Genetic Algorithms

and their Application to the Artificial Evolution of Genetic Regulatory Networks

Tutorial ICSB 2007

Johannes F. Knabe, Katja Wegner, and Maria J. Schilstra

University of Hertfordshire, UK

Overview

- Fundamentals
  - Biological evolution
  - Evolutionary algorithms
    - Genetic algorithms
- Modelling Gene Regulatory Networks (GRNs)
- Evolution of biological clocks with GRNs
- Evolution in NetBuilder‘

Part 1: Fundamentals

Biological Evolution

Biological Evolution

- Evolution = change in the **gene pool** of a population over time

- Gene = hereditary unit - can be passed on unaltered for many generations
- Gene pool = set of all genes in a species or population

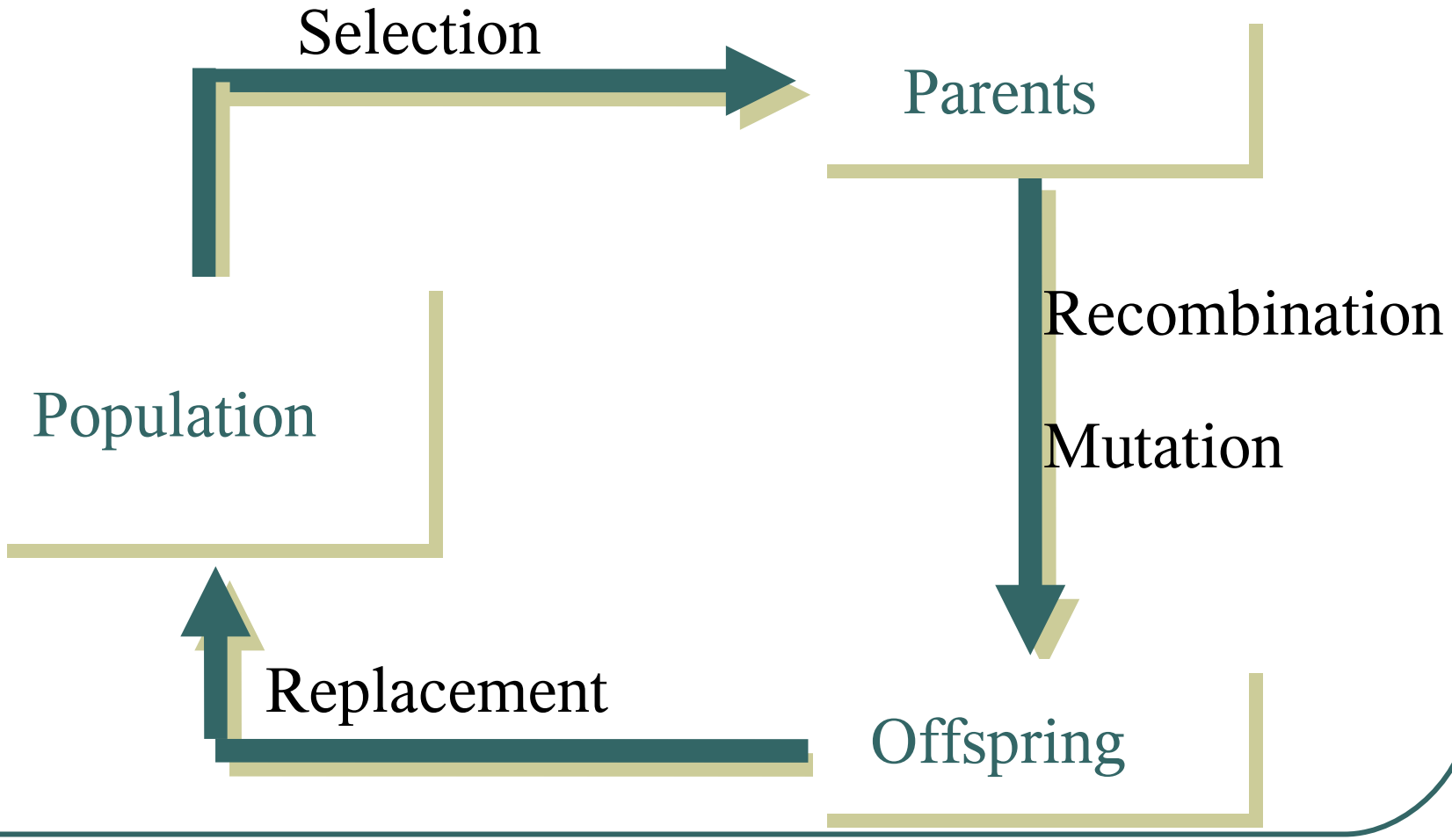
Biological Evolution - Example

- **English moth**, *Biston betularia*:
  - Two color morphs: light and dark
  - 1848 dark moths <= 2% of the population
  - Frequency of the dark morph increased:
    - 1898 95% were dark in Manchester and other highly industrialized areas
  - Why? → ***natural selection***
    - Soot from factories darkened birch trees the moths landed on
    - Birds could see the ligther colored moths better and ate more of them → more dark moths survived
  - http://www.talkorigins.org/faqs/faq-intro-to-biology.html

Biological Evolution – cont'd

- **Natural selection**:
  - Favors those species for further survival and evolution that are best adapted to their environment → see English moth
- Population is evolving → **ratio** of different genetic types is changing and new types are created
  - *Not each* individual !!
- **Darwin**:
  - Evolution through random variations of heritable characteristics and natural selection

Evolutionary cycle : Generation



Dictionary 1

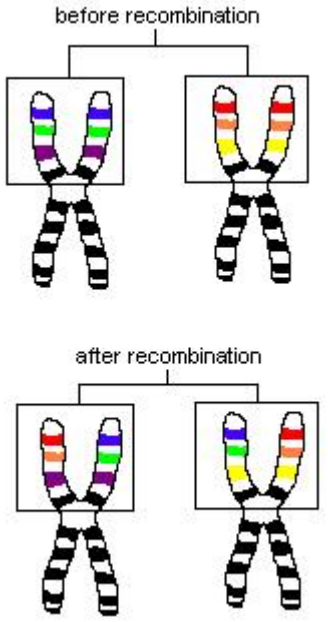
- **Gene** – smallest unit with genetic information
- **Genotype** – collectivity of all genes
- **Phenotype** – expression of genotype in environment
- **Individual** – single member of a population with genotype and phenotype
- **Population** – set of several individuals
- **Generation** – one iteration of evaluation, selection and reproduction with variation

Selection and Reproduction

- Selection does not act on genotype at all but on the performance of the phenotype (**fitness**)
- There is *differential reproduction* → phenotypes better adapted to the environment are likely to produce more offspring
- Slightly unfaithful reproduction creates genotypic variations → affect traits of the phenotype, which in turn affect fitness
- These genotypic variations are heritable

Recombination (crossover)

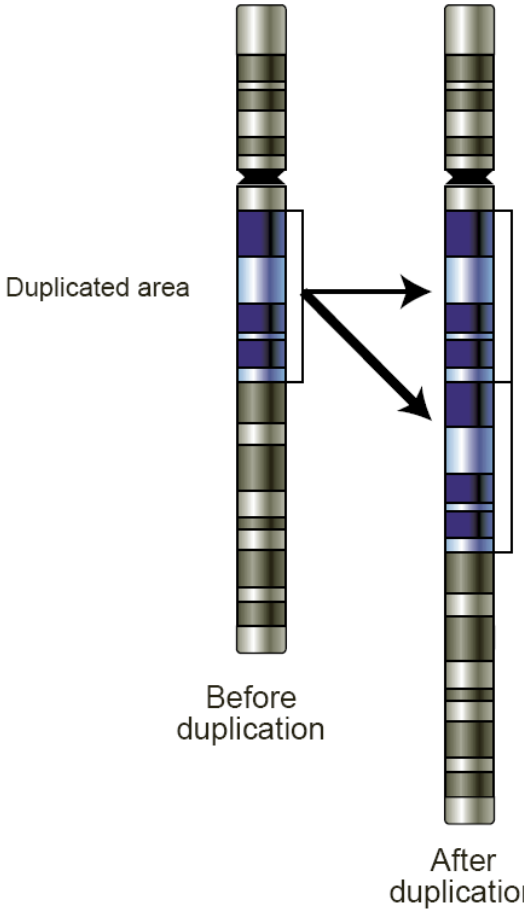
- Choose two individuals from current population → parents
- New combination of the genetic material of these individuals → offspring
- No new genetic information, only reshuffling of existing information
- But can have strong effects on phenotype



http://student.biology.arizona.edu/honors2001/group12/introduction.html

Duplication

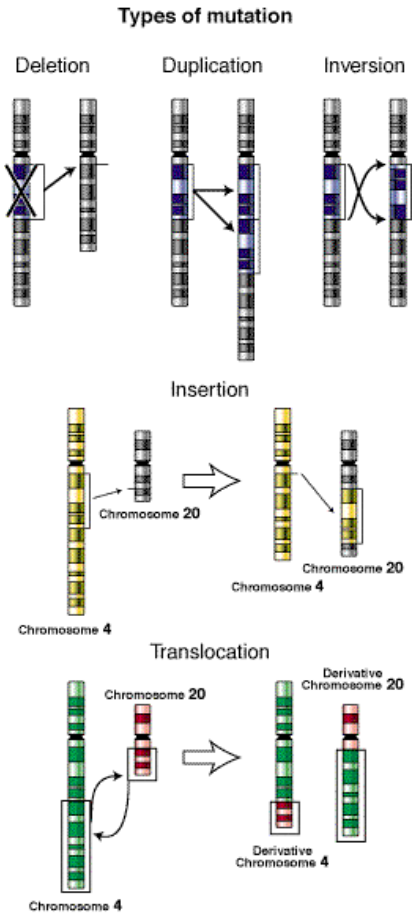
- Any doubling of a certain region, e.g. through unequal recombination
- If this region consists of a gene, it is called gene-duplication



http://en.wikipedia.org/wiki/Gene_duplication

Mutation

- Permanent changes to genetic material
- Can be caused by errors during reproduction of DNA
  - Mutation rate: i.e. 1 in 10.000 bases is incorrectly reproduced
- Brings variability into reproduction
- Usually small changes at individual level but strongly depends on “importance” of mutated base to phenotype



http://www.biocrawler.com/encyclopedia/Mutation

The Evolutionary Mechanisms

- Selection and differential reproduction
  - **DECREASE** diversity in population
- Genetic operators (mutation, recombination)
  - **INCREASE** diversity of population

Part 1: Fundamentals (2)

Evolutionary Computation

Genetic Algorithms (GA)

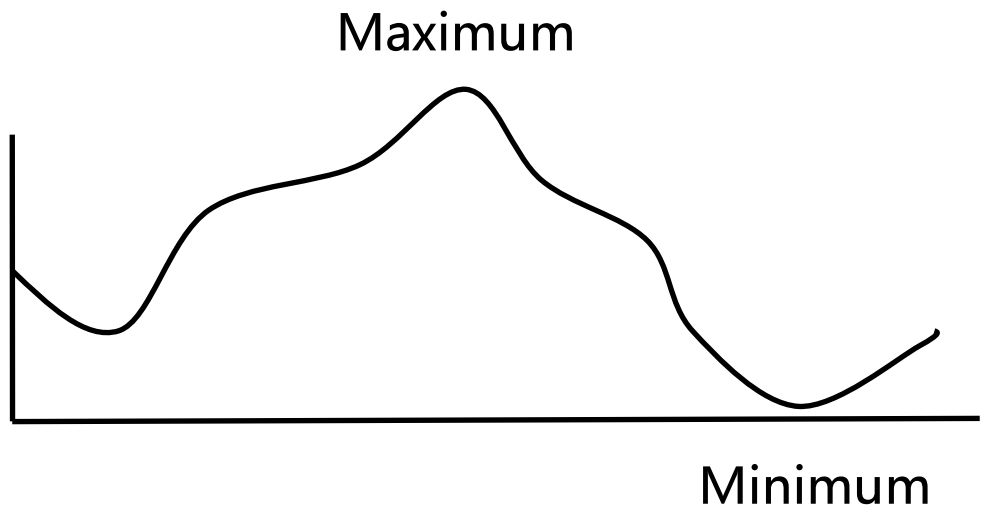
Evolutionary Computation

Evolutionary Computation

- Exploitation of concepts of natural evolution for problem solving using computers
- Simulation of evolutionary processes (recombination, mutation, selection) for solving a desired problem
- Particularly well-suited to complex, multidimensional problems too big to search exhaustively (non-linear optimization problems)
- Cannot solve all problems perfectly, but has fewer restrictions than most problem-solving algorithms

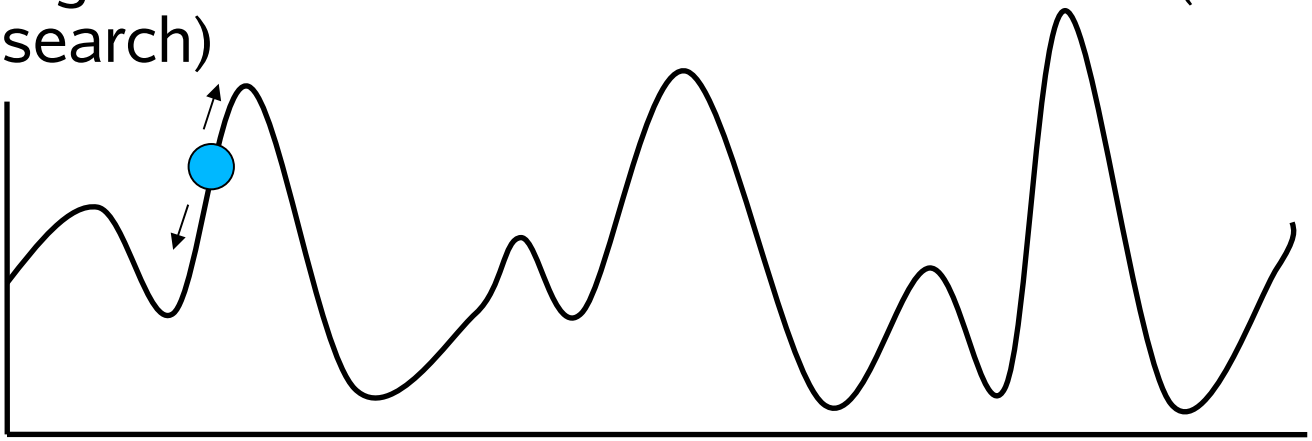
Optimization

- Finding the best solution to a problem
- Mathematically: finding the minimum or maximum of a function (**optimum**)



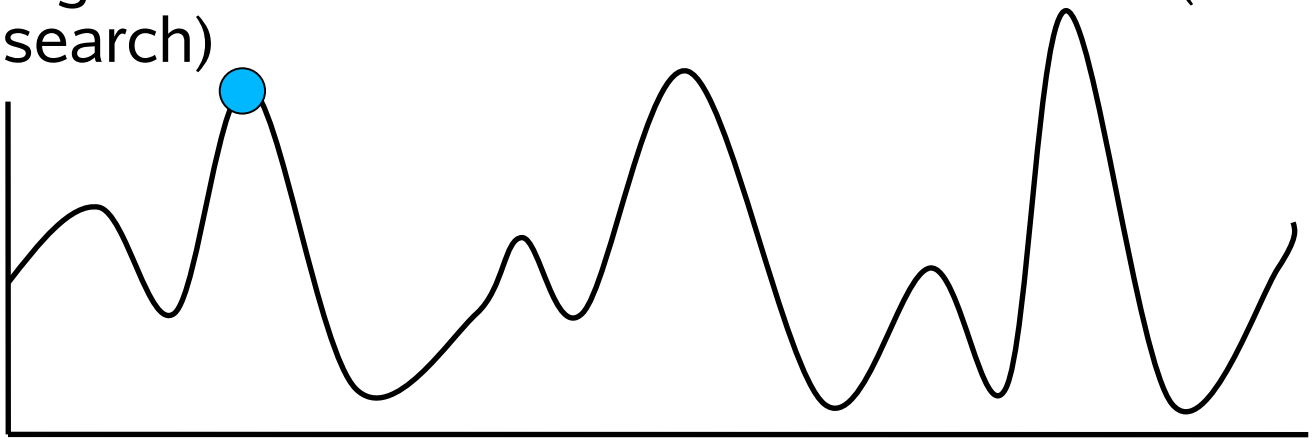
Optimization - Problems

- Example: hill-climbing
  - Start with estimate of global maximum
  - Try to improve by finding other solutions that have a greater value than the current estimate (local search)



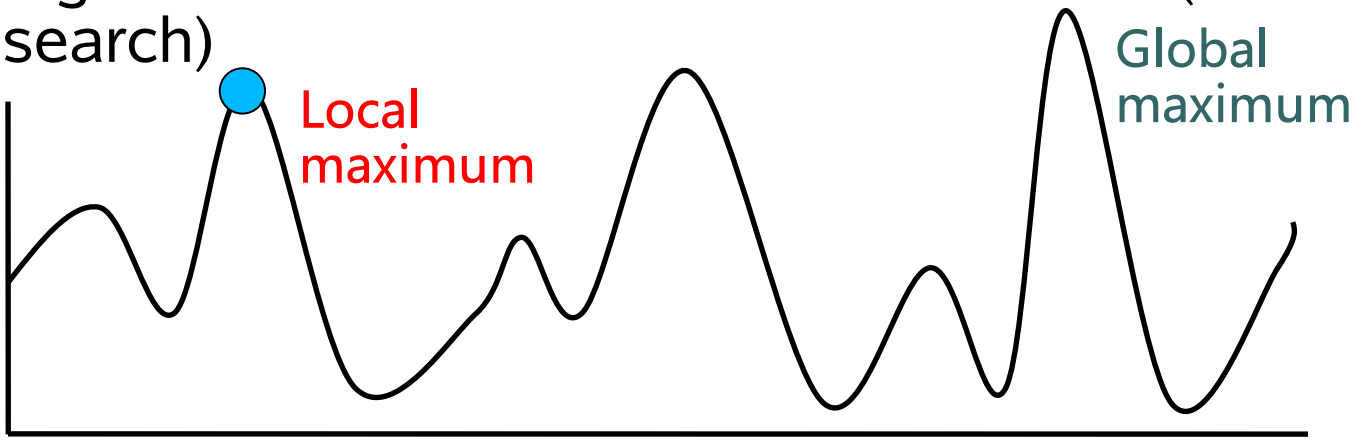
Optimization - Problems

- Example: hill-climbing
  - Start with estimate of global maximum
  - Try to improve by finding other solutions that have a greater value than the current estimate (local search)



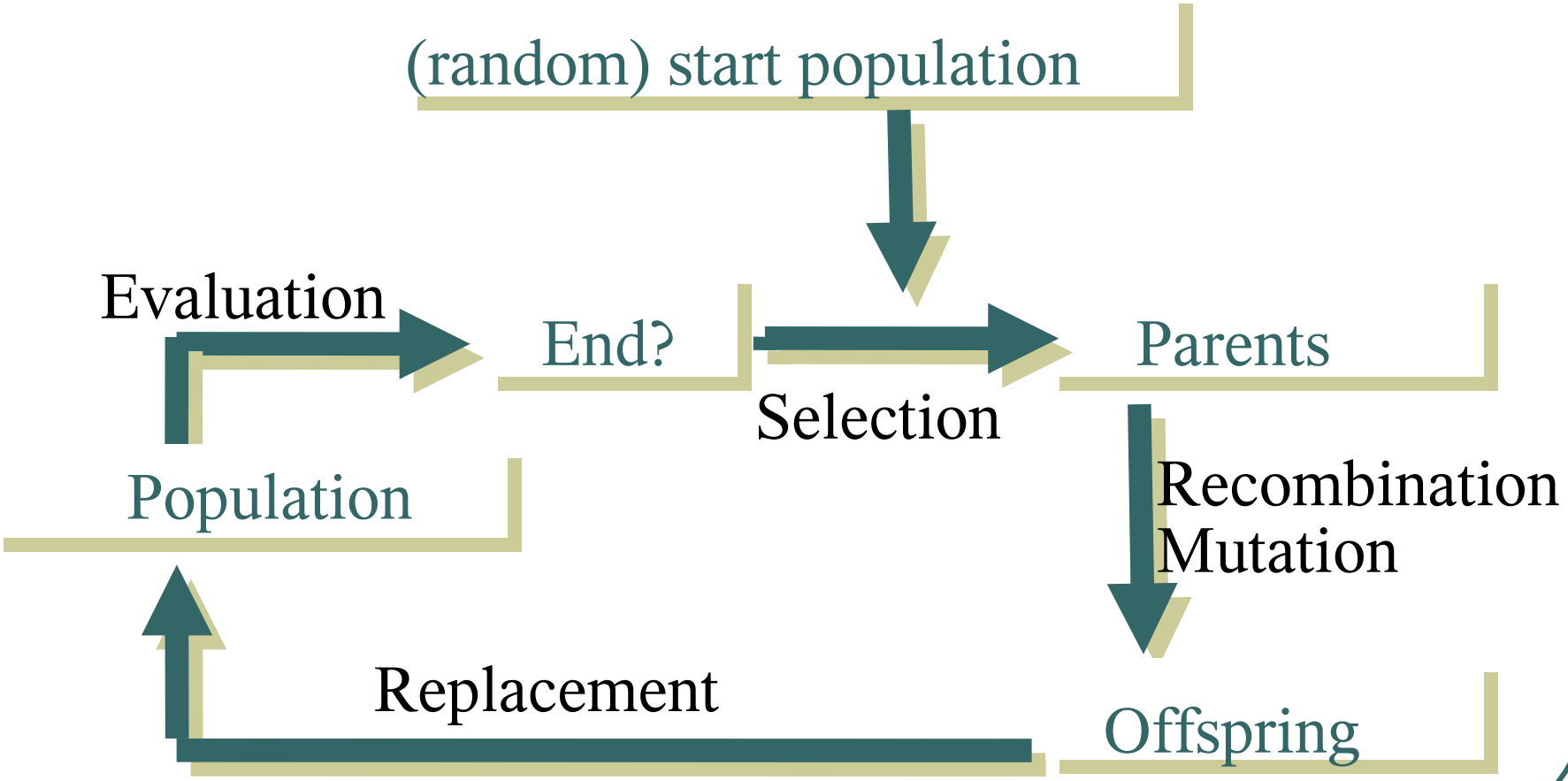
Optimization - Problems

- Example: hill-climbing
  - Start with estimate of global maximum
  - Try to improve by finding other solutions that have a greater value than the current estimate (local search)



  - Local maxima = hazards → could converge to local maximum instead of global

Evolutionary cycle - revisited



Dictionary 2

- **Individual** - one candidate solution
- **Population** - set of individuals
- **Genotype** - encoded representation of individual
- **Phenotype** - decoded representation of individual
- **Mapping** - decodes the phenotype
- **Mutation** - variability operator that modifies a genotype
- **Recombination/Crossover** - variability operator mixing genotypes
- **Fitness** - performance of a phenotype with regard to objective
- **Iteration** - Generation

EC - General properties

- Exploit collective “learning“ process of a population (each individual = one solution = one search point)
- Evaluation of individuals in their environment = measure of quality = fitness → comparison of individuals
- Selection favors better individuals who reproduce more often than those that are worse
- Offspring is generated by random recombination and mutation of selected parents

Main trends

- Genetic algorithms (GAs)
- Genetic programming (subform of GAs)
- Evolutionary strategies (ES)
- Evolutionary programming (EP)

Genetic Algorithms

GAs - Simple Example

Simple example – $f(x) = x^2$

- **Finding the maximum of a function:**
  - **$f(x) = x^2$**
  - Range [0, 31] → Goal: find max ($31^2 = 961$)
  - Binary representation: string length 5 = 32 numbers (0-31)

genotype	0 0 1 0 1	
mapping	2^8 2^4 2^2 2^1 2^0 16 8 4 2 1	
phenotype	0*16+0*8+1*4+0*2+1*1 = 5	
fitness	25	= f(x)

F(x) = x² - Start Population

	binary	value	fitness
String 1	00110	6	36
String 2	00011	3	9
String 3	01010	10	100
String 4	10101	21	441
String 5	00001	1	1

F(x) = x² - Selection

	binary	value	fitness
String 1	00110	6	36
String 2	00011	3	9
String 3	01010	10	100
String 4	10101	21	441
String 5	00001	1	1

- Worst one removed

F(x) = x² - Selection

	binary	value	fitness
String 1	00110	6	36
String 2	00011	3	9
String 3	01010	10	100
String 4	10101	21	441
String 5	00001	1	1

- Best individual: reproduces twice → keep population size constant

F(x) = x² - Selection

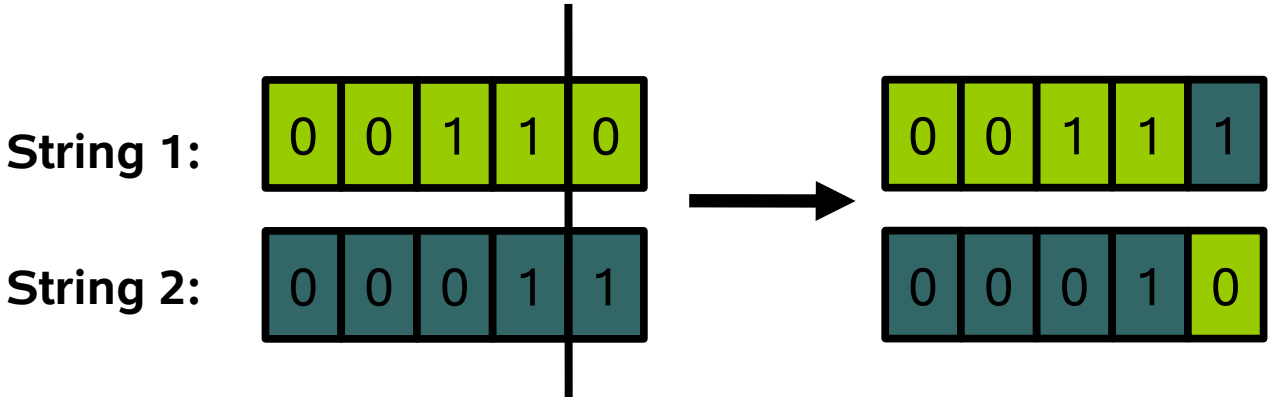
	binary	value	fitness
String 1	00110	6	36
String 2	00011	3	9
String 3	01010	10	100
String 4	10101	21	441
String 5	00001	1	1

- All others are reproduced once

F(x) = x² - Recombination

- Parents and x-position randomly selected (equal recombination)

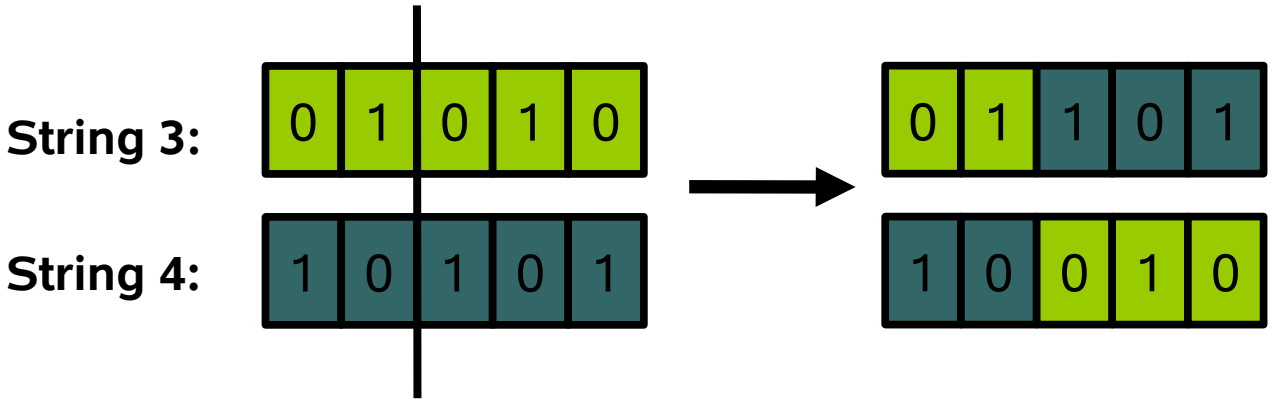
	partner	x-position
String 1	**String 2**	**4**
String 3	String 4	2



F(x) = x² - Recombination

- Parents and x-position randomly selected (equal recombination)

	partner	x-position
String 1	String 2	4
String 3	**String 4**	**2**



F(x) = x² - Mutation

- bit-flip:
  - Offspring -String 1: **0**0111 (7) → **1**0111 (23)
  - String 4: 10**1**01 (21) → 10**0**01 (17)

$F(x) = x^2$

- All individuals in the parent population are replaced by offspring in the new generation
  - (generations are discrete!)
- **New population (Offspring)**:

	binary	value	fitness
String 1	10111	23	529
String 2	00010	2	4
String 3	01101	13	169
String 4	10000	16	256
String 5	10001	17	289

$F(x) = x^2$ - End

- Iterate until termination condition reached, e.g.:
  - Number of generations
  - Best fitness
  - Process time
  - No improvements after a number of generations
- Result after one generation:
  - Best individual: 10111 (23) – fitness 529

$F(x) = x^2$ - Animation

- Java-Applet (html page) with examples

GAs - General

Genetic algorithms

- Differences to other search and optimization algorithms:
  - GAs search from a ***population of points*** (possible solutions), *not* from a single point
  - GAs use ***probabilistic***, *not* deterministic rules

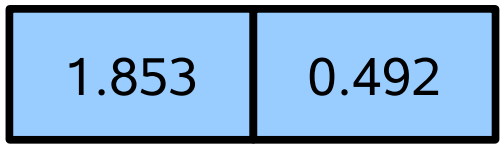
History

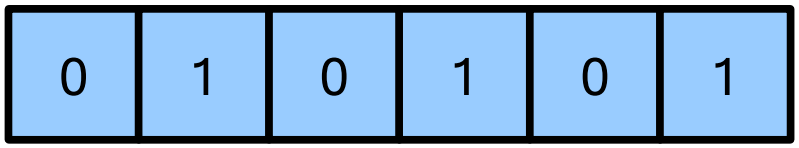
- In 1960s John H. **Holland**, University of Michigan:
  - Abstraction and generalisation of the population concept with genetic coding and operators
- Use in **Bioinformatics**, e.g.:
  - motif discovery,
  - sequence alignment,
  - protein structure prediction etc.

