# EDAA40

# Discrete Structures in Computer Science

## A few words on

# Amoebas – a Game of Clones

Jörn W. Janneck, Dept. of Computer Science, Lund University

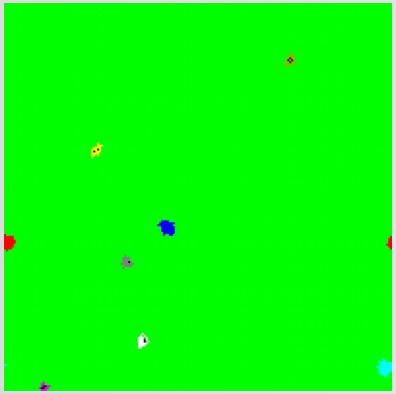
# some things to keep in mind...

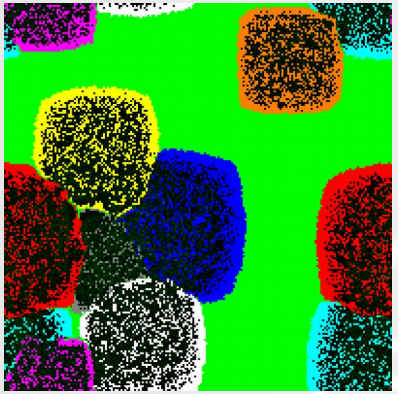
1. The objective of the entire exercise is to get you to tinker with Clojure. That's the idea behind a programming task that has no clear "best" solution. The point is to come up with ideas, translate them into Clojure, evaluate their effects experimentally. Repeat.

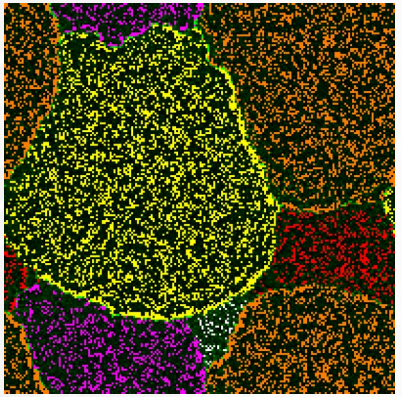
2. This is the first time we do this. Things are bound to blow up here and there.

