Special Forms of Evolution

Lecture 13

MONASH UNIVERSITY CLAYTON'S SCHOOL OF INFORMATION TECHNOLOGY

Co-Evolution

Reading: Eiben& Smith Chapter 13

- Sewall-Wright's "fitness landscape" analogy is limiting in relation to biological evolution.
- The adaptive value of an organism is determined by its environmental niche (this includes other organisms)
- Co-adaptation: **mutualism**, **symbiosis** (mutual benefit, e.g., plants pollination by insects), **predation**, **parasitism** (antagonism, e.g., intestinal worms)
- Co-evolutionary EAs may use both **cooperation** and **competition**, with **single** and **multiple** species.

Co-operative Co-Evolution

- Multiple species, each representing a partial solution co-operate to find a total solution. Examples include high-dimensional function optimisation and job shop scheduling.
- User must be able to partition the problem into sub-problems which will be solved by different species / individuals.
- **Endosymbiosis** – where two species become so interdependent they become physically linked. (e.g., components of Eukaryote formation)

Co-operative Co-Evolution (cont.)

- Bull & Fogarty examined coevolving symbiotic systems with "linkage flags" to denote solutions from different populations that should stay together.
- Strategies depend on the inter-effect of each populations fitness-landscape, with linkage preferred in highly interdependent situations.
- How should a solution from one population be paired with others to gain a fitness evaluation?

Co-operative Co-Evolution (cont.)

- Options for fitness evaluation:
  - Generational GA in each subpopulation, with different species taking turns to undergo a round of selection, recombination and mutation. Evaluation performed using the current best from each of the other species. (Potter and DeJong).
  - Steady-state GGA: new individual undergoes 20 "encounters" with solutions selected from the other population. Fitness set as mean of these encounters (Paredis).
  - Husbands used a diffusion EA model, with one member on each grid point.
  - Bull compared pairing strategies: best, random, stochastic fitness-based, joined and distributed as per Husbands diffusion model. No one model was the best, however random is robust for generational GA, distributed did best for steady-state GA. "Best" is robust if used with fitness sharing (prevents premature convergence).