Coding and Mapping

Genetic coding

- Finite strings (= genome, represents genotype)
- Strings consists of units with information (unit = gene)
- One string (→ individual) = one possible solution of the problem
- Genotype often real numbers or bit string:





Genetic coding and mapping

- What should the phenotype look like and how to encode it as a genotype?
- How does one map from genotype to phenotype, considering the sources of variation (mutation and recombination)?

  - *Highly problem dependent!*
  - Hint: small changes to genotype should **often** result in small changes to phenotype, i.e. similar performance: *heritability of traits!*
  - heritability of traits is important → otherwise GA becomes only random search

Mapping – Example

- Binary coding versus Gray coding of a number
- *Hamming distance:*
  - Number of bits that have to be changed to map one string into another one
  - *E.g. 00**0** and 00**1** → distance = 1*
- *Remember: small changes in genotype should cause small changes in phenotype*

Mapping – Example cont'd

- **Binary** coding of 0-7 (phenotype):

0	000
1	001
2	010
3	011
4	100
5	101
6	110
7	111

Mapping – Example cont'd

- **Binary** coding of 0-7 (phenotype):

0	000
1	001
2	010
3	011
4	100
5	101
6	110
7	111

- Hamming distance, e.g.:
  - 000 (0) and 001 (1)
    - Distance = 1 (optimal)
  - 011 (3) and 100 (4)
    - Distance = 3 (max possible)

Mapping – Example cont'd

- **Gray** coding of 0-7:

0	000
1	001
2	011
3	010
4	110
5	111
6	101
7	100

Mapping – Example cont'd

- **Gray** coding of 0-7:

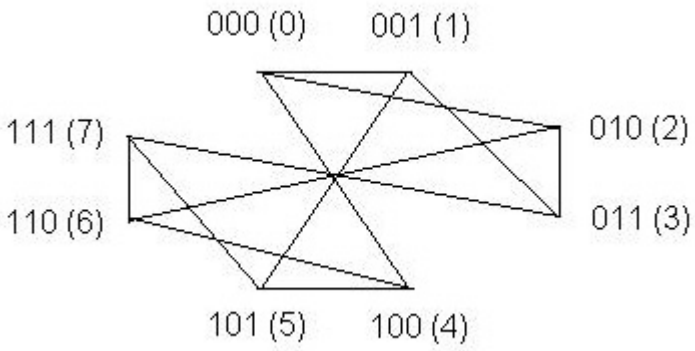
0	000
1	001
2	011
3	010
4	110
5	111
6	101
7	100

- Hamming distance:
  - Two neighboring numbers (phenotypes) have always a genotype distance of 1 (all differ only by one bit flip) – **OPTIMAL mapping**

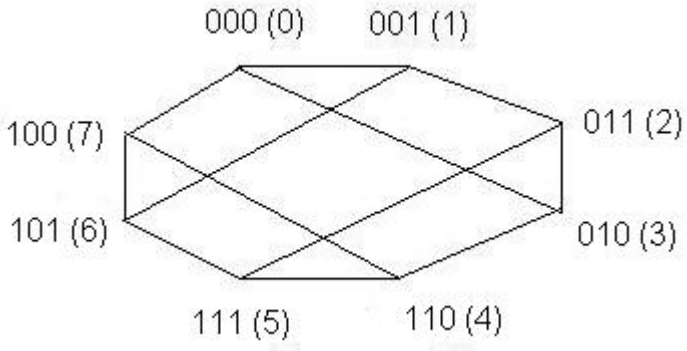
Mapping – Example cont'd

- Comparing kinship with distance = 1

- **Binary**:



Gray:



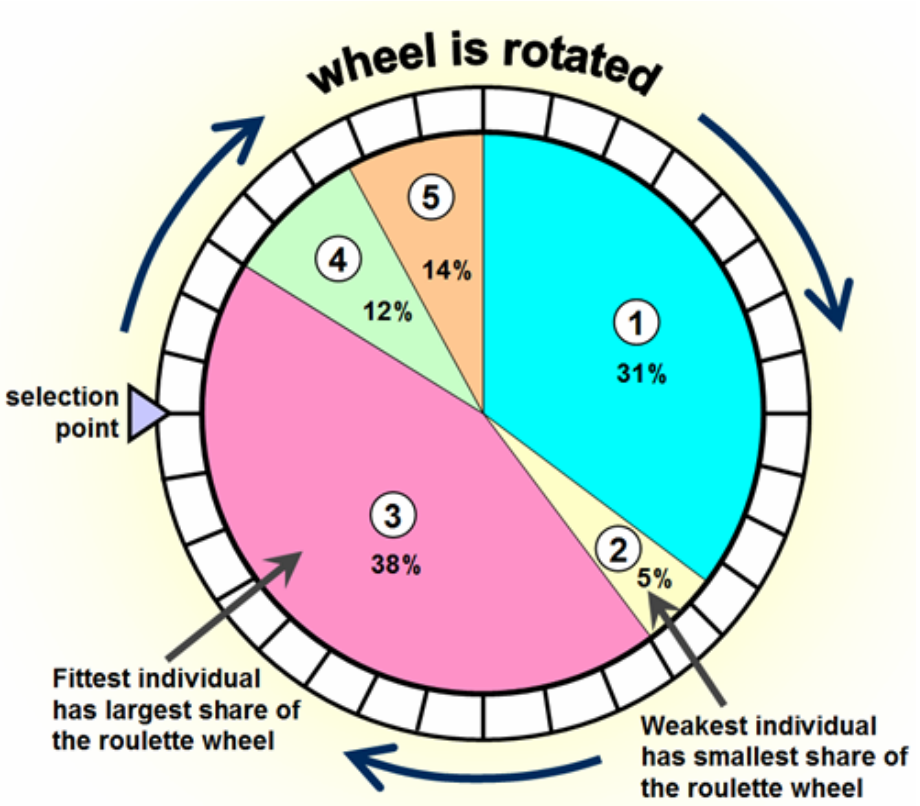
Selection

Selection

- Based on fitness function:
  - Determines how „good“ an individual is (**fitness**)
  - Better fitness, higher probability of selection
- Selection of individuals for differential reproduction of offspring in next generation
- Favors better solutions
- Decreases diversity in population

Selection - Roulette-Wheel

- Each solution gets a region on a roulette wheel according to its fitness
- Spun wheel, select solution marked by roulette-wheel pointer
- **stochastic selection** (better fitness = higher chance of reproduction)



http://www.edc.ncl.ac.uk/highlight/rhjanuary2007g02.php

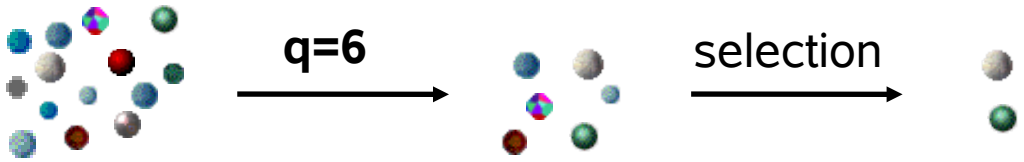
Selection - Elitism

- Individual(s) kept ***unchanged*** for next population

- **Example**:
  - Selection based on fitness values
  - Keep the best individual of current population
  - unrealistic but ensures best fitness of a generation never decreases → decrease of diversity

Selection - Tournament

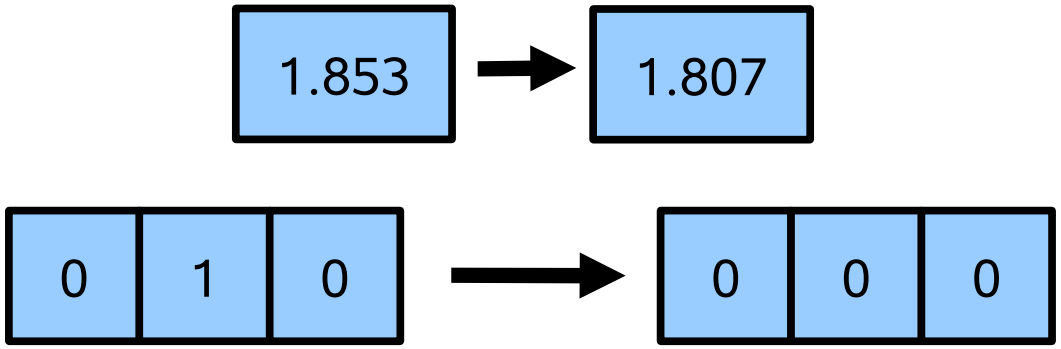
- *randomly* select **q** individuals from current population
- **Winner**: individual(s) with best fitness among these q individuals
- **Example**:
  - select the best two individuals as parents for recombination



Genetic variability operators

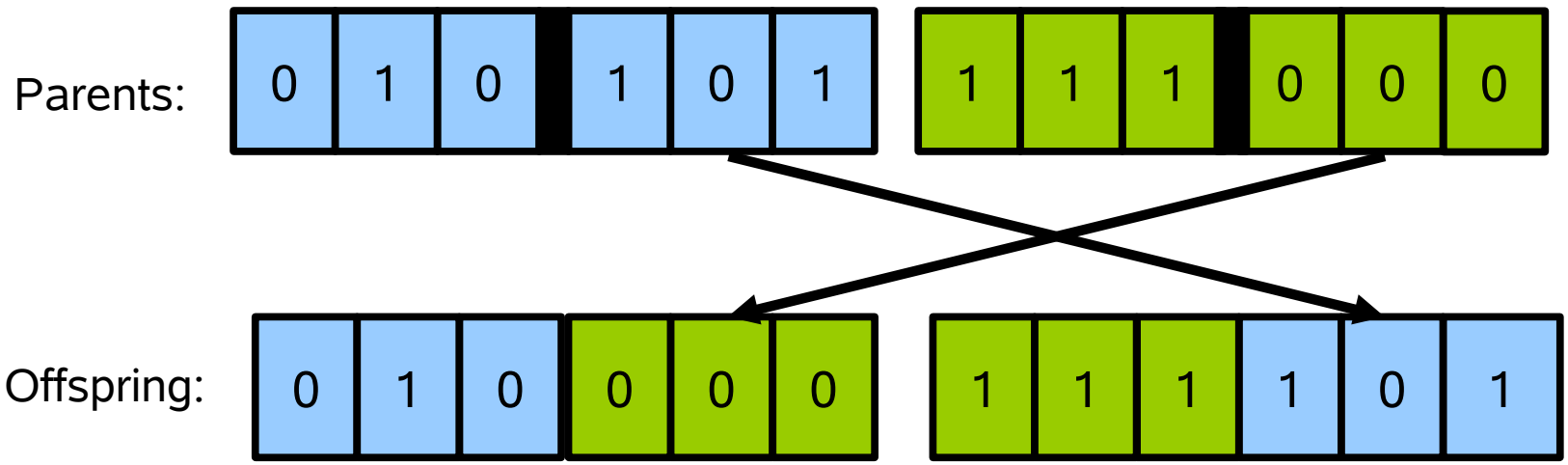
Mutation

- Varies details, usually exploitive
- Changes *one* position in the string
  - each position same small probability of undergoing a mutation
- **Goal**: search around existing good solution, possibly leave local optima



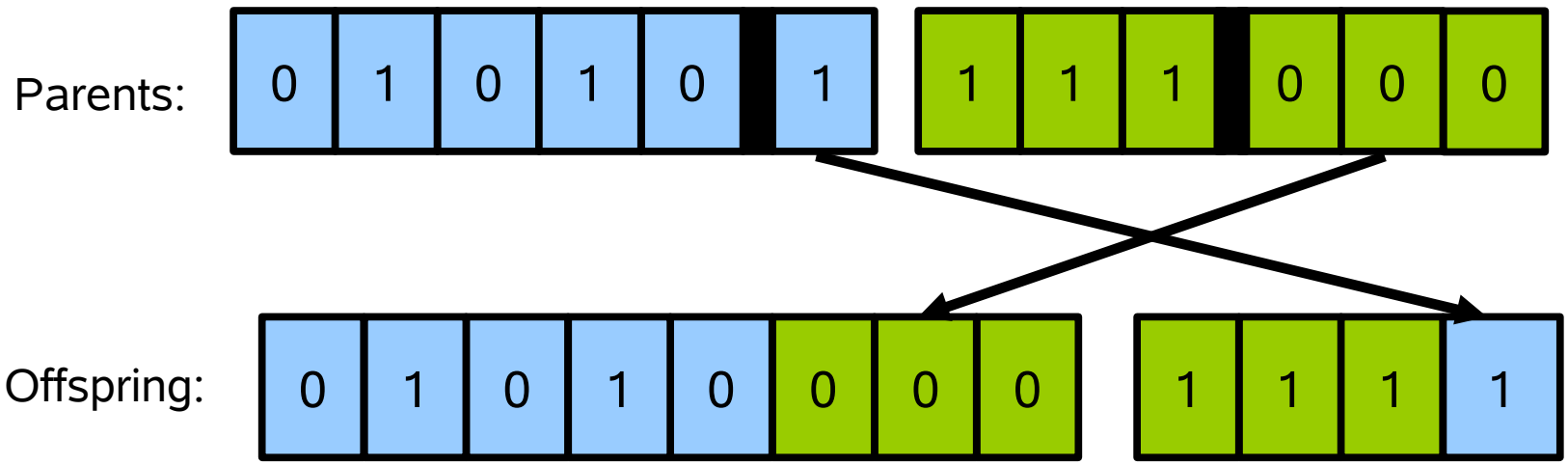
Recombination/Crossover

- Usually explorative
- Creates new strings by combining parts of two existing strings



Recombination

- Unequal:
  - Crossover points independent for each string chosen

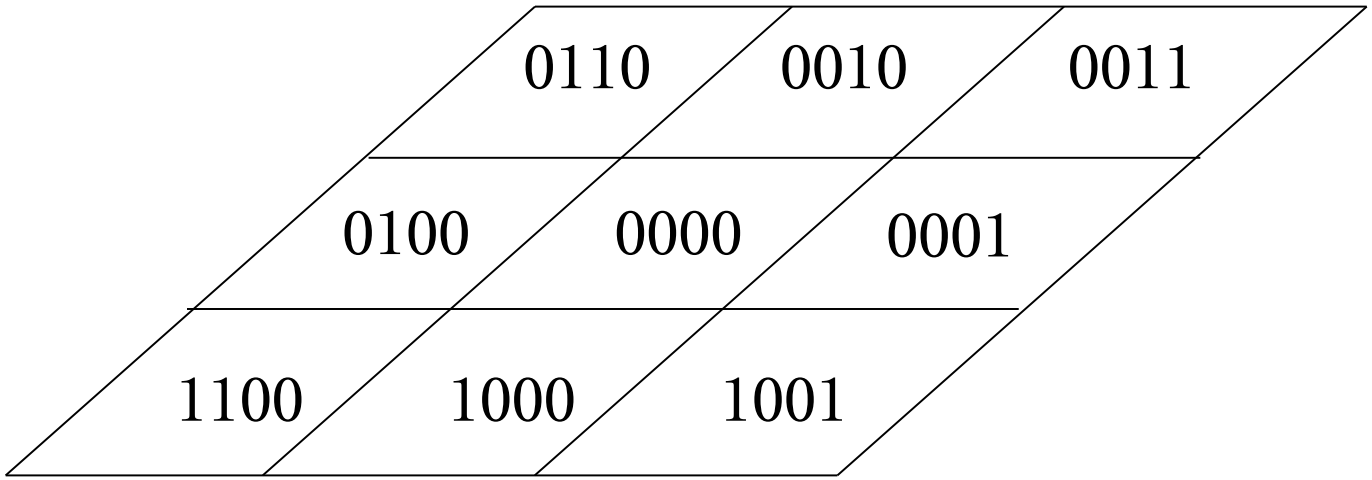


Fitness

Fitness function

- **Nature**:
  - only survival and reproduction count
  - how well do I do in my environment
- **Fitness space** structure:
  - *Defined by kinship of genotypes and fitness function*
  - ***Advantage***: visual representation can be useful when thinking about model design
  - ***Limitation***: ideas might be too simplistic when not working on toy-problems - complex spaces and movements (think crossover!)

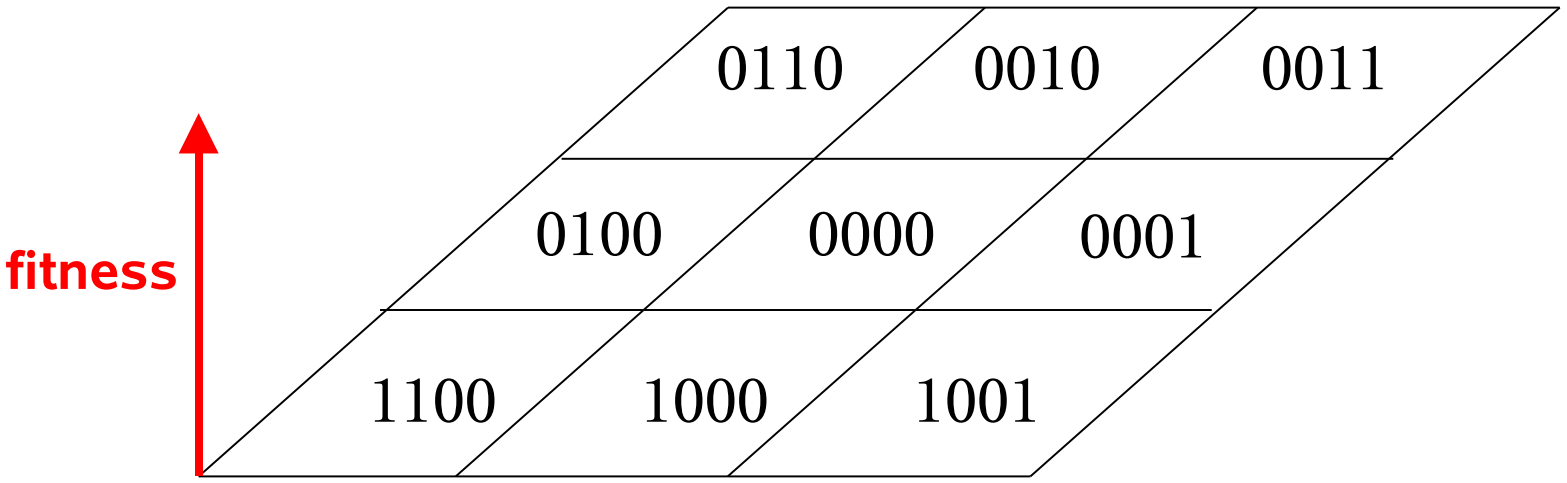
Fitness space or landscape



Schema of genetic kinship

- How we "move" in that landscape over generations is defined by our variability operators, usually mutation and recombination
- ***Now add fitness...***

Fitness space or landscape



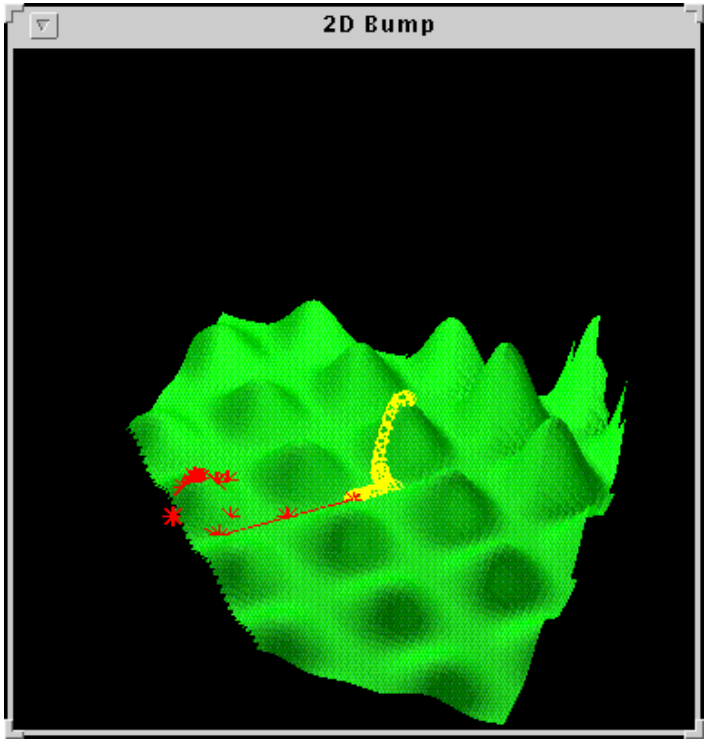
Schema of genetic kinship

- How we "move" in that landscape over generations is defined by our variability operators, usually mutation and recombination
- ***Now add fitness...***

Fitness landscapes contd.

