

EECS498-008

Formal Verification

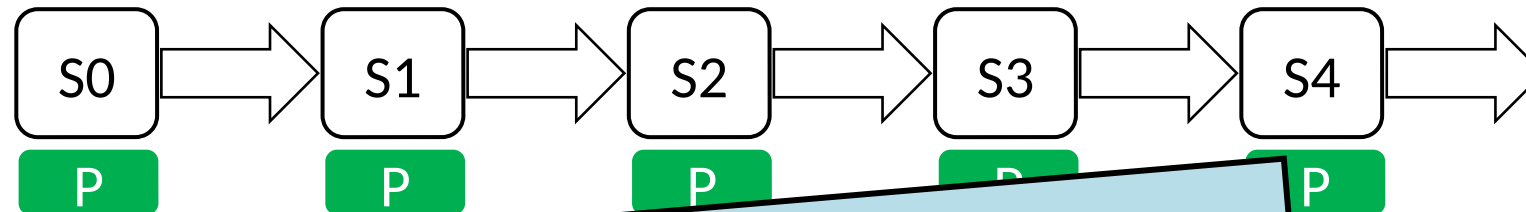
of Systems Software

Material and slides created by
Jon Howell and Manos Kapritsos

Inductive invariants

Safety property (a.k.a.
invariant):

