Stream Fusion to Perfection

O. Kiselyov[1], A. Biboudis[2], N. Palladinos[3], Y. Smaragdakis[2]
Tohoku University[1]   University of Athens[2]   Nessos IT[3]

(under review)

Aggelos Biboudis
International Summer School on Metaprogramming 2016
Monday 8/8/2016

Stream Fusion to Perfection

O. Kiselyov, A. Biboudis, N. Palladinos, Y. Smaragdakis

- stream fusion is still an open (and super interesting) area for investigation
    - Duncan Coutts et al. (Stream Fusion 2007) (a pull-based approach)
    - Andrew Farmer et al. (Hermit in the Stream 2014) perform stream fusion (concatMap included {use Hermit not just GHC RULES})
    - “Implement list fusion using streams instead of foldr/build” (ticket opened 9 years ago, “*we close this ticket as requiring more research*”)

Stream Fusion to Perfection

O. Kiselyov, A. Biboudis, N. Palladinos, Y. Smaragdakis

- the ultimate challenge

  ✓ design a *library* …

  ✓ … that supports *many* and *complex combinations of operators* …

  ✓ … and generates *loop-based*, *fused* code with *zero allocations*

Stream Fusion to Perfection

O. Kiselyov, A. Biboudis, N. Palladinos, Y. Smaragdakis

```
of_arr .⟨arr⟩.
  ▷ map (fun x → .⟨~x * ~x⟩.)
  ▷ sum
```

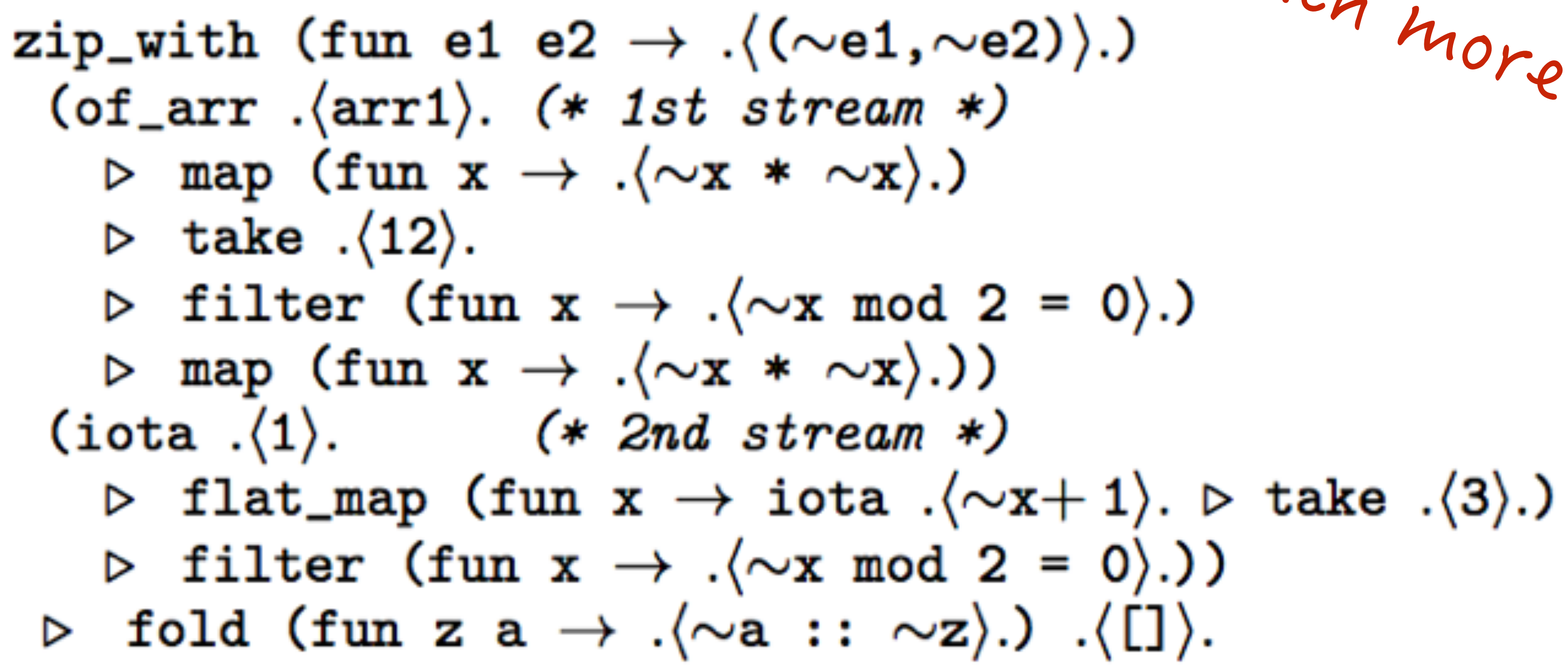
staging $\downarrow$

```
let s_1 = ref 0 in
let arr_2 = arr in
 for i_3 = 0 to Array.length arr_2 -1 do
   let el_4 = arr_2.(i_3) in
   let t_5 = el_4 * el_4 in
   s_1 := t_5 + !s_1
 done;
!s_1
```

