\downarrow{\phi} & \searrow{\bar{\phi}} & \Lambda \\
\Lambda & \xleftarrow{\phi} & X
\end{array}
\]

with \(\phi\) identity on objects and \(\bar{\phi}\) 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 $\text{Lens}(Y)$ has:
objects are asymmetric lenses to $Y$;
arrows (using the representation above) are comm diagrams:

\[
\begin{array}{c}
\Lambda & \xrightarrow{\bar{H}} & \Lambda' \\
\downarrow{\phi} & & \downarrow{\phi'} \\
X & \xrightarrow{H} & X'
\end{array}
\]

\[
\begin{array}{c}
\Lambda & \xleftarrow{H} & \Lambda' \\
\downarrow{\phi} & & \downarrow{\phi'} \\
X & \xleftarrow{G} & X'
\end{array}
\]

\[
\begin{array}{c}
\Lambda & \xleftarrow{\bar{H}} & \Lambda' \\
\downarrow{\phi} & & \downarrow{\phi'} \\
Y & \xleftarrow{G} & Y
\end{array}
\]

$\text{Lens}(Y)$ has products
Spans of asymmetric lens

- There is a forgetful functor $\text{Lens}(Y) \rightarrow \text{cat}$ sending an object to domain of $G$.
- The head of the pullback diagram defines the hom categories for a bicategory $\text{SpnLens}$.
- Morphisms of $\text{SpnLens}(X, Y)$ (2-cells of $\text{SpnLens}$) are span morphisms of $\text{Gets}$ compatible with cofunctor parts.
Symmetric lens adjunctions

- There is forgetful functor $\text{Meal}(X, Y) \rightarrow \text{Span}(\text{cat})(X, Y)$ from the span representation above.
- Further, there is $\text{Meal}(Y, X) \rightarrow \text{Span}(\text{cat})(X, Y)$ by first reversing the span representation.
- The head of the pullback diagram defines the hom categories for a bicategory $\text{SymLens}$.
Symmetric lens adjunctions

Theorem (Clarke, ACT20 paper)

*There is an adjoint triple*

\[
\begin{array}{c}
\text{SymLens}(X, Y) \\
\downarrow L \\
\downarrow M \\
\downarrow R \\
\end{array}
\end{array}
\begin{array}{c}
\text{SpnLens}(X, Y) \\
\end{array}
\]

with \( R \) reflective and (hence) \( L \) coreflective

Using functor/cofunctor representations, define \( M \) on objects by

\[
\begin{array}{c}
\Lambda \\
\phi \\
\Lambda' \\
\end{array}
\end{array}
\begin{array}{c}
\phi' \\
\Lambda \\
\Lambda' \\
\end{array}
\end{array}
\begin{array}{c}
Z \\
G \\
Z \\
\end{array}
\end{array}
\begin{array}{c}
X \\
Y \\
X \\
\end{array}
\end{array}
\begin{array}{c}
Y \\
Y \\
X \\
\end{array}
\end{array}
\begin{array}{c}
\Lambda \\
G' \phi \\
\Lambda' \\
\end{array}
\end{array}
\begin{array}{c}
\Lambda \\
\Lambda \\
\Lambda' \\
\end{array}
\end{array}
\begin{array}{c}
G' \phi \\
\Lambda \\
\Lambda' \\
\end{array}
\end{array}
\begin{array}{c}
X \\
Y \\
X \\
\end{array}
\end{array}
\end{array}
\end{array}
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*

\[
\begin{align*}
\text{SymLens} & \xrightarrow{L} \text{SpnLens} \\
\text{SpnLens} & \xleftarrow{M} \text{SymLens} \xrightarrow{R}
\end{align*}
\]

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 \( f : A \to B \) by \((a, f(a))\) pairs (training data) parameterized by \( P \), then allow updates

- Learner is \((P, I, U, R) : A \to B\) with
  - \(I : P \times A \to B\) (implementation),
  - \(U : B \times P \times A \to P\) (update),
  - \(R : B \times P \times A \to A\) (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...