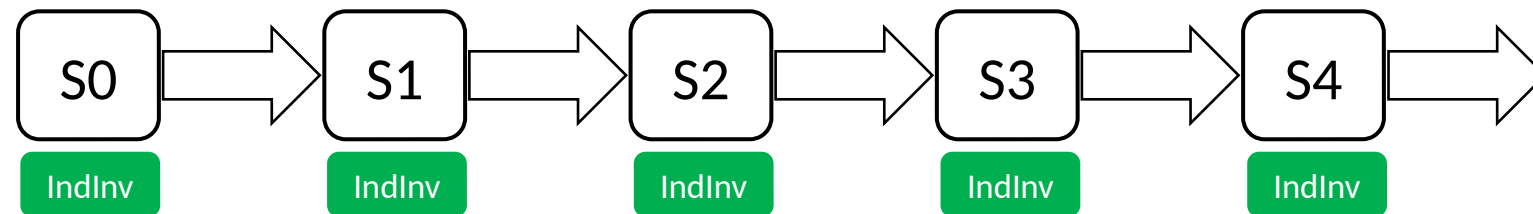
a property that **always** holds



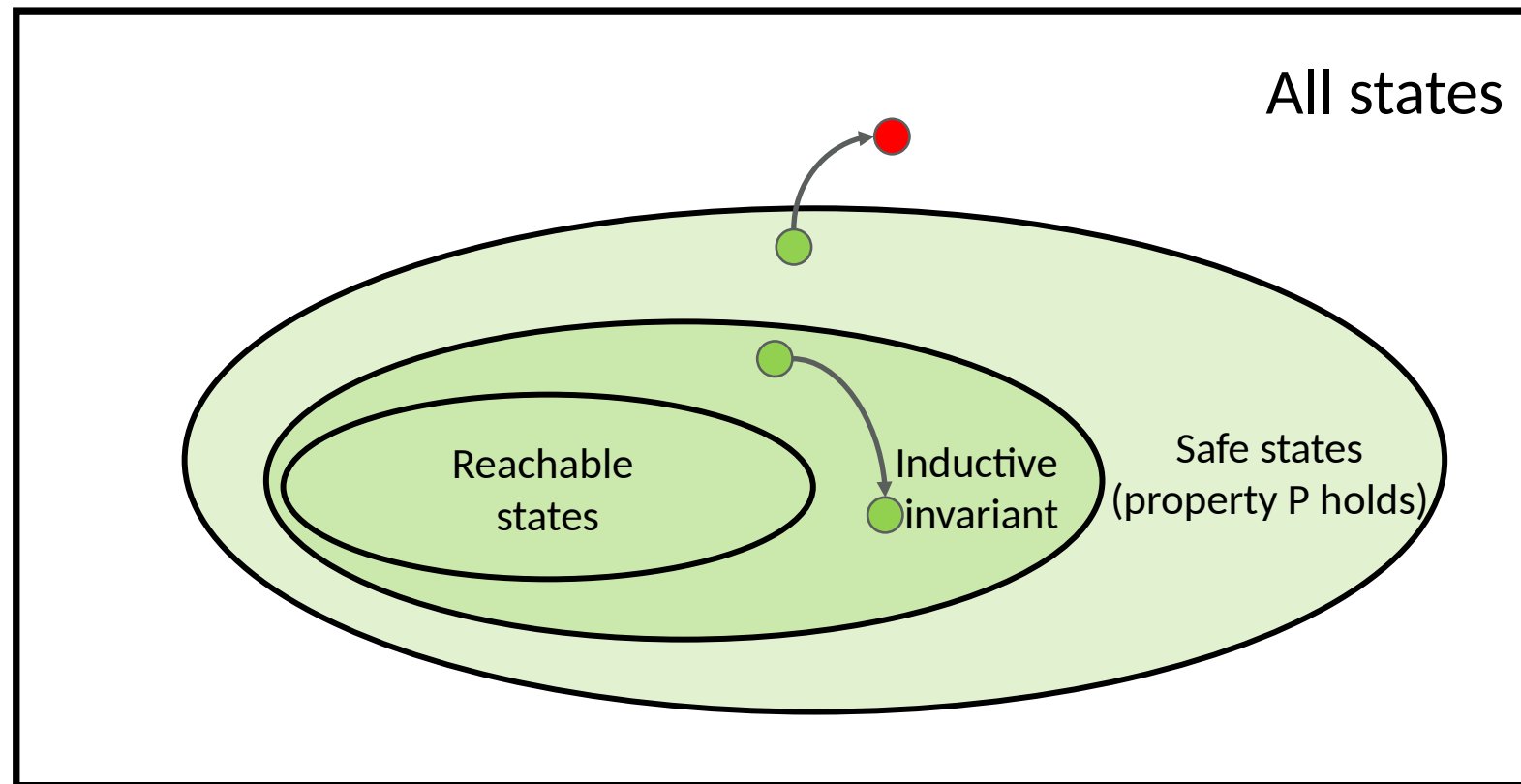
The problem:
Property P may **not** be inductive!

$$P(v) \ \&\& \ \text{Next}(v, v') \\ \implies P(v')$$

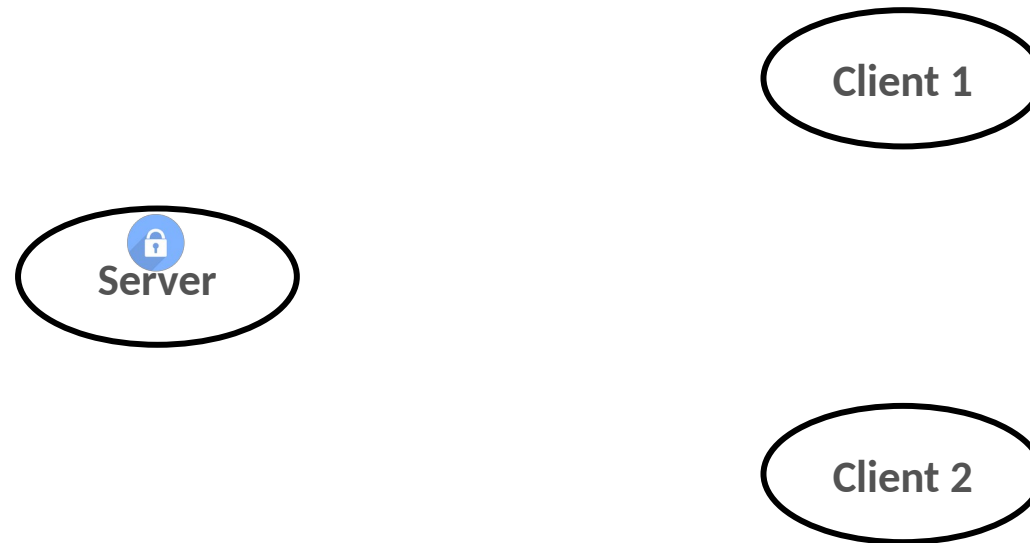
Proving safety with inductive invariants

$$\text{IndInv}(v) \implies \text{Safety}(v)$$
$$\text{Init}(v) \implies \text{IndInv}(v)$$
$$\text{IndInv}(v) \ \&\& \ \text{Next}(v, v') \implies \text{IndInv}(v')$$