# about the amoeba world

amoebas live in a 200x200 cell world that wraps horizontally and vertically





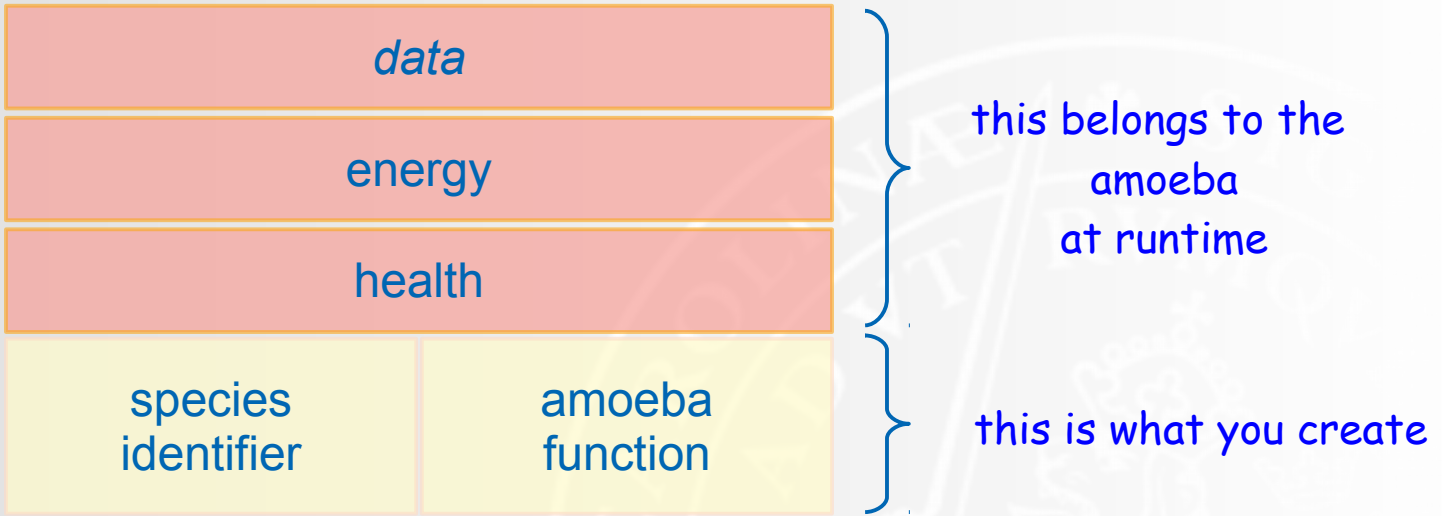


simulation happens in rounds – every amoeba gets to take one action:

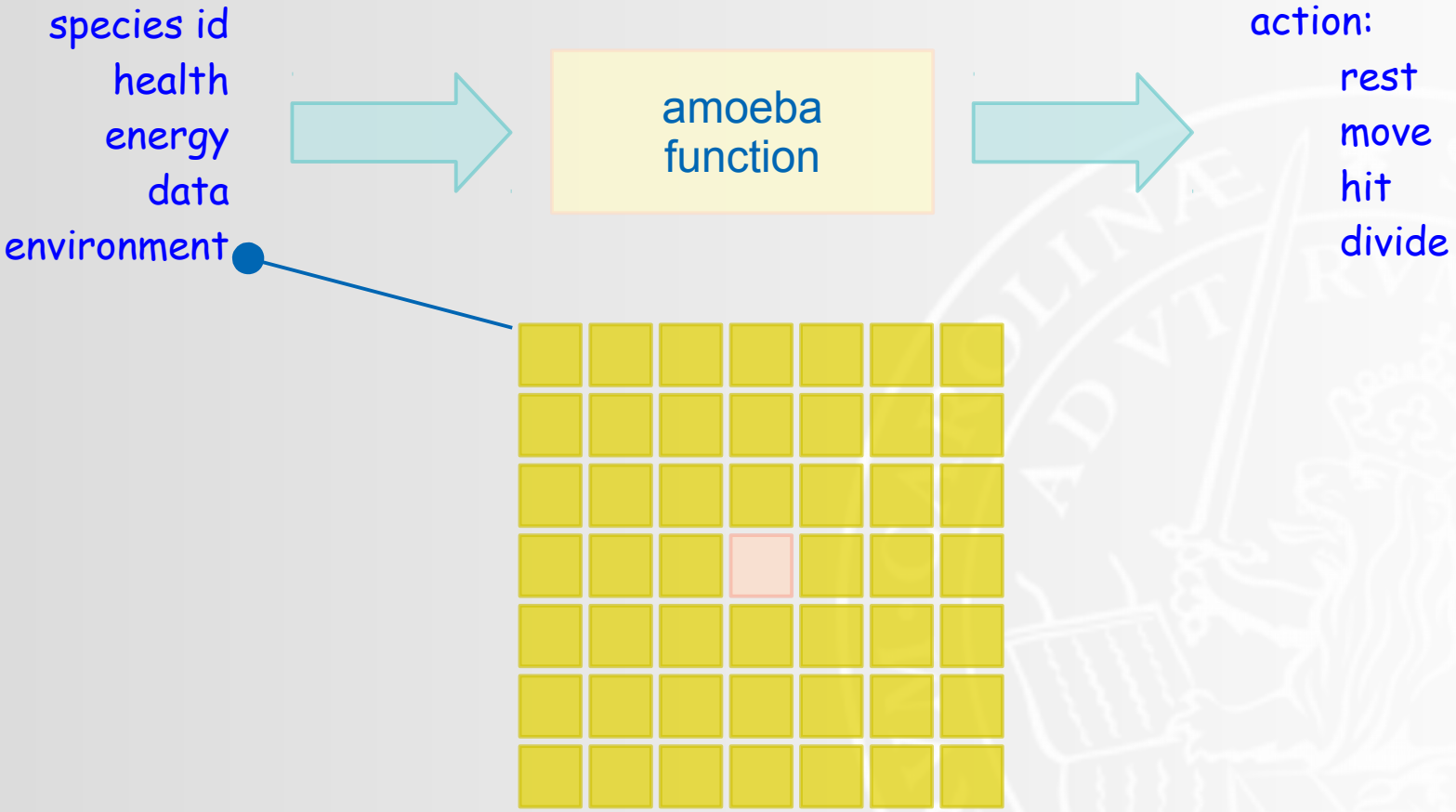
rest - move - hit - divide

every action requires energy, resting consumes fuel in the local cell (and heals)
after each round, the sun shines on empty cells and replenishes the fuel

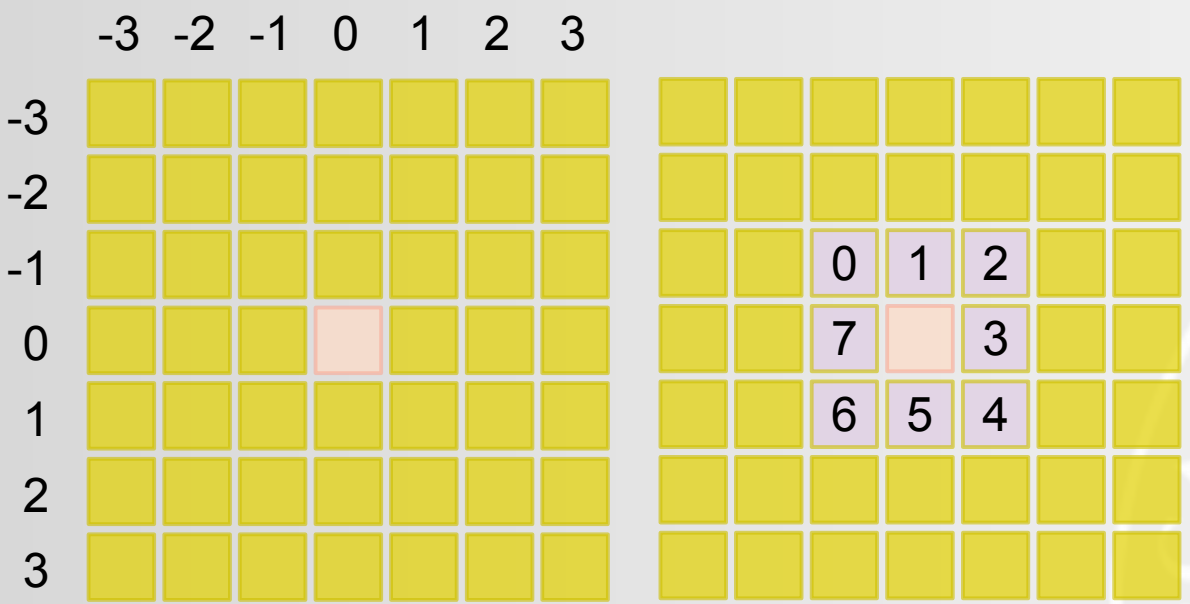
# parts of an amoeba



# amoeba functions



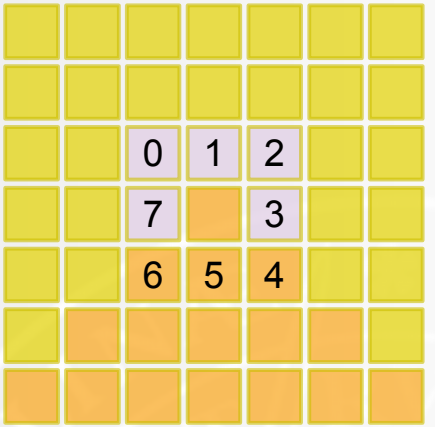
# environments, neighbors, sections



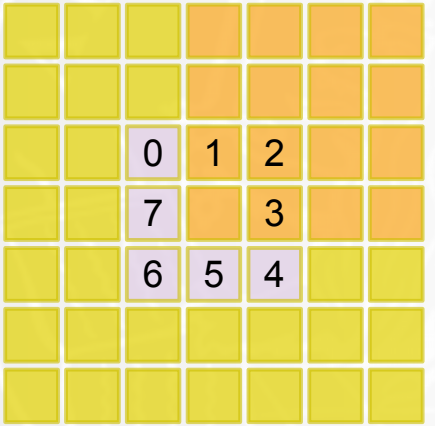

environment
(with coordinate system)




