Evolutionary and Estimation of Distribution Genetic Programming: The Use of Grammars: Why and How?

Bob McKay

School of Computer Science and Engineering

Seoul National University, Korea

Seoul National University
서울 국립 대학교

- Elite University of South Korea
  - ~30,000 Students
  - ~2,000 Faculty
  - http://snu.ac.kr

SNU Computer Science & Eng.

- 30 Faculty
  - About half in Computer Science
  - Half in Computer Engineering

  http://cse.snu.ac.kr

SNU Structural Complexity Lab

- One faculty member, one postdoc
- Four PhD Students
- Three ME students
- Two interns

https://sc.snu.ac.kr

SC Lab Research

- Evolutionary Computation
  - Genetic Programming
    - Grammars and GP
    - EDA-GP
    - Evolution, Development and Evaluation (EDE) in GP
    - Extracting Theories from EDE
    - XML Output Standards for GP
    - Entropy Metrics in GP
    - Regularity Metrics in GP
    - Generalisation in GP
  - Dynamic Optimisation
  - Modelling of Combinatorial Chemistry
  - Automated Debugging

SC Lab Research Applications

- Web Retrieval using Hypernetworks
- Ecological Modelling
  - River Modelling
  - Daisyworld Models
- Personalised Adaptive Input Devices
- Adaptive Constraint Programming
- Software Methodologies for Future Hardware
- Document Clustering and Classification

SC Lab Collaborative Research

- Communications Network Optimisation
- Sports Prediction
- Crime Scene Modelling
- Data Mapping for Machine Learning
- Semantic Operators in Evolutionary Computation
- Spanning Tree Optimisation
- Quasi-random initialisation in EC
- Associative Mechanisms in Language Learning

Outline

- Introduction
  - Background: Genetic Programming
    - Typical Applications
  - Background: Grammars
- Grammar Guided Genetic Programming
- Grammatical Evolution
- Tree Adjunct Grammars and Genetic Programming
- Estimation of Distribution Algorithms, Grammars and Genetic Programming
- Developmental Evaluation and Genetic Programming

Outline

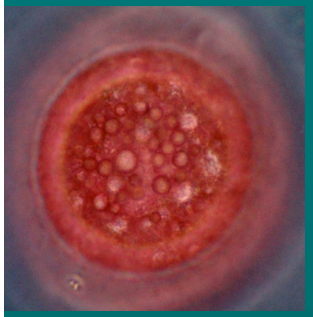
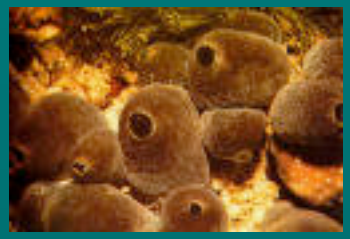
- What is GP?
  - Grammar-based GP
- What is EDA-GP?
  - PIPE and other PPT-based EDA-GP
    - What's wrong with PPT?
  - Grammar-based EDA-GP
    - Fixed Grammars
    - Annotation-learning Grammar EDA-GP
    - Grammar-learning EDA-GP
      - GMPE
- Current and Future Directions

Our Lab's Contributors

- Peter Whigham (UNSW)
- Nguyen Xuan Hoai (UNSW)
- Yin Shan (UNSW)
- Kangil Kim (SNU)

Evolution in Biology

Evolution







Darwinian Evolution (1850s)

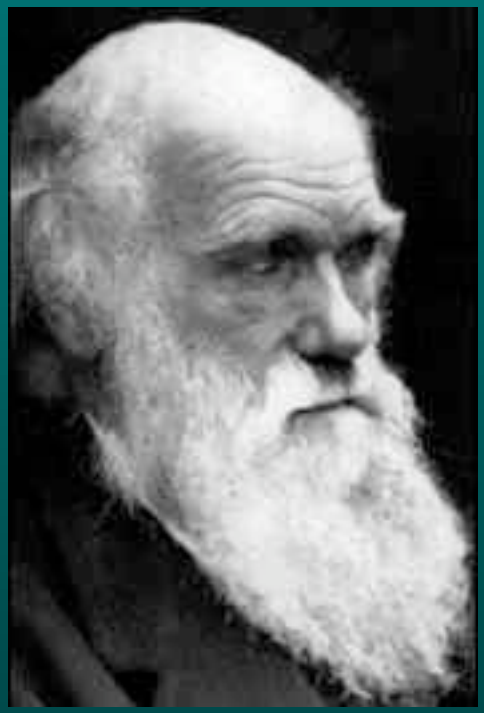
Multiple populations competing for limited resources

Dynamically changing populations with births and deaths

Inheritance: children are like their parents

Variation: children are not exactly the same as their parents

Fitness: different individuals have different probabilities to survive and reproduce

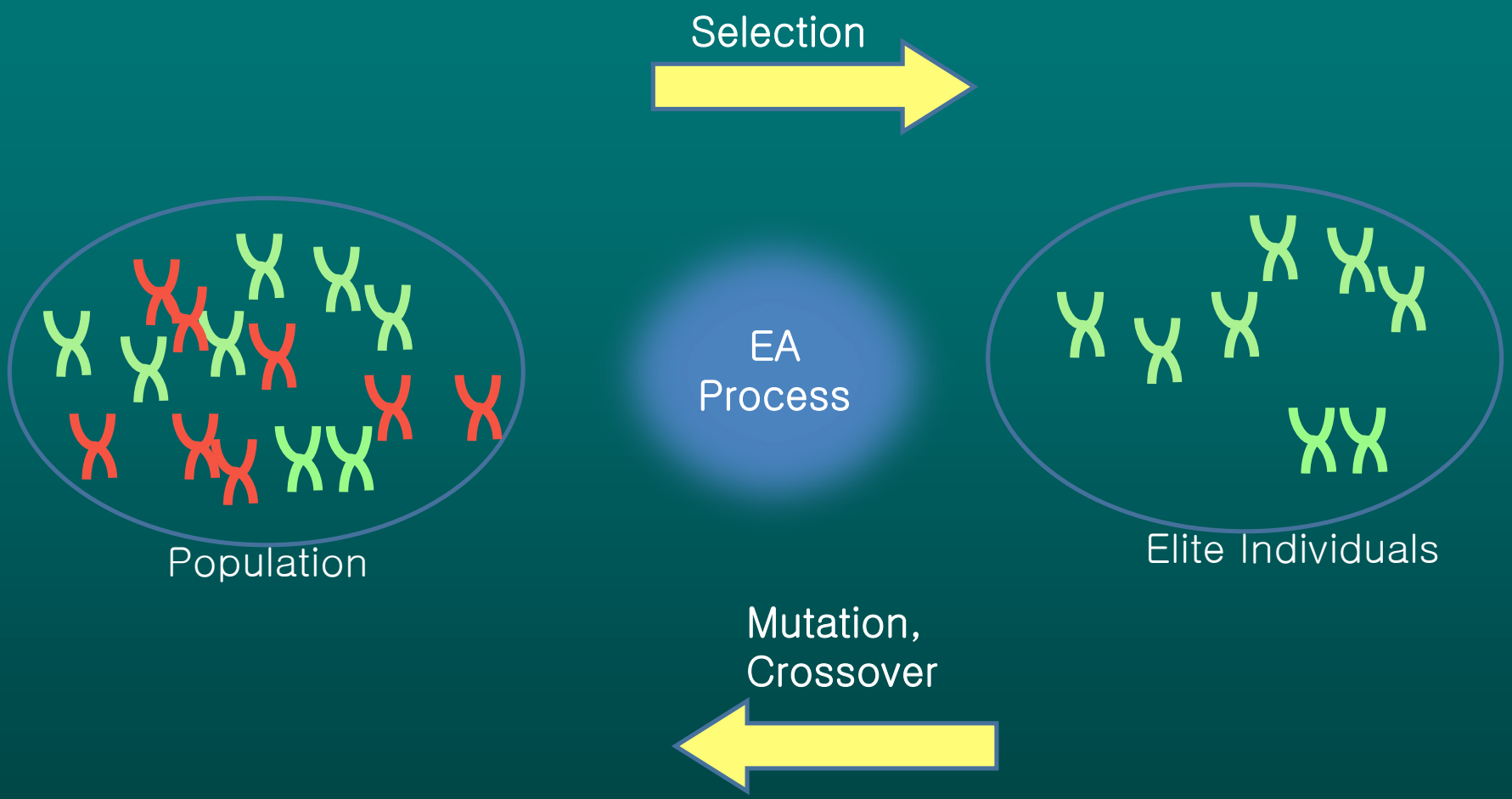


Evolution as an Algorithm

A Generic Evolutionary Algorithm

- Initialise a population of individuals
- Repeat
  - Evaluate the fitness of the individuals for the given problem
  - Repeat
    - Probabilistically select parents according to their fitness
    - Apply sources of variation to parents to generate children
  - Until there are sufficient children for the next population
- Until the problem is solved, evolution stagnates, or resources are exhausted

- Evolutionary Algorithm



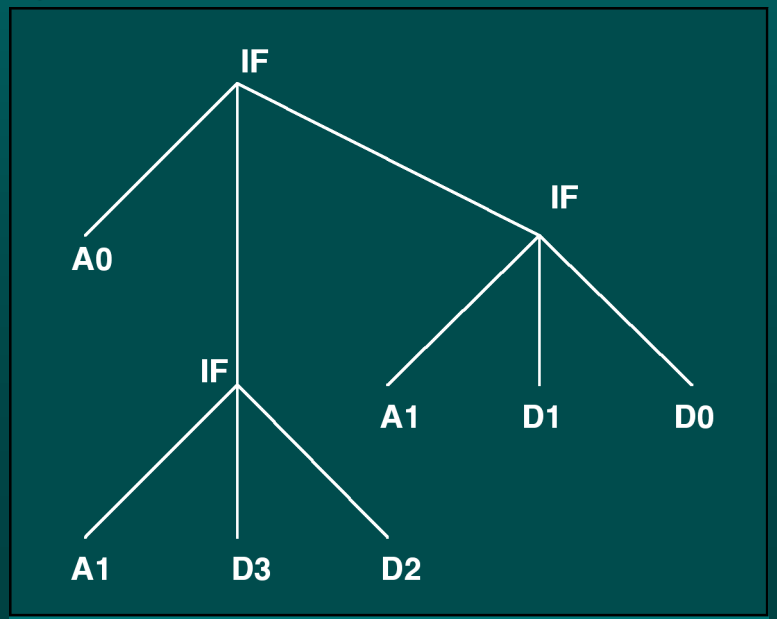
A Generic Evolutionary Algorithm

- Initialise a population of individuals
- Repeat
  - Evaluate the fitness of the individuals for the given problem
  - Repeat
    - Probabilistically select parents according to their fitness
    - Apply sources of variation to parents to generate children
  - Until there are sufficient children for the next population
- Until the problem is solved, evolution stagnates, or resources are exhausted