Invariants vs Inductive invariants



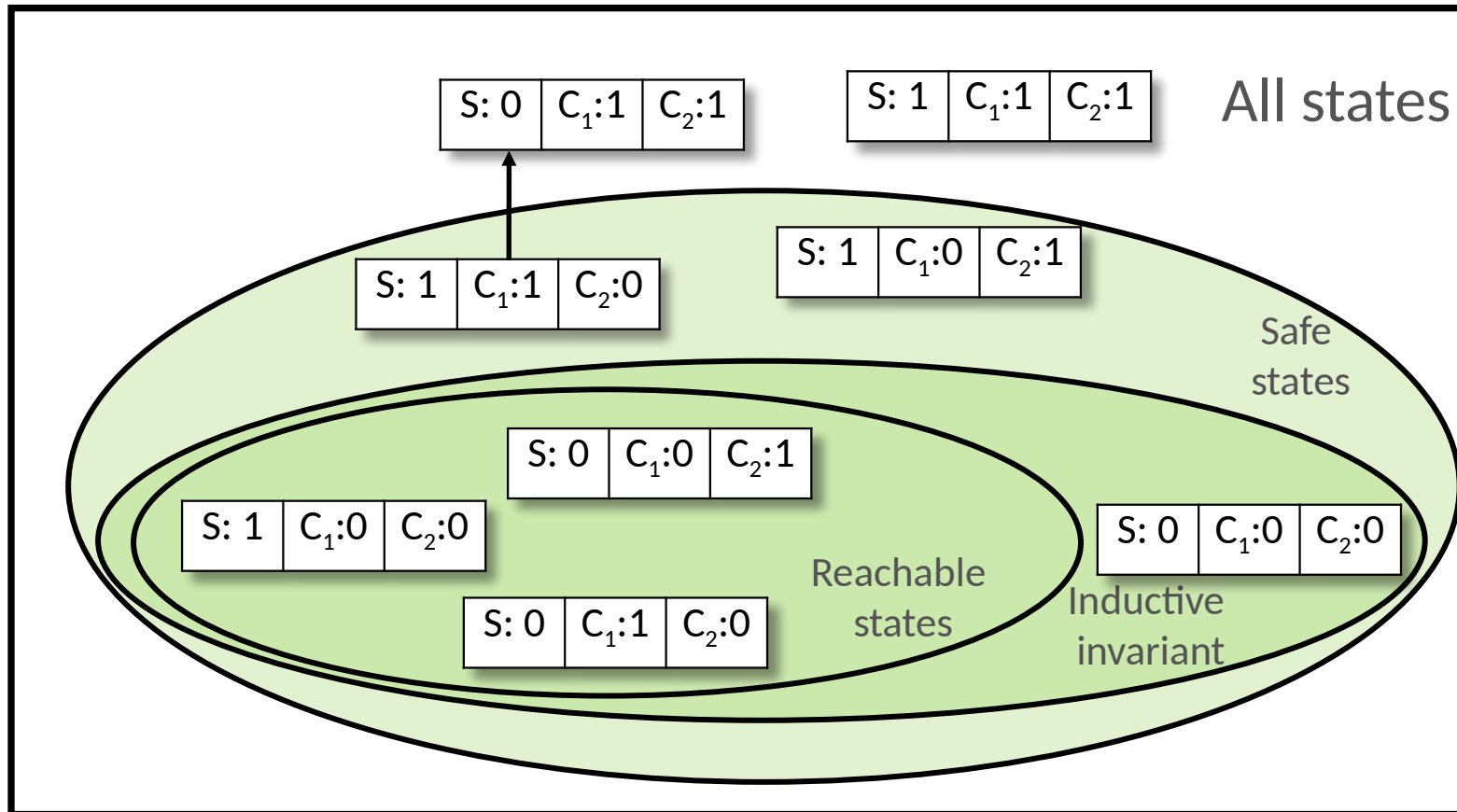
Example: lock server



Safety property: $\neg(C1 \wedge C2)$

Both clients cannot hold the lock at the same time

Example: lock server



Some useful boilerplate

```
datatype Constants = Constants(capacity:int)
```

```
datatype Variables = Variables(numCokes:int)
```

```
predicate Init(c:Constants, v:Variables) { ... }
```

```
predicate Next(c:Constants, v:Variables, v':Variables) { ... }
```