neighbors
(with numbering)




section 5




section 2

# some amoebas in action...

```
(defn create-mindless-divider
    [division-prob]
    (fn [energy health species env data]

        (if (< energy (+ MinDivideEnergy (/ (- MaxAmoebaEnergy MinDivideEnergy) 2)))
            {:cmd :rest}
            (if (< (rand) division-prob)
                {:cmd :divide, :dir (rand-int 8)}
                {:cmd :move, :dir (rand-int 8)}
            )
        )
    )
)
```

# some amoebas in action...

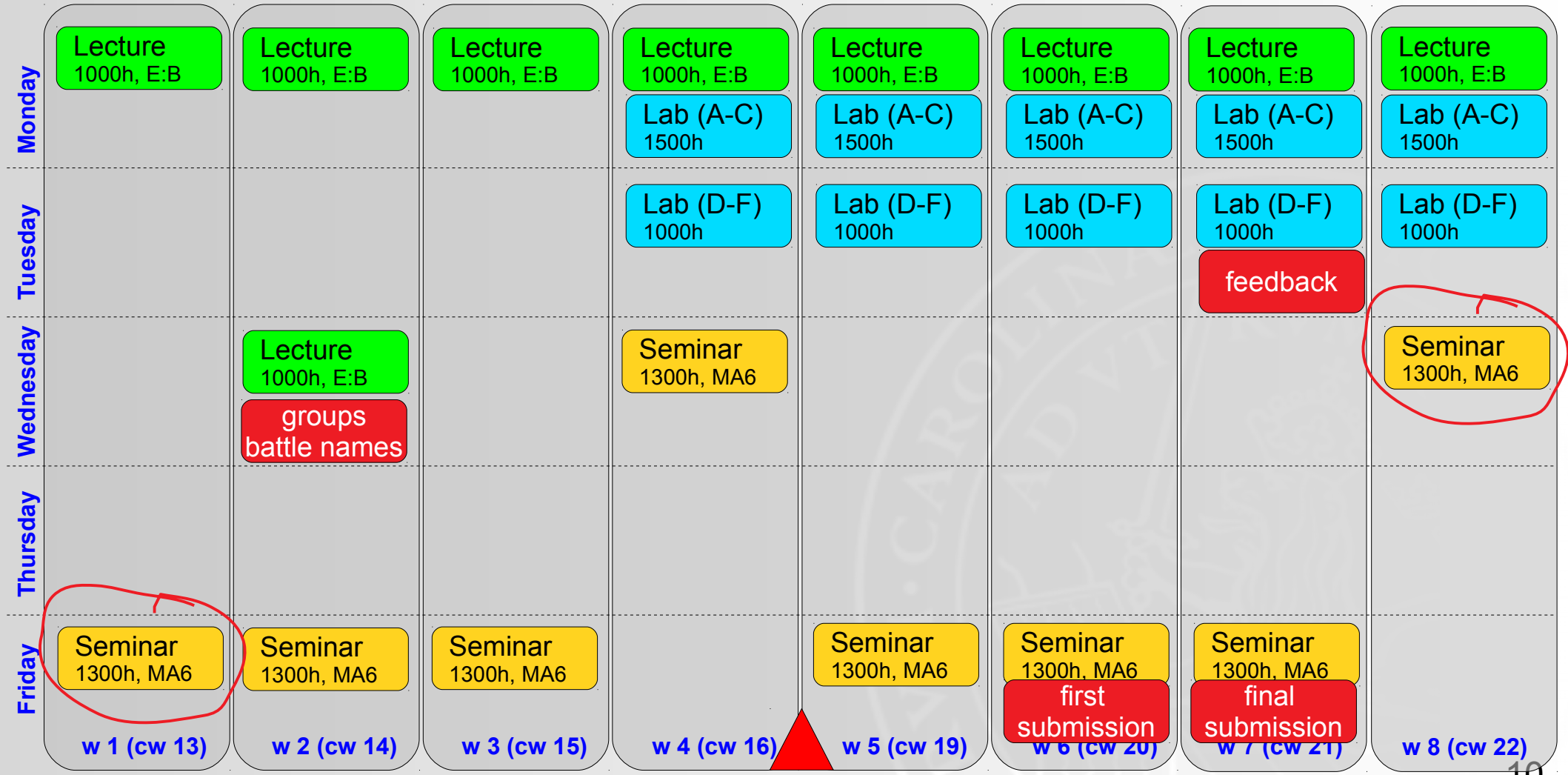
```
(defn create-nasty
    [division-prob select-target]

    (let
        [md (create-mindless-divider division-prob)]

        (fn [energy health species env data]

            (let
                [ hs     (hostiles species Neighbors env) ]

                (if (empty? hs)
                    (md energy health species env data)
                    {:cmd :hit :dir (Neighbor-To-Dir (select-target hs species env))}
                )
            )
        )
    )
)
```