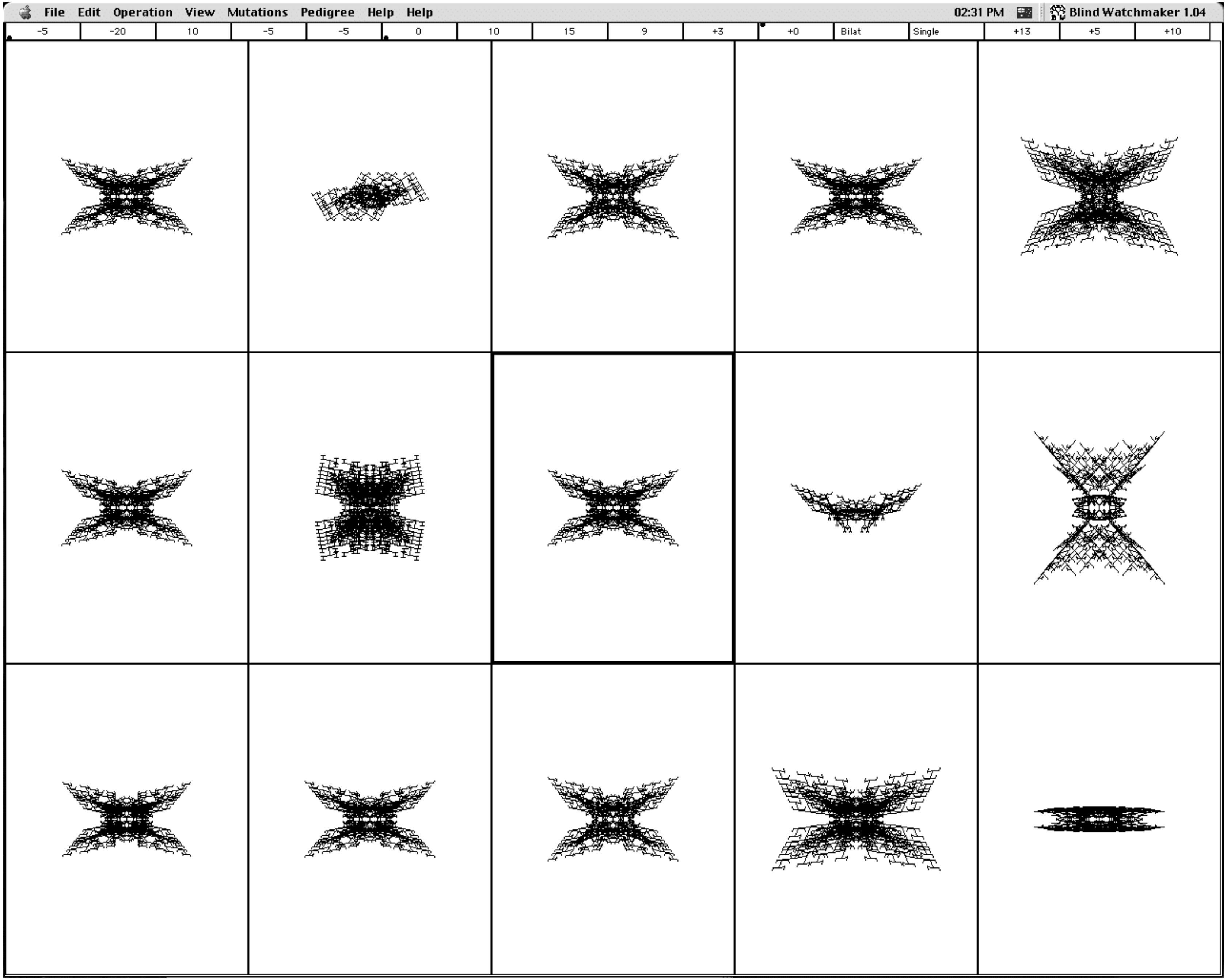
Competitive Co-Evolution

- Individuals compete against each other to gain fitness *at each other's expense*.
- Individual competition or species competition is possible.
- The classic experiment is Axelrod's *Iterated Prisoners Dilemma (IPD)*. A two-player game where each participant must decide to co-operate or defect at each iteration, the payoffs dependent on the decision of both players. A Payoff matrix determines the reward received.

Competitive Co-Evolution (cont.)