One Kind: Genetic Programming

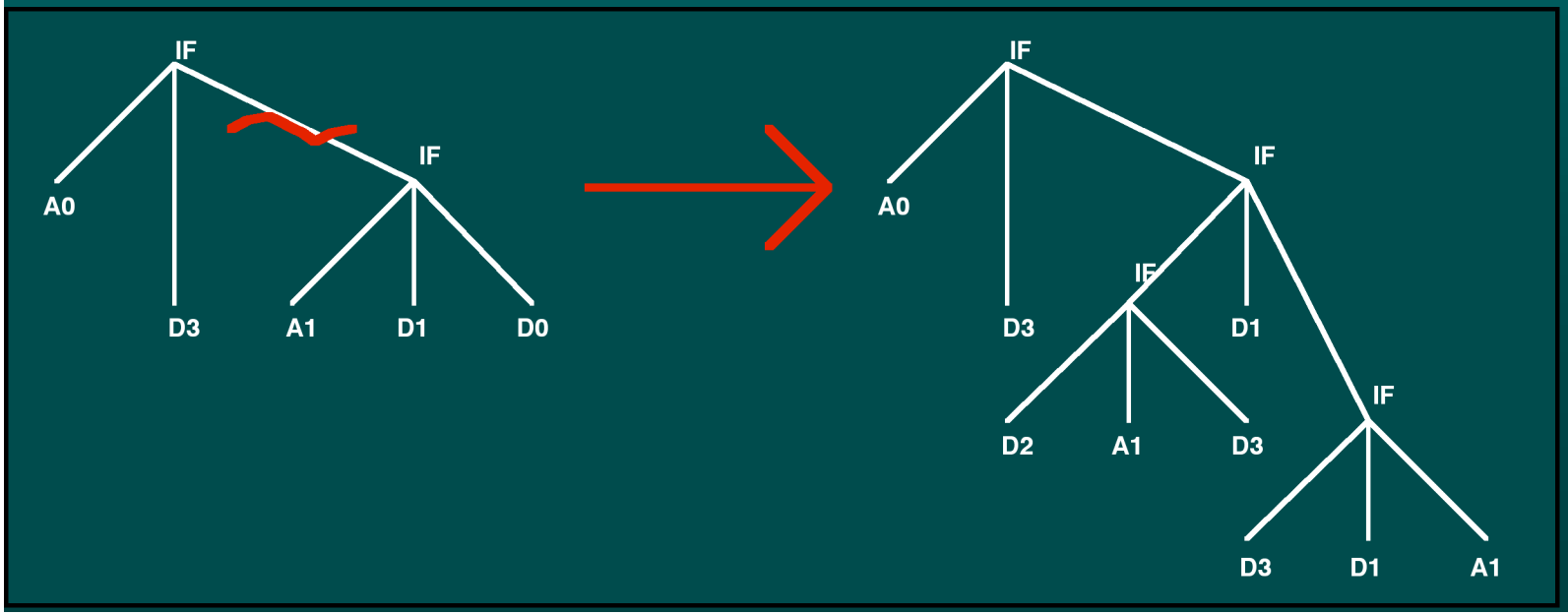
What is Genetic Programming?

- Original Idea:
  - Evolve populations of expression trees representing problem solutions
  - Cramer (1985); Schmidhuber (1987); Koza (1992)
    - Closure assumption: any function can apply to any argument



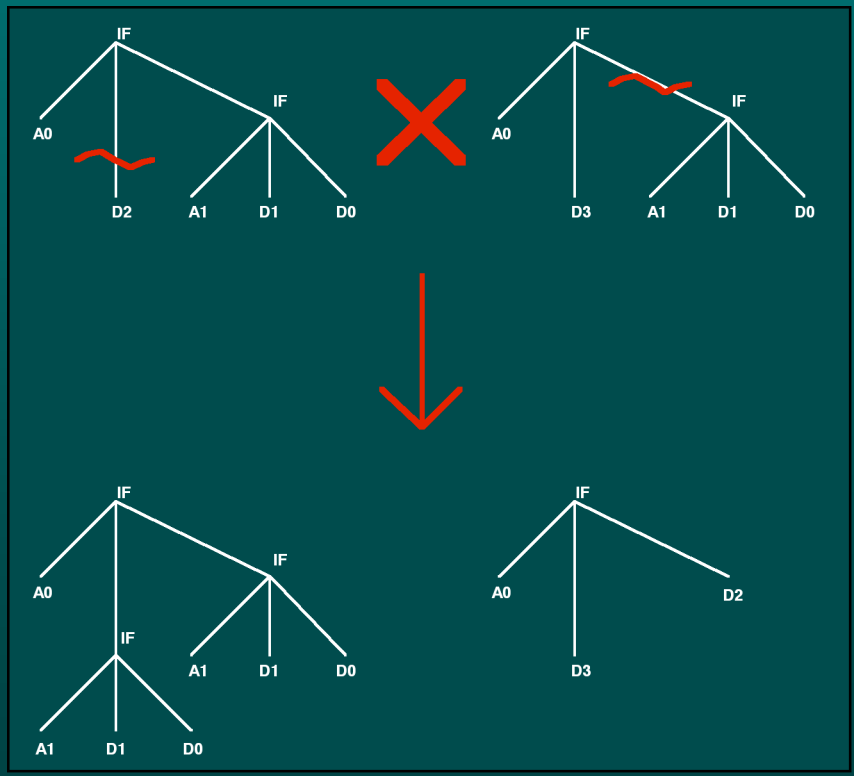
Stochastic Variation Operator: Mutation

- Randomly choose a node in the parent tree
- Delete the sub-tree below that node
- Generate a new random sub-tree



Stochastic Variation Operator: Crossover

- Randomly choose a node in each parent tree
- Exchange the sub-trees rooted at those points



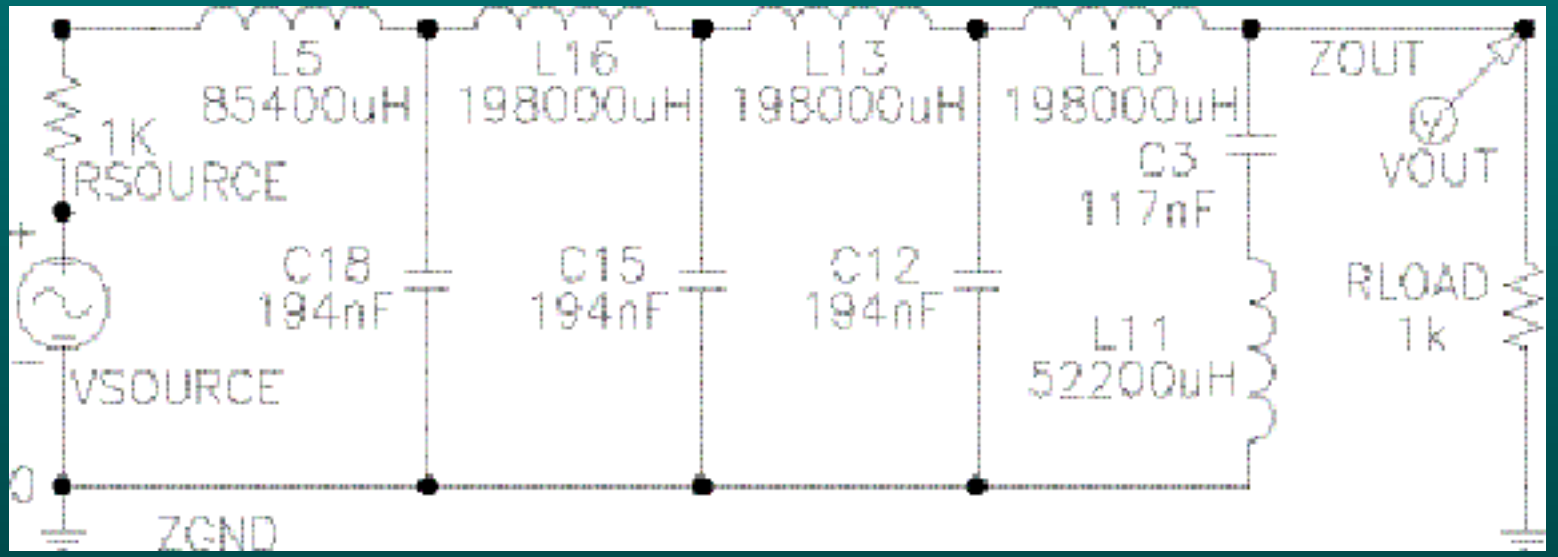
Why should we care?

Why should we care?

- The only serious candidate for automated solution of a wide range of important real-World problems
  - Noisy relational problems
  - Unbounded design problems