# some amoebas in action...

```
(defn create-slightlybrainy
    [low-energy divide-energy select-target]

    (fn [energy health species env data]
        (let
            [
                do-move (fn []
                            (let                                        ;; otherwise we gotta move...
                                [
                                    empty-nb     (empty-neighbors env)
                                    by-fuel      (sections-by-fuel empty-nb env)
                                ]

                                (if (empty? empty-nb)
                                    {:cmd :rest}            ;; hunker down, we can't move --- FIXME: perhaps we should hit someone?
                                    {:cmd :move :dir (last by-fuel)}    ;; move toward the most fuel
                                )
                            )
                        )
                do-fuel (fn []
                            (if (< MaxFuelingEnergy (:fuel (env Here)))     ;; are we *at* a McDonald's?
                                {:cmd :rest}                                ;; chomp chomp
                                (do-move)
                            )
                        )
                do-hit  (fn []
                            (let
                                [h  (hostiles species Neighbors env)]

                                (if (empty? h)      ;; nobody to hit?
                                    (do-fuel)       ;; eat
                                    {:cmd :hit :dir (Neighbor-To-Dir (select-target hs species env))}
                                )
                            )
                        )
                do-div  (fn [empty-nb]
                            {:cmd :divide :dir (rand-nth empty-nb)}
                        )
            ]

            (cond
                (< energy low-energy)           ;; need some chow?
                    (do-fuel)
                (< divide-energy energy)                ;; parenthood!
                    (let
                        [empty-nb   (empty-neighbors env)]

                        (if (empty? empty-nb)       ;; nowhere to put that crib?
                            (do-hit)                ;; then screw parenthood, hit someone
                            (do-div empty-nb)       ;; oooh, look, it's... an amoeba :-(
                        )
                    )
                (hostiles species Neighbors env)            ;; someone looking at us funny?
                    (do-hit)                    ;; whack 'em
                :else
                    (do-fuel)                   ;; let's eat some more
            )
        )
    )
)
```

```
(defn most-energy-target-selector
    "picks a target with the highest amount of energy stored"
    [hs species env]

    (last (sort-by #(:energy (:occupant (env %))) hs))
)
```

# timeline


Monday
Tuesday
Wednesday
Thursday
Friday
Lecture 1000h, E:B
Lab (A-C) 1500h
Lab (D-F) 1000h
feedback
Seminar 1300h, MA6
groups battle names
first submission
final submission
w 1 (cw 13)
w 2 (cw 14)
w 3 (cw 15)
w 4 (cw 16)
w 5 (cw 19)
w 6 (cw 20)
w 7 (cw 21)
w 8 (cw 22)