Transducing for fun and profit

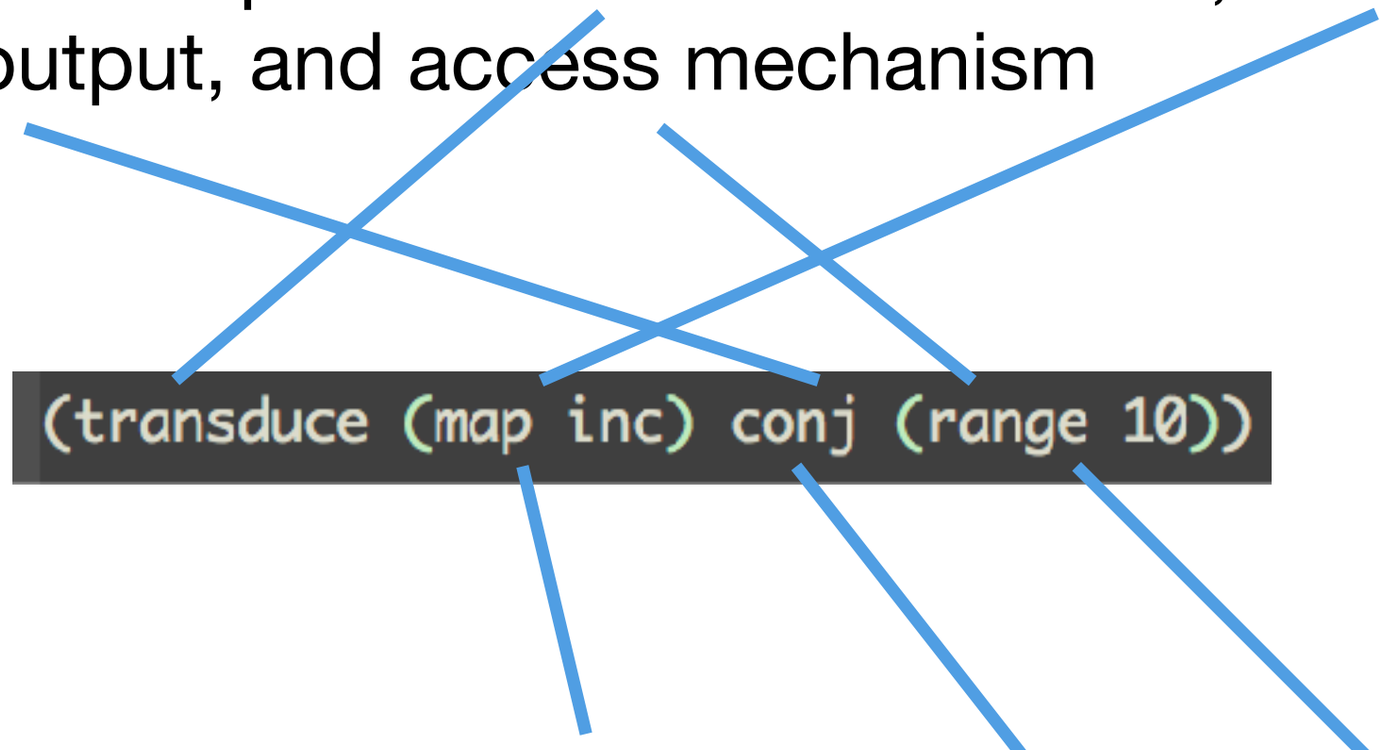
simon@metabase.com
@sbelak

Clojure at a glance

- `(lisp (running-on :JVM))`
- Functional, dynamic, immutable
- Excellent concurrency and state-management primitives
- Unparalleled data manipulation

Anatomy of a transducer

- Transducers decomplect recursion mechanism, transformation, building the output, and access mechanism



- 3 user-facing “protocols”: transducer, reducing fn, CollReduce

transducer and reducing function

```clojure
(defn my-map [f]
  (fn [rf]
    (fn
      ([] (rf))
      ([result] (rf result))
      ([result input]
       (rf result (f input))))))
```

```clojure
(defn my-min
  ([] Double/POSITIVE_INFINITY)
  ([acc] acc)
  ([acc e]
   (min acc e)))
```

transducer and reducing function

```clojure
(defn my-map [f]
  (fn [rf]
    (fn
      ([] (rf))
      ([result] (rf result))
      ([result input]
       (rf result (f input))))))
```

```clojure
(defn my-min
  ([] Double/POSITIVE_INFINITY)
  ([acc] acc)
  ([acc e]
   (min acc e)))
```