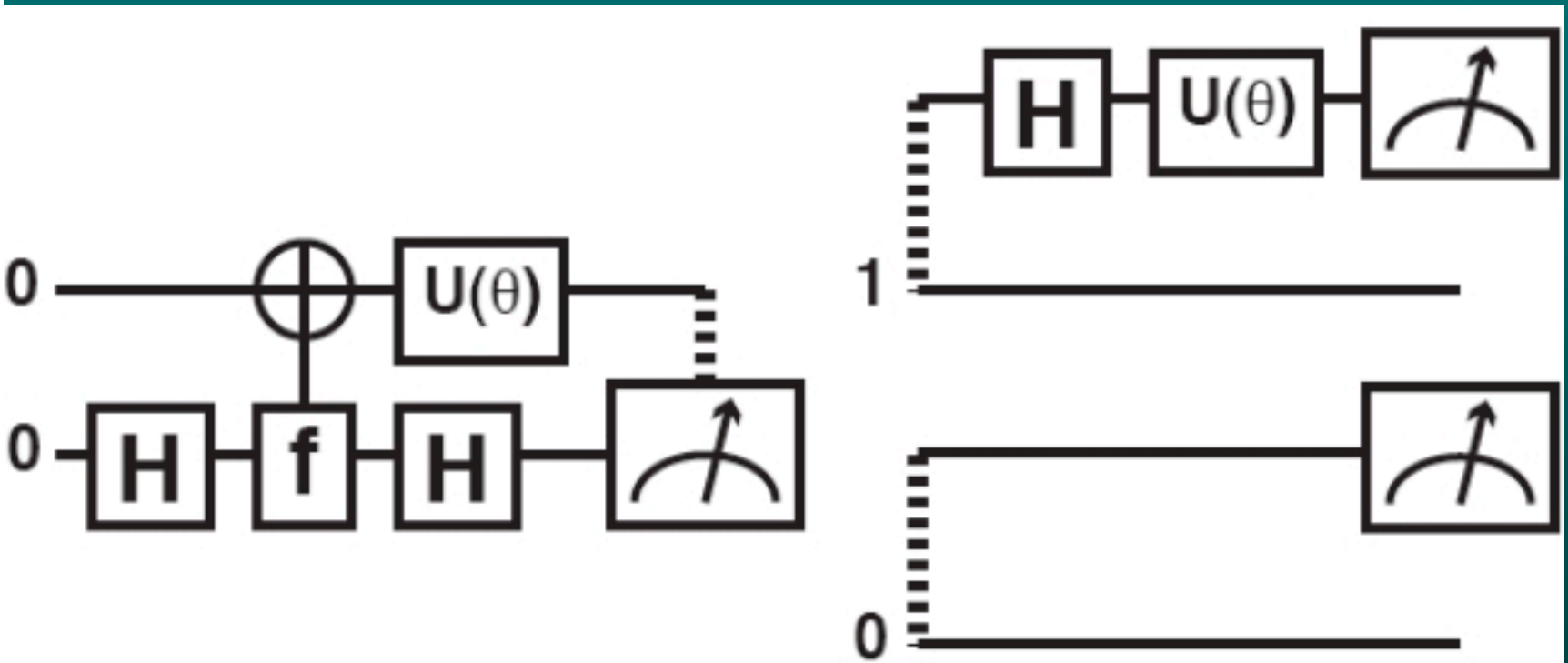
Electronic Design

- Koza et al: Zobel filter



Quantum Algorithms

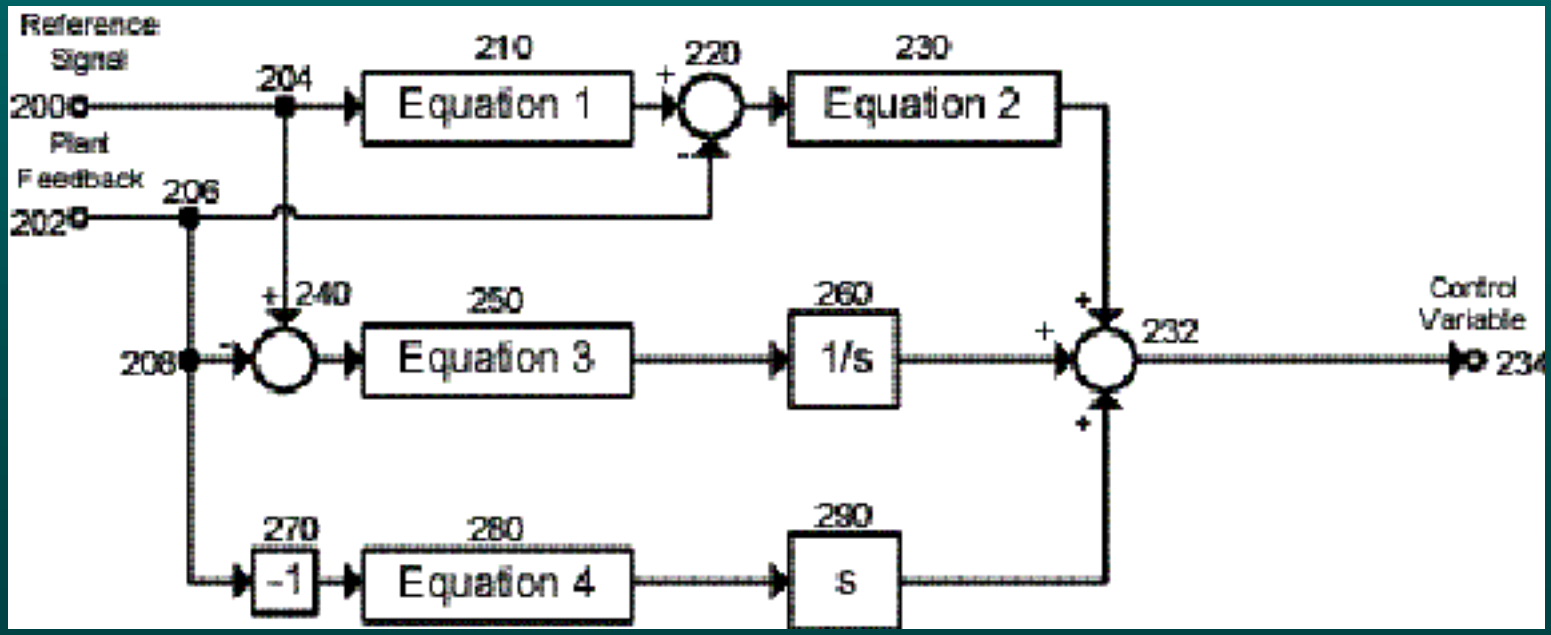
- Barnum, Bernstein and Spector Depth One OR Query



θ=5.96143477

Control System Parameters

- Koza et al:
  - Parameter Equations for Proportional Integral Derivative (PID) Controller

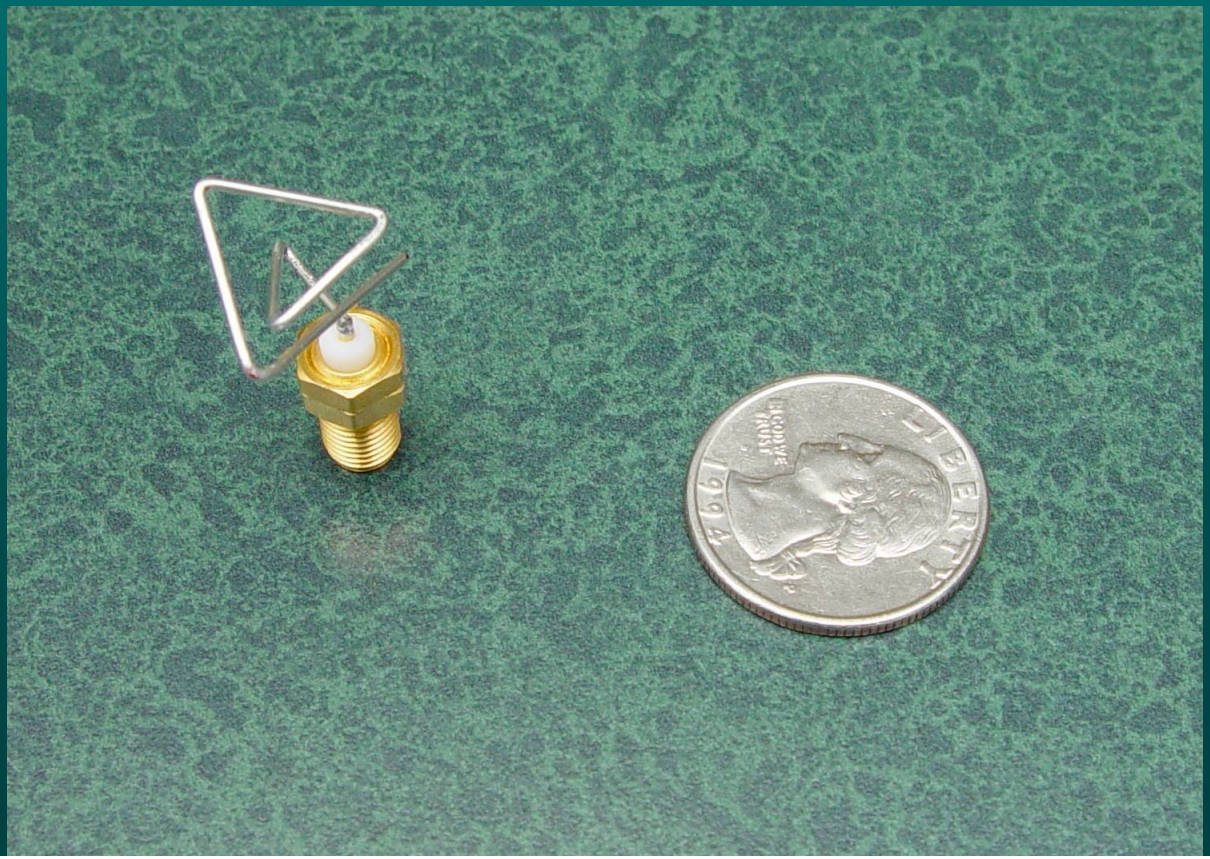


Bioinformatics

- Wide variety of applications
- Well known: Motif detection for gene families
  - D-E-A-D
  - manganese superoxide dismutase
  - Koza et al 1999

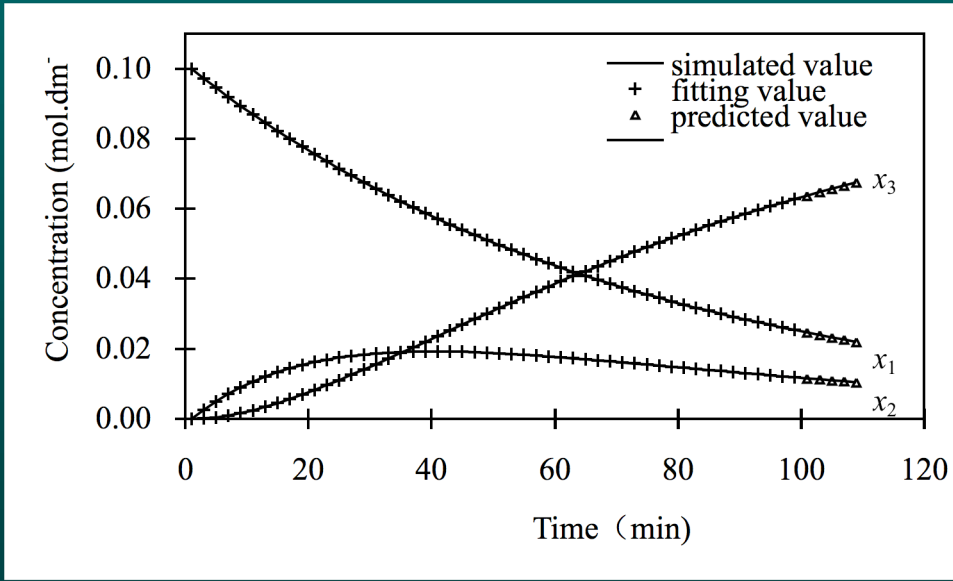
Antenna Design

- Lohn et al (2003)
  - Design of wire antenna for NASA spacecraft



Chemical Dynamics Modelling

- Evolving systems of differential equations
  - Predicting discharge behaviour of a battery
  - Cao et al 2000

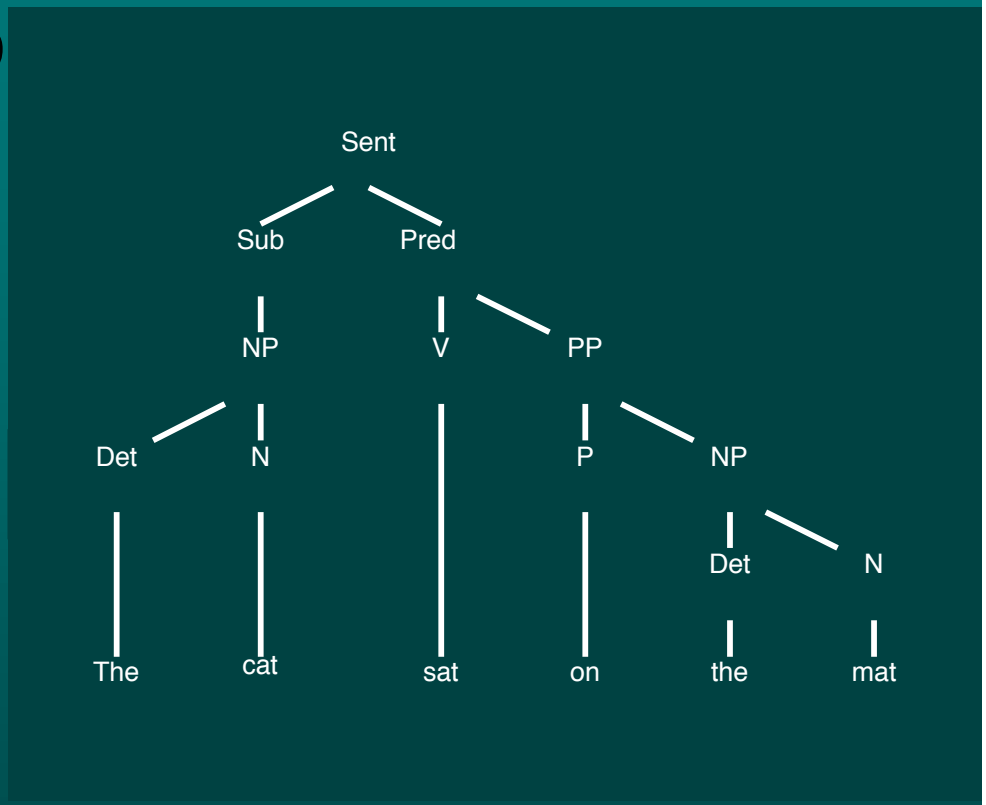


Grammar-Guided GP

Context Free Grammars

- Sentences are made up from simpler sub-structures:

Sent	→	Sub Pred
Sub	→	NP
Pred	→	V PP
PP	→	P NP-
NP	→	Det N
V	→	sat l stood l …
Det	→	the l a l ….
N	→	cat l mat l …



Derivation Tree

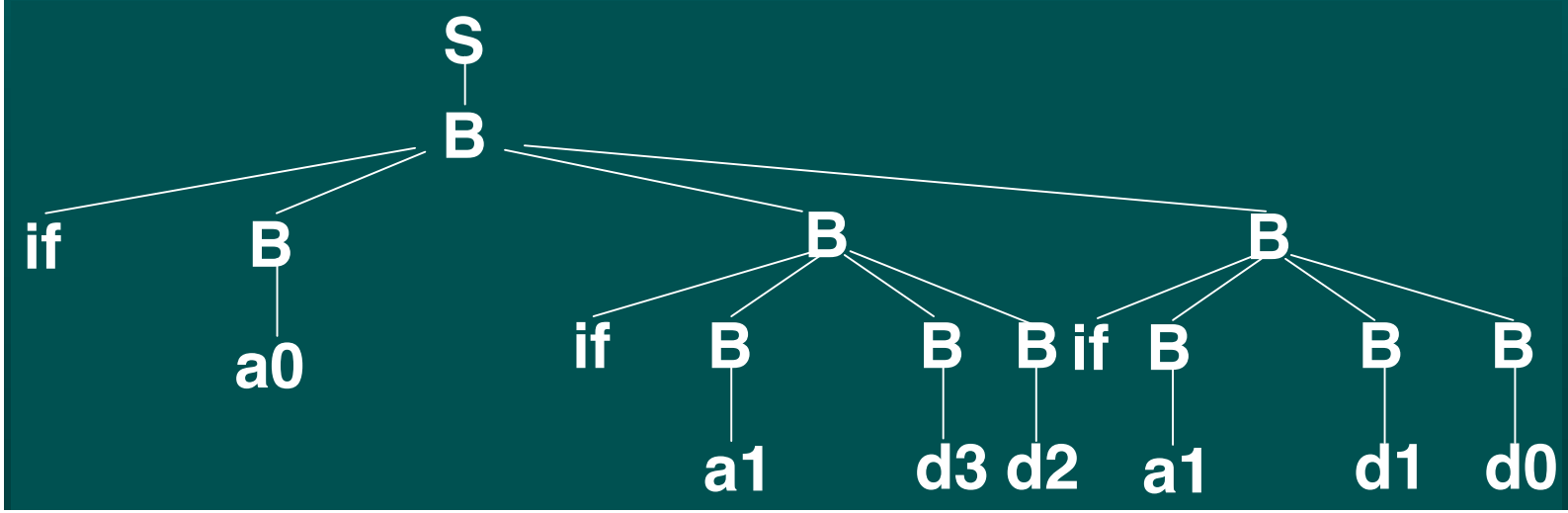
The Idea….

- Why not combine the ideas of Genetic Programming and Context Free Grammars?
  - Wong and Leung 1995
  - Whigham 1995
  - Schultz 1995
  - Gruau 1996
- Evolve populations of
  - Grammar derivation trees

    Instead of
  - Expression trees

Grammar-Based Representations

- The chromosome is a derivation tree in a predefined grammar
  - S → B
  - B → B or B
  - B → B and B
  - B → not B
  - B → if B B B
  - B → a0 | a1
  - B → d0 | d1 | d2 | d3



Grammar Guided Genetic Programming (GGGP)

- Problem space represented by a Context Free Grammar G
  - Individuals are derivation trees in G
  - Crossover uses sub-tree crossover as in GP
    - But the root nodes must have the same label
  - Mutation uses sub-tree mutation as in GP
    - But the generated sub-tree must be consistent with the grammar
- Highly successful
  - Now one of the dominant forms of GP

Major Variants

- Grammatical Evolution: Linearised Chromosome
  - Very widely used
- Tree Adjoining Grammar GP
  - Many technical advantages over CFG for GP
    - Support for long-distance dependence
    - Less constrained search space
- Grammars with Semantic Component
  - Christensen Grammar Evolution

Why should we care?

What do Grammars offer to Genetic Programming?

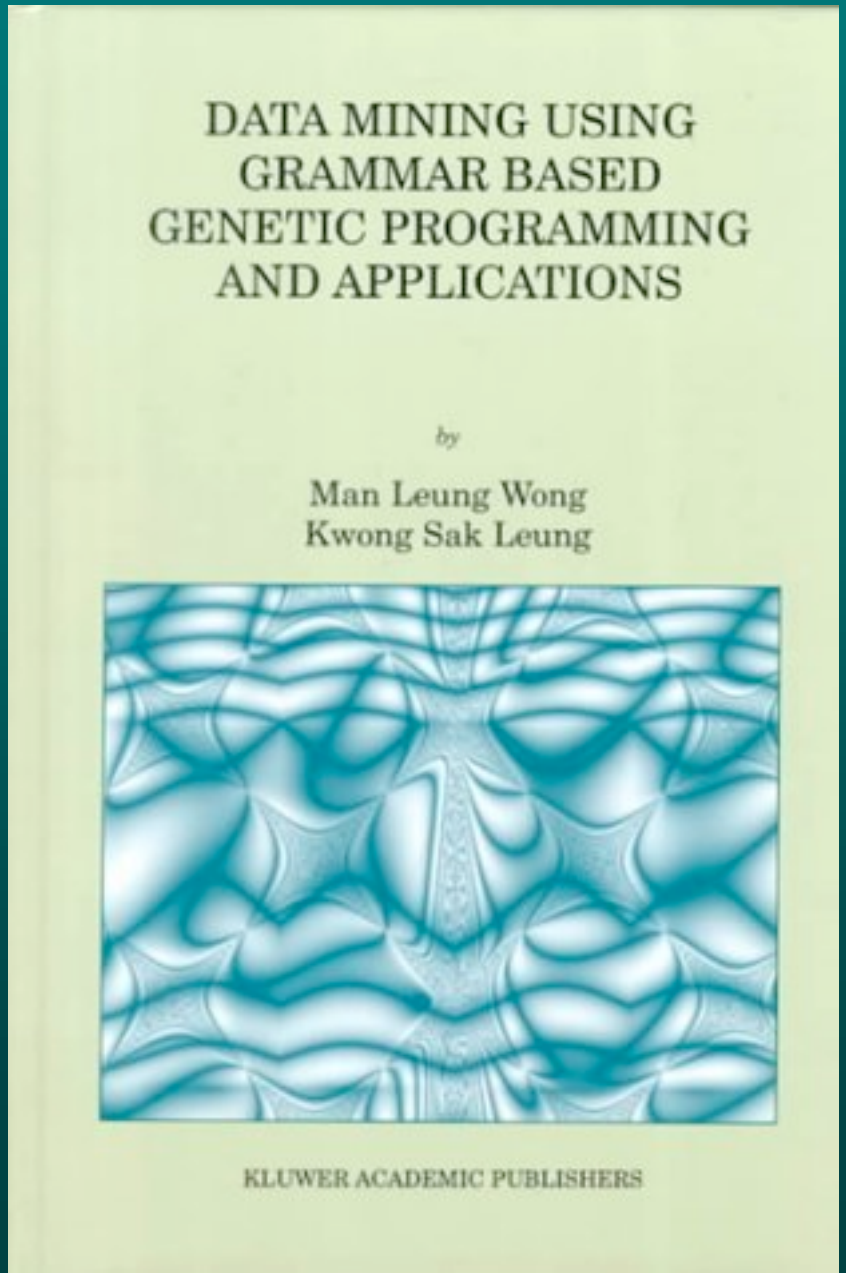
- Declarative Search Space Restriction
  - Grammars offer a systematic, declarative way to incorporate user knowledge of restrictions on the search space
- Homologous Operators
  - In GGGP crossover and mutation, a subtree is replaced by another with the same syntactic function
    - And hopefully, similar semantic function

What do Grammars offer to Genetic Programming (2)?

- Solution Models and Incremental Learning
  - The incremental refinement process may be automated
    - Grammars describing progressively smaller sub-spaces of the solution space
- Incremental Model Exploration
  - Supports an interactive style of model development, in which the user gradually refines the grammar describing the search space, until it is sufficiently small for effective GP solutions to emerge

