Zoology of lockfree concurrent data structures

Clément Allain

INRIA Paris

June 16, 2024

Zoology

Specimen ①: Michael-Scott queue

Specimen 2: KCAS

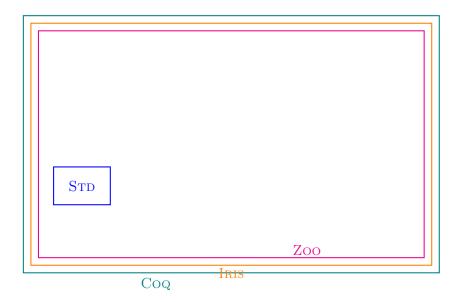
```
HEAPLANG (modified)
+ ADTs
+ DLS
+ exceptions
+ algebraic effects
+ relaxed memory
(planned before Osiris,
in case you were wondering)
                      Zoo
```

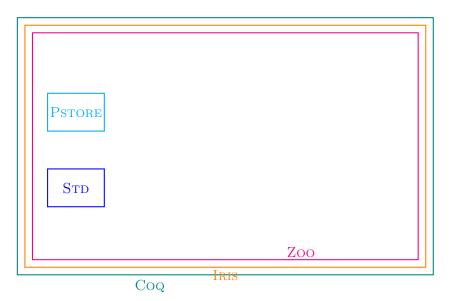
Coq

```
MetaCoq Run (zoo_variant "list" [
  "Nil" ;
  "Cons"
1).
Definition map : val :=
  rec: "map" "fn" "xs" :=
    match: "xs" with
    | Nil =>
        §Nil
    | Cons "x" "xs" =>
        let: "y" := "fn" "x" in
        'Cons{ "y", "map" "fn" "xs" }
    end.
```

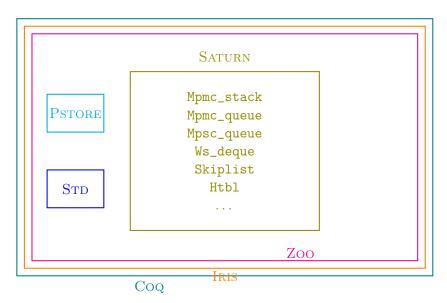
```
HEAPLANG (modified)
+ ADTs
+ DLS
+ exceptions
+ algebraic effects
+ relaxed memory
(planned before Osiris,
in case you were wondering)
                      Zoo
```

Coq

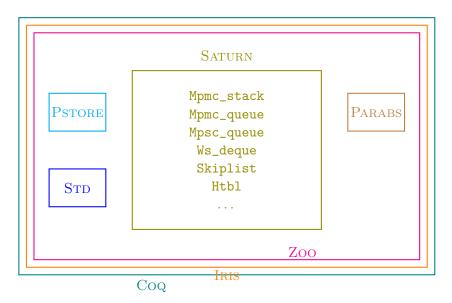


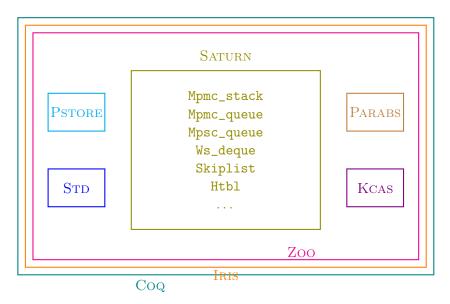


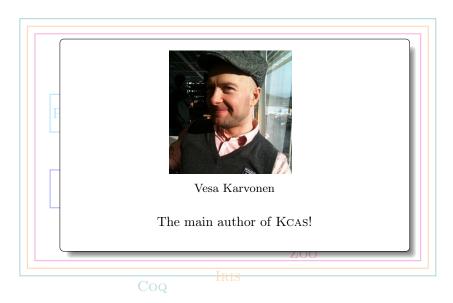


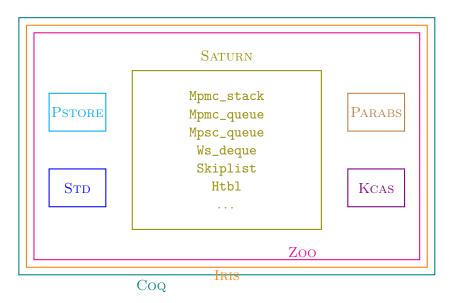












Zoology: why is it fun?

Lockfree algorithms typically exhibit complex behaviors:

- \triangleright physical state \neq logical state,
- external linearization points,
- ▶ future-dependent linearization points.

IRIS is a good match for verifying them thanks to advanced mechanisms:

- invariants to enforce protocols,
- atomic updates to materialize linearization points,
- prophecy variables to reason about the future.

Zoology

Specimen ① : Michael-Scott queue

Specimen 2: KCAS

Michael-Scott queue

TODO

Zoology

Specimen $ext{ } ext{ } ex$

Specimen ②: KCAS

KCAS

TODO

Thank you for your attention!