Stream Fusion to Perfection

O. Kiselyov, A. Biboudis, N. Palladinos, Y. Smaragdakis

and much more complex...

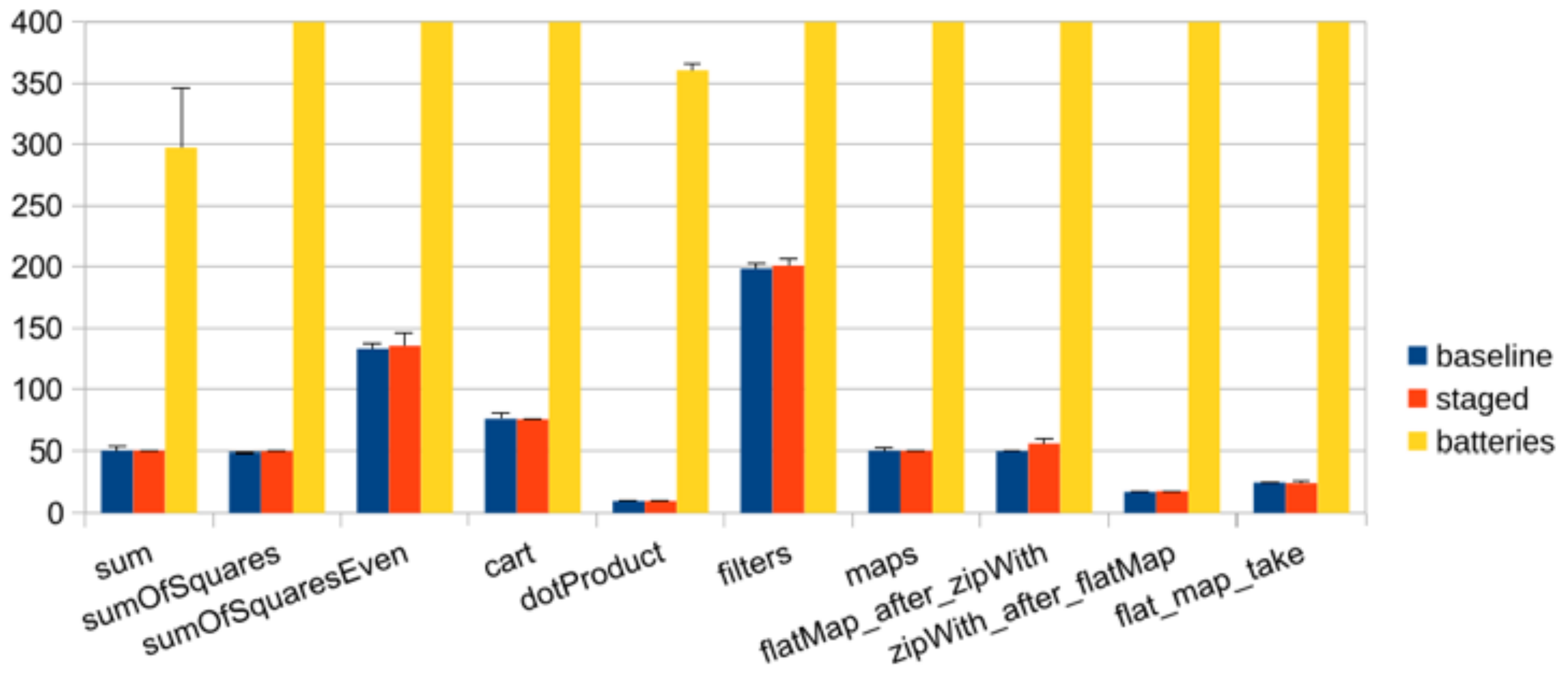


```
zip_with (fun e1 e2 → .⟨(~e1,~e2)⟩.)
 (of_arr .⟨arr1⟩. (* 1st stream *)
   ▷ map (fun x → .⟨~x * ~x⟩.)
   ▷ take .⟨12⟩.
   ▷ filter (fun x → .⟨~x mod 2 = 0⟩.)
   ▷ map (fun x → .⟨~x * ~x⟩.))
 (iota .⟨1⟩.        (* 2nd stream *)
   ▷ flat_map (fun x → iota .⟨~x+1⟩. ▷ take .⟨3⟩.)
   ▷ filter (fun x → .⟨~x mod 2 = 0⟩.))
 ▷ fold (fun z a → .⟨~a :: ~z⟩.) .⟨[]⟩.
```