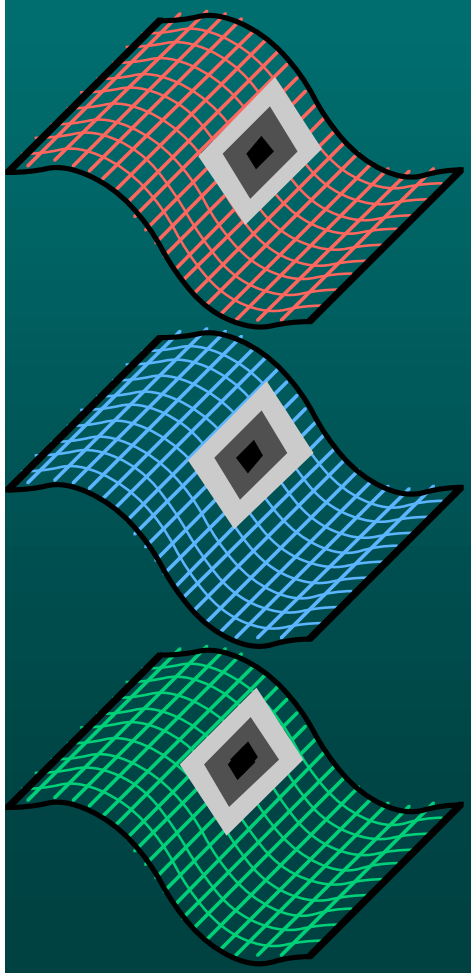
Applications

Medical Data Mining



Species Distribution Modelling

Inexpensive Layers

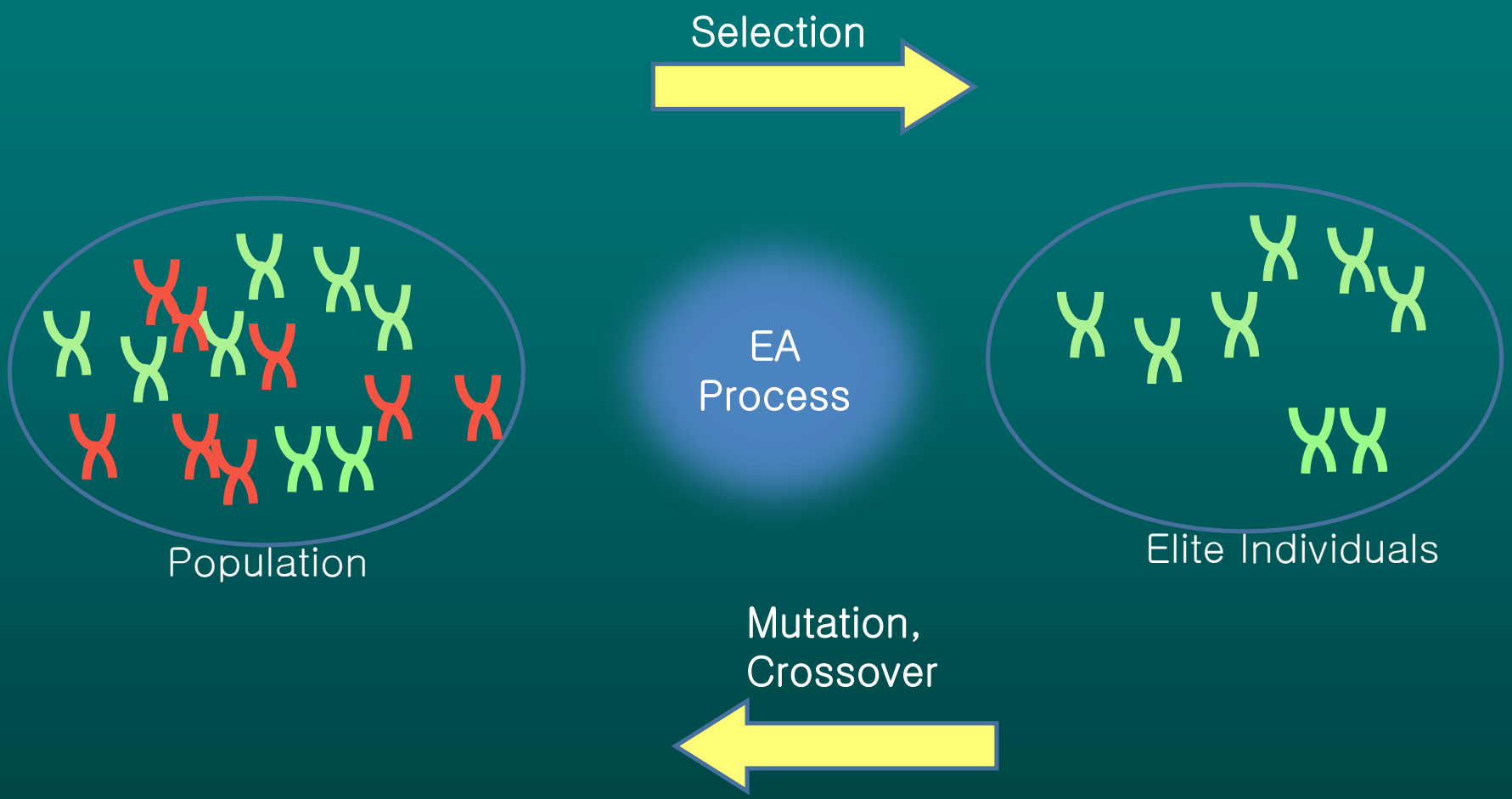


Expensive Layer

Modelling Algal Growth (Nakdong R)

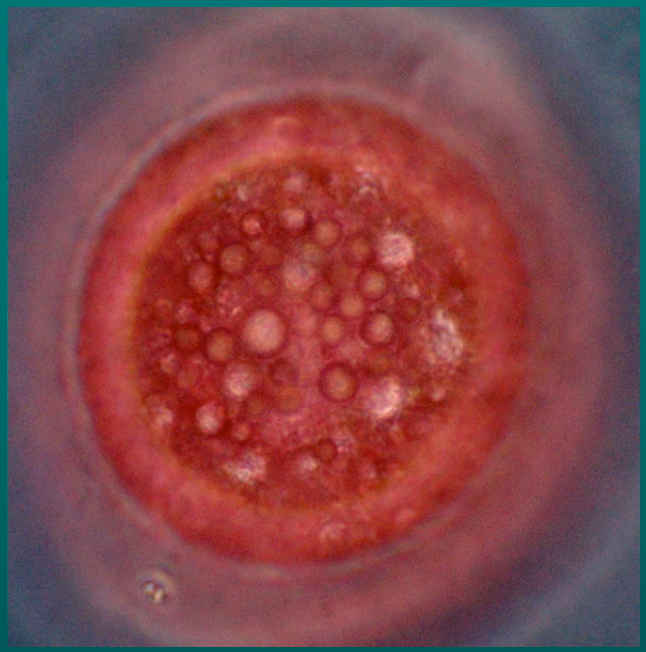
Generalising Evolutionary Operators

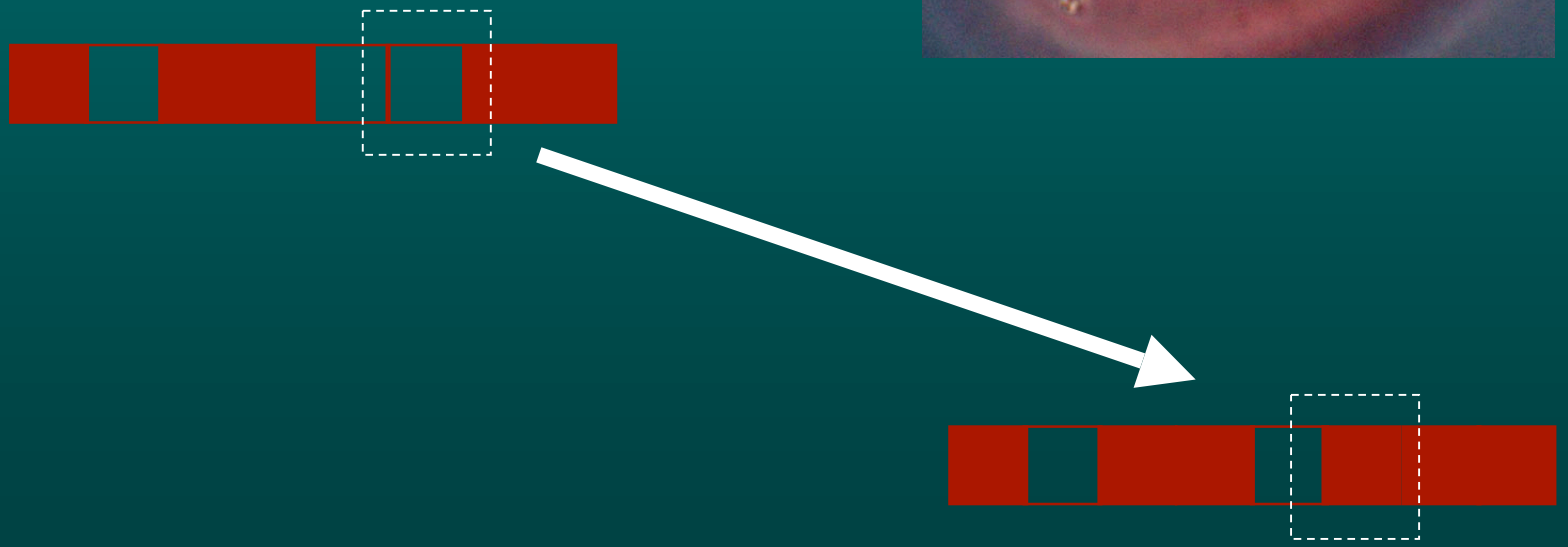
- Reminder: Evolutionary Algorithm



Mutation

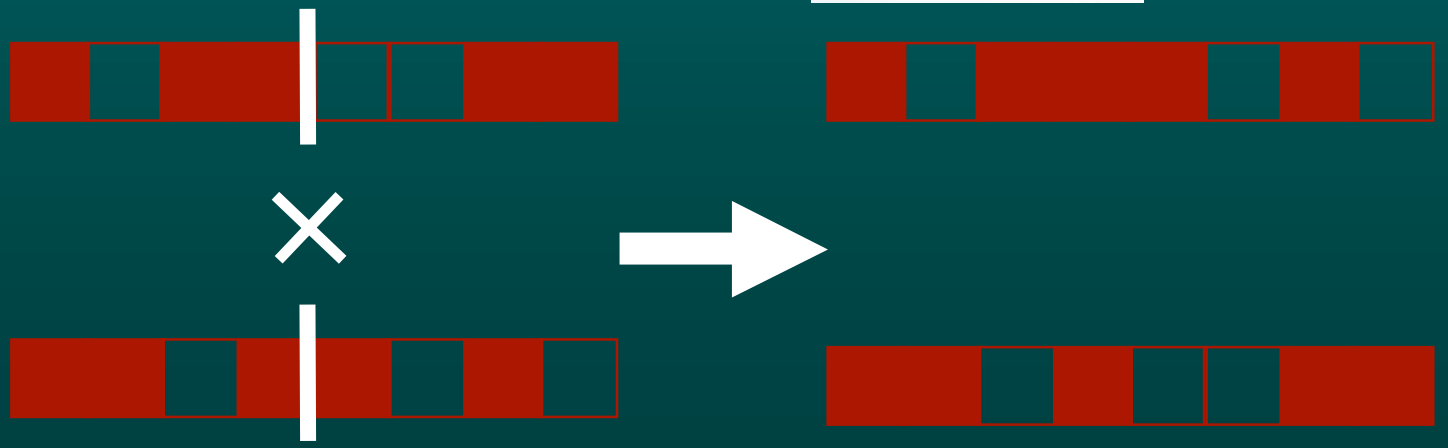
- As in Bacteria





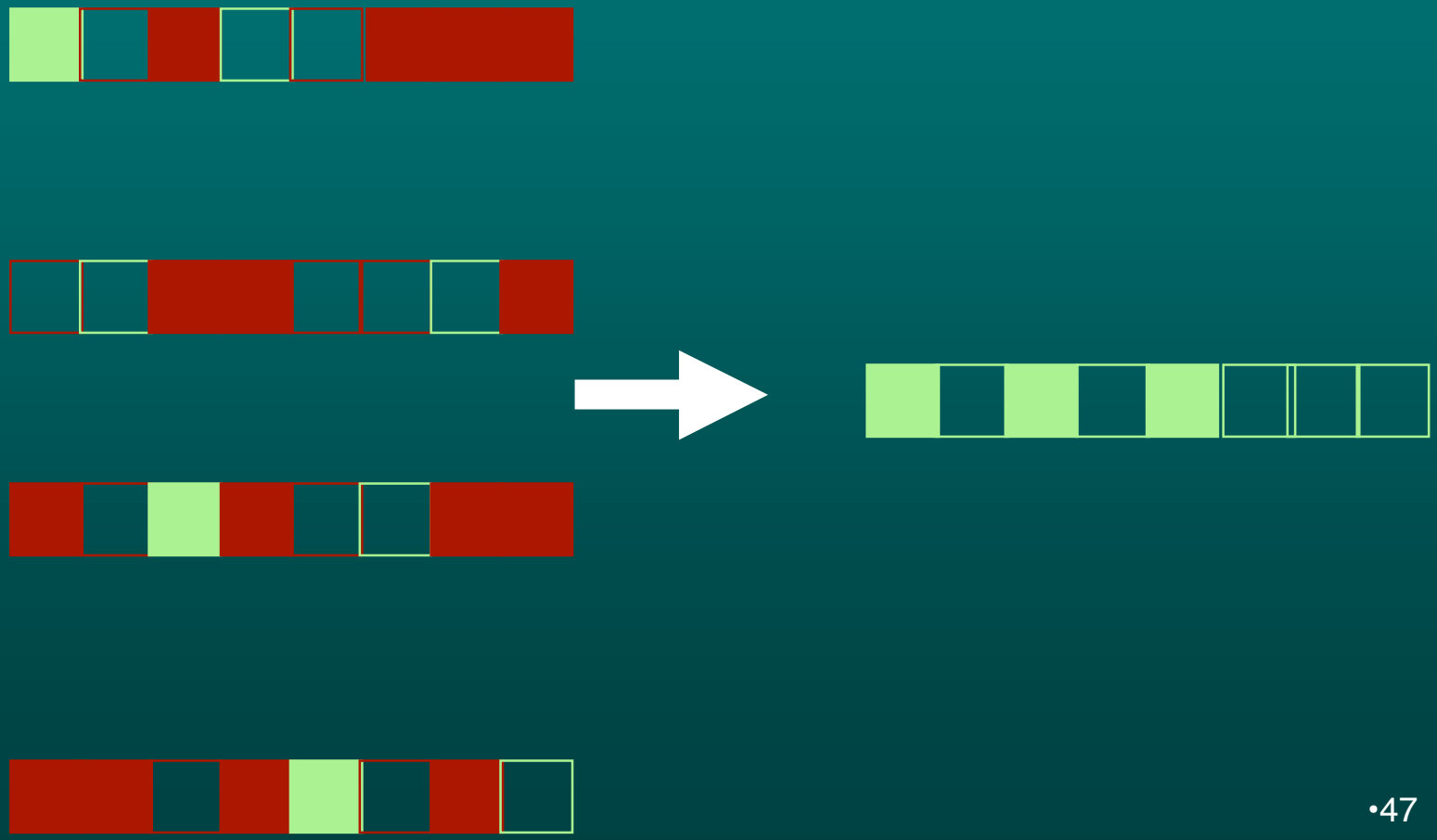
Crossover (Sexual Reproduction)