Using a transducer to wrap/keep state

```clojure
(defn my-drop [n]
  (fn [rf]
    (let [nv (volatile! n)]
      (fn
        ([] (rf))
        ([result] (rf result))
        ([result input]
         (let [n @nv]
           (vswap! nv dec)
           (if (pos? n)
             result
             (rf result input)))))))))
```

Wrap Java

```
1 (import 'com.clearspring.analytics.stream.cardinality.HyperLogLogPlus)
2
3 (defn cardinality
4   ([] (HyperLogLogPlus. 14 25))
5   ([^HyperLogLogPlus acc] (.cardinality acc))
6   ([^HyperLogLogPlus acc x]
7    (.offer acc x)
8    acc))
```

CollReduce protocol

- Get the next element
- Makes transducing data structure-agnostic allowing us to (re)use transducers for things such as clojure.async channels

Transducing an async channel

```
(let [ch (async/chan 1)]
  (async/onto-chan ch (range 10))
  (async/<!! (async/transduce identity + 0 ch)))
```

```
(let [ch (async/chan 1 my-count)]
  (async/onto-chan ch (range 10))
  (async/<!! (async/into □ ch)))
```

Composing transducers

1. `comp` transducers

```
(transduce (comp (filter odd?)
                 (map inc))
           conj
           (range 10))
```

2. Reducing function and transducer

```
((remove nil?) +)
```

3. github.com/henrygarner/redux

`post-complete`

```
(transduce identity
           (redux/post-complete
            +
            (fn [sum]
              (Math/sqrt sum)))
           (range 10))
```

`fuse`

```
(transduce identity
           (redux/fuse {:sum  +
                        :conj conj})
           (range 10))
```

Data structure that can be manipulated like any other

On-line/streaming analysis

Metabase ❤️

github.com/metabase/metabase

- Open source analytics tool (runs on-premises)
- Building a “data scientist in a box”
- Hundreds to billions of rows
- Some DBs optimised for analytics, some not

Many batch algorithms can be turned into online ones

$$\bar{x} = \frac{1}{n}\left(\sum_{i=1}^{n} x_i\right) = \frac{x_1 + x_2 + \cdots + x_n}{n}$$

$$\sigma^2 = \frac{\sum_{i=1}^{N} (X - \mu)^2}{N}$$

```clojure
(defn mean
  ([] [0 0])
  ([[s c :as acc] e]
   [(+ s e) (inc c)])
  ([[s c]]
   (when-not (zero? c)
     (/ s c))))
```

```clojure
(defn variance
  ([] [0 0 0])
  ([[c m ss :as acc] e]
   (let [c' (inc c)
         m' (+ m (/ (- e m) c'))]
     [c' m' (+ ss (* (- e m') (- e m)))]))
  ([[c m ss]]
   (when-not (zero? c)
     (let [c' (dec c)]
       (if (pos? c')
         (/ ss c') 0)))))
```

Parallelize independent computations

Find a recursive relation

github.com/MastodonC/kixi.stats

- Count
- (Arithmetic) mean
- Geometric mean
- Harmonic mean
- Median
- Variance
- Interquartile range
- Standard deviation
- Standard error
- Skewness
- Kurtosis
- Covariance
- Covariance matrix
- Correlation
- Correlation matrix
- Simple linear regression
- Standard error of the mean
- Standard error of the estimate
- Standard error of the prediction
- …

Single-pass analysis

```
(require '[redux.core :as redux])
(require '[kixi.stats.core :as stats])

(transduce identity
           (redux/post-complete
            (redux/fuse {:sum   +
                         :mean  stats/mean
                         :sd    stats/standard-deviation
                         :count stats/count
                         :zeros ((filter zero?) stats/count)})
            (fn [{:keys [mean sd] :as summary}]
              (-> summary
                  (assoc :rsd (when-not (zero? mean)
                                (/ sd mean))))))
           (range 10))
```

Data = code

Using transducers is worth it for the composition alone

Annoyances

- Can only transduce one coll at a time
- Always have to pass in an xf (especially annoying when using redux)
- Having functions that return a transducer or not is error prone
- Inconsistent support for transducers in core library

Questions

simon@metabase.com
@sbelak