- x/y axes: kinship, i.e. the more genetic resemblance the closer together
- z axis: fitness

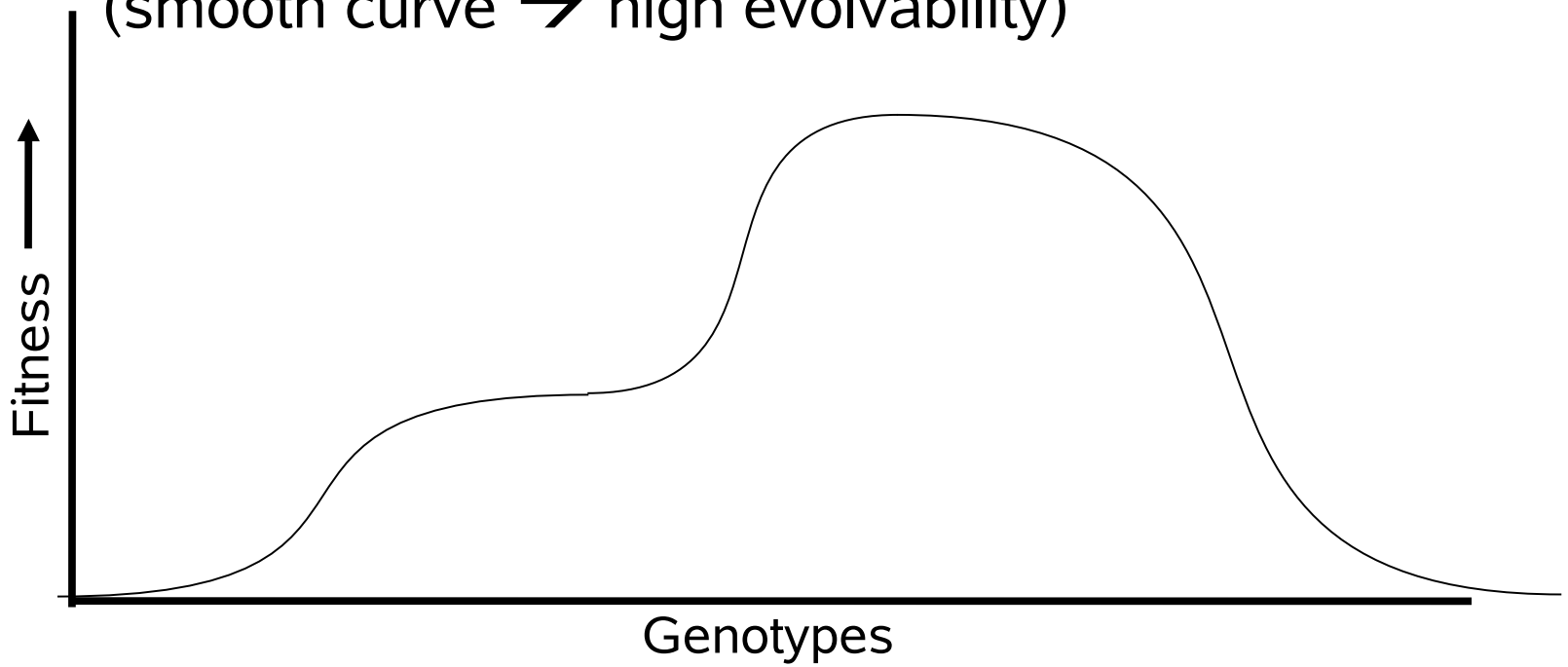
Every “snowflake” one individual, search focuses on “promising” regions (due to differential reproduction)



Animation adapted from Andy Keane, Uni. Of Southampton

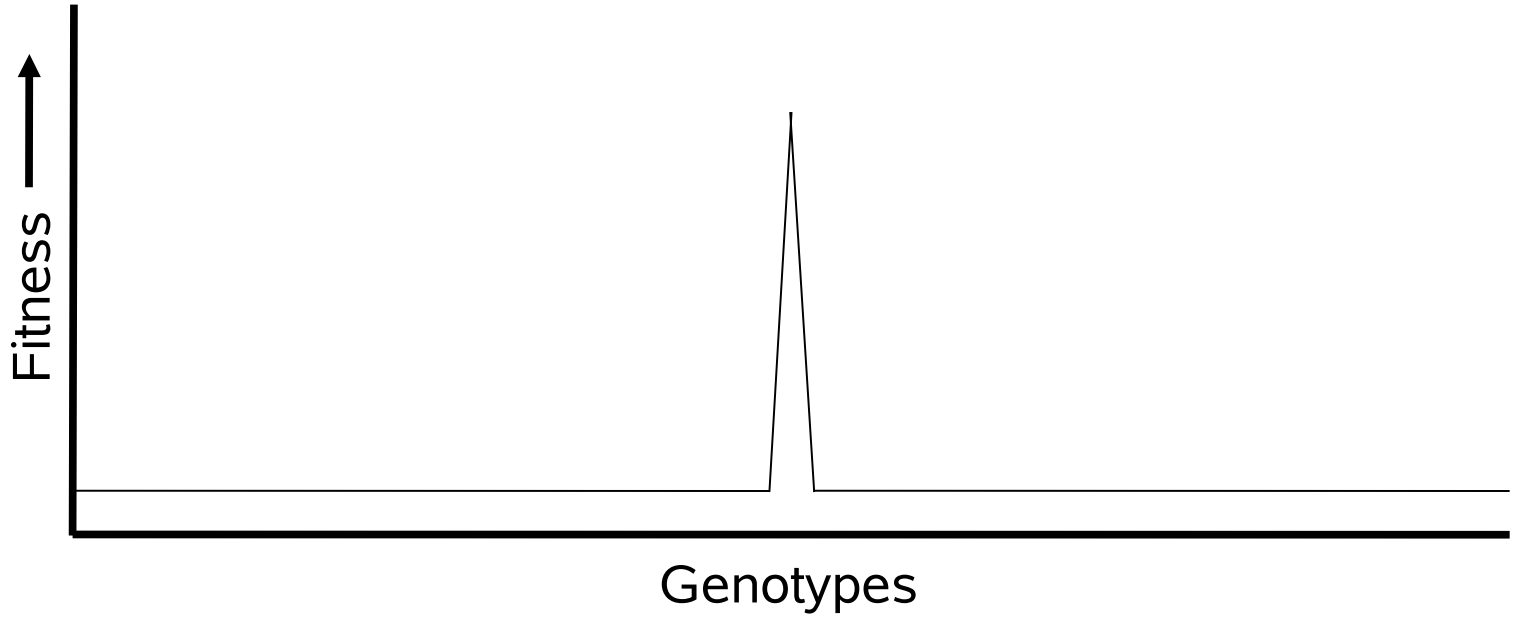
Fitness space – Good design

- Easy to find the optimum by local search
- neighboring genotypes have similar fitness (smooth curve → high evolvability)



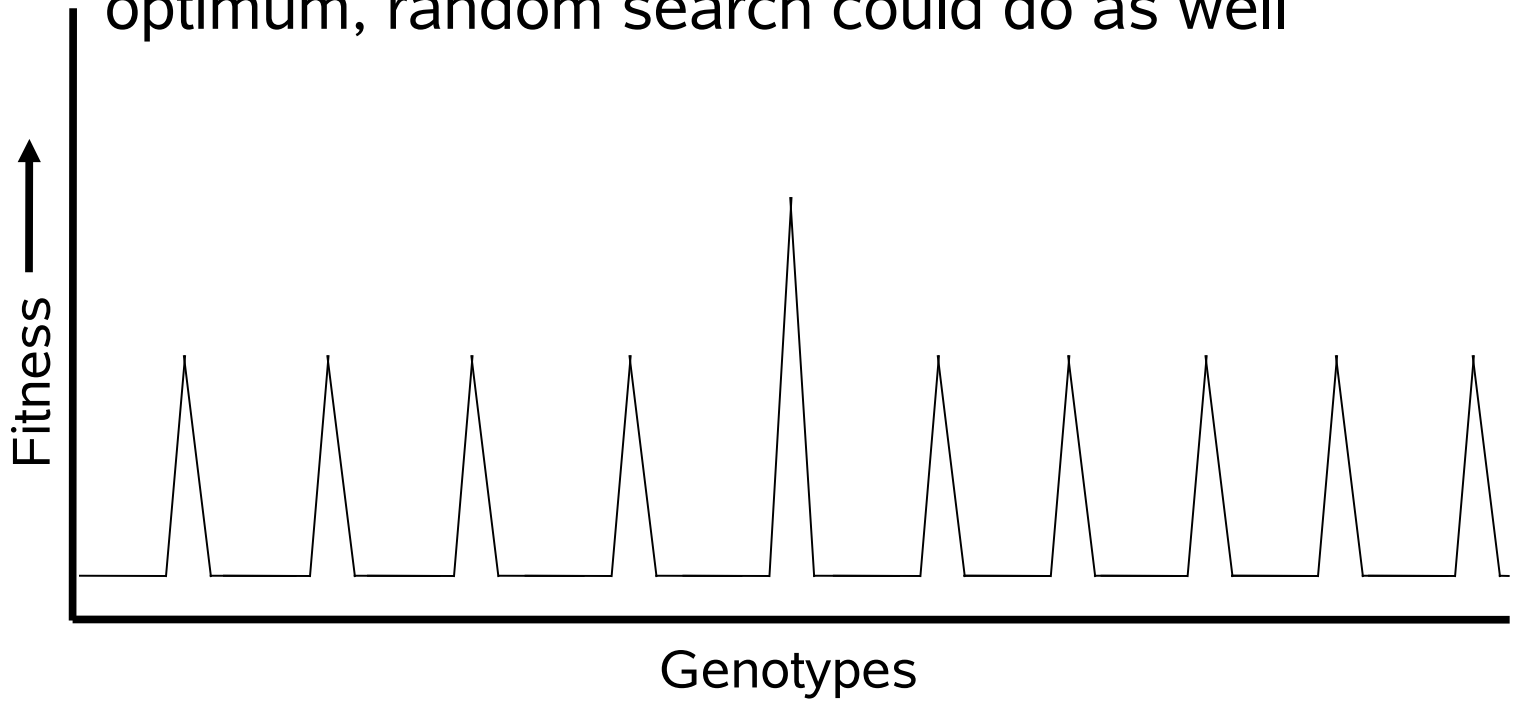
Fitness space - Bad design

- Here we will have a hard time finding the optimum
- Low evolvability (fitness is right/wrong)
- Either problem not well suited for GA or bad design



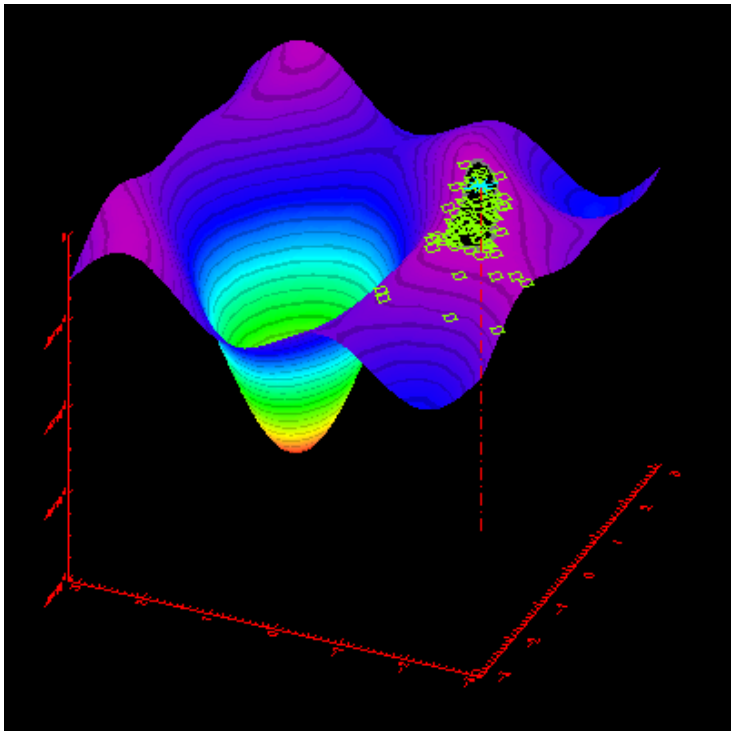
Fitness space – Mediocre design

- Many local optima, so we are likely to find one
- However not much of a gradient to find global optimum, random search could do as well



Dynamic fitness landscape

- Fitness does not need to be static over generations
- Can allow to reach regions otherwise uncovered
- Natural fitness certainly very dynamic



Animation by Michael Herdy, TU Berlin

Design issues

Integrating problem knowledge

- Always to some degree in representation/ mapping
- Create more complex fitness function
- Start population chosen instead of a uniform random one
  - Useful e.g. if constraints on range of solutions
  - Possible problems: Loss of diversity and bias

Design decisions

- GAs: high flexibility and adaptability because of many options:
  - Problem representation
  - Genetic operators with parameters
  - Mechanism of selection
  - Size of the population
  - Fitness function
- Decisions are highly problem dependent
- Parameters not independent, you cannot optimize them one by one

Hints for the parameter search

- Find **balance** between:
  - **Exploration** (new search regions)
  - **Exploitation** (exhaustive search in current region)
- Parameters can be adaptable, e.g. from high in the beginning (exploration) to low (exploitation), or even be subject to evolution themselves
- Balance influenced by:
  - *Mutation, recombination*:
    - create indiviuals that are in new regions (**diversity**!!)
    - fine tuning in current regions
  - *Selection*: focus on interesting regions

Keep in mind

- Start population has a lot of diversity
- “Invest” search time in areas that have proven good in the past → Loss of diversity over evolutionary time
- Premature convergence: quick loss of diversity poses high risk of getting stuck in local optima
- Evolvability:
  - Fitness landscape should not be too rugged
  - Heredity of traits
  - Small genetic changes should be mapped to small phenotype changes

Wrapping up Part 1

GA- Summary

- Selection:
  - Focus on fittest individuals
- Recombination:
  - Adds alternative solutions to population
- Mutation:
  - Makes sure that most of the search space is reached

GA- Summary cont'd

- **Advantages**:
  - Basic method simple and broadly applicable
  - No need for very detailed understandung of the problem
  - But can be adjusted to problem if knowledge present
  - Fast and can be scheduled in parallel

- **Disadvantages**:
  - No guarantee to find best solution
  - High computational demands
  - Adapting to problems at hand can be hard, e.g. finding suitable representation/mapping and evolutionary operators
  - Search can get “caught” in local optima

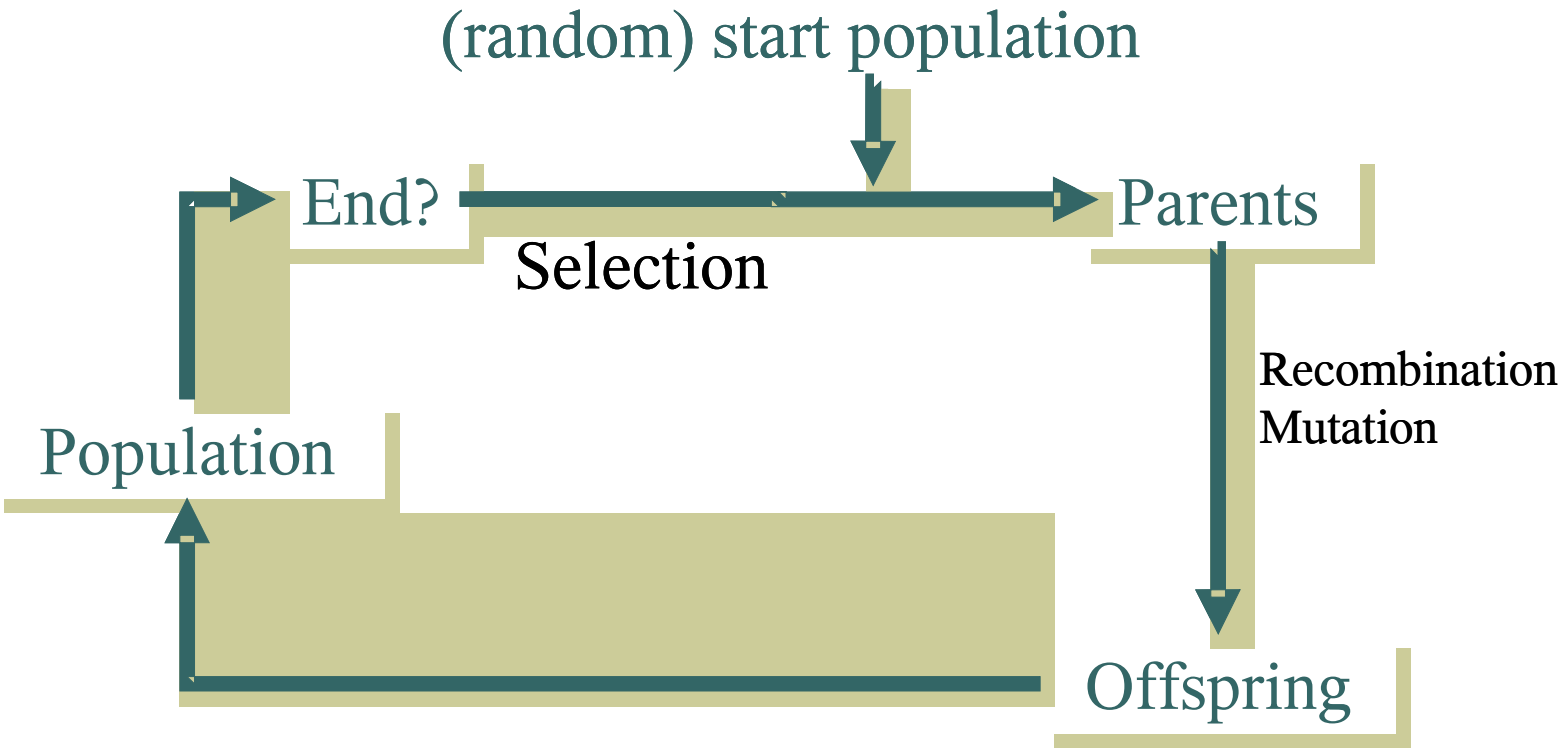
More recent inputs from Biology

- Populations are *spatial, e.g. for* "speciation"
  - interaction (mating, competition) localized to maintain diversity
- Populations have *structure,* e.g. niche protection
  - *competition will be stronger if many individuals do the same to maintain diversity*
- Diploidy with dominance / recessivity
- N-point crossover and other variants
- Morphogenesis instead of simple function mapping (allowing for modularity, making crossover less fatal)

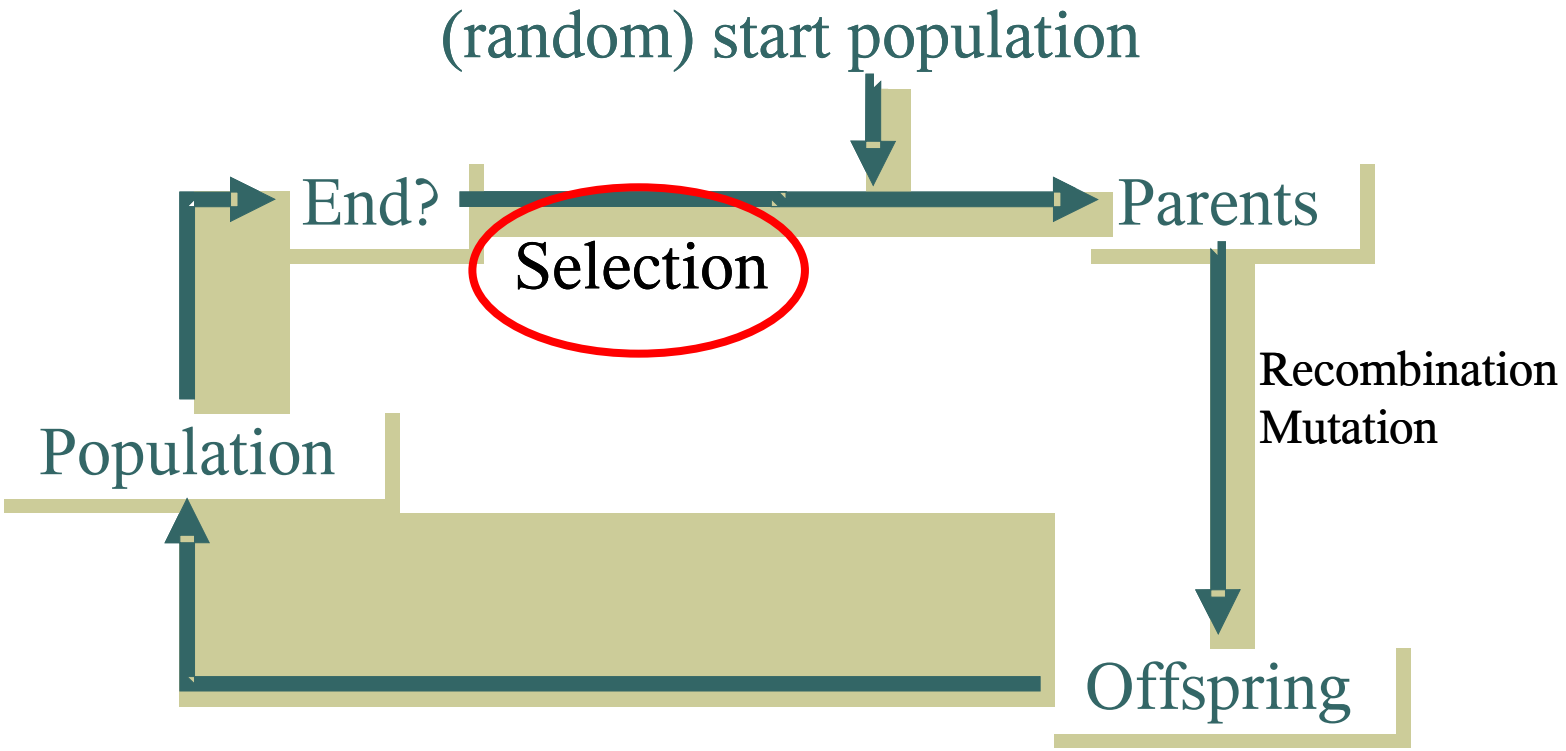
Part 2: Modelling GRNs

Gene Regulatory Networks (GRNs)

Evolutionary cycle - again



Evolutionary cycle - again



Selection

- Selection based on comparison of individual phenotypes with target phenotype
- For phenotype read: input/output system
  - Individual strings in population contain parameter sets
  - Each parameter set is used to build a different input/output system

Decoding, Evaluating, Comparing

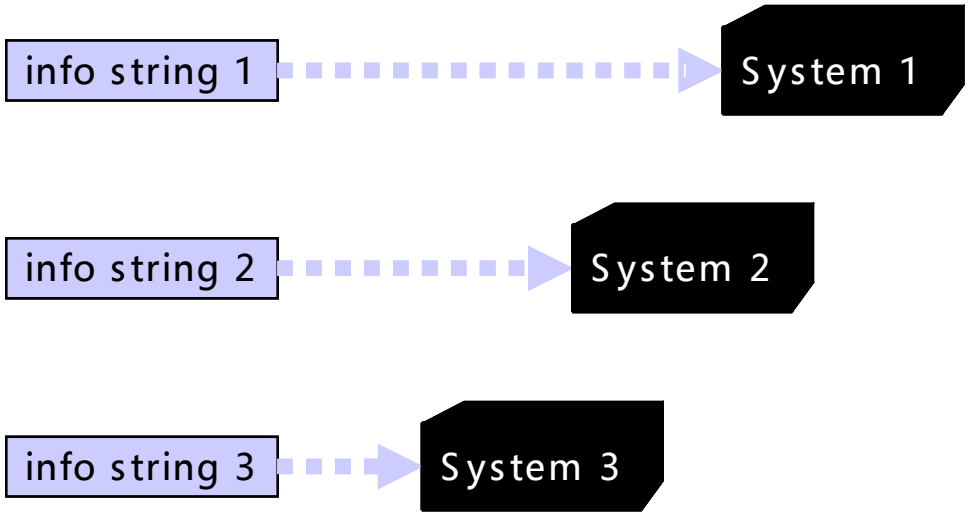
Decoding

info string 1

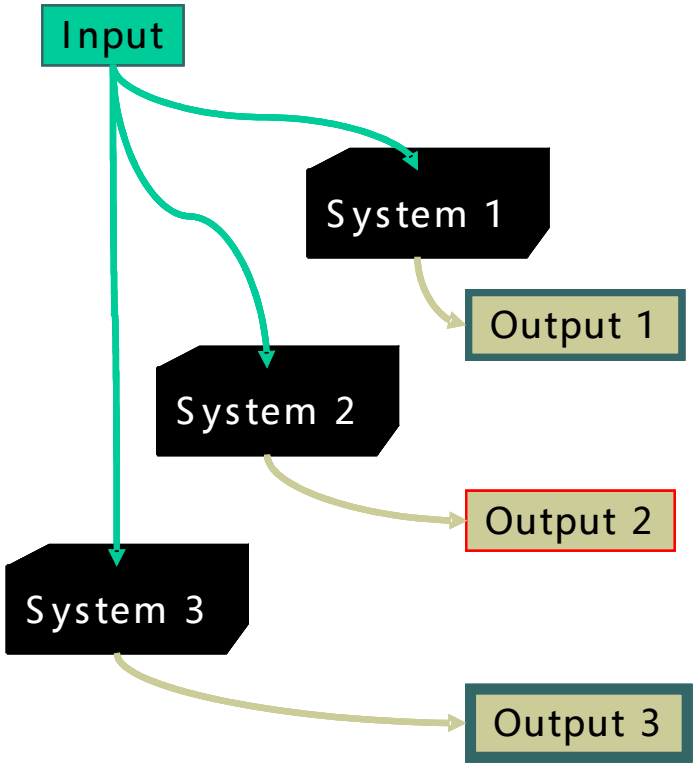
info string 2

info string 3

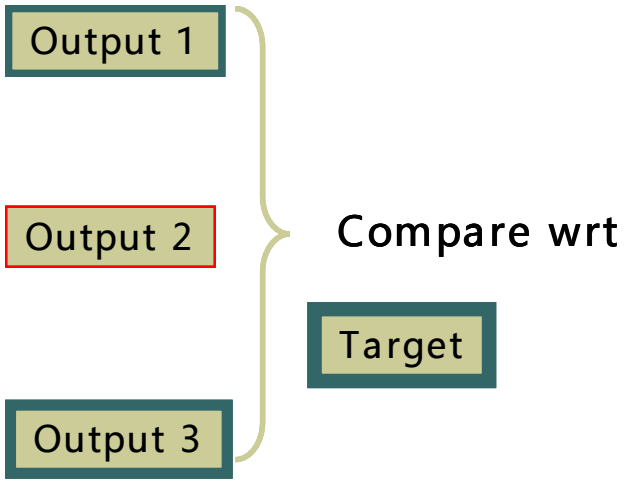
Decoding


info string 1
System 1
info string 2
System 2
info string 3
System 3

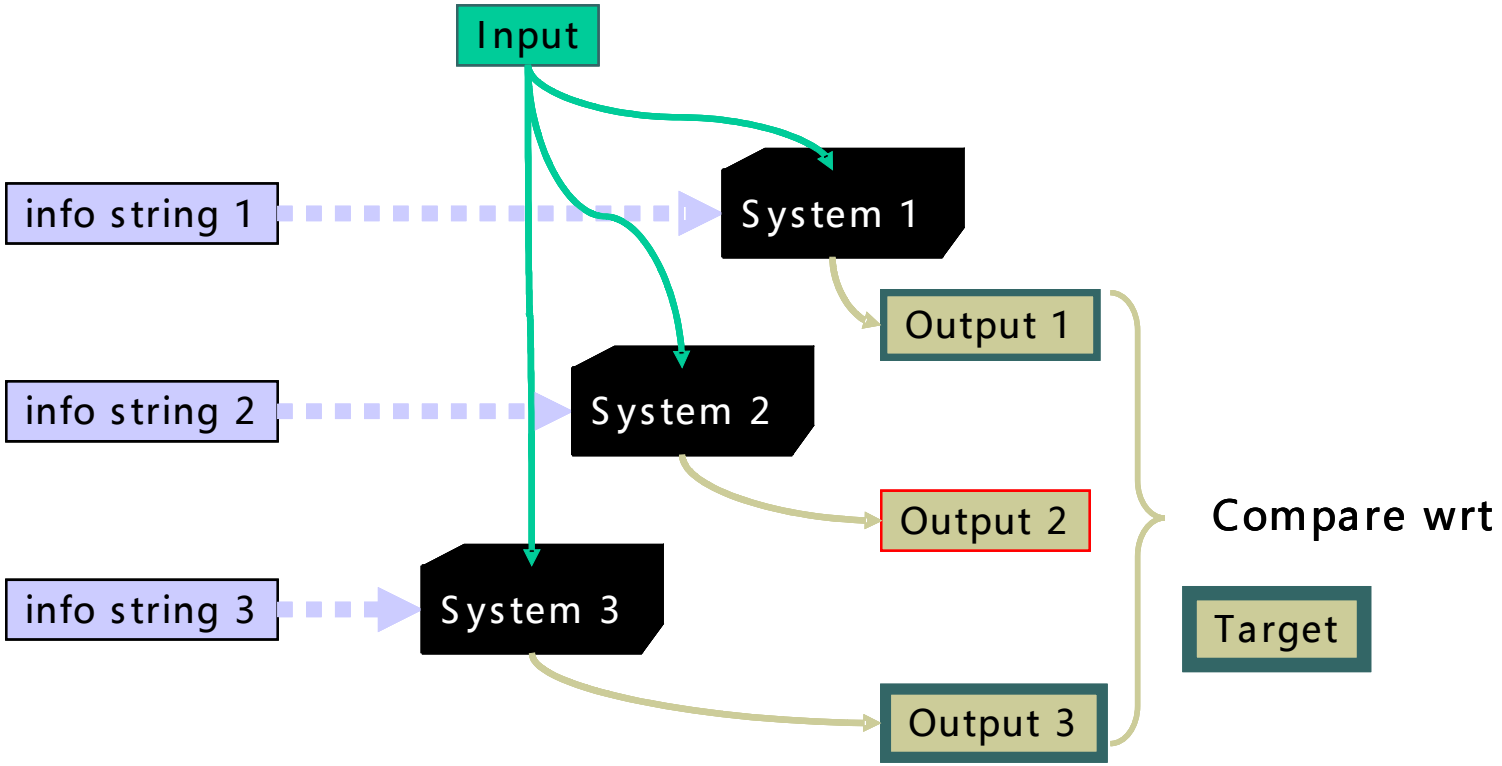
Evaluating



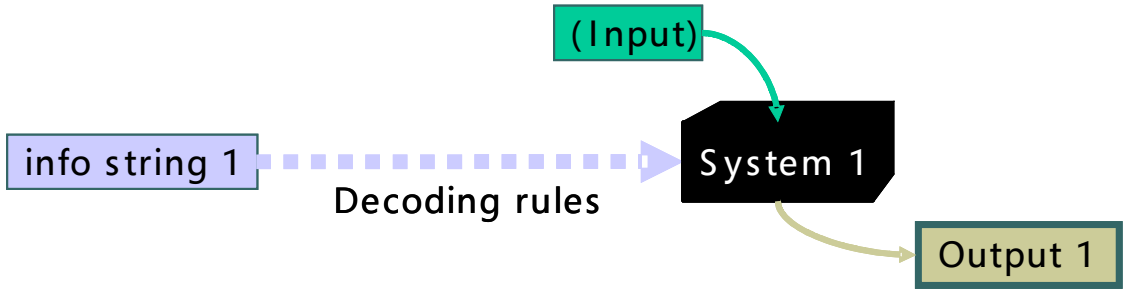
Comparing



Decoding, Evaluating, Comparing

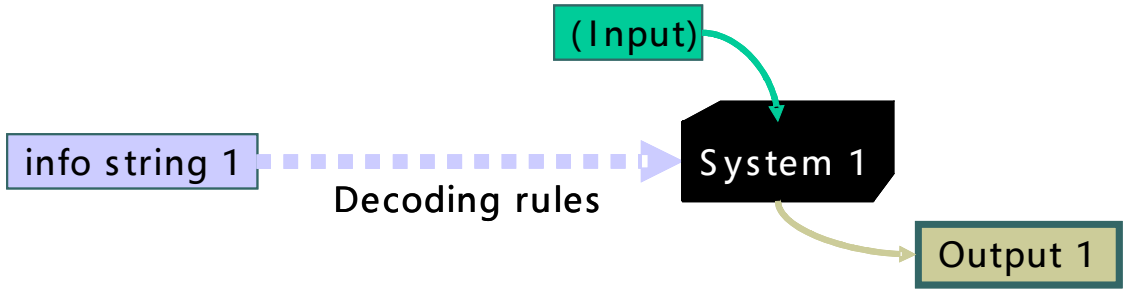


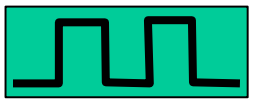
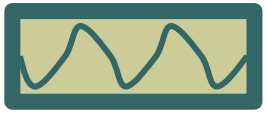
Decoding, Evaluating, Comparing