- Most Higher Animals and Plants



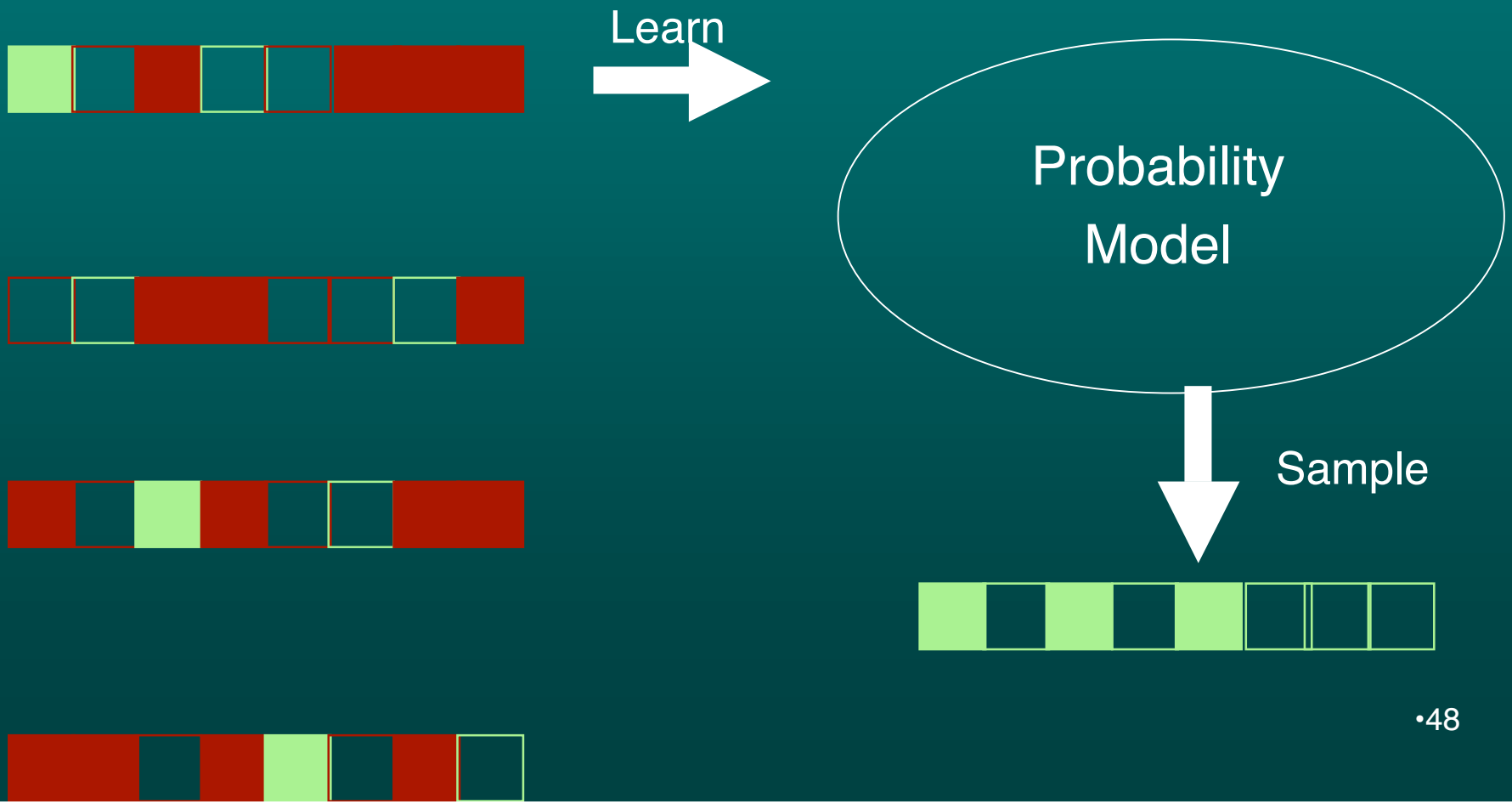
Multi-Parent Crossover
An Engineering Variant

- Not (as far as I know) in nature



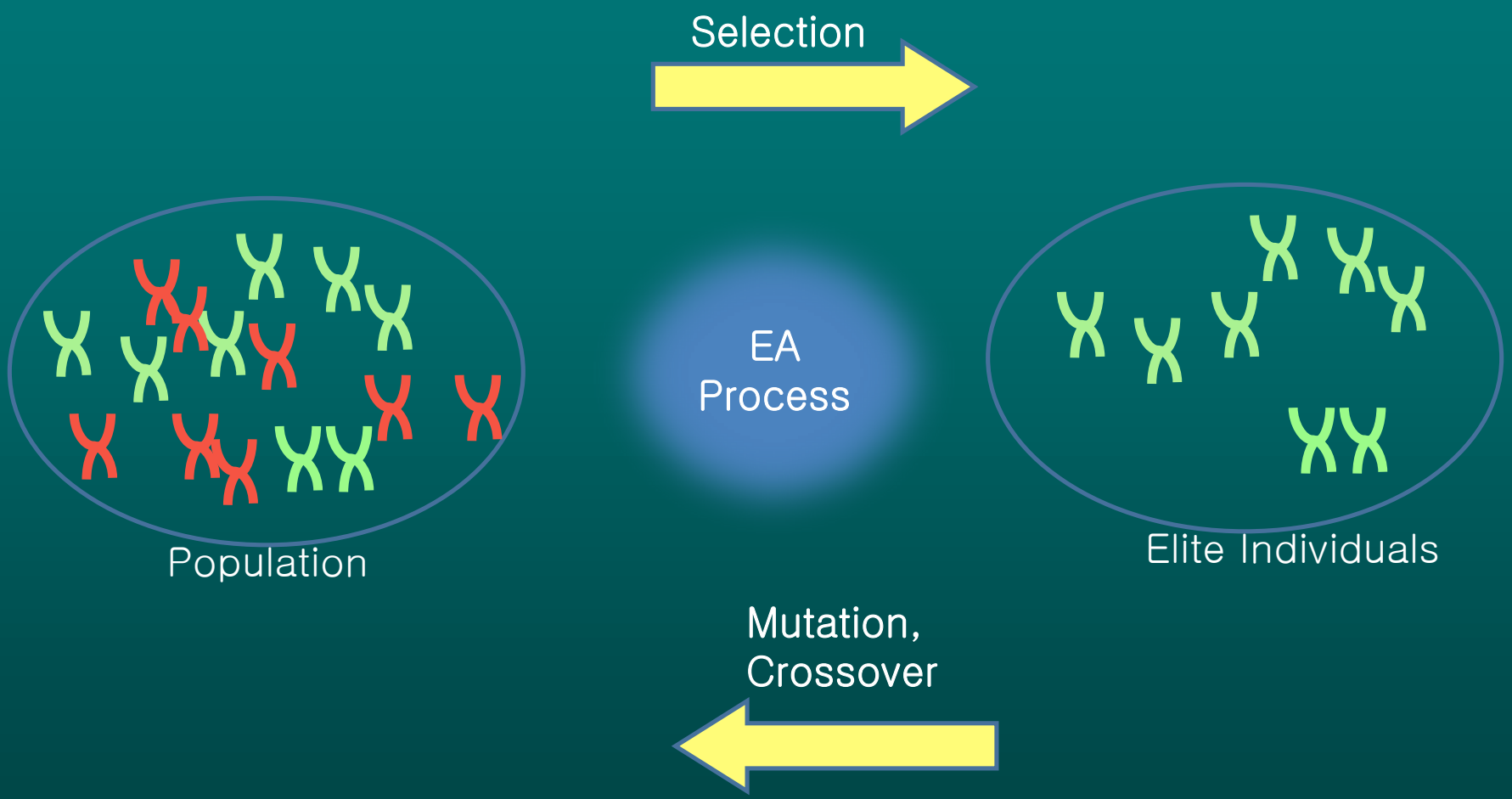
Generalising Multi-Parent Crossover

- Why not:
  - Use the whole population
  - Use a probability distribution rather than frequency table

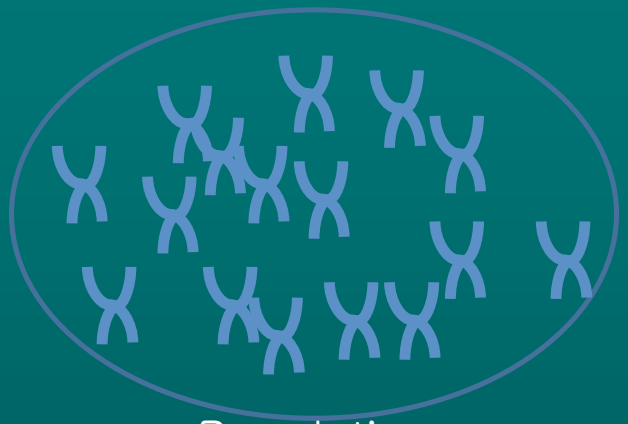


Estimation of Distribution Algorithms

- Evolutionary Algorithm



• Estimation of Distribution Algorithm



Population

Selection

A	T	C	G
T	T	G	G
G	T	G	C
C	T	C	A

Chromosomes

EDA Process

Sampling

Model Update

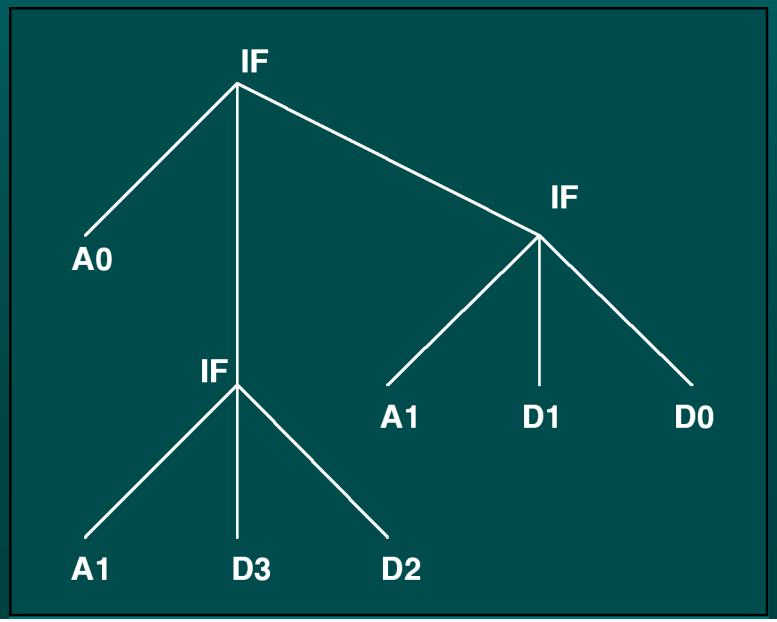
Probability model (ex.PBIL)