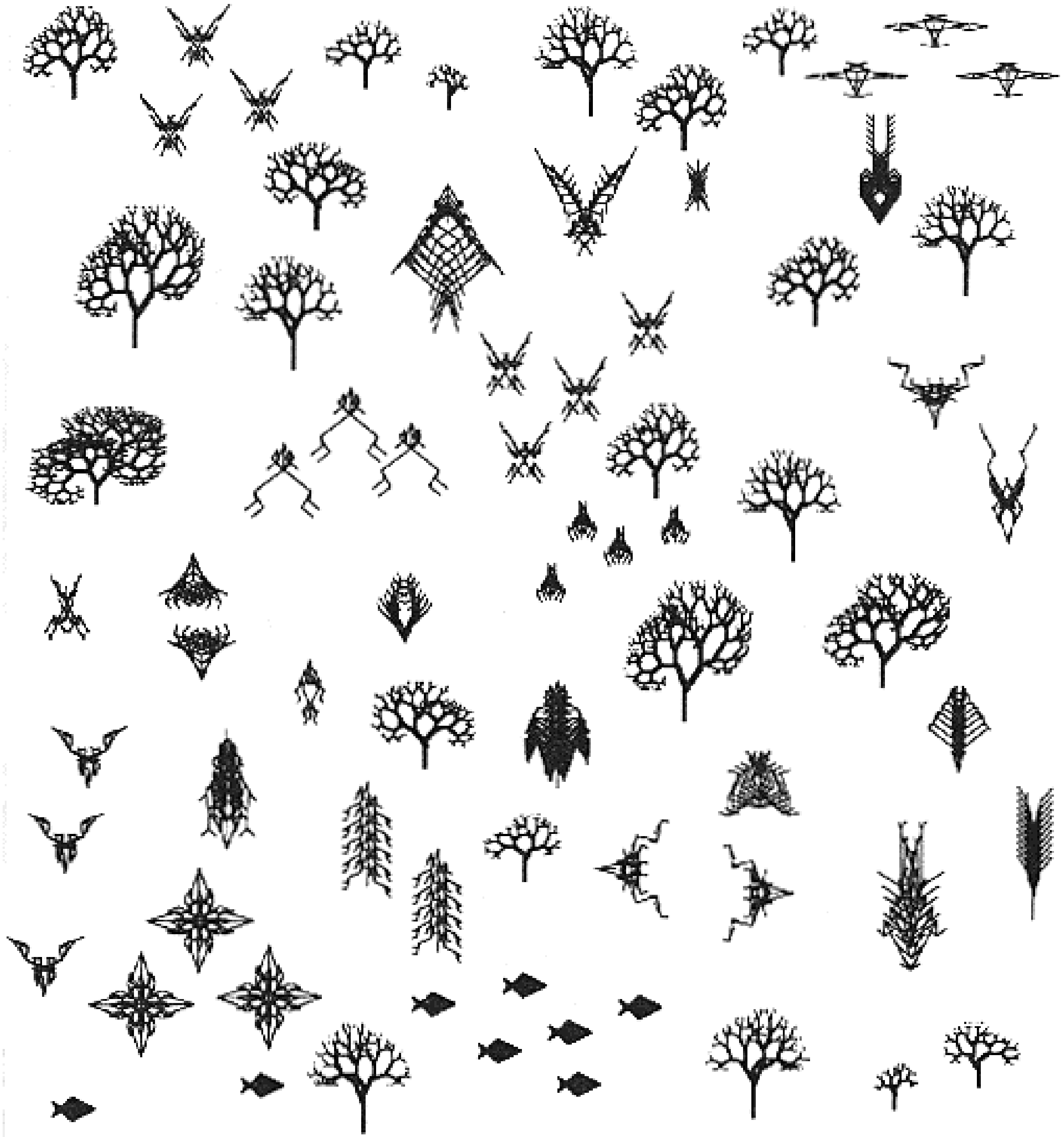
- Hillis used a two-species model with pairing strategy determined by colocation on a grid in a diffusion model EA to solve the Bachelor Sort problem where populations represented sorting networks (fitness is assigned based on each networks ability to sort a series of test cases)
- Music composition: have a population of composers and critics who co-evolve (Todd and Werner)
- As with co-operative evolution, fitness landscapes will change as the different populations evolve – pairing strategies effect observed behaviour. (different strategies for inter- and intra-population competition).