Example	Info	Rules	System	Input	Output	Target
$F(x) = x^2$	11011	27	f(x)	-	729	Max

Decoding, Evaluating, Comparing



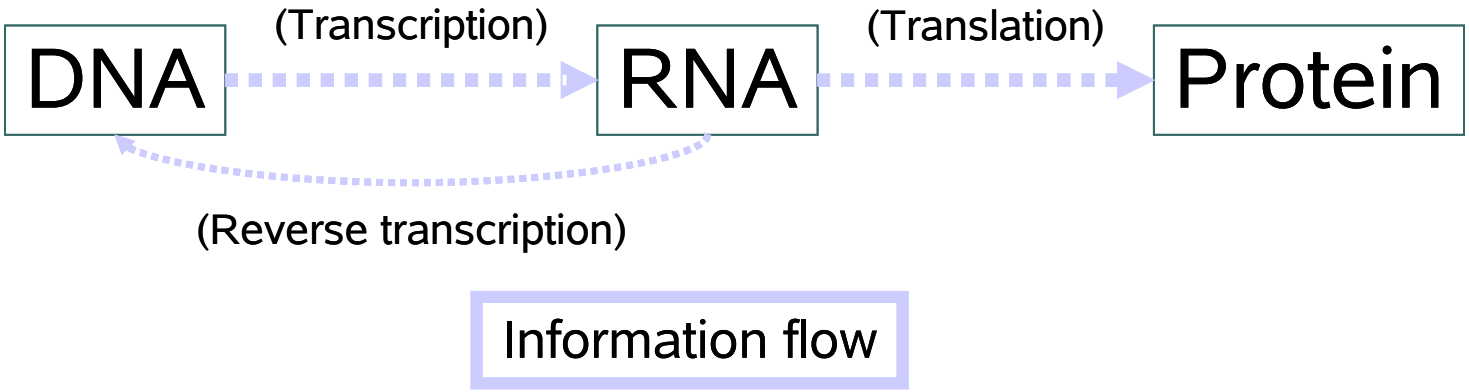
Example	Info	Rules	System	Input	Output	Target
$F(x) = x^2$	11011	27	f(x)	-	729	Max
GRN	Connectivity; parameter values	build GRN	GRN			

(Artificial) Genetic Regulatory Networks (aGRN)

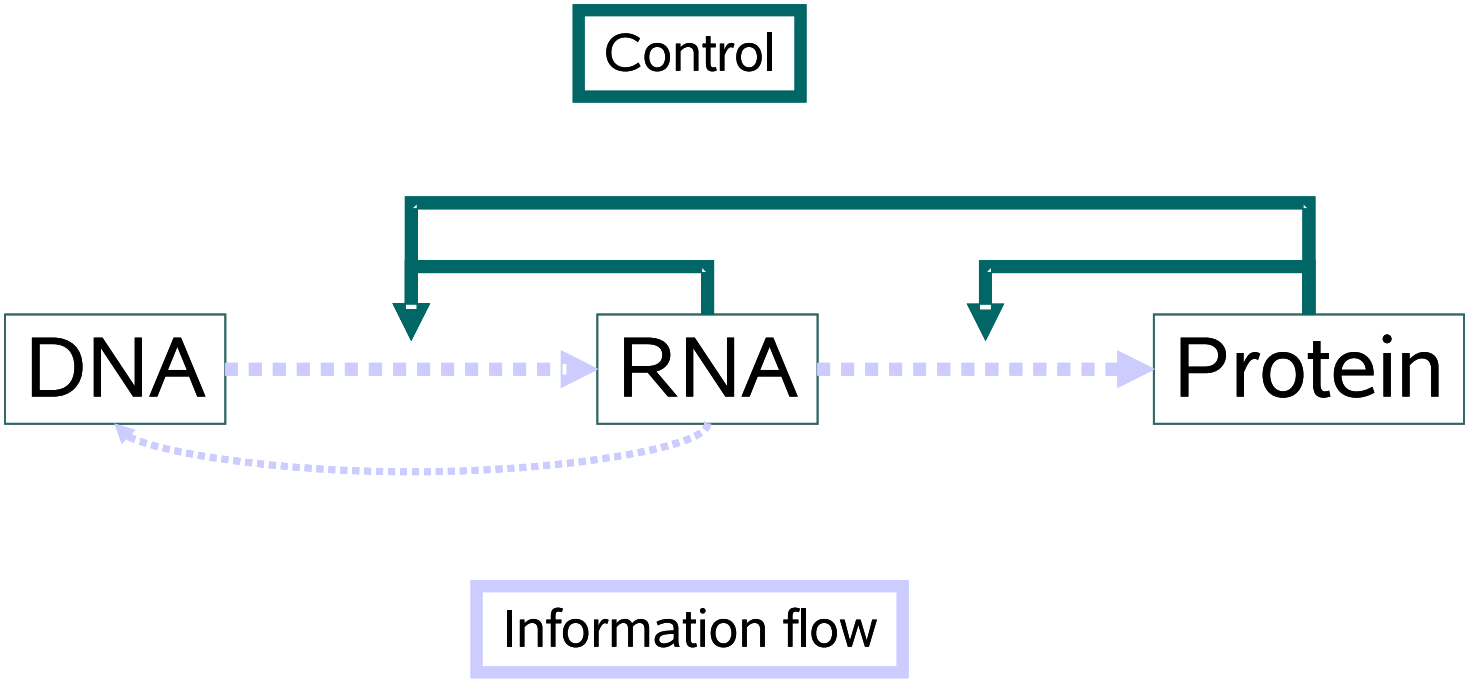
GRNs and aGRNs

- Genetic Regulatory Networks (GRN)
  - The basic control networks that underlie the development and responses of organisms
  - Involve interactions between genes, RNA, proteins
- Artificial Genetic Regulatory Networks
  - Input – output transformation systems
  - Built using concepts taken from “GRN-theory” – assumptions on how GRNs work
- Why aGRNs?
  - As (very crude) models of the “real thing”
  - Interest in “computational potential”

GRN-theory – Central Dogma



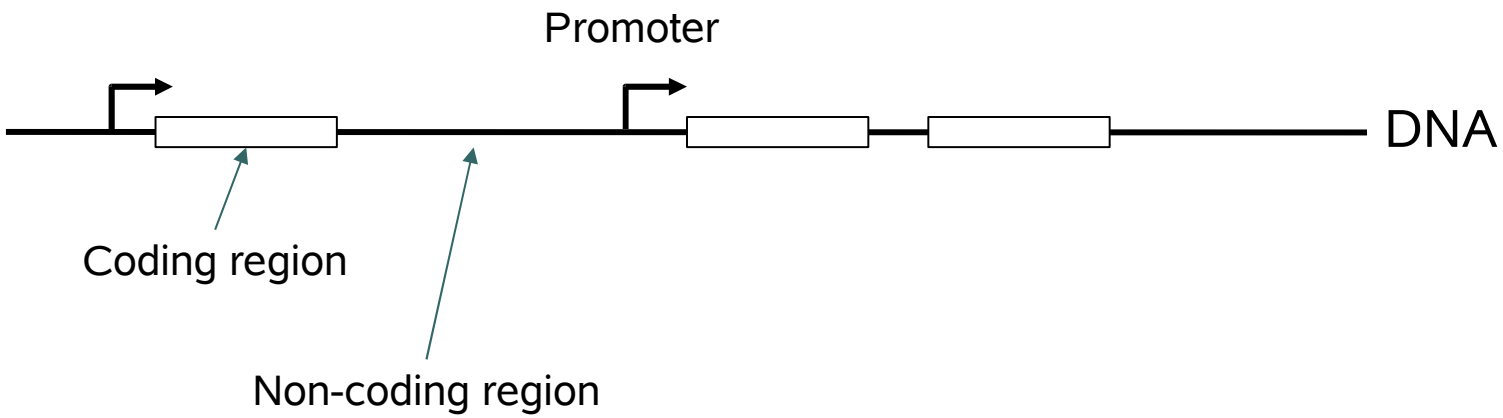
GRN-theory – Central Dogma



GRN Control

Control of gene expression

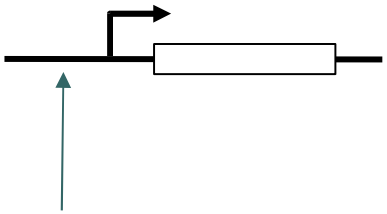
"Standard" notation



Control of gene expression

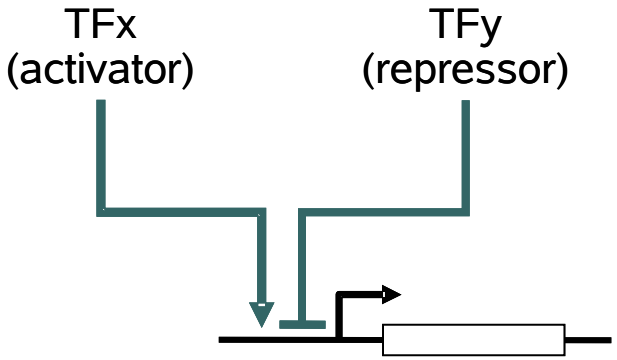
Single gene

Promoter:
Starting point for transcription



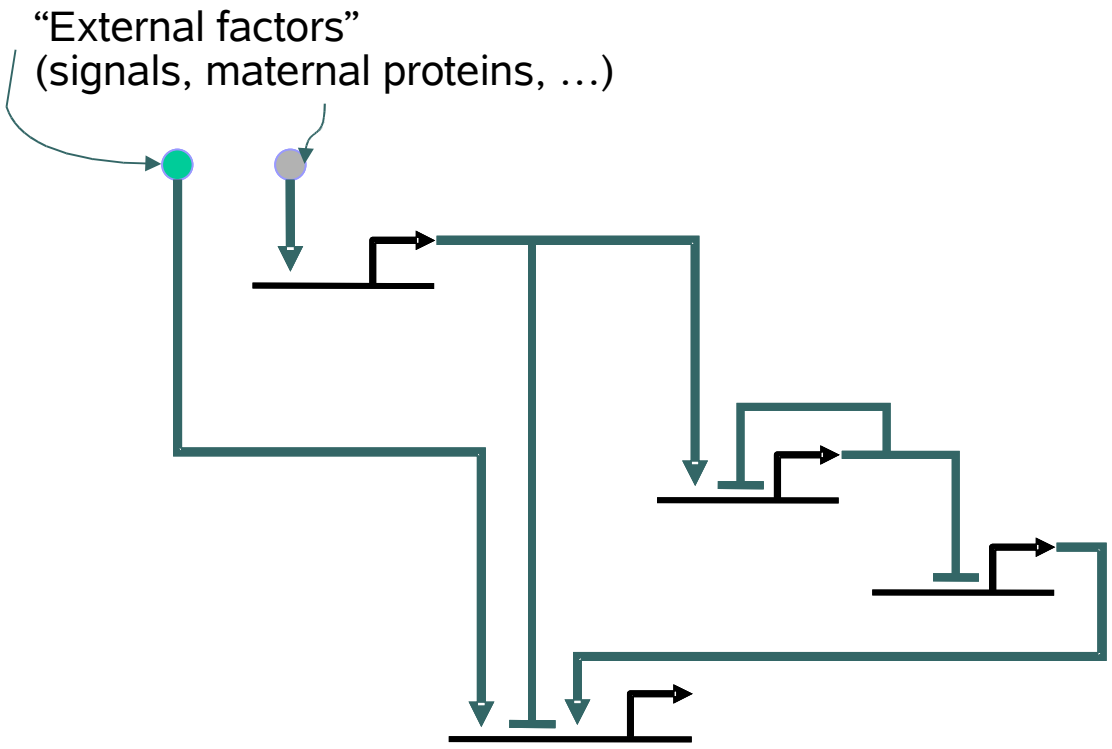
"Upstream" region:
Often contains interaction points for Transcription Factors: proteins that repress or activate transcription

Control of gene expression



Control of gene expression

Genetic Regulatory Network

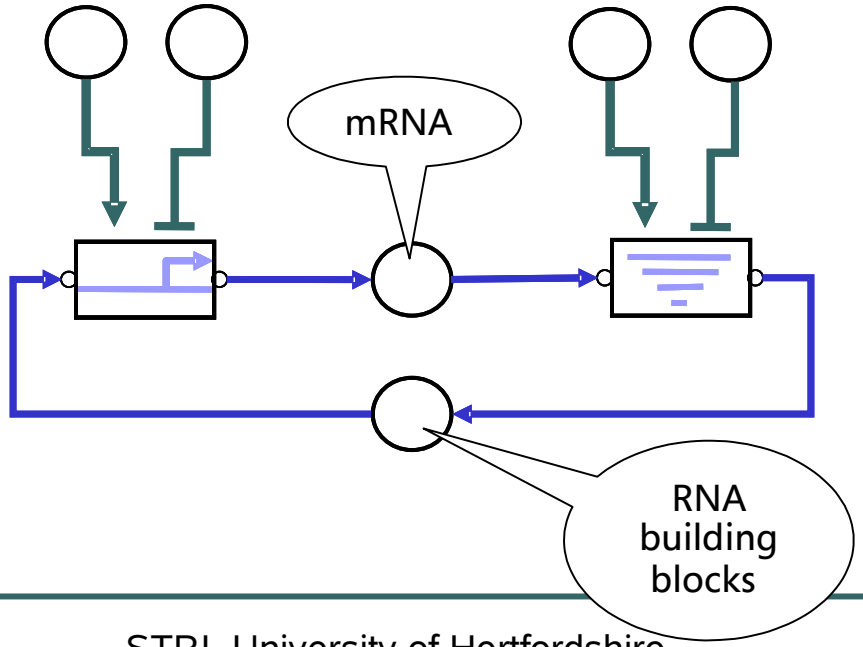


- No limit to number of activators or repressors
- Protein that acts as an activator for one gene may act as a repressor for another