Gene1		Gene2		Gene3		Gene4	
A	0.25	A	0	A	0	A	0.25
T	0.25	T	1	T	0	T	0
G	0.25	G	0	G	0.5	G	0.5
C	0.25	C	0	C	0.5	C	0.25

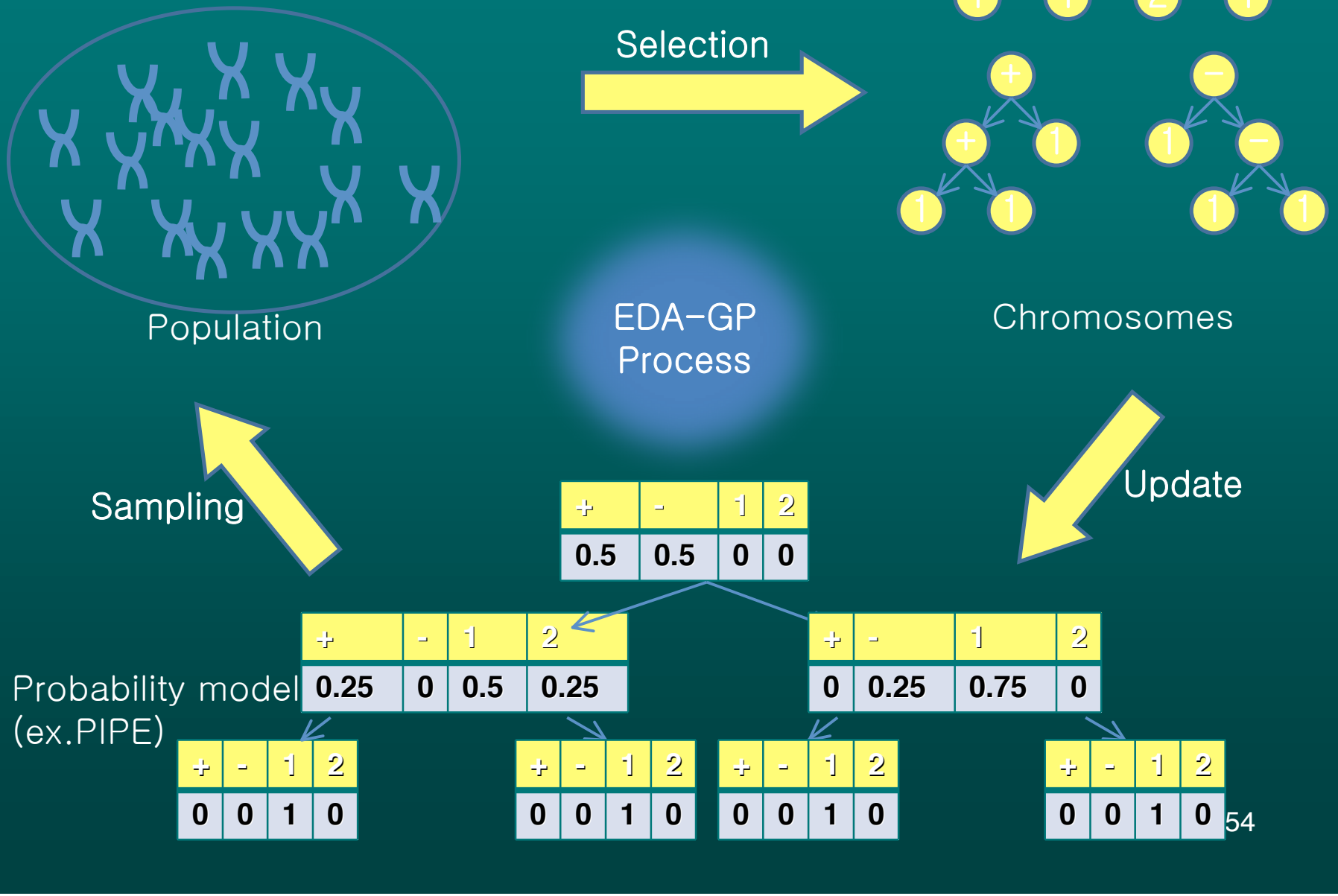
What is EDA-GP?

What is EDA-GP?

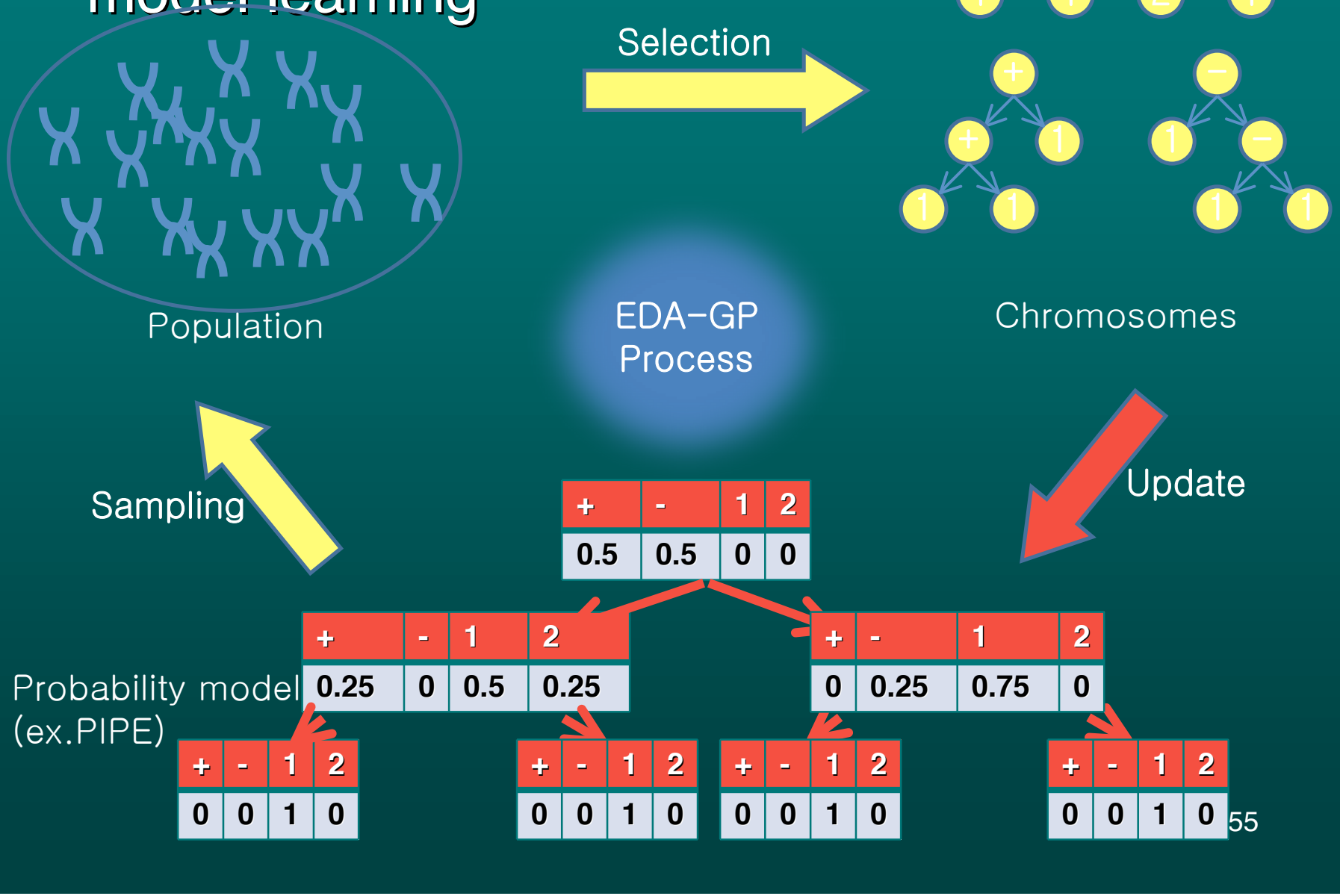
- Applying EDA algorithms to GP problem spaces



• EDA-GP


Selection
Population
EDA-GP
Process
Chromosomes
Sampling
Update
Probability model
(ex.PIPE)
+ - 1 2
0.5 0.5 0 0
+ - 1 2
0.25 0 0.5 0.25
+ - 1 2
0 0.25 0.75 0
+ - 1 2
0 0 1 0
+ - 1 2
0 0 1 0
+ - 1 2
0 0 1 0
+ - 1 2
0 0 1 0

- Will emphasise model and model learning



What is EDA-GP?

- Requires
  - Probabilistic models for distributions over sets of expression trees
  - Methods for sampling such models
  - Methods for updating or learning such models

Why should we care?

Potential Benefits of EDA-GP?

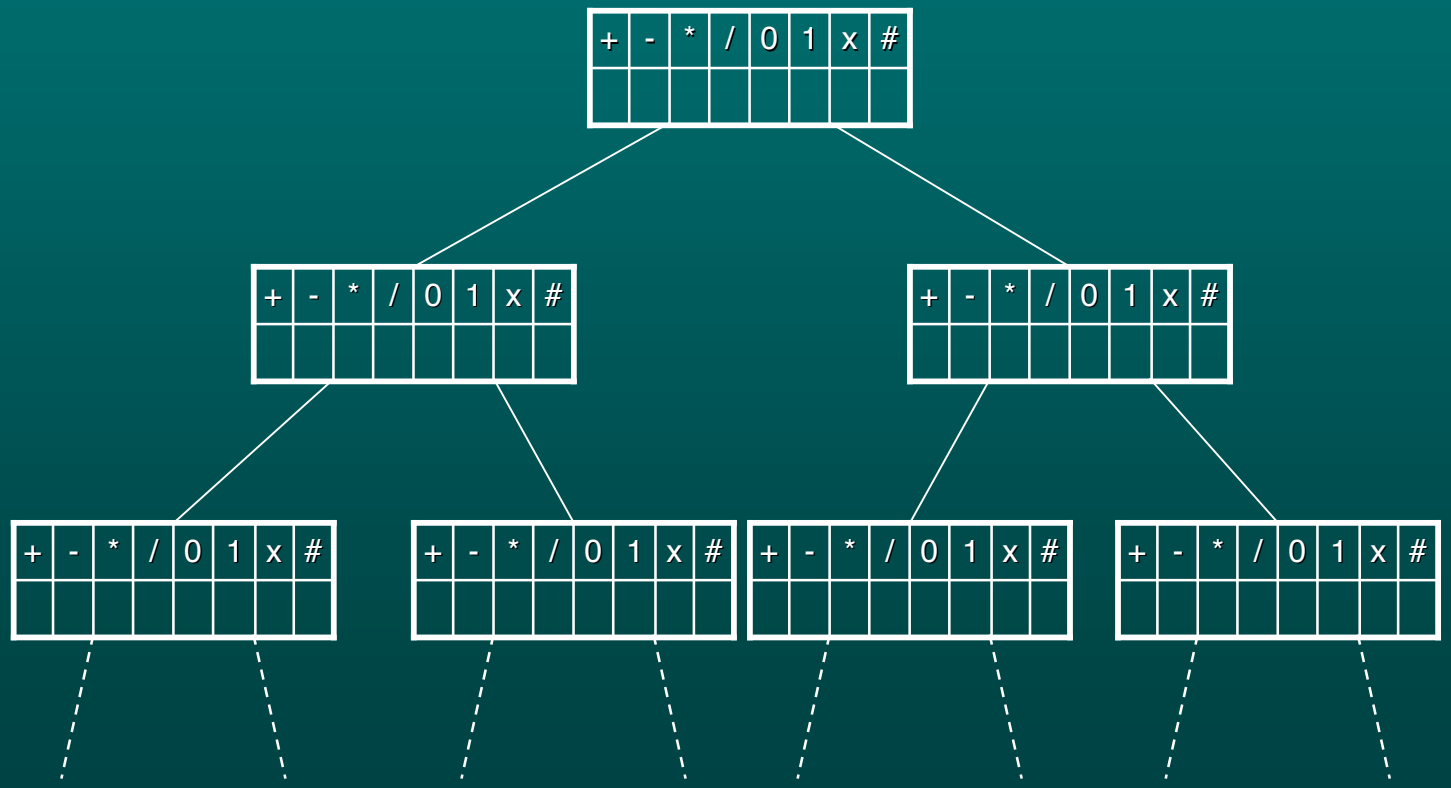
- Model of Space of Good Solutions
  - Probability model is more informative than a single best individual
  - Potentially can distinguish essential from inessential parts of a scientific theory
- Efficiency
  - In benchmark problems, often exhibits much faster learning than GP
  - Unfortunately doesn't seem to scale well to real-world problems
    - At present