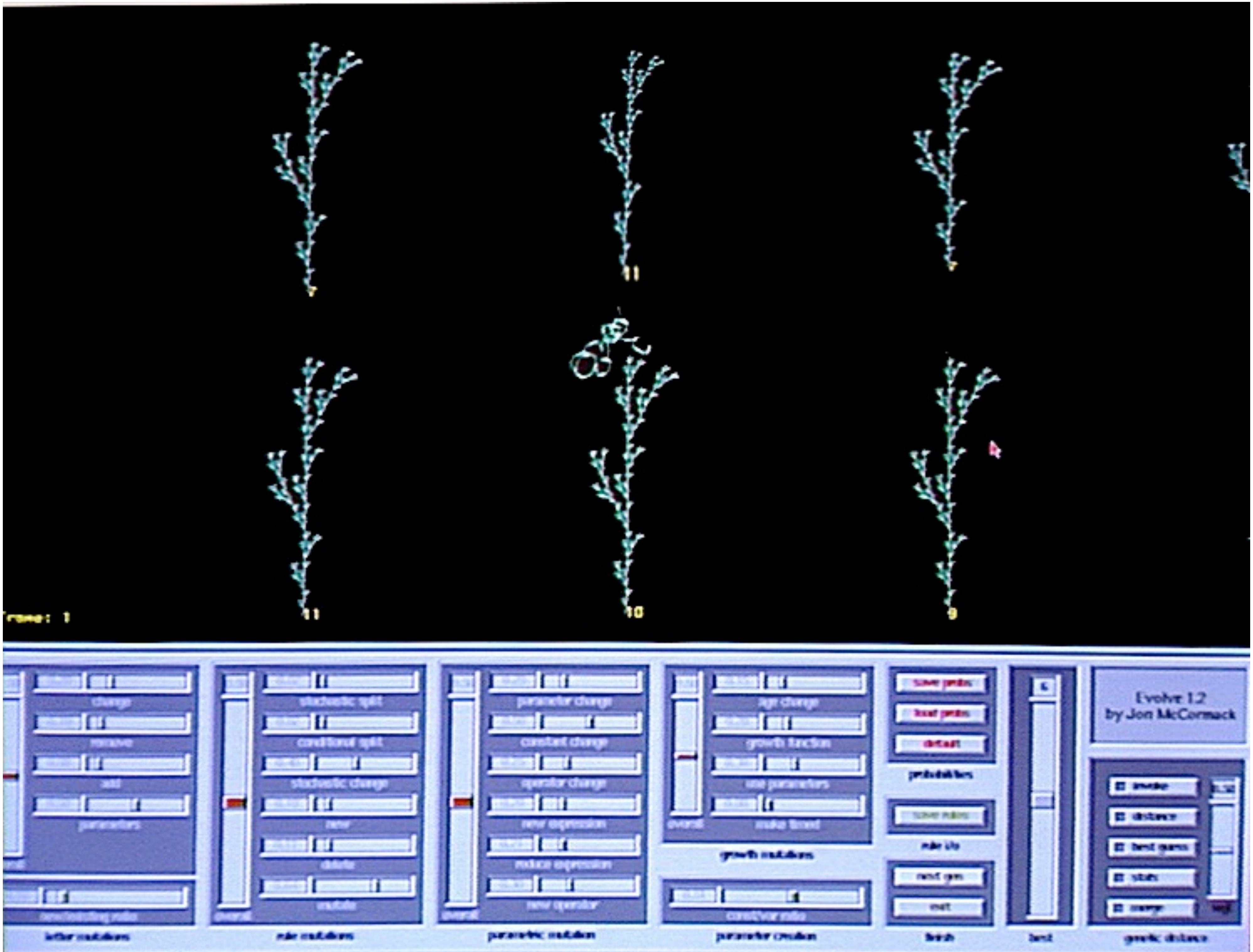
Interactive Evolution



Dawkins' *Biomorphs* (1986)

Interactive Evolution (cont.)