GRN Dynamics

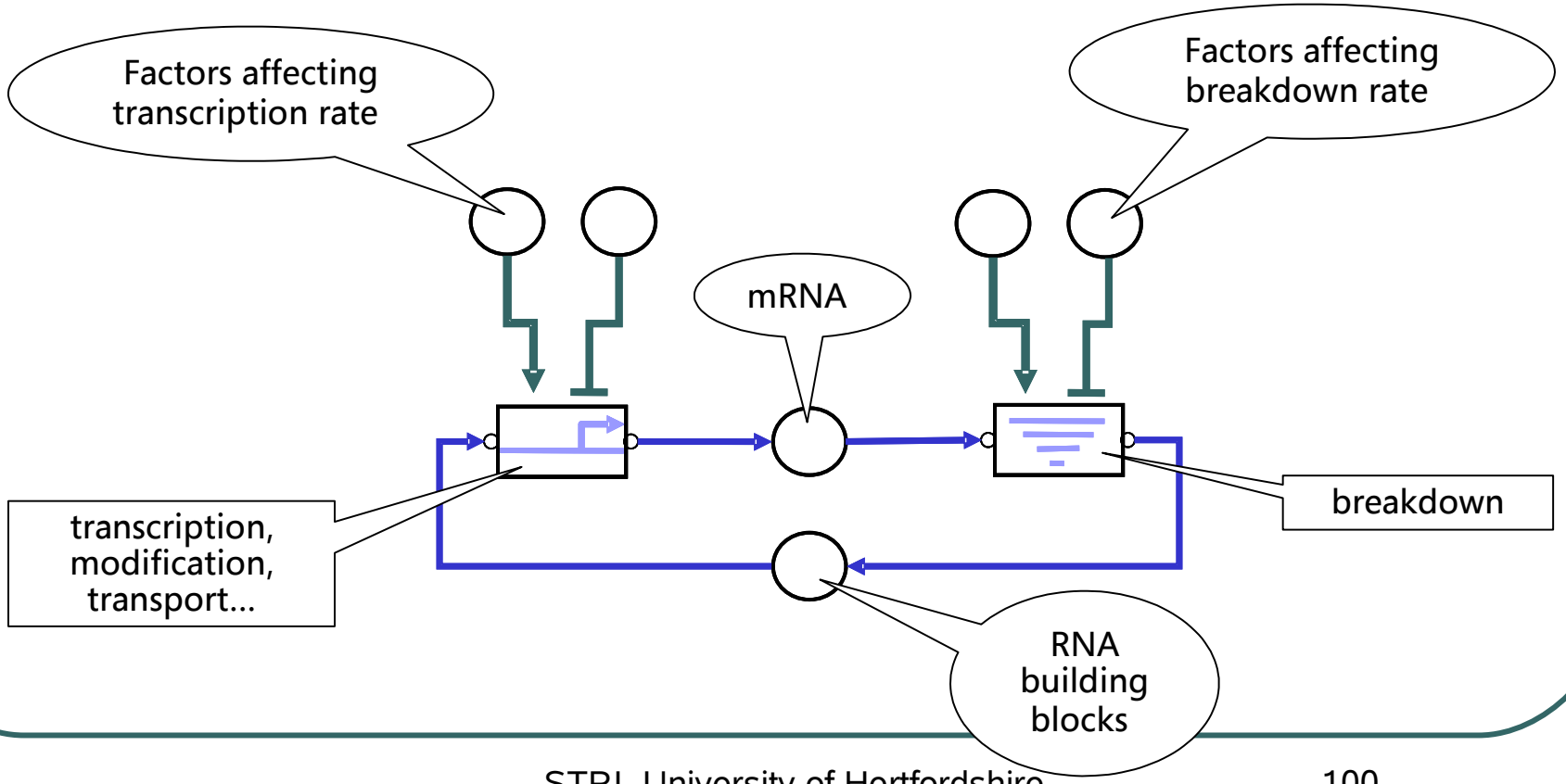
Dynamics: Petri-net notation

Messenger RNA dynamics

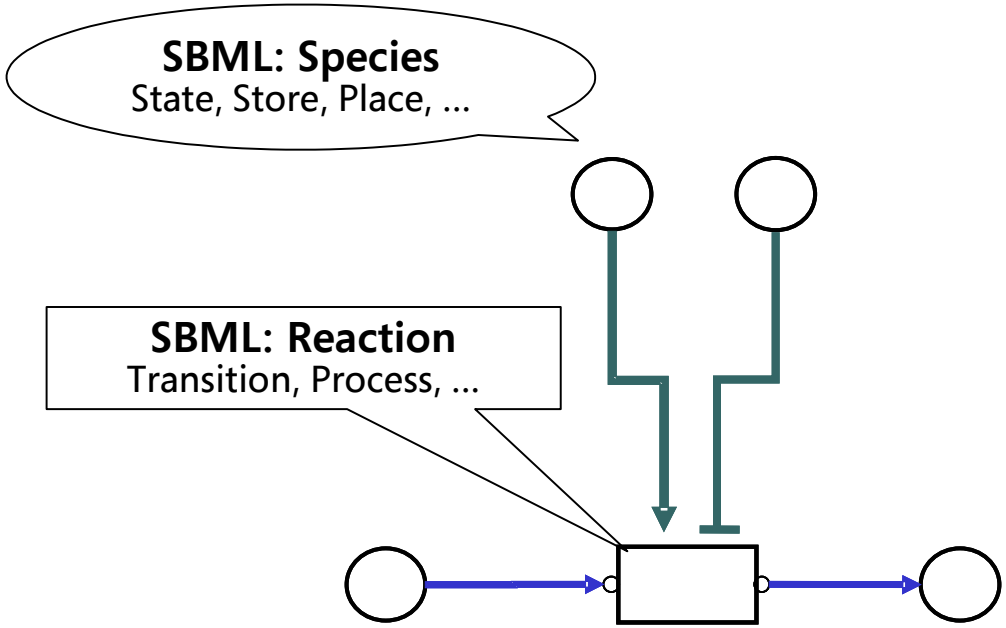


Dynamics: Petri-net notation

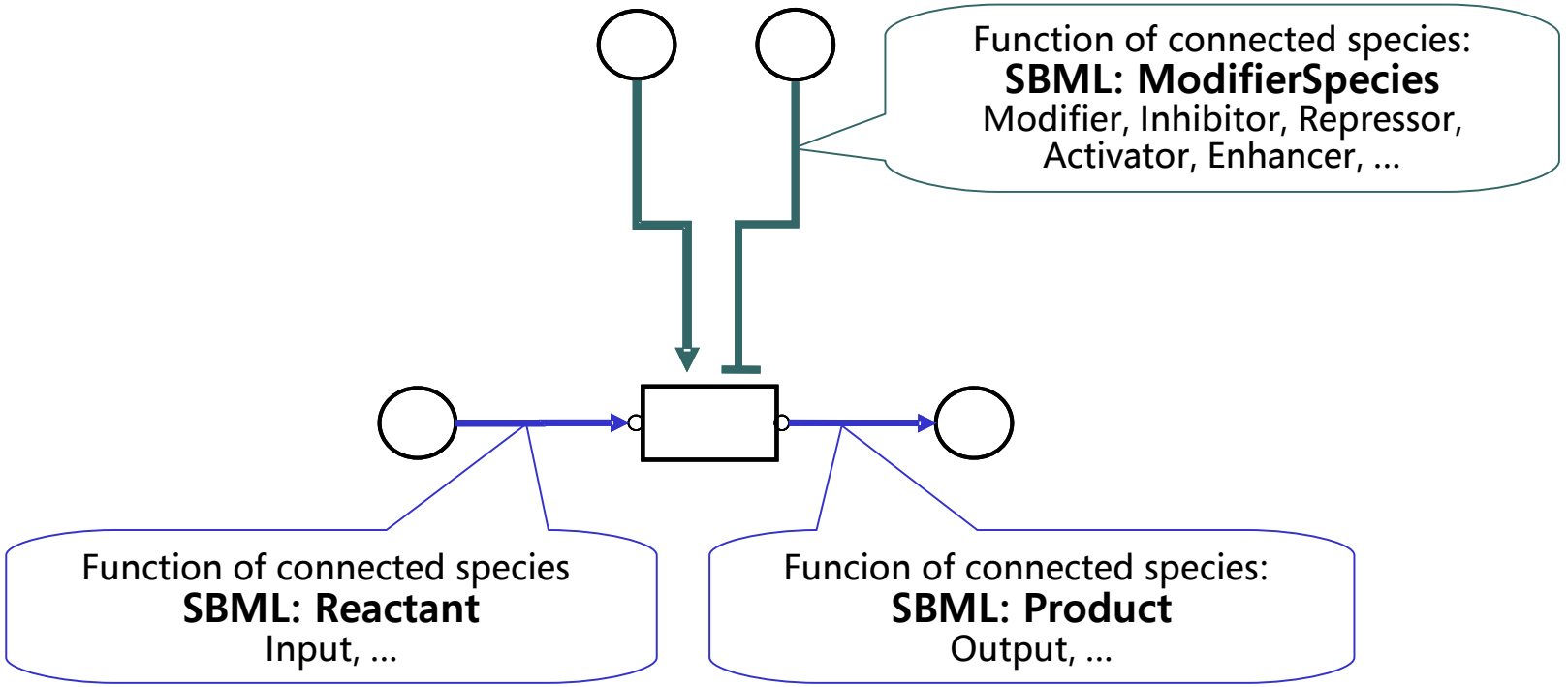
Messenger RNA dynamics



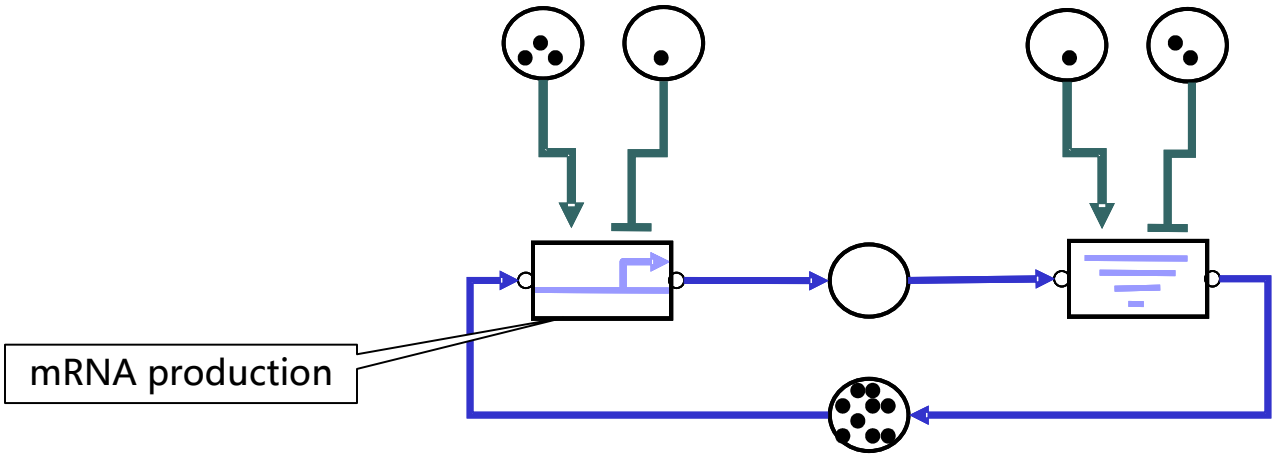
Symbols: Nodes



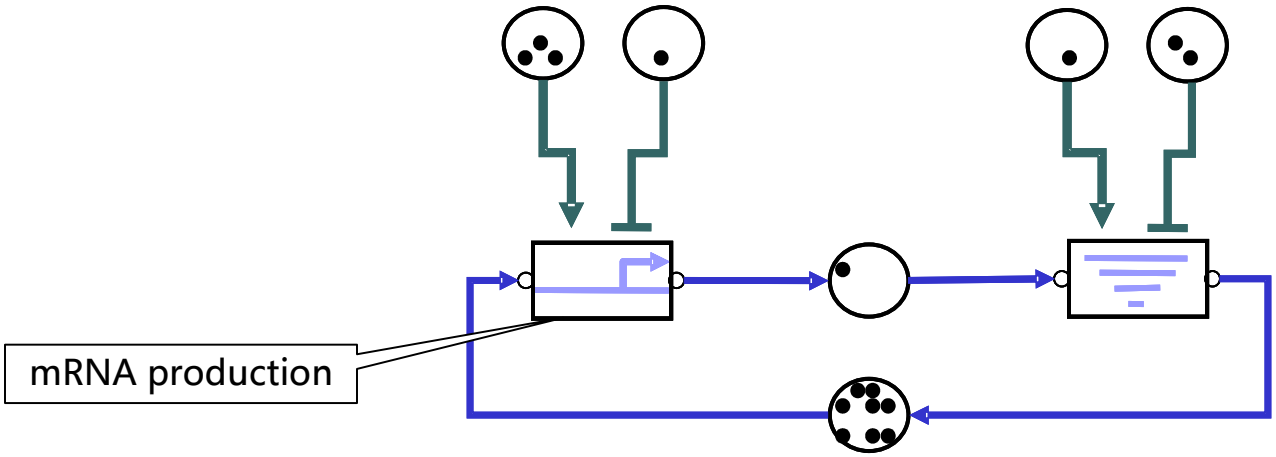
Symbols: Arrows



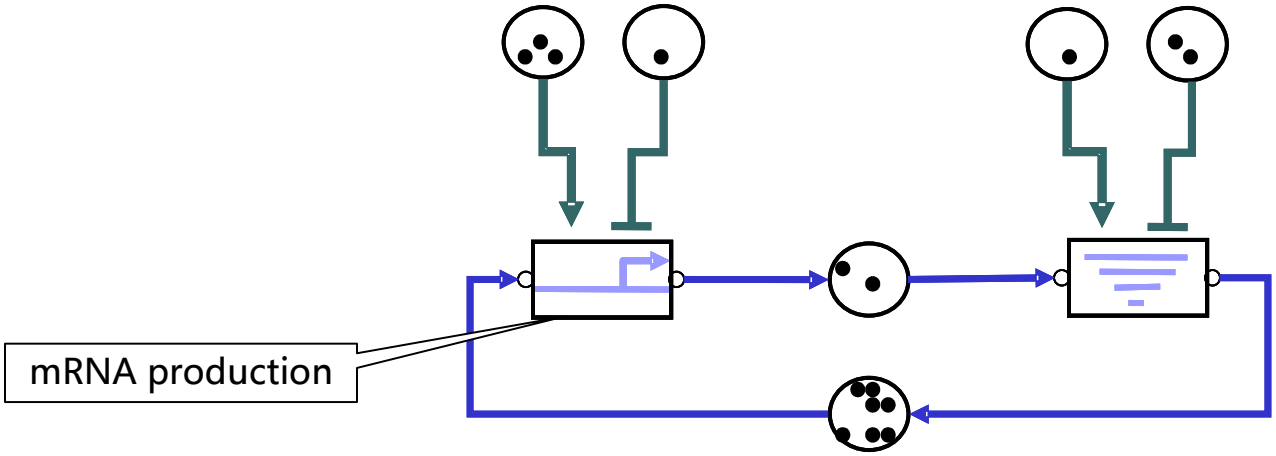
Dynamics: what happens?



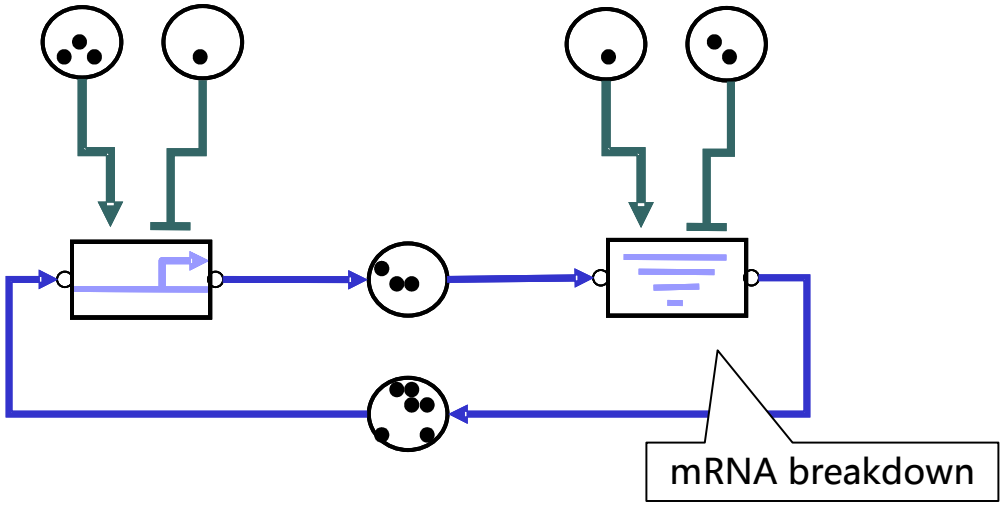
Dynamics: what happens?



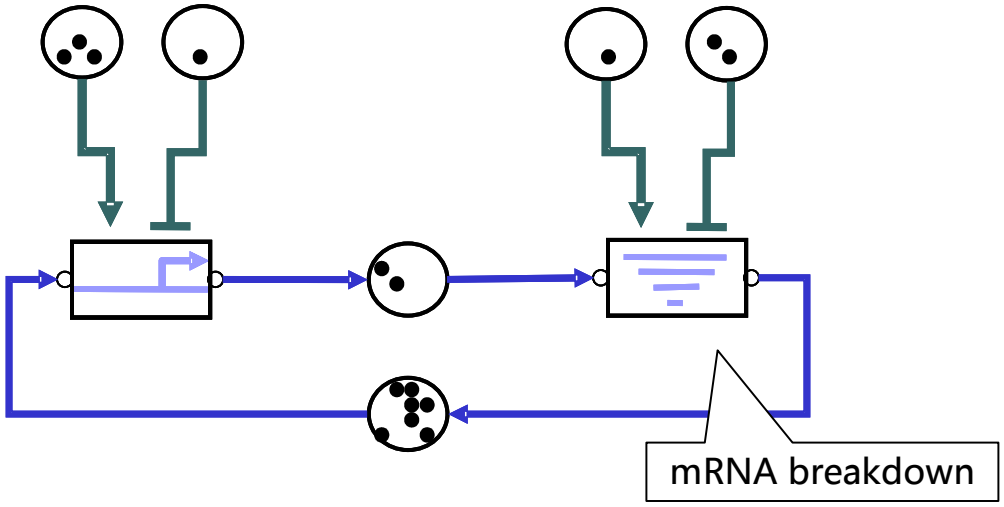
Dynamics: what happens?



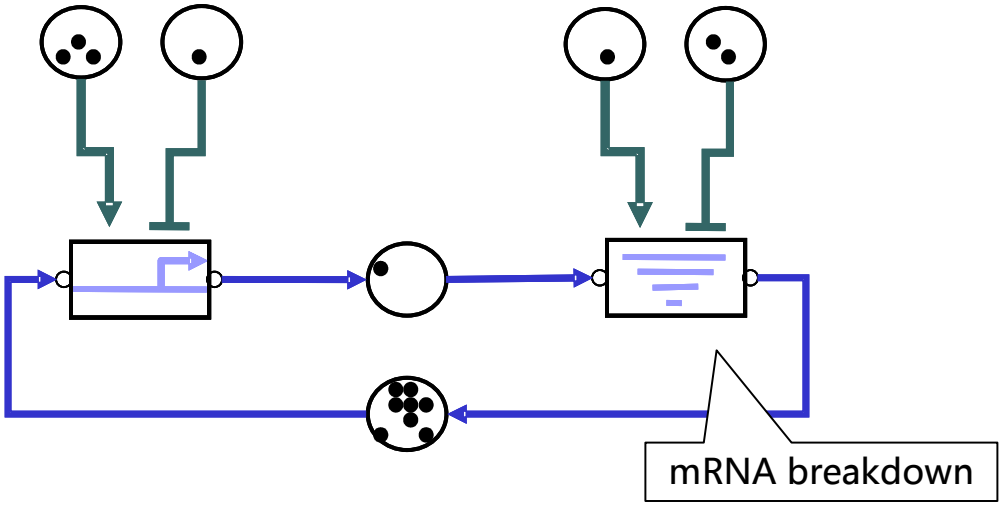
Dynamics: what happens?



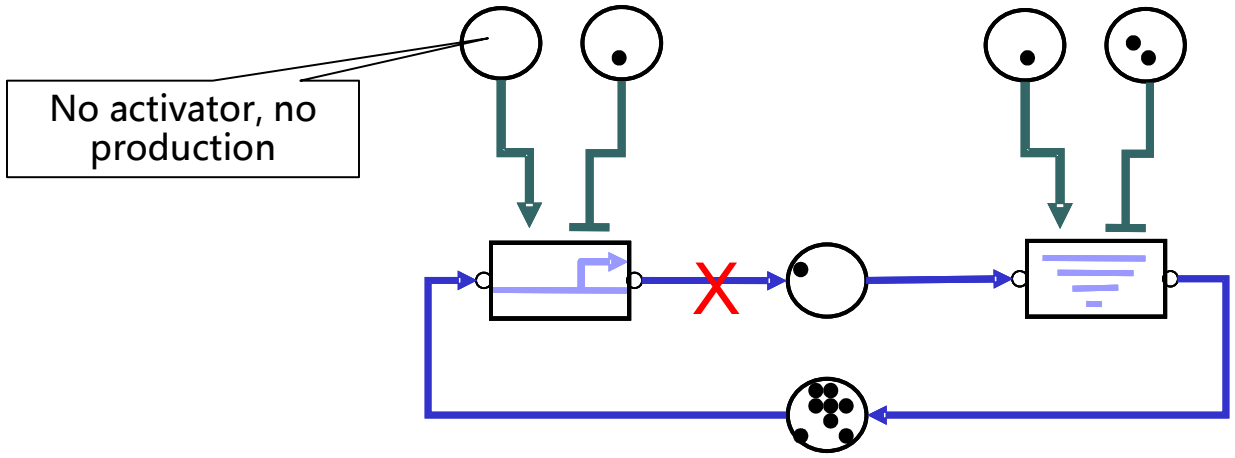
Dynamics: what happens?



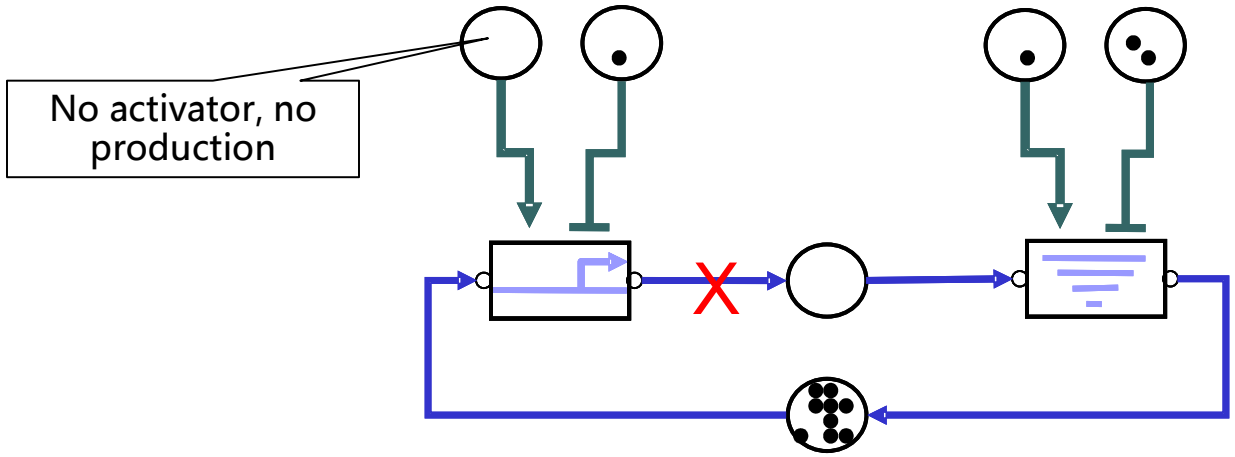
Dynamics: what happens?



Dynamics: what happens?



Dynamics: what happens?

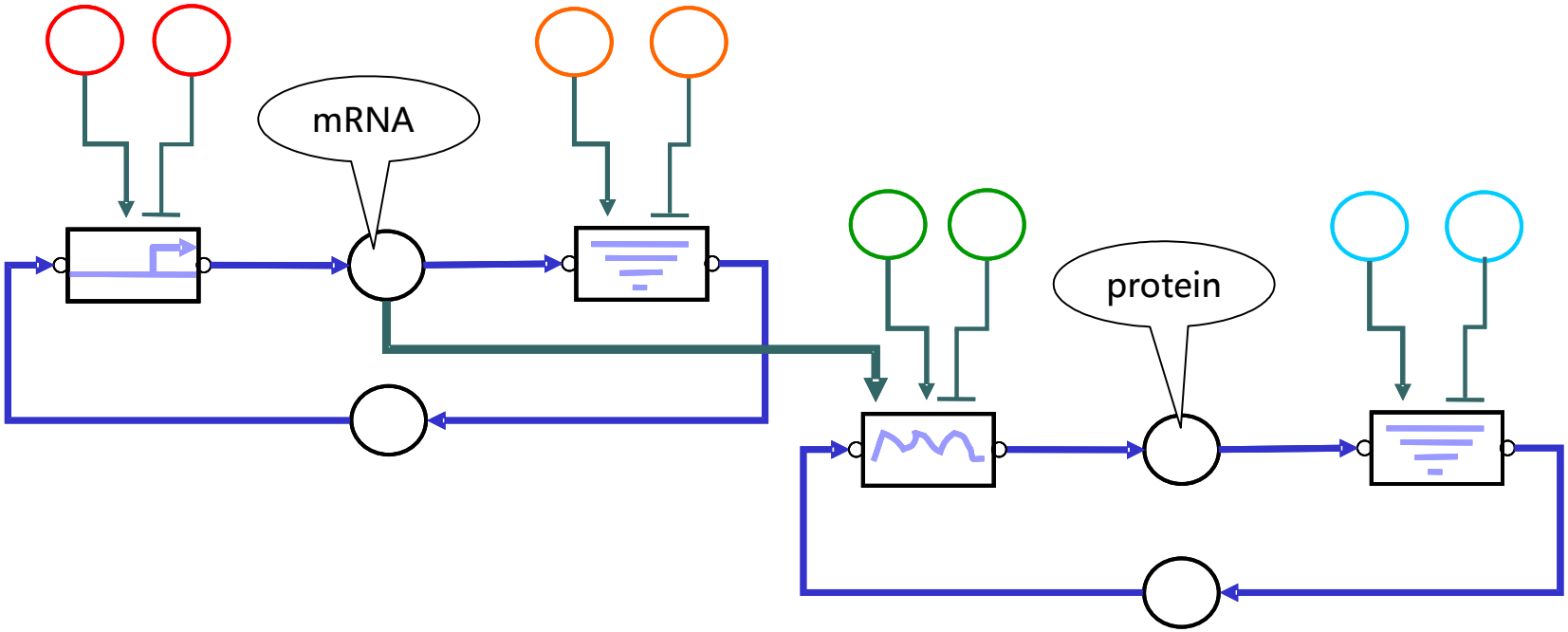


Worth remembering (1)

- Reactant - Consumed during reaction
  - Reactant stoichiometry > 0
- Product - Produced during reaction
  - Product stoichiometry > 0
- Modifier - Neither: only affects reaction *rate*
  - Stoichiometry = 0
  - Dependence defined in the reaction's rate equation (SBML: KineticLaw; transfer function, …)
  - Shape of rate equation depends on process (and how much we know about it…)

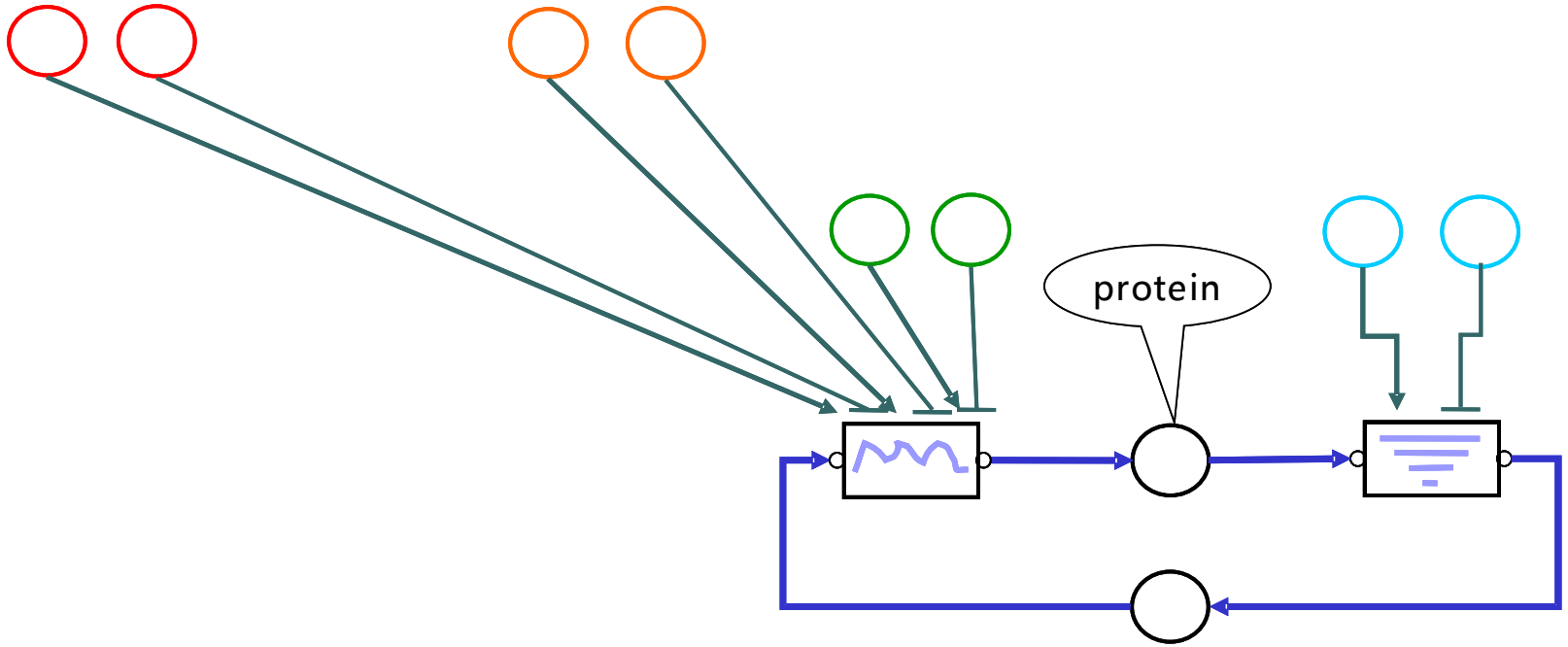
Dynamics

Messenger RNA and protein product dynamics



Dynamics

Messenger RNA and protein product dynamics



Worth remembering (2)

- Constant production without breakdown:
  - Amount will increase linearly over time
- Production plus breakdown:
  - Amount will reach plateau (usually; there are exceptions)
- Time to reach plateau determined by breakdown rate, not production rate
  - In general: breakdown processes determine how rapidly a system can respond (adapt) to new external conditions

Combining control and dynamics

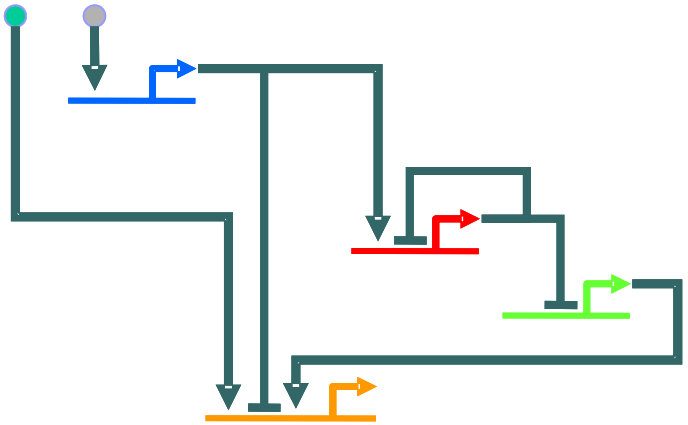
Creating a dynamic GRN model - Questions (1)

- Aim:
  - Model as “realistic” as possible? Computational tool? Control network?
- GRN constituents
  - Genes (if so: what do they represent)? Transcription factors? Other regulators (e.g. regulatory RNA)? Intermediates (e.g. mRNA)? Signals?
- Processes to incorporate
  - Transcription? Translation? Breakdown? Signalling?
- Multicellularity

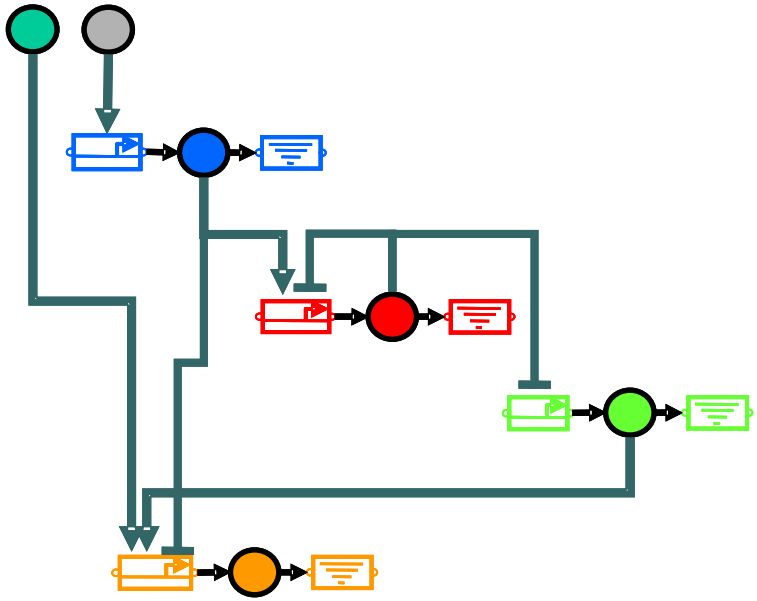
Creating a dynamic GRN model – Questions (2)

- Equations that describe process dynamics
  - Mass-action type rate equations?
- Rules for (combined) effects of regulatory interactions
  - Saturable? Additive? Logical?
- Numerical representation of component values
  - Continuous? Discrete multivalued? Boolean (if so: what do 0 and 1 represent)?
- Representation of time
  - Continuous? Discrete?
- Evaluation method
  - Numerical integration of rate equations (if so: stochastic or deterministic)? Finite state automaton? Boolean switching?