Stream Fusion to Perfection

O. Kiselyov, A. Biboudis, N. Palladinos, Y. Smaragdakis

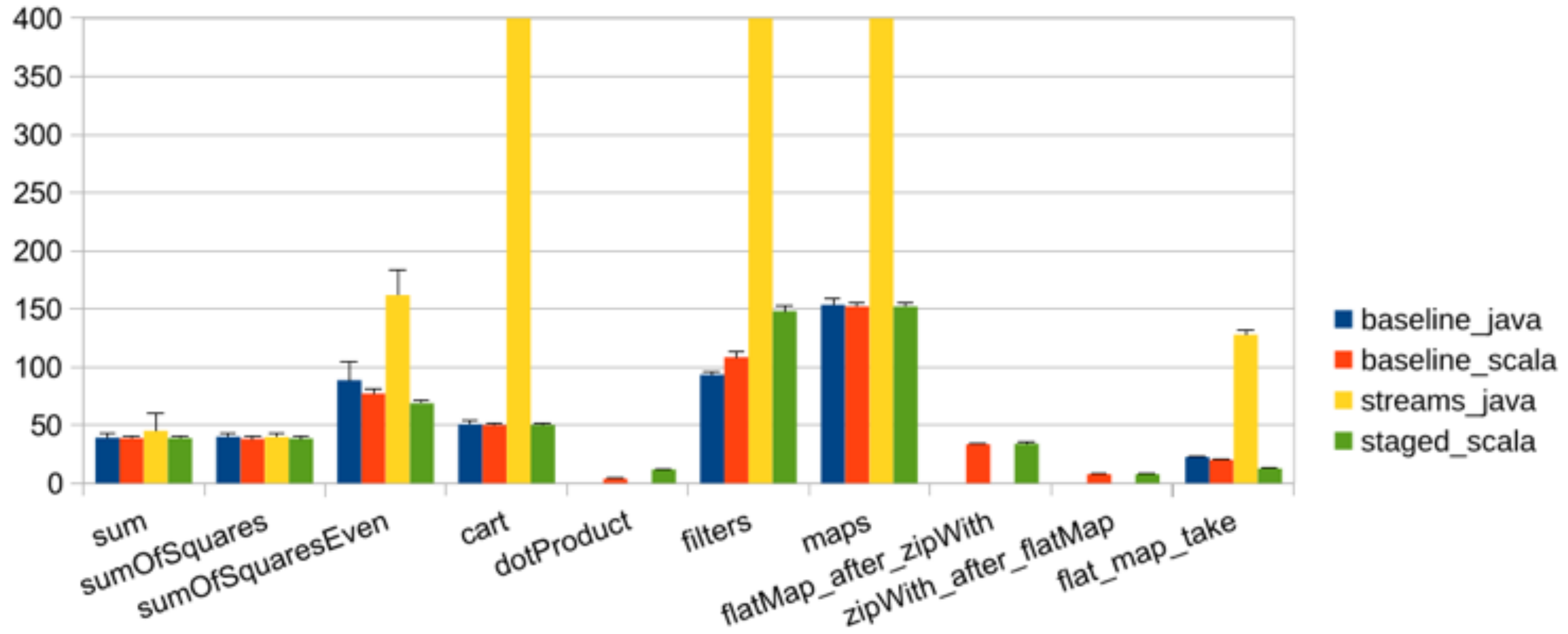
- MetaOCaml



Stream Fusion to Perfection

O. Kiselyov, A. Biboudis, N. Palladinos, Y. Smaragdakis

- Scala with Lightweight Modular Staging (LMS) (also compared to Java streams)


sum
sumOfSquares
sumOfSquaresEven
cart
dotProduct
filters
maps
flatMap_after_zipWith
zipWith_after_flatMap
flat_map_take
🟦 baseline_java
🟥 baseline_scala
🟨 streams_java
🟩 staged_scala