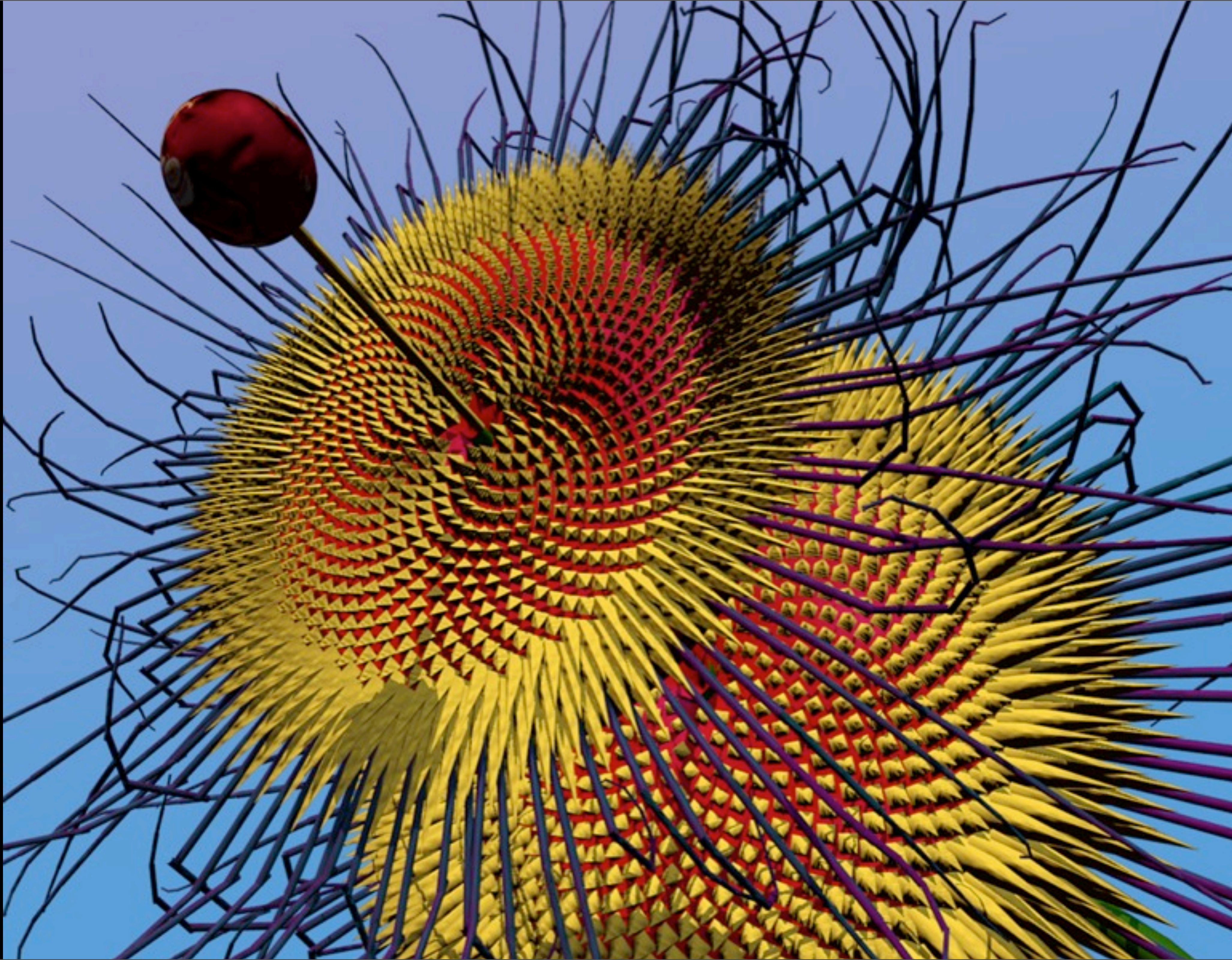


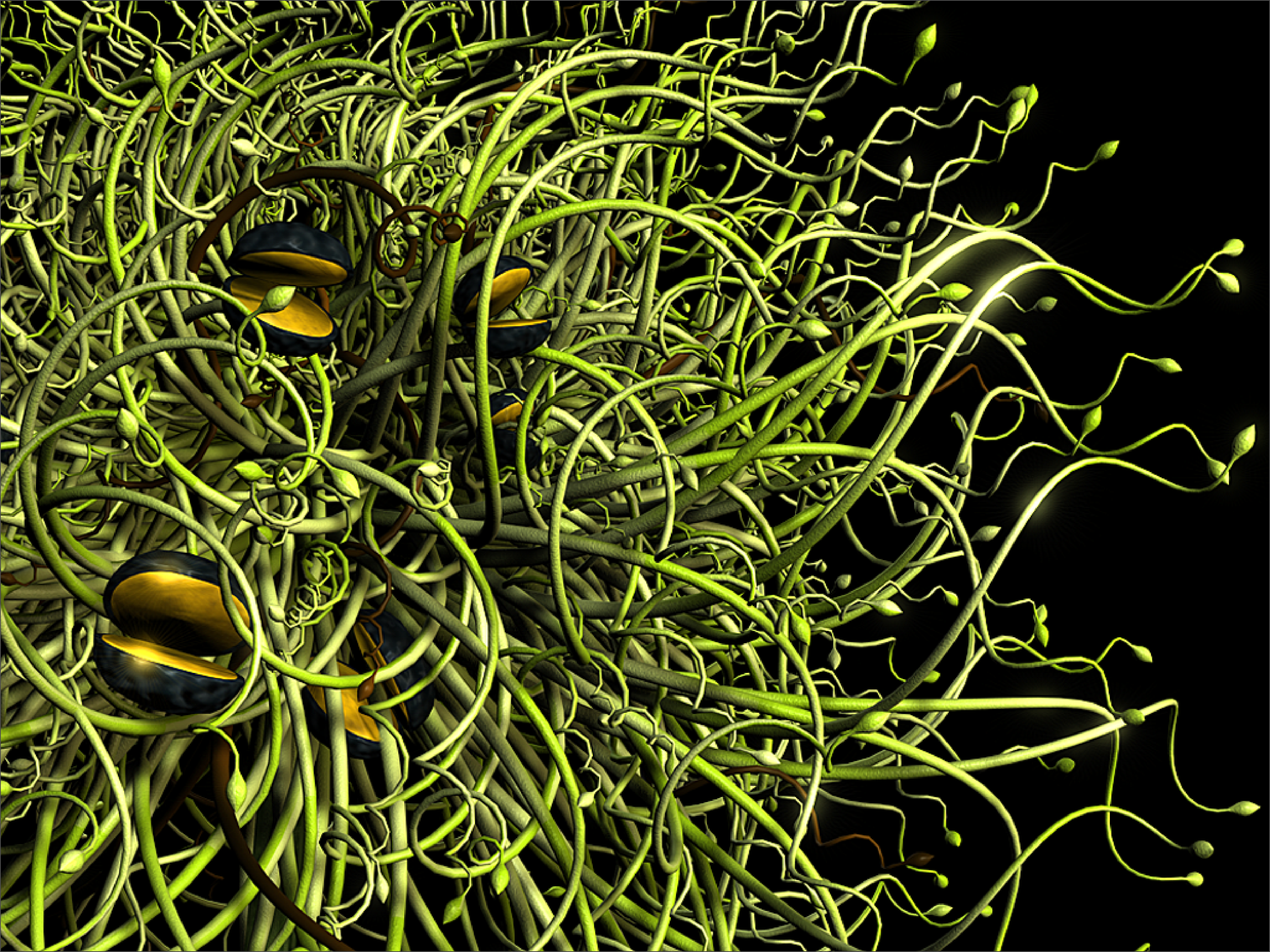
Frame: 1
11
10
9
change
remove
add
parameters
stochastic split
conditional split
stochastic change
delete
mutate
parameter change
constant change
operator change
new expression
reduce expression
new operator
age change
growth function
save genes
load genes
default
next gen
exit
Evolve 1.2
by Jon McCormack

```
a(t) : t>0 : -> [&(70)L]/(137.5)I(10)a(t-1);
a(t) : t==0.0 : -> [&(70)L]/(137.5)I(10)A;
A -> [&(18)u(4)F F I(10)I(5)X(5)K K K K]/(137.5)I(8)A;
I(t) : t>0 : -> F I(t-1);
I(t) : t<=0 : -> F;
u(t) : t>0 : -> &(9)u(t-1);
u(t) : t<=0 : -> &(9);
L -> ['{.-F I(7)+F I(7) + F I(7)}]['{.+ F I(7) - F I(7) - F I(7)}];
K -> [&"{.+F I(2)- - F I(2)}][&"{.-F I(2)+ + F I(2)}]/(90);
X(t) : t > 0 : -> X(t-1);
X(t) : t <= 0 : -> ^(50)[[- G G G G + + [G G G[+ + G"' {.].].]. + + G G G G .
                - - G G G . - - G .}]%;

object k2039583 {
        length = 0.1;
        radius = 0.025;
        level = 30;
        delta = 20;
} 'I(9)a(13);

scene { [i,4,k948271^(i*6,0,0,40 - i * 20)]; }
scene { [i,1,k2039583(38)];}
```





Generation 0