Basic GRN model



Control



Dynamics

Possible dynamic representation

Rate equations

Product production rate:

$$v_p = k_p \times f(\text{modifiers})$$

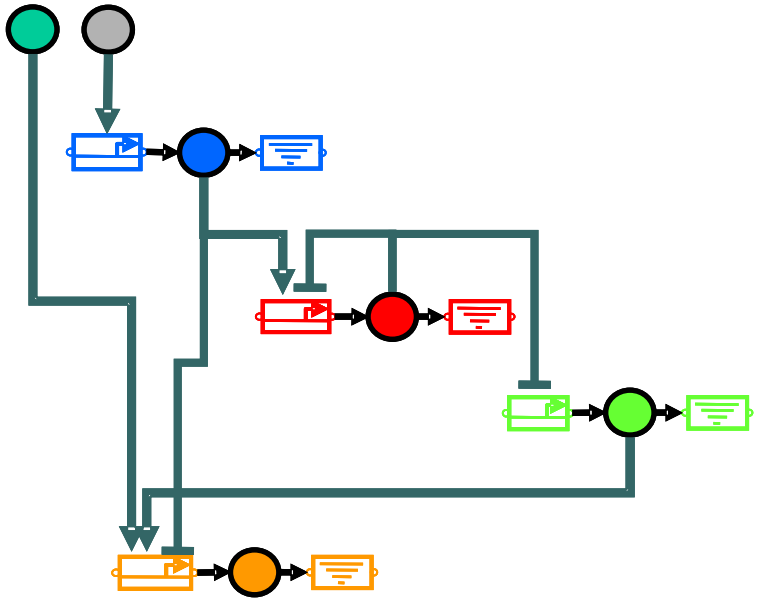
Product breakdown rate:

$$v_b = k_b \times [P]$$

Plateau value (steady state):

$$[P] = \frac{k_p}{k_b} f(\text{modifiers})$$

k_p, k_b: production and breakdown rate constants
[x]: concentration or amount of species x
P: gene product



Rules for activation and repression

Saturable activation:

$$\mu^{S,A} = \frac{m[M]^p}{1 + m[M]^p}$$

Saturable repression:

$$\mu^{S,R} = \frac{1}{1 + m[M]^p}$$

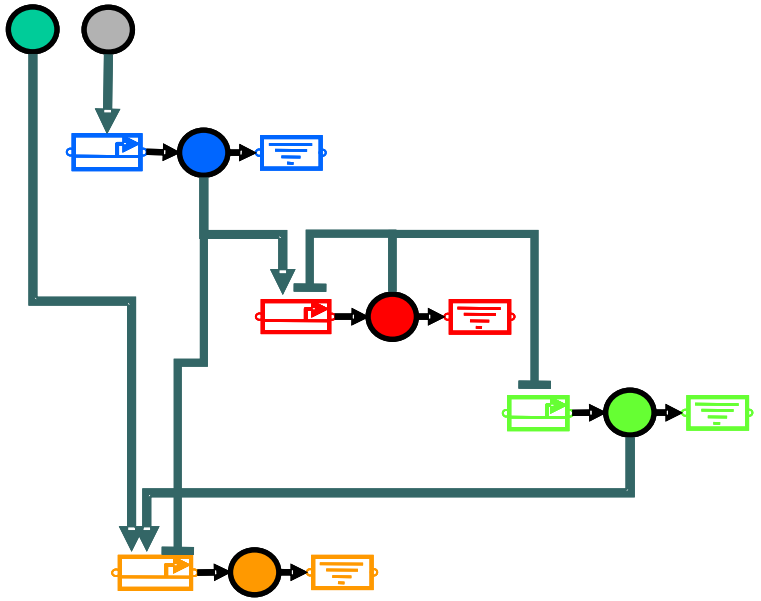
Linear activation:

$$\mu^{L,A} = m[M]^p$$

Linear repression:

$$\mu^{L,R} = -m[M]^p$$

m: multiplier
p: exponent



Rules for combinatorial modifier effects

Logical disjunction ("or") (saturable):

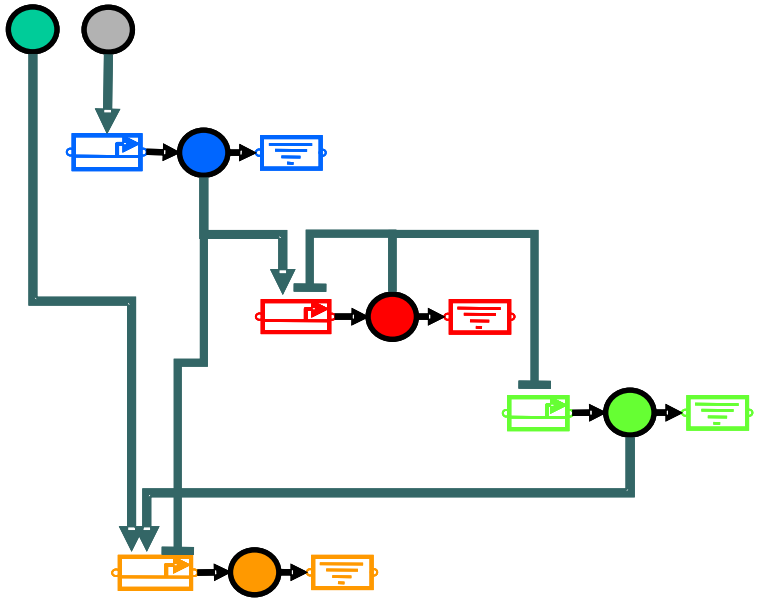
$$f(\text{modifiers}) = \mu_1^S + \mu_2^S(1-\mu_1^S)$$
$$f(\text{modifiers}) = \mu_1^S \vee \mu_2^S$$

Logical conjunction ("and") (saturable):

$$f(\text{modifiers}) = \mu_1^S \times \mu_2^S$$
$$f(\text{modifiers}) = \mu_1^S \wedge \mu_2^S$$

Addition, multiplication (linear):

$$f(\text{modifiers}) = \mu_1^L + \mu_2^L$$
$$f(\text{modifiers}) = \mu_1^L \times \mu_2^L$$



Note on GRN representation

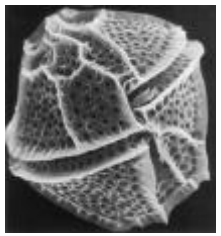
- No single “proper” representation method
- Chosen method depends on aim of study and personal preference
- However, be aware of:
  - the way your representation relates to the “real thing”
  - the type of simplifications that have been made
- Tips:
  - usually, amounts or concentrations do not have negative values
  - simple negative feedback systems are often oscillatory, but the oscillations may well disappear when using smaller time steps, or a less crude representation

Part 3: Evolving biological clocks

with artificial genetic regulatory networks (aGRNS)

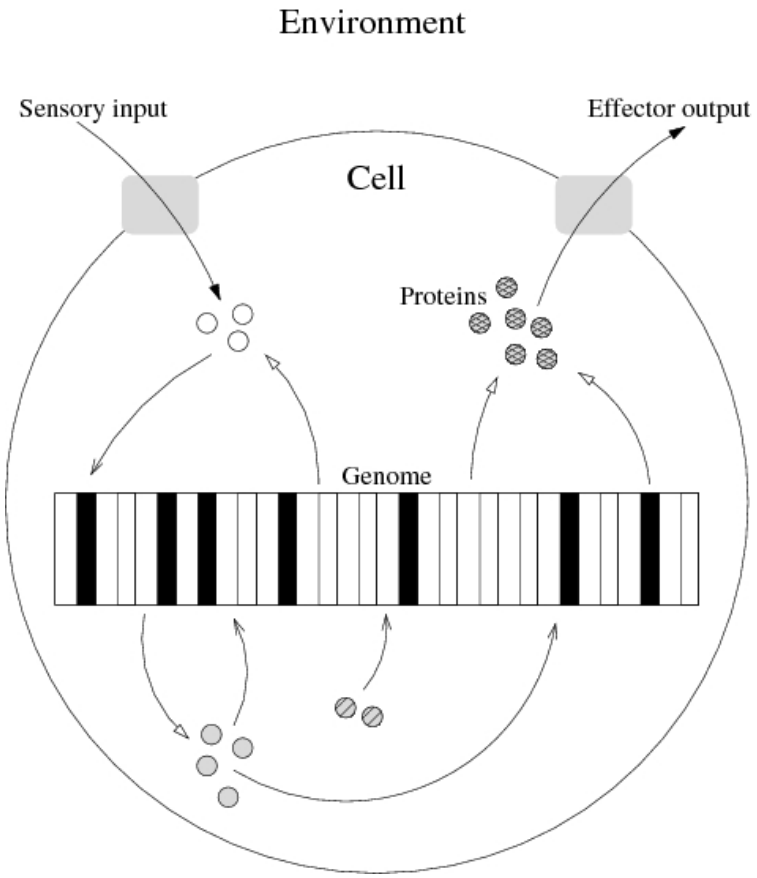
Introduction

- Biological clocks abound in all organisms, even the simplest single celled ones like *Gonyaulax'* (red tide organism):
  - characteristic example of responsiveness of life on earth



- Evolvability of Genetic Regulatory Networks (GRNs) as a paradigm for novel computation
  - developmental mapping
  - parallelism

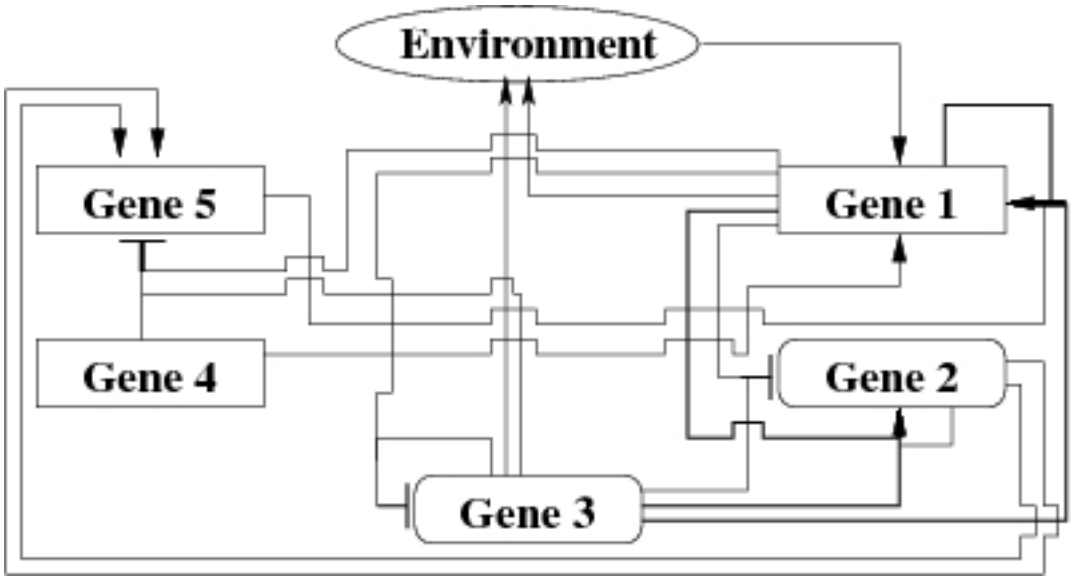
Schematic drawing



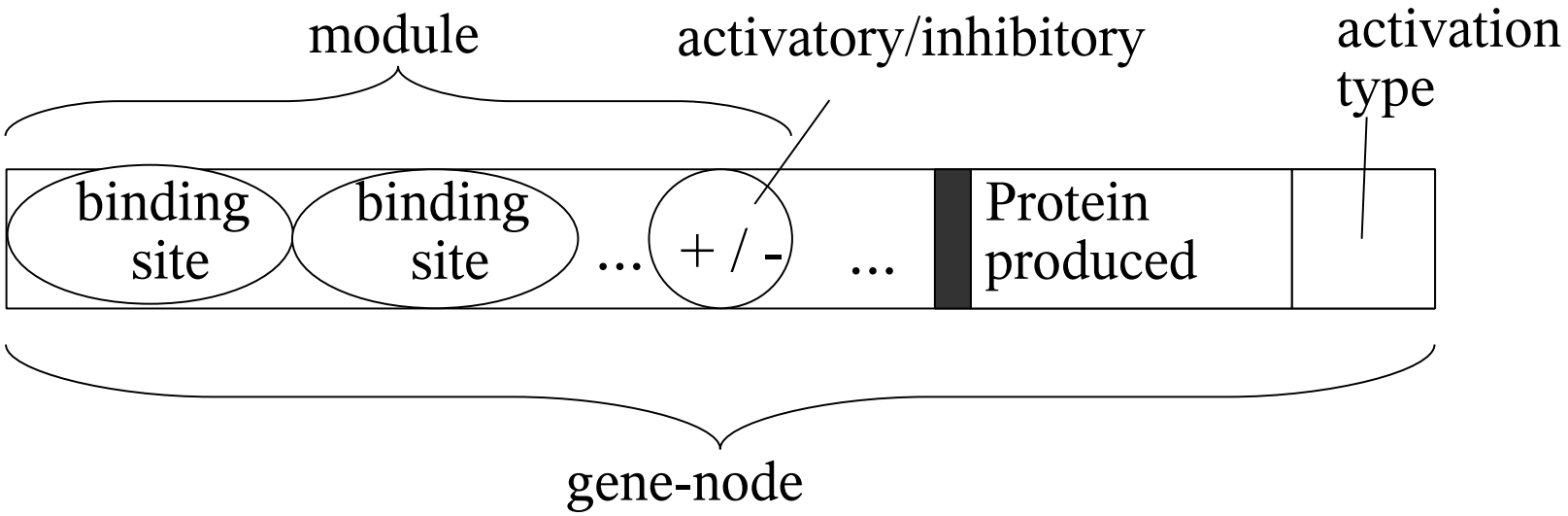
- Note: no proper genotype/phenotype distinction is made in this drawing!

Schematic drawing - Phenotype

- A “gene” is a gene-node in a regulatory network here
- This is the phenotype!!!



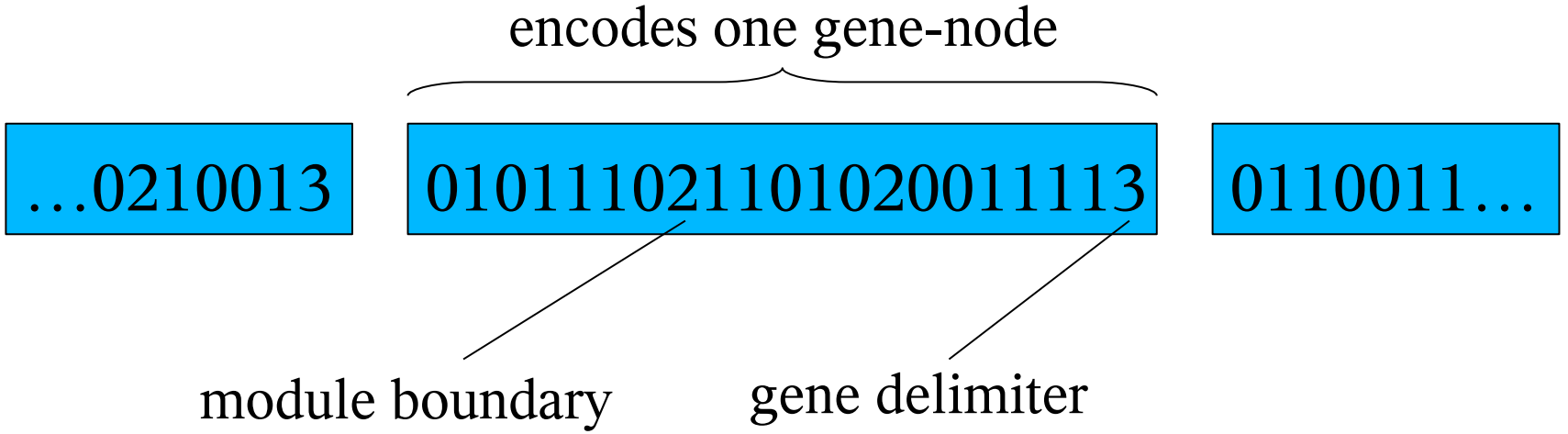
What does a gene-node consist of?



- ***binding sites***: protein input from other genes
- ***module***: "grouping" of inputs
- *any number* of binding sites, any number of modules
- one protein output type and one activation type

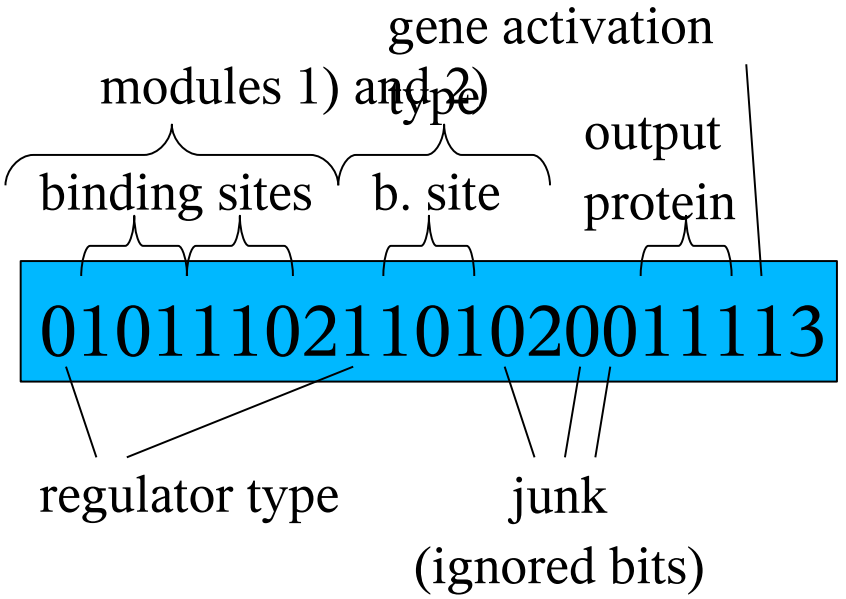
Encoding of gene-nodes and genome

- Genome consists of a number of genes plus a compartment coding parameters global to the cell
- '0'/'1' code, '2' module boundary, '3' gene boundary; used for compartmentalization



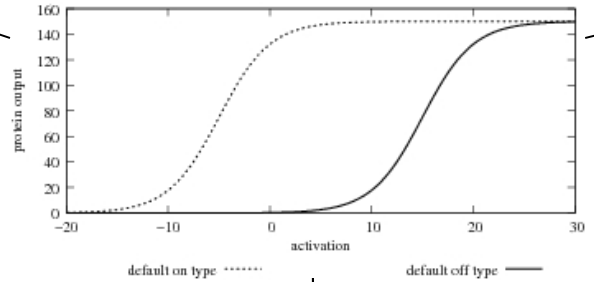
Encoding of gene-nodes and genome – cont'd

- Two modules:
  1) inhibitory (starting with 0) binding a combination of proteins 5 (101) and 6 (110)
  2) activatory cis-module (starting with 1) to which protein 5 (101) will bind.
- Last three zeros of 11010200 are ignored → junk.
- Will produce protein 7 (111) and is off by default (last bit is 1).

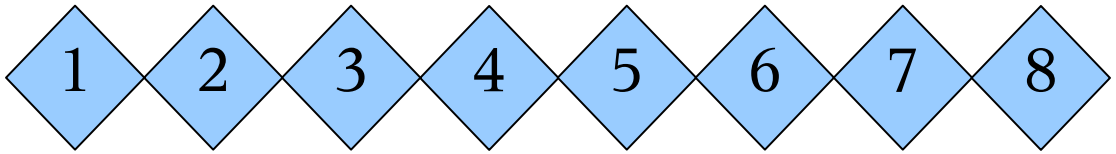


Activation

- binding sites: **AND**( )
- modules: **OR**( )
- gene (activiation function):



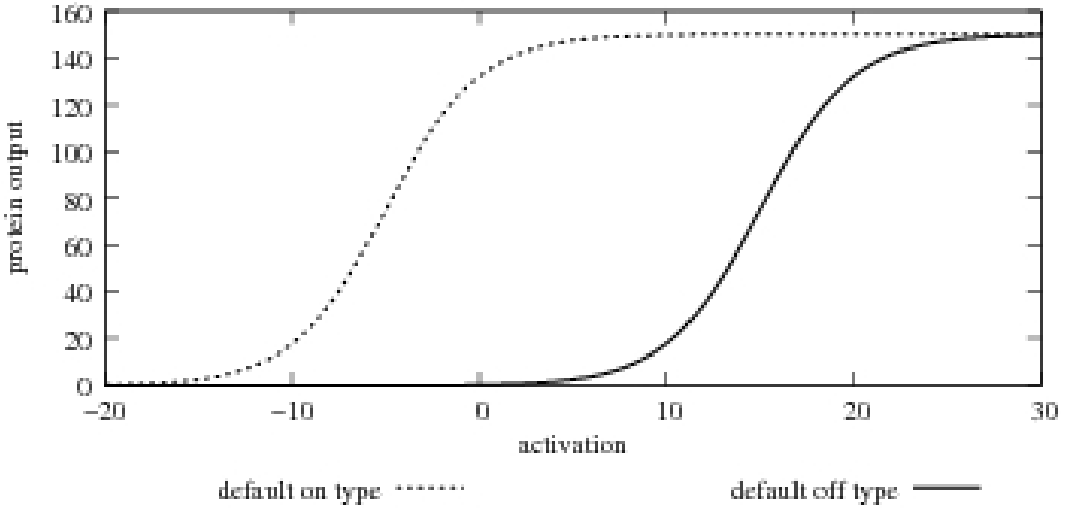
activation increases one protein level



- protein levels: (global, 8 here)

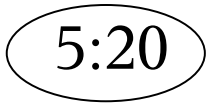
Activation – cont'd

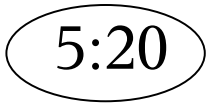
- As protein values are not boolean, AND is actually minimum and OR is a sum, but effect very similar

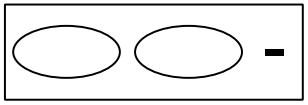


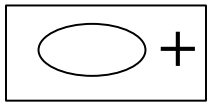
- every gene is either of constituitive ("default on", dotted line) or induced ("default off", continuous line) type

Gene-node activation example

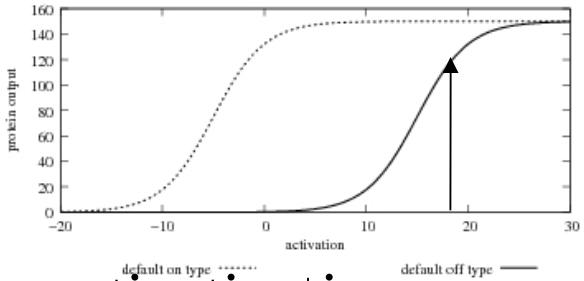
- protein 5: 20 per site and type 6: 1 per site bind
- binding sites: 

[=1] 

[=20]
- modules: 

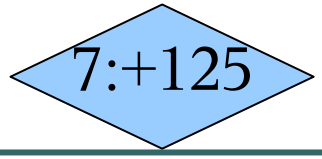
[-1] 

[+20]
- gene: (activation function f)



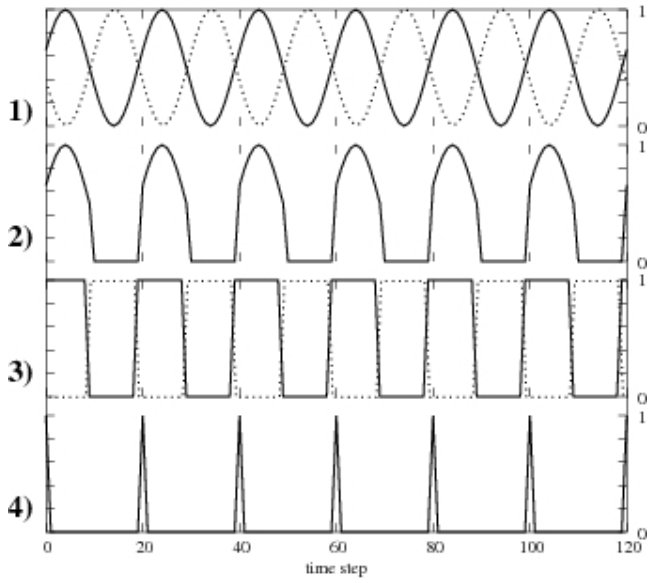
[f(-1+20)=~125]

activation increases

- protein level: 

Environmental in-/output

- simple protein in-/output
- periodic functions used:
  - input 1)-4), 400 time steps
  - wavelength 20 time steps
- variations with perturbations:
  - +/- noise with std. dev. 0.1
  - +/- 2x blackout of 20 steps
- desired behaviors 1) or 3)
  closeness of match = fitness

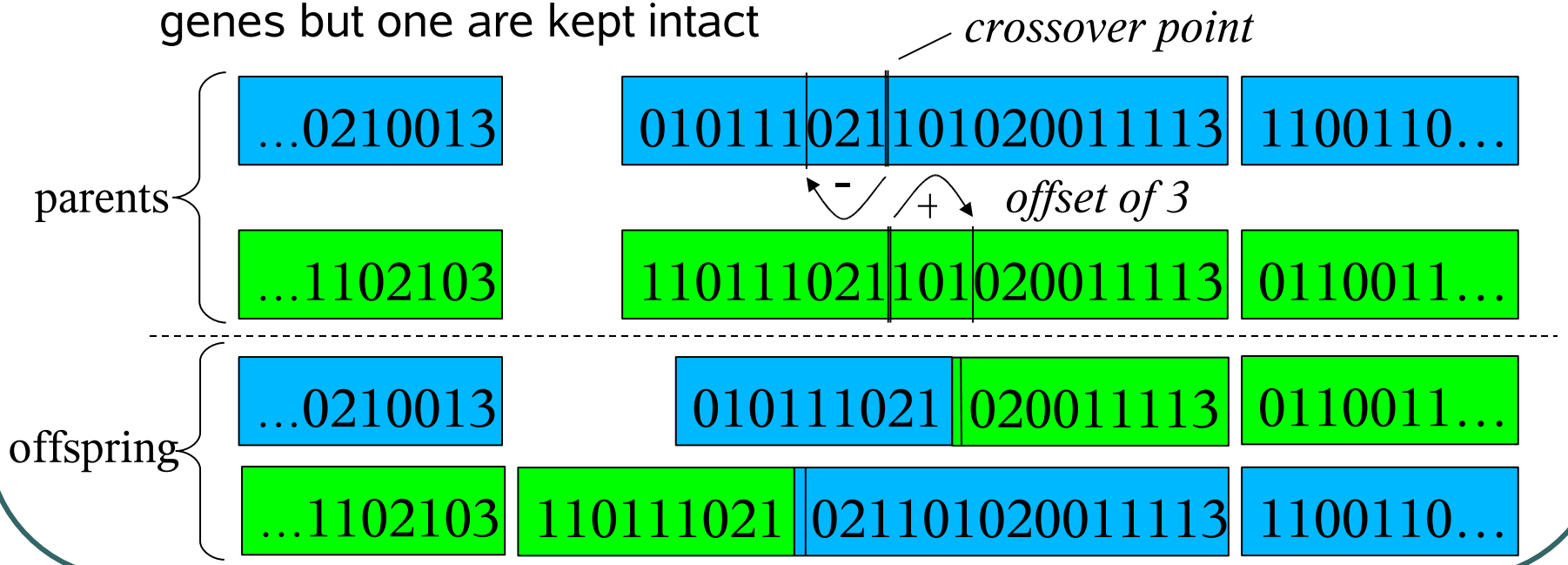


Selection and Variation

- Tournament selection:
  - 15 individuals randomly picked from population, best two of them chosen as parents
  - weak elitism (only the best individual copied over)
- Mutation:
  - 1% chance for every bit to flip 0->1, 1->0
- Recombination:
  - unequal crossing over, always two parents for two children
  - Length of genes might change, number of genes is held constant in these experiments