Some useful boilerplate

```

datatype Constants = Constants(tableSize:nat)
{
  predicate WellFormed() {
    && 0 < tableSize
  }
}

datatype Variables = Variables()
{
  predicate WellFormed(c:
    Constants) {
    && c.WellFormed()
  }
}

```

Typical examples:

- Length constraints on sequences
- Indices fit into a sequence length
- Domains of maps

Non-linear arithmetic

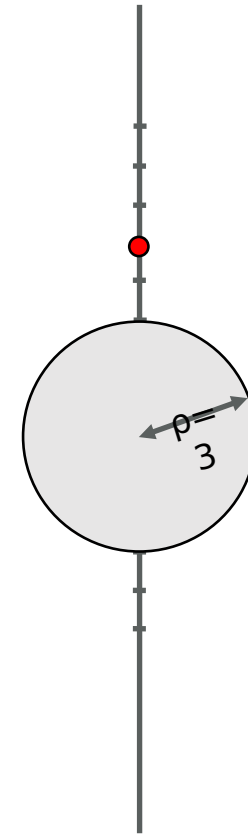
Dafny runs without non-linear reasoning by default

Beware of modulo operations

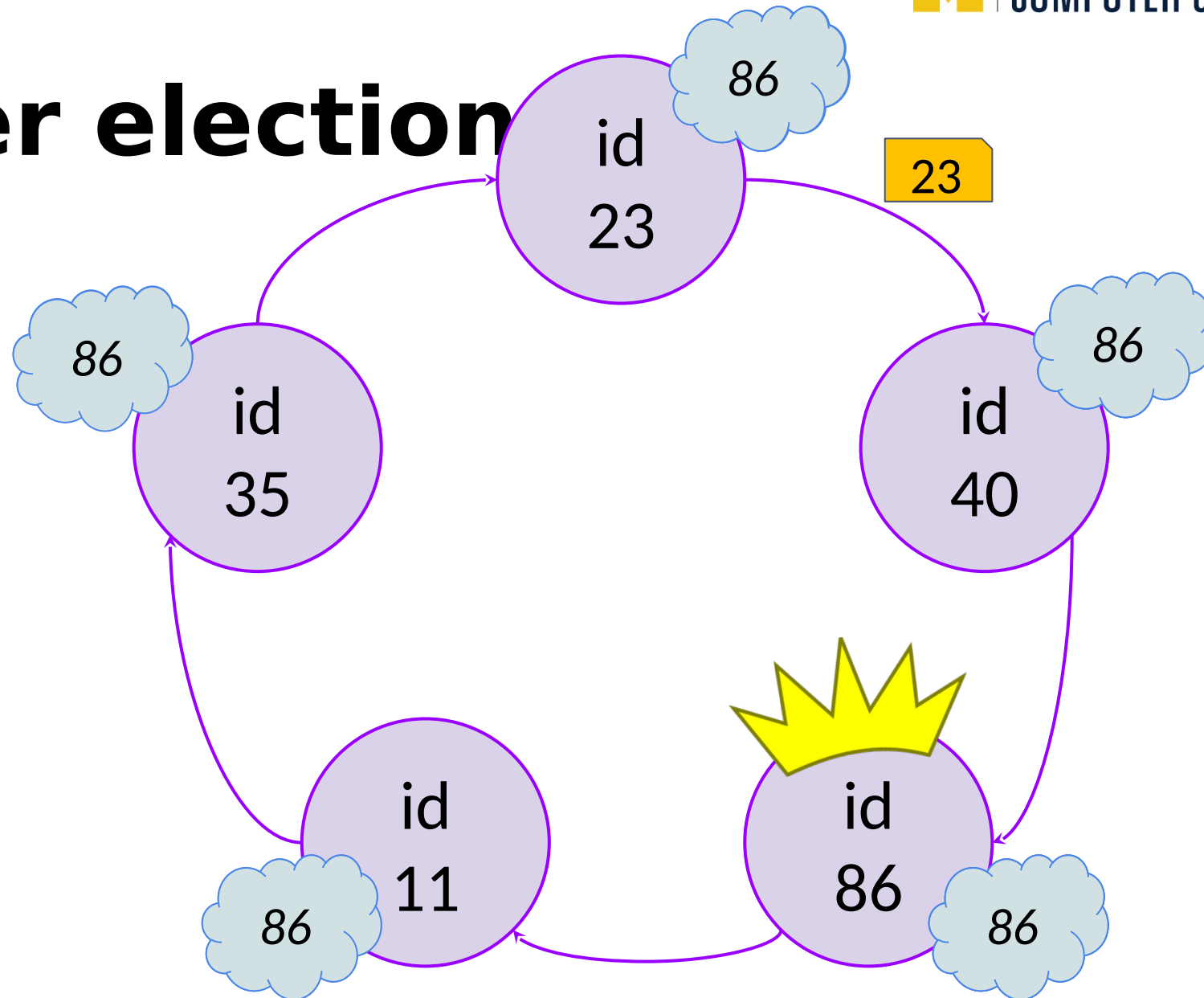
- Think of alternatives, if you run into trouble