PIPE: the first EDA-GP

PIPE: Salustowicz & Schmidhuber 1995

- Extremely early in the history of EDA
- A PBIL-like probability model
  - Basic model is a tree of maximum allowed depth
  - Branching factor is the maximum arity of the function set
  - Nodes in tree each hold a probality table for what symbols is at that node

PIPE's Probabilistic Prototype Tree (PPT) Model



PIPE Sampling Strategy

- Sample from the root
  - If a child doesn't need to be sampled
    - E.g. second child of negation node
  - Don't sample it or its children

PIPE Selection Strategy

- Truncation selection
  - Fixed selection ratio

PIPE Model Update

- Increase probabilities of node values that occur in selected trees
  - Exact update formula varies with the version of PIPE

PIPE Evaluation

- Successfully applied to a few “toy” benchmark problems
- Also to some small real-World problems
- Only represents independent probabilities
  - Unable to learn solutions where dependency between nodes is required
- Some later variants (Sastry, Iba and his group)
  - Build dependency models between nodes
  - Still limited follow-up
  - Why?

Interlude: Some Issues from GP

Importance of Context

- Problem solutions often require complex context
  - The wrong context can give a potentially good component very poor fitness
  - Negation in Boolean problems
  - Minus in arithmetic problems

Repetition of Building Blocks

- Solutions usually contain repeated blocks of code
  - This is why crossover is so important in GP
  - The extent of repetition generally scales with the problem size

Relocatability of Building Blocks

- GP problem spaces generally include ineffective blocks of operators
  - NOT NOT, --
- Thus the same building block in different locations may have exactly the same semantic effect
  - (one of the key issues in GP bloat)

GP Considerations and PPTs

- All these considerations suggest that we generally do *not* want dependence on absolute position in an EDA-GP model
  - Yet this is exactly what is forced on us by a PPT representation
  - Probably the prime motivator for use of stochastic grammars in EDA-GP
  - Now the dominant approach to EDA-GP

Grammar-based EDA-GP

Why Grammar-Based?

- Reasons given above for grammar-based GP
- Also an additional reason:
  - In CS, what would we normally use to represent an infinite set of node-labelled trees (especially, if we think they have some systematic relationship)?
    - Almost universally in CS, we use a grammar
    - We could use a PPT with Boolean annotations
      - BUT WE DON'T - for very good reasons
- So what should we use to represent a distribution over a set of trees in similar circumstances?
  - Stochastic grammars seem obvious to me

General Idea

- Use stochastic grammars (generally context free grammars, CFG) for the EDA probability model
  - Stochastic CFG: each production of the grammar has an attached probability
    - The probability, given an instance of the LHS nonterminal, that this production will be used to expand it
- Other (non-CFG) stochastic grammars have also been used

Grammar-Based EDAs

- The underlying model is a stochastic Context-Free Grammar:
  - B → B or B 0.6
  - B → B and B 0.3
  - B → not B 0.1

Grammars and Building Blocks

- Supports position-independent building blocks
  - B → B or B 0.05
  - B → C or D 0.90
  - B → not B 0.05
  - C → C and C 0.95
  - C → C or C 0.04
  - C → not C 0.01
  - D → D and D 0.03
  - D → D or D 0.02
  - D → not D 0.95
- B generates
  - **((? and ?) or (not ?))**
- with high probability
- Wherever B occurs

Fixed Context Grammar-based EDA-GP

Ratle and Sebag 2001: Scalar Stochastic Grammar SGGP

- Model:
  - Stochastic Context Free Grammars as above
- Learning:
  - Probability learning only
  - The required context structure must be in the grammar supplied by the user
  - Specifically, the only context learning is through grammar probabilities

Shan et al 2002: Ant-TAG

- Model:
  - Stochastic Tree Adjoining Grammars
    - Different kind of grammar with some advantages for GP
      - No need to check feasibility constraints
- Learning:
  - Probability learning only
  - No context learning

Evaluation

- Quite successful for the right problem
- But most problem domains require more learning of context
  - Example: max problem
    - Find the largest expression of a given depth that can be built from *, +, 0.5
      - (full tree, 0.5 in the leaves, + in the next two rows, * everywhere else)
  - Solution can't be represented in SGGP

Annotation-Learning SGGP

Learning Grammar Annotations

- Most commonly-used annotation is depth
  - E → E * E 0.99 depth < max_depth - 2
  - E → E + E 0.005 depth < max_depth - 2
  - E → 0.5 0.005 depth < max_depth - 2
  - E → E * E 0.005 max_depth - 2 < depth < max_depth
  - E → E + E 0.99 max_depth - 2 < depth < max_depth
  - E → 0.5 0.005 max_depth - 2 < depth < max_depth
  - E → E * E 0.005 depth = max_depth
  - E → E + E 0.005 depth = max_depth
  - E → 0.5 0.99 depth = max_depth
- Complete solution to max problem

Annotated

R&S: Vectorial SGP (2001) (also PEEL1 2002)

- Learn depth annotations in Stochastic CFGs
  - Successfully solved max problem and other similar problems
  - Able to learn position dependence vertically in trees
  - Unable to learn horizontal position dependences

PCFG-LA Hasegawa & Iba (2009)

- Learns general annotations in CFGs
  - Very powerful mechanisms
  - Limitations not well understood (by me)

Grammar Learning EDA-GP

Learning New Building Blocks

- But what if the right building blocks don't occur in the original grammar?

• Learning new building blocks requires learning new, more specific, grammar models:

- Start with
  - B → B or B
  - B → B and B 0
  - B → not B
- Learn
  - B → C or D
  - C → C and C
  - D → not D

Learning New Building Blocks

- Use Grammar Learning methods from natural language processing
  - Grammar Learning is extraordinarily computationally intensive
  - Current methods have been developed for noise-free single-shot learning
  - There is a need to develop efficient incremental grammar-learning methods for multi-shot learning in noisy environments

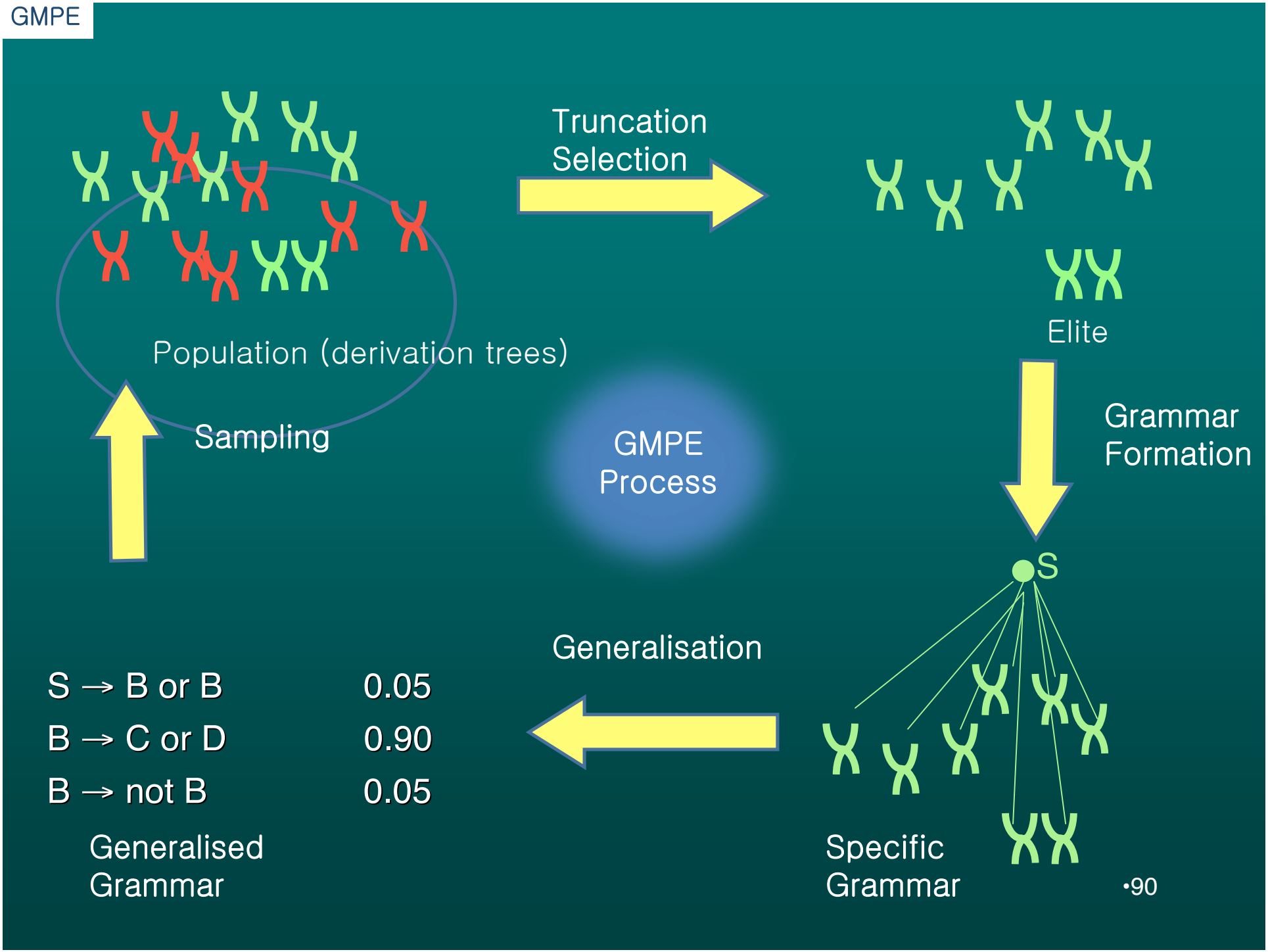
Learning New Grammars

- Grammar learning can be
  - Top-down or bottom-up (or inside-out)
  - Specific to general or general to specific (or both)
  - We have experimented with
    - Inside-out, general to specific
      