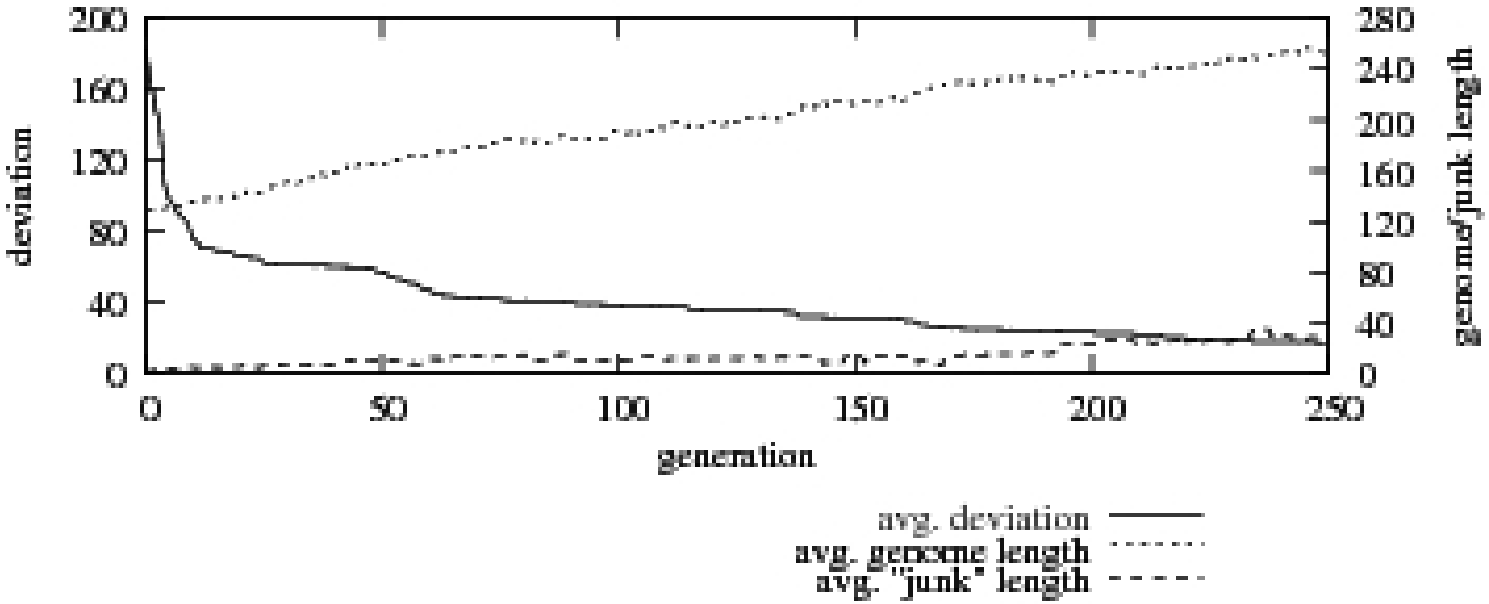
Recombination

- **Unequal** crossing-over allows for genomes of varying length, important for varying number of binding sites and modules
- Unequal crossover point is shifted by an offset
- Note that offset always stays within the compartment, so all genes but one are kept intact



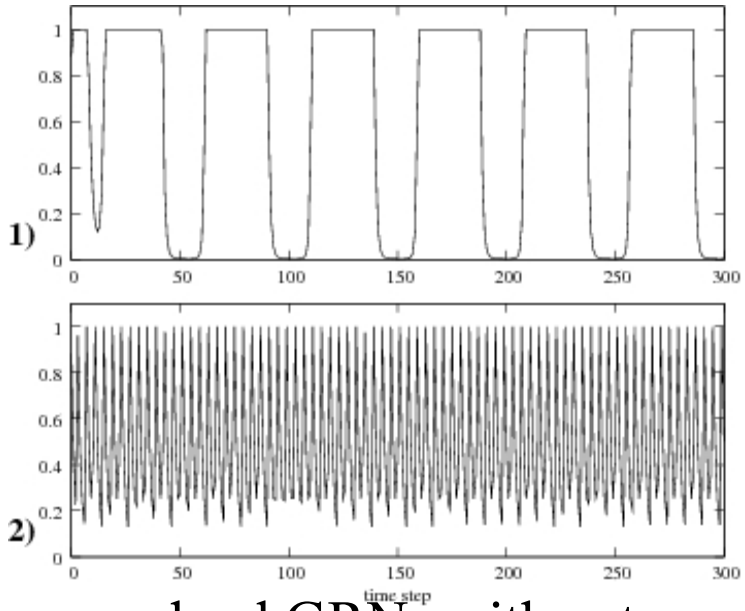
Population development example

- Can easily evolve to show cyclic behavior
- Genome length and junk length increase on average (average over 10 runs)



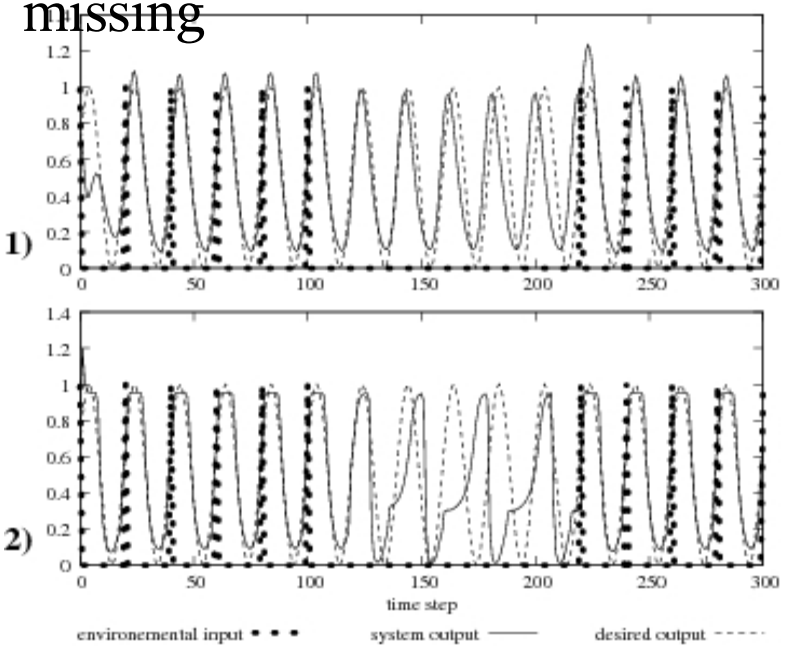
Individual dynamics analysis

- internalization of (quasi-) periodic behaviour in many cases
- the more unreliable the input the more this was found

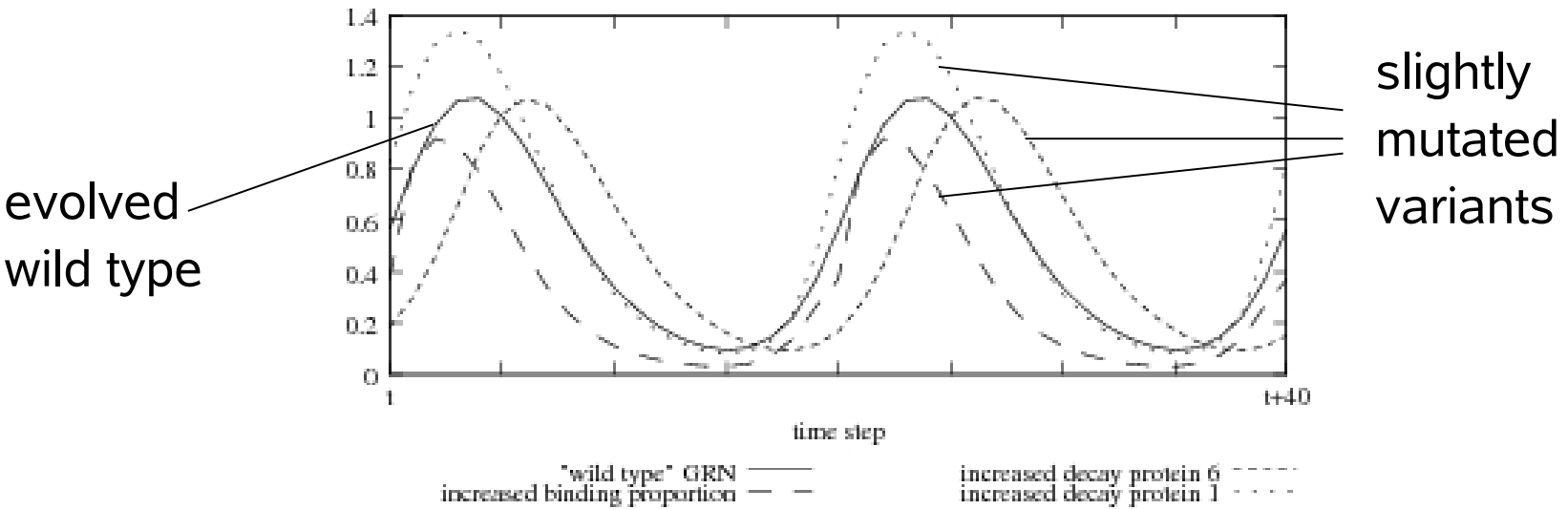


two evolved GRNs with extremely different internalized wavelengths

two evolved GRNs getting out of synchrony when stimuli are missing



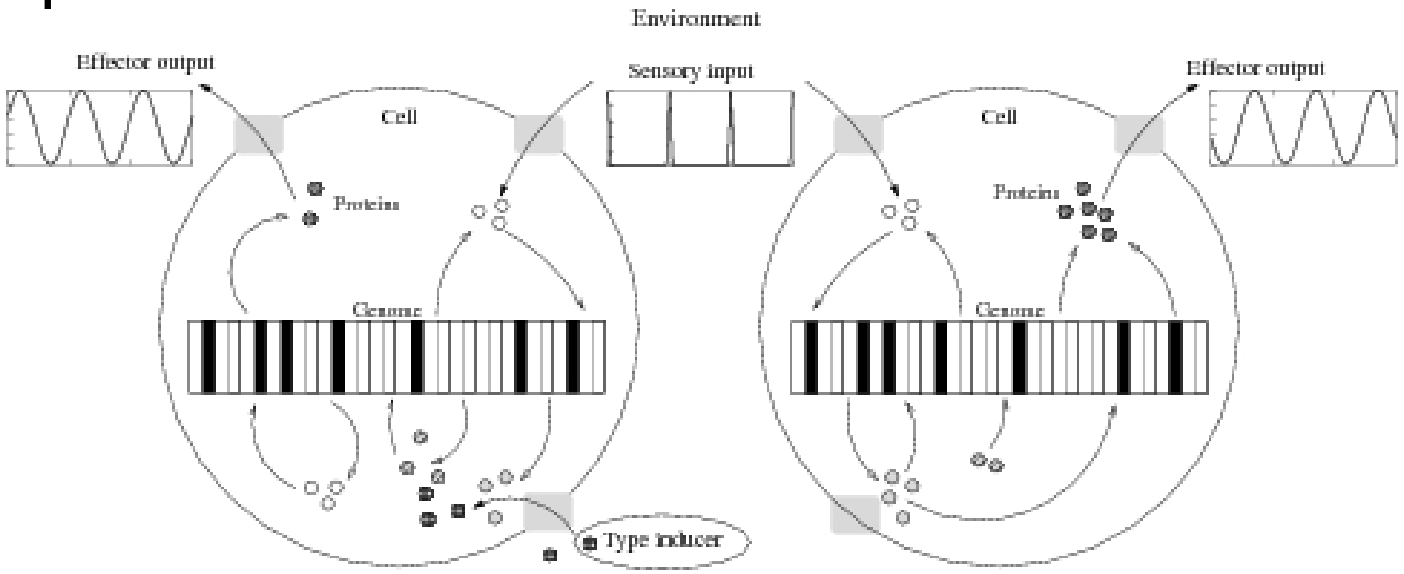
Evolvability - Heterochronicity



- all variants are one or two bit flips away from each other
- can allow heterochronic control: changes in timing are possible while preserving general dynamics
- remember: small genotype changes usually should cause small phenotype changes for smooth adaptation

Differentiation – schematic

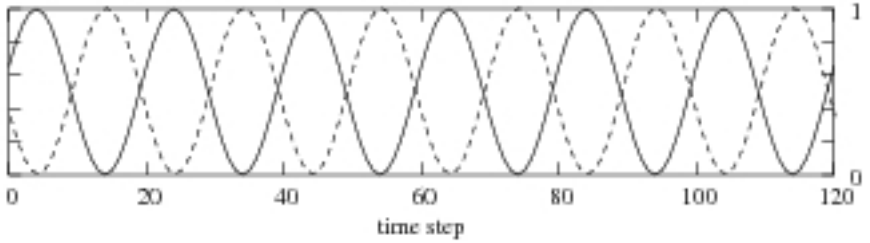
- Individual has two cells with same genome and almost same input, but different behavior required



- Note: Again no clear seperation of genotype and phenotype in this drawing!

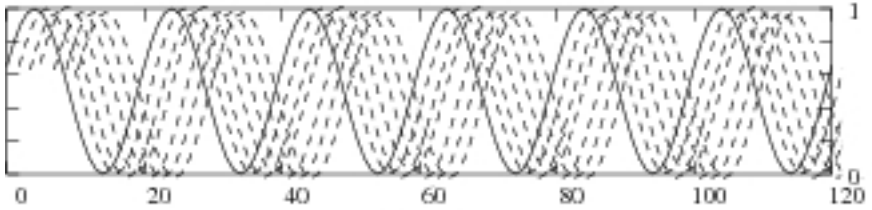
Differentiation: two pathways

- Same ultimate goal functions (periodic and inverse)
- but different fitness evaluations over initial generations
  - **Immediate setting:**



fitness is final objective from beginning: one cell reproduces phase of input while the other has to produce inverse

  - **gradual setting:**

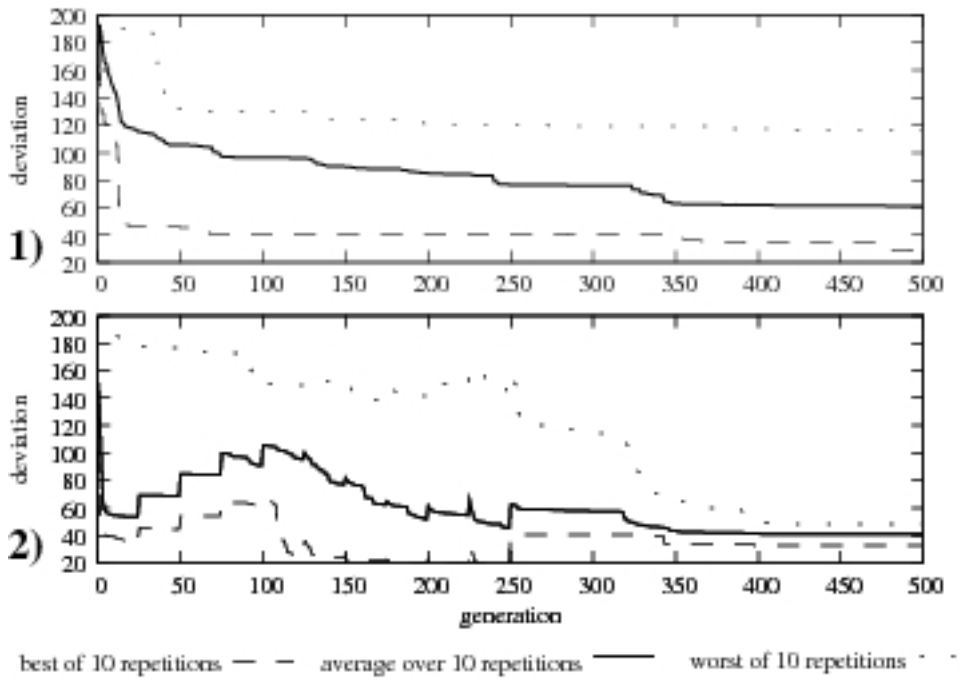


fitness is initially the same phase reproduction from beginning, but target phase shifts for one of the cells over generations
→ changing fitness landscape!

Typical example runs for the two settings

1) immediate

6) gradual evaluation



- Best results similar, but average much better for 2), *robustly* finds good results

Part 4: Netbuilder'

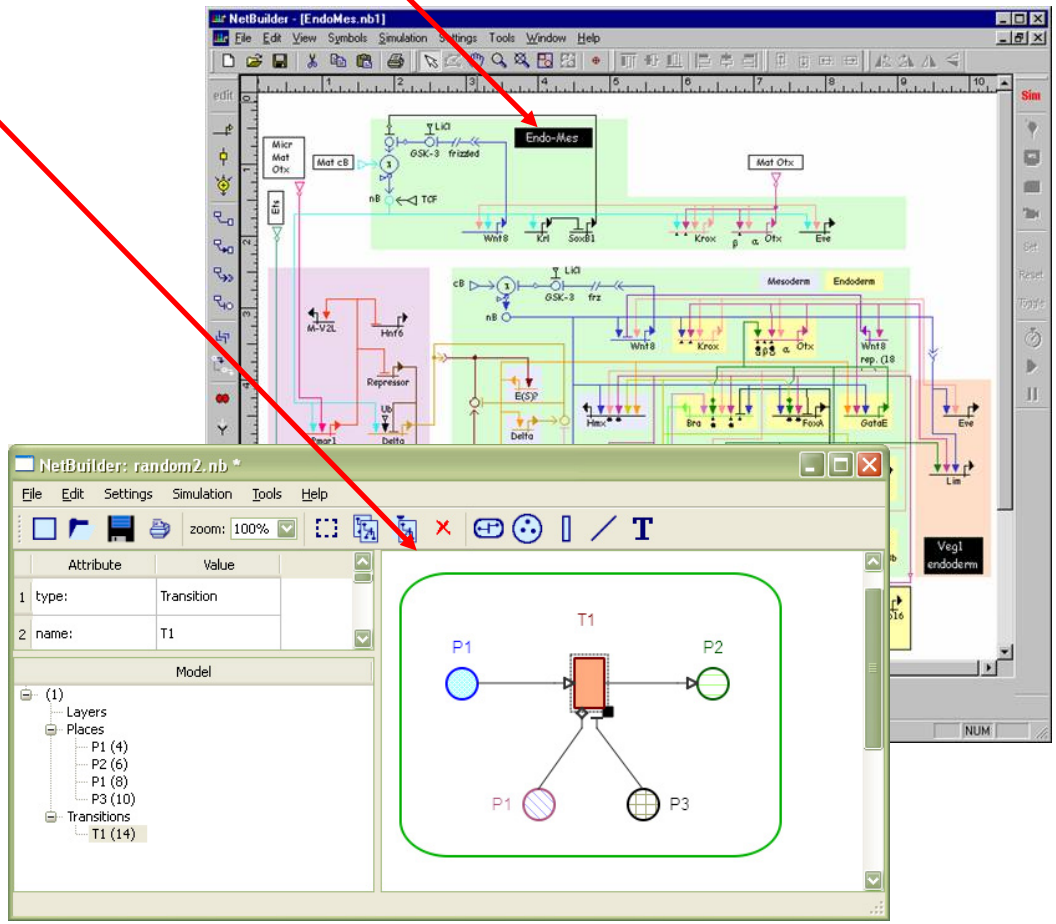
a tool for construction,
simulation and evolution
of GRNS

NetBuilder′ (Project Apostrophe)

- A tool for
  - construction
  - modelling
  - simulation (stochastic, deterministic, hybrid)
  - evolution
  - future: Analysis of GRNs
- Uses the Petri-net formalism
- Download:
  - http://strc.herts.ac.uk/bio/maria/Apostrophe/

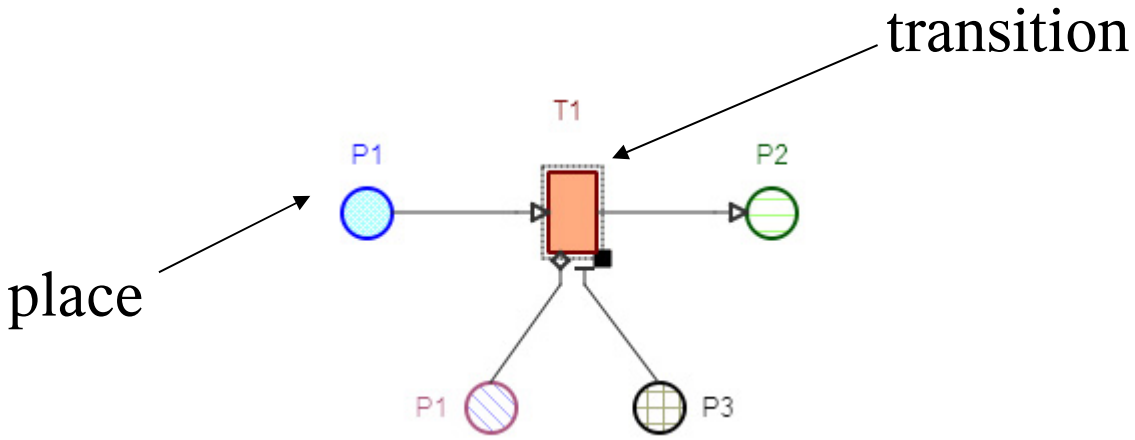
NetBuilder′ ≠ NetBuilder

- NetBuilder':
  - completely overhauled version
  - different model visualisation
  - more simulation and analysis methods

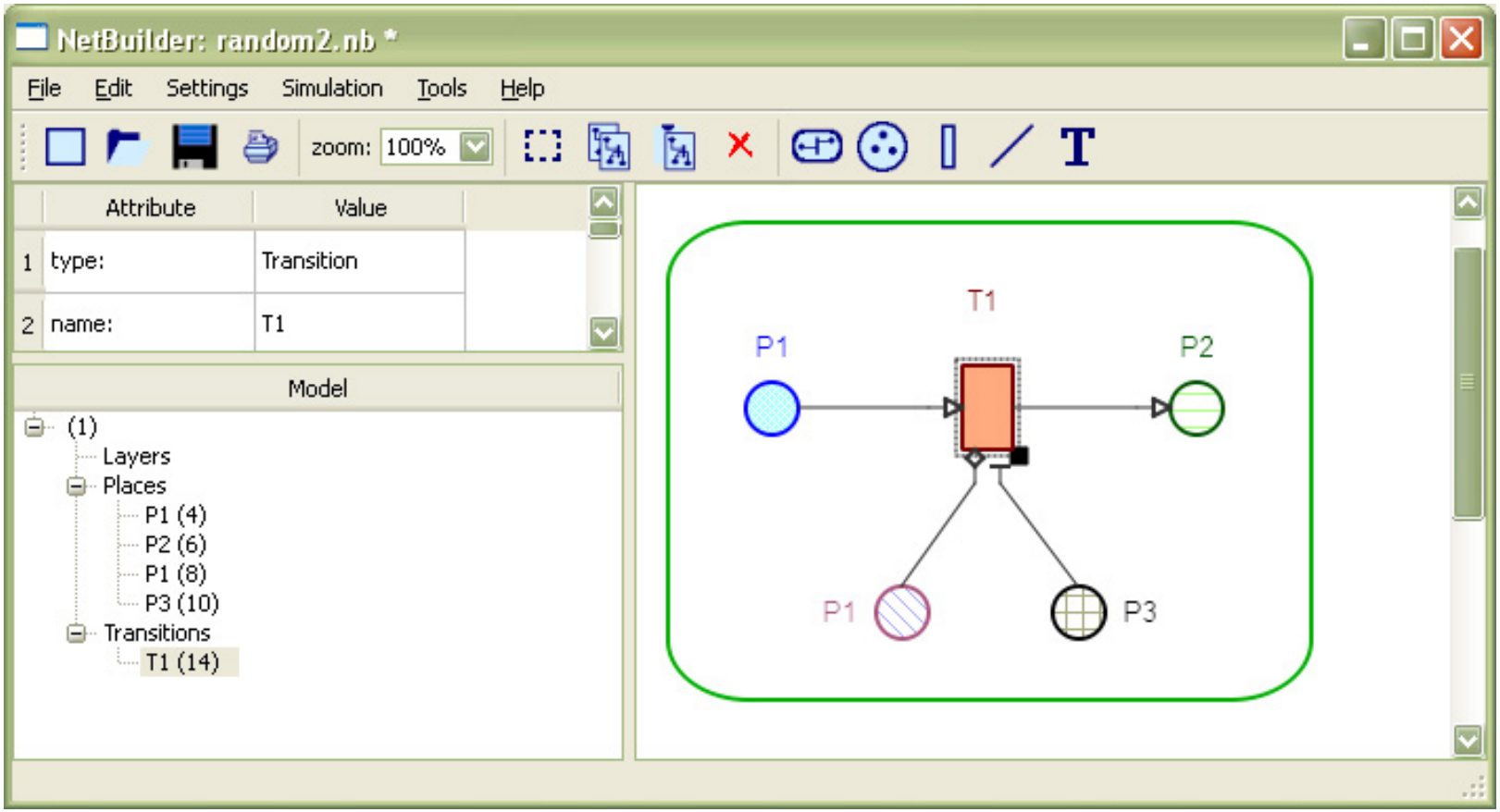


NetBuilder′ - Petri net

- bipartite graph
  - place – e.g. proteins (circle)
  - transition – e.g. reaction, gene (rectangle)
  - arc - connection between a place and a transition (what is consumed to produce what)