Crawler 2: Revenge of the inductive invariant

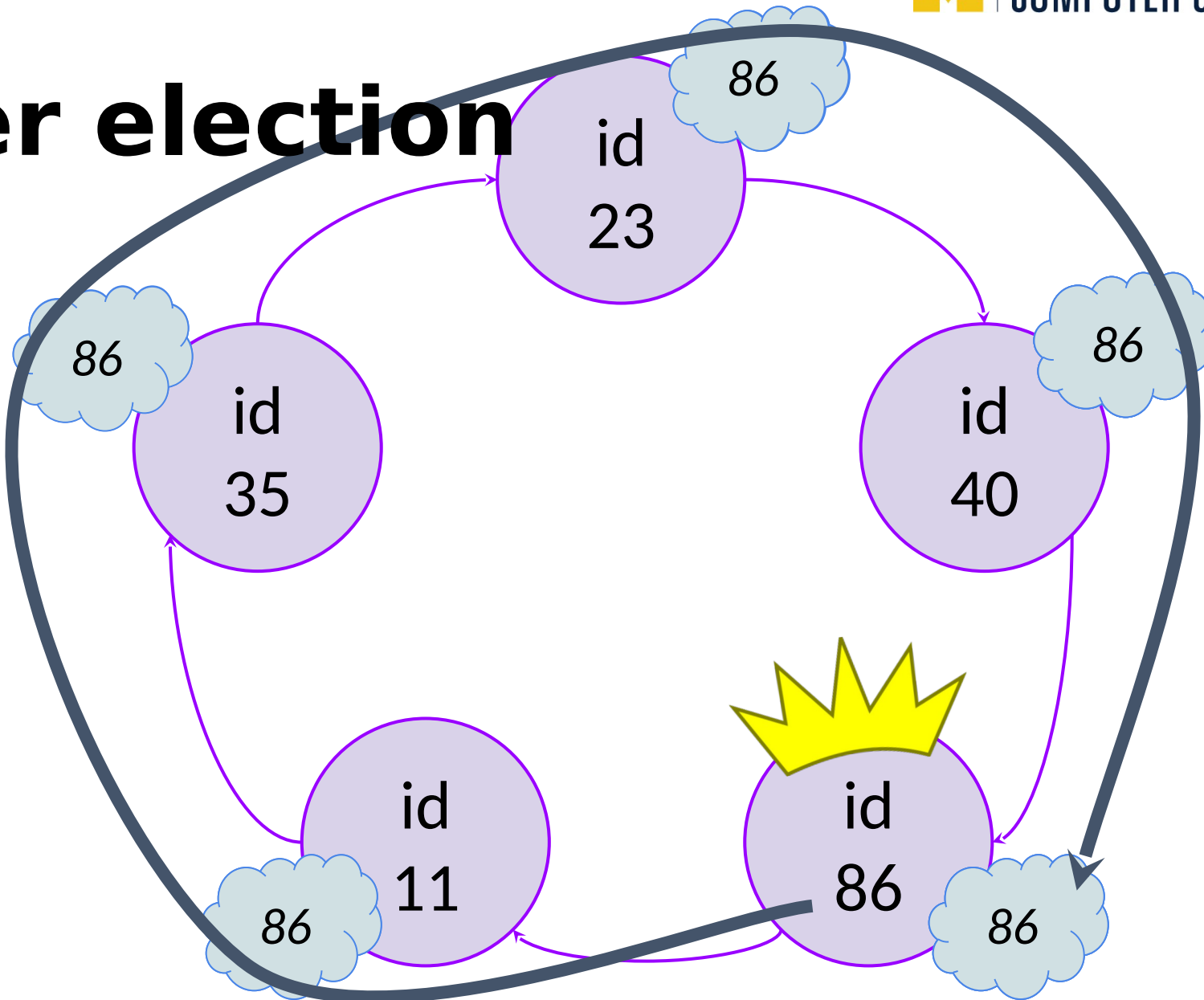
- The crawler can now only move North/South
 - Initially it can only move North
- It can also Flip(), teleporting to the symmetric point on the y-axis and changing direction of movement



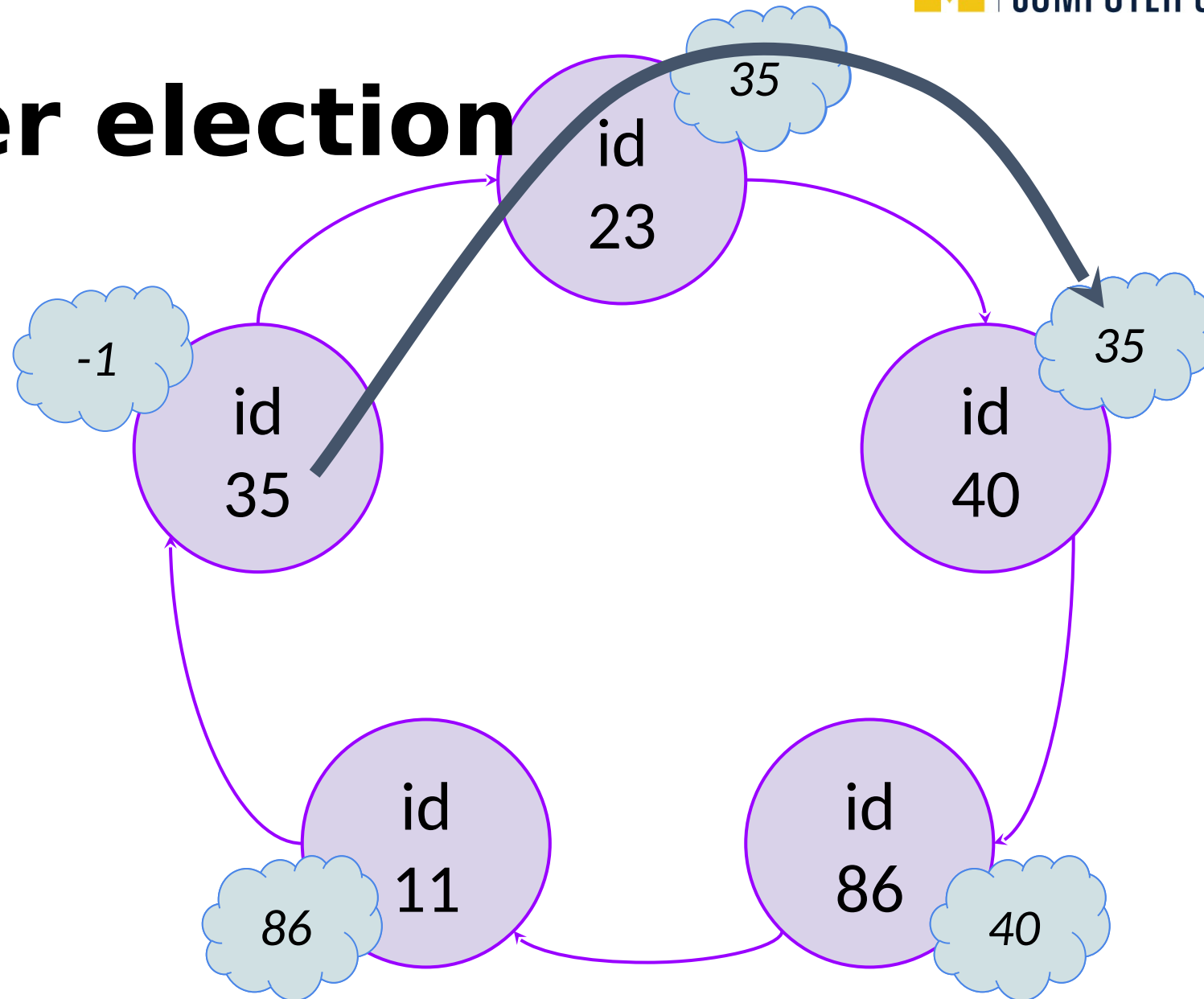
Leader election



Leader election



Leader election



Leader election