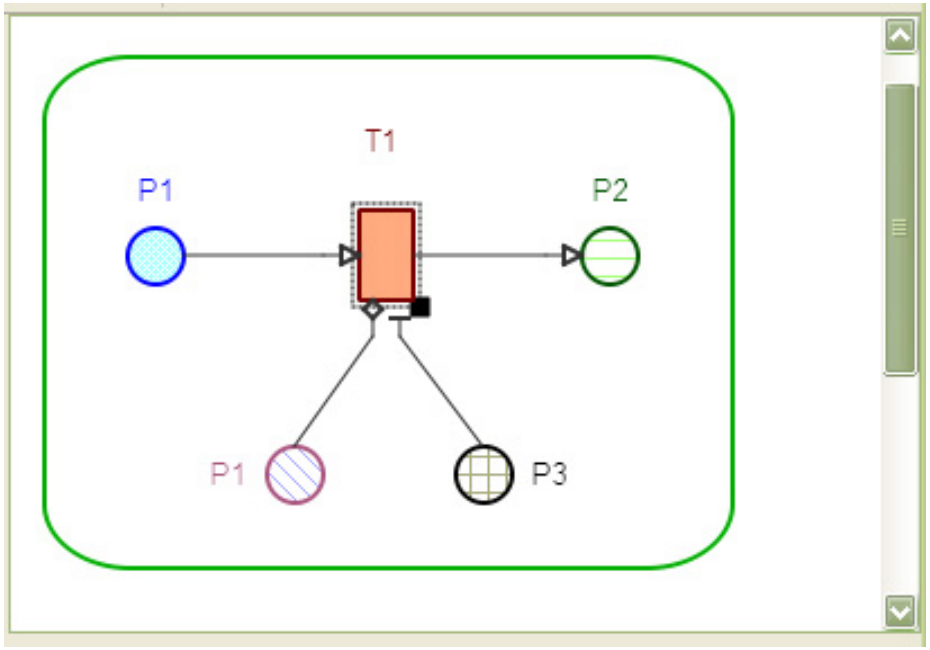


NetBuilder' - GUI



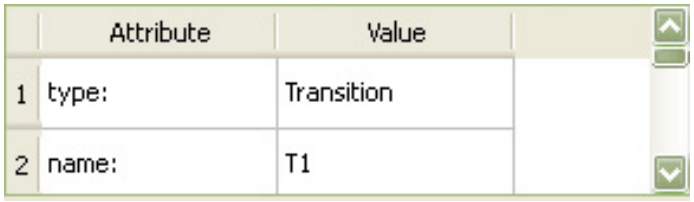
NetBuilder' - GUI

- Drawing area
  - arcs
  - layers
  - places
  - transitions
  - text objects



NetBuilder' - GUI

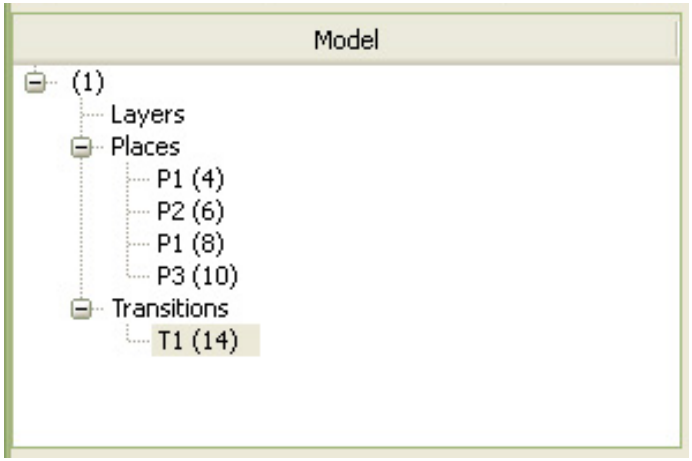
- Table of attributes of an selected object



	Attribute	Value
1	type:	Transition
2	name:	T1

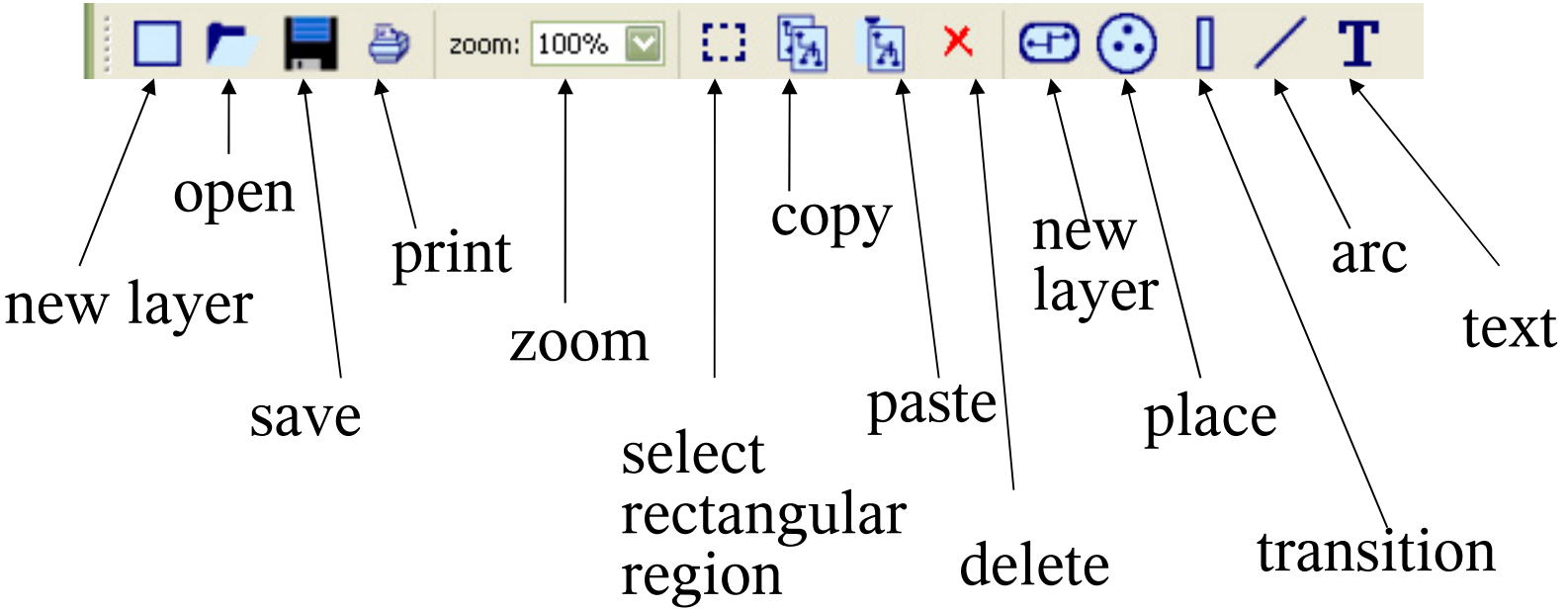
NetBuilder' - GUI

- Model hierarchy
  - overview about which place/transition belongs to which layer



NetBuilder' - GUI

- Tool bar

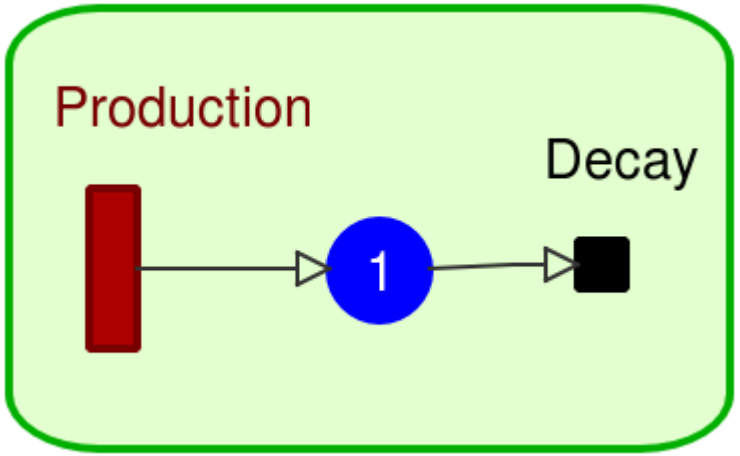


Evolution in NetBuilder'

- Based on Johannes' GA
- Differences:
  - *Genotype*: list of arcs, places, transitions, parameters
  - *Phenotype*: Petri net
  - *Mapping*: Petri net construction
  - *Fitness*: likeness to target function (sum square error)

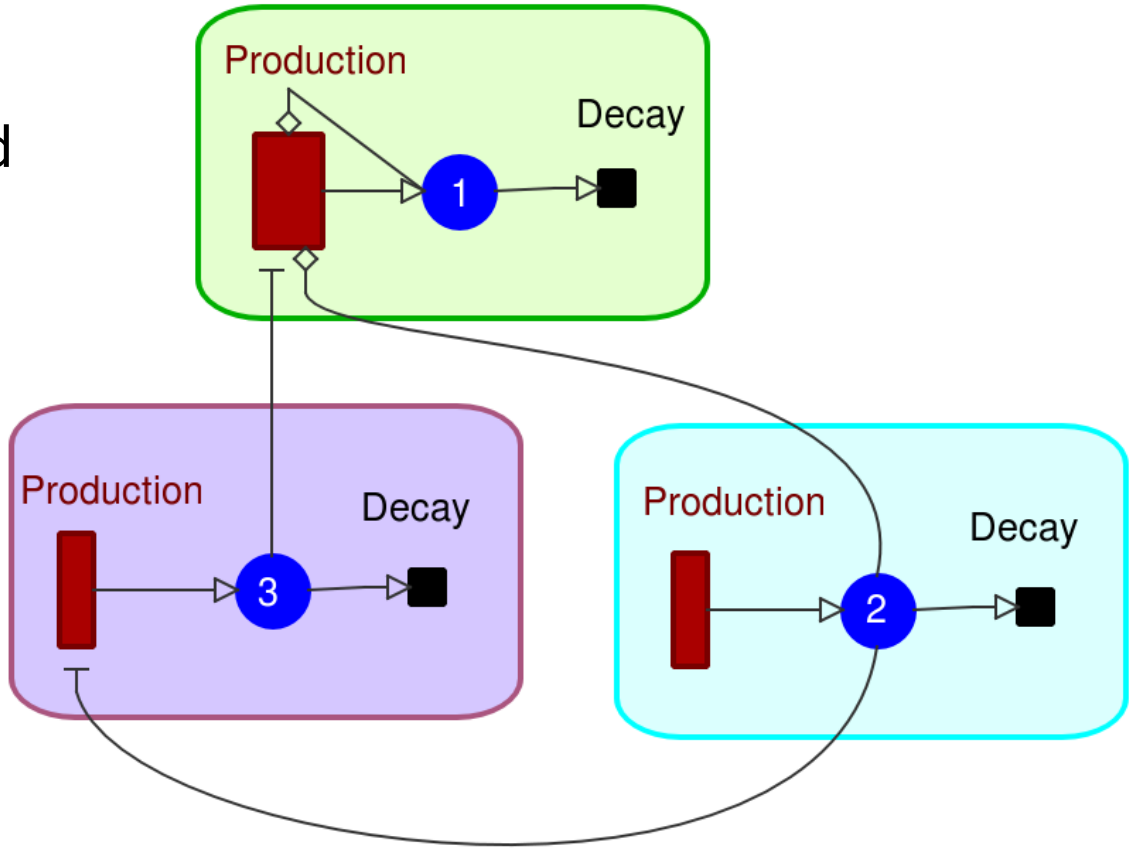
Genotype - Phenotype

- **Gene**:
  - These arcs or nodes in a gene are fixed and cannot be removed by evolutionary operators but attributes can be adjusted



Genotype – Phenotype (2)

- Network:
  - Activation and inhibition of protein production between genes
  - *Changeable*



Furthermore

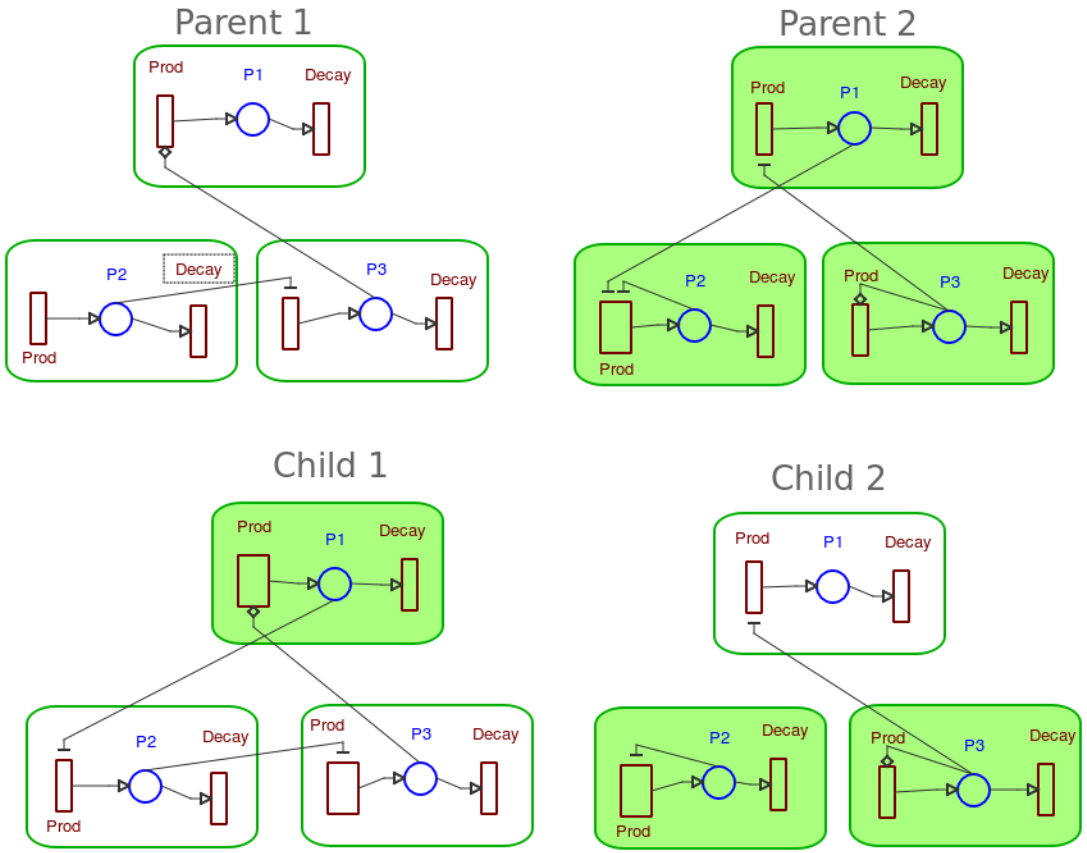
- **Remember**:
  - No limit to number of activators or repressors
  - Protein that acts as an activator for one gene may act as a repressor for another
- **Mathematical description:**
  - Automatically created by NetBuilder'
  - or users add their own function to each transition
- **Environmental input:**
  - Each place can have any input function (e.g. sine or step functions like in Johannes' GA)

Selection

- Tournament:
  - Select randomly 15 networks (default)
  - The two best networks of these 15 are recombined
- Elitism:
  - By default the best network is kept in the next generation (fitness never decreases)

Recombination

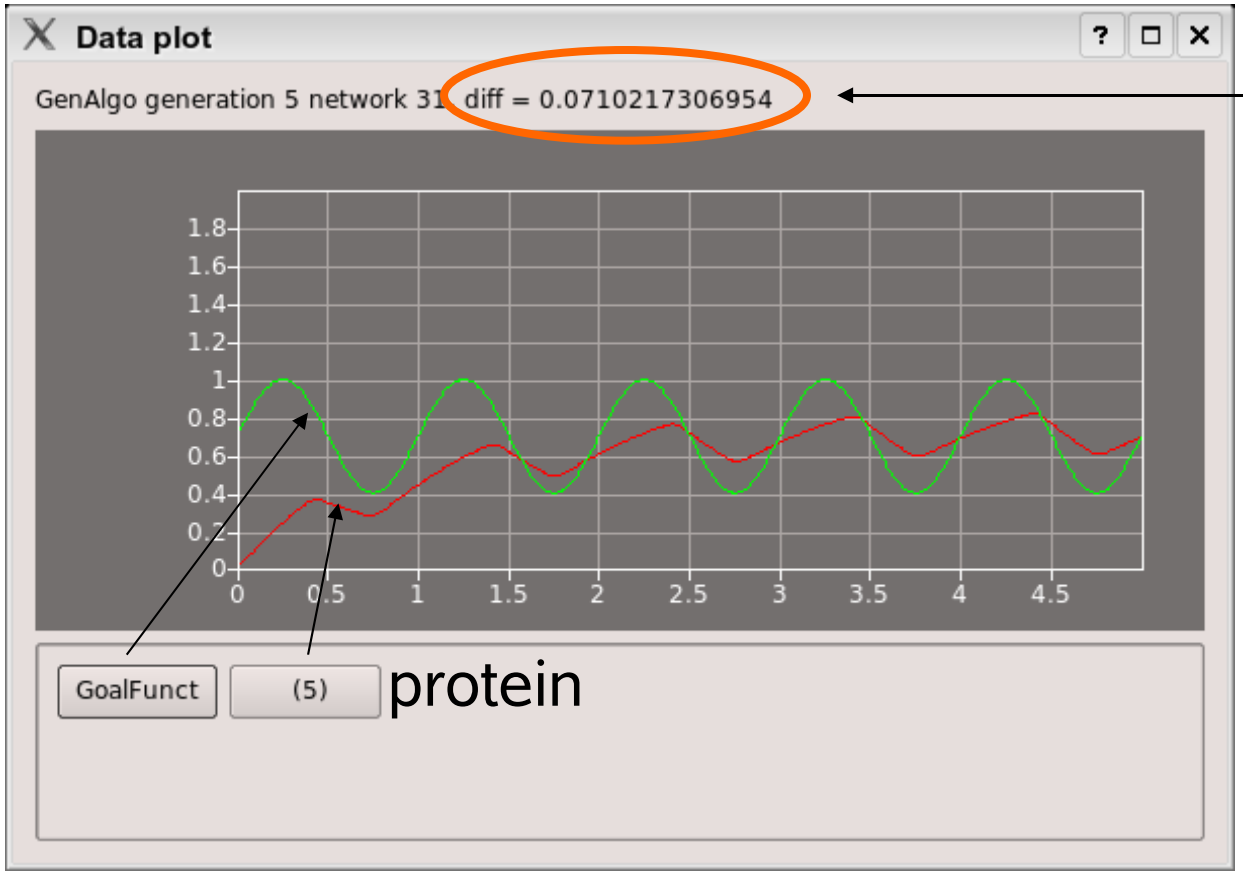
- Two networks recombined
- A gene and its arcs go into the child
- Probability: 90% (default)



Mutations

- Add or remove arcs
- Increase or decrease arc attributes (e.g. arc weight = stochiometry)
- Duplicate or remove genes (inluding arcs)
- Increase or decrease transition rate
- Probability: 1% (default)

Fitness

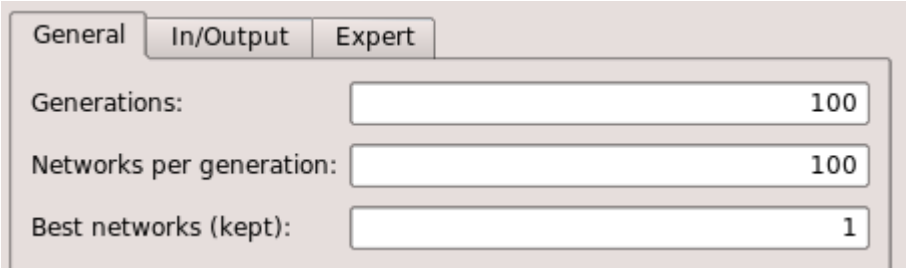


Fitness: Likeness between target function and current results

Parameters

- as adjustable as possible

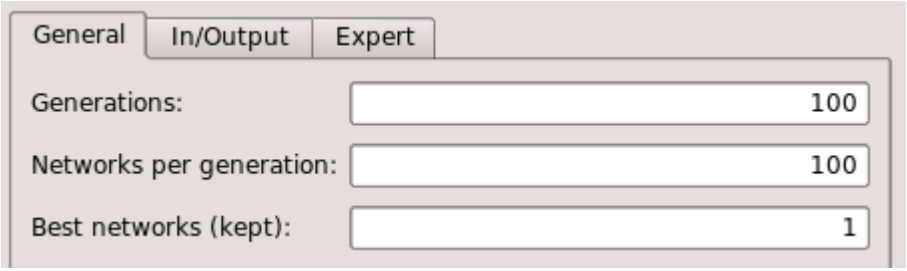
General parameters:



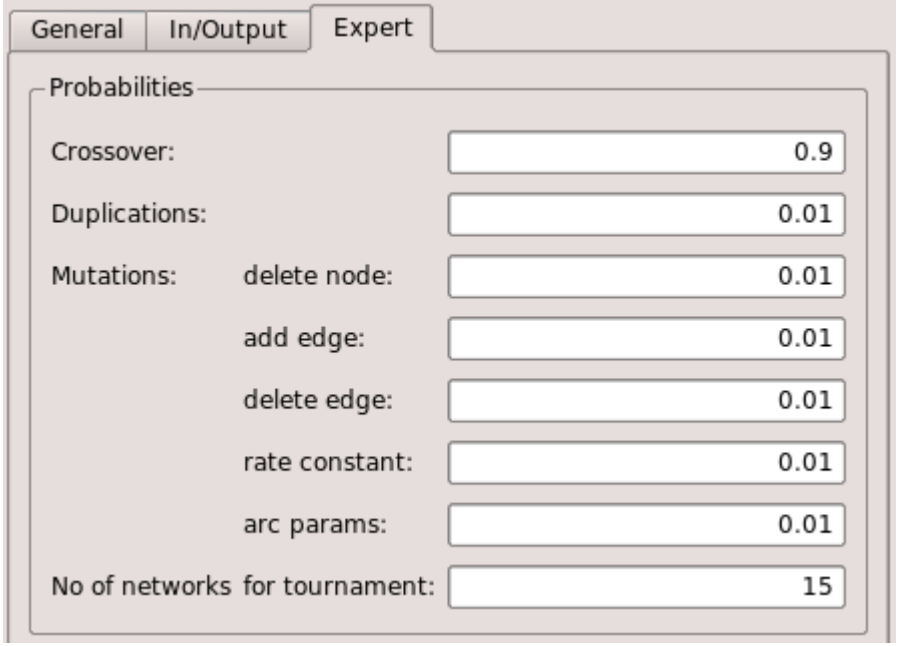
Parameters

- as adjustable as possible

General parameters:



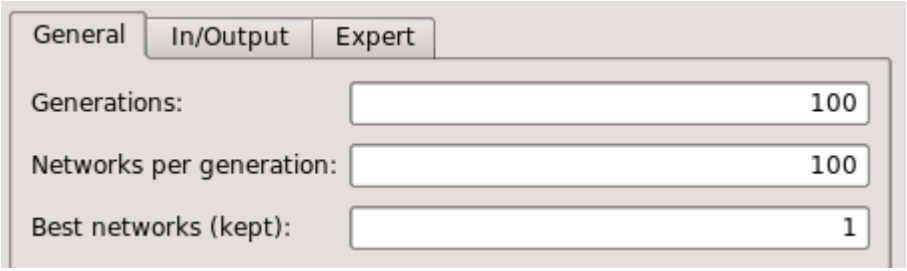
Probabilities:



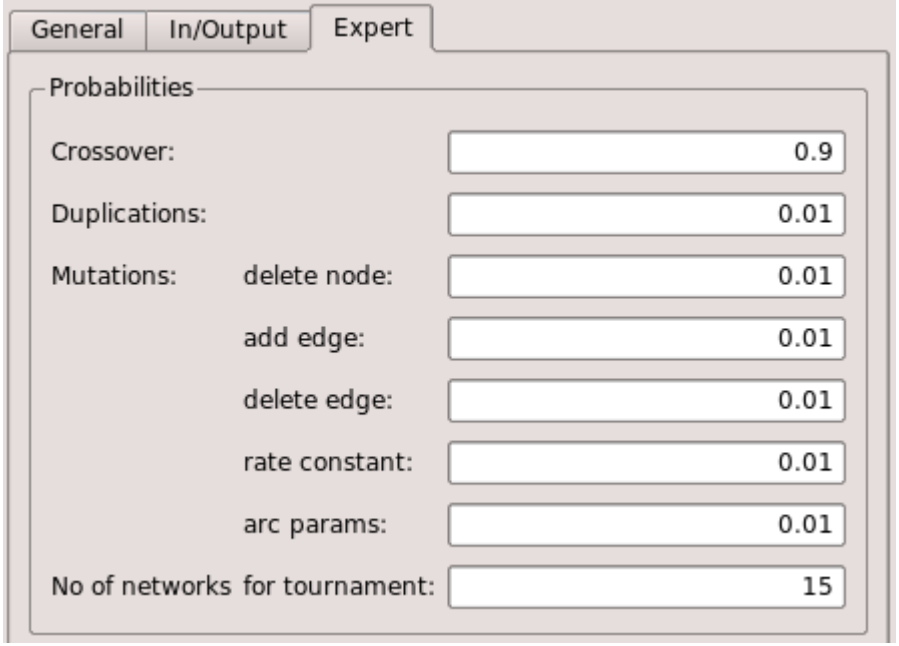
Parameters

- as adjustable as possible

General parameters:



Probabilities:



- Setting parameters to 0 turns operators off

Summary – NetBuilder'

- Easy-to-use GUI
- Free and open-source
- Create network, simulate (and evolve) it
- Evolution adjustable to user's needs

- Evolutionary algorithm in test phase now and will be available in NetBuilder' soon !!
- https://lists.sourceforge.net/lists/listinfo/apostrophe-users

ICSB 2007 - Posters

- For further discussions meet us at our posters (Tuesday and Wednesday):
  - **Johannes**: *In Silico Evolution Of Biological Clocks With Genetic Regulatory Networks*, **F5**
  - **Katja**: *Netbuilder' - A Tool For The Modeling And Simulation Of Genetic Regulatory Networks*, **G8**

Acknowledgement

- Chrystopher L. Nehaniv
- Ralph Gauges
- Mark Robinson

- Slides:
  http://strc.herts.ac.uk/bio/maria/Apostrophe/

Resources – Evolutionary Algorithms

- http://www.talkorigins.org/faqs/faq-intro-to-biology.html
- Evonet flying circus
  http://evonet.lri.fr/CIRCUS2/node.php
- The on-line tutorial on evolutionary computation
  http://www.lcc.uma.es/~ccottap/semEC/
- Bäck, T, Fogel, D B and Michalewicz, Z, ed.: *Evolutionary Computation 1 & 2*. Taylor & Francis 2000
- Goldberg, D. E. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley 1989
- Langdon, W.B. and Poli, R. *Foundations of Genetic Programming*. Springer 2002

References –
Evolving biological clocks with aGRNs

- Knabe, J. F., Nehaniv, C. L. and Schilstra, M. J. *Genetic Regulatory Network models of Biological Clocks: Evolutionary history matters*. In *Artificial Life,* 2007 (in press). http://panmental.de/GRNclocks/
- Knabe, J. F., Nehaniv, C. L. and Schilstra, M. J. *Evolutionary Robustness of Differentiation in Genetic Regulatory Networks. In Proceedings of the 7th German Workshop on Artificial Life 2006 (GWAL-7), pages 75-84, Akademische Verlagsgesellschaft Aka, B*erlin, 2006. http://panmental.de/GWALdiff/
- Knabe, J. F., Nehaniv, C. L. and Schilstra, M. J. *The Essential Motif that wasn't there: Topological and Lesioning Analysis of Evolved Genetic Regulatory Networks*. In IEEE Symposium on Artificial Life (CI-ALife'07), pages 69-76, Omnipress, 2007. http://panmental.de/GRNmotifs/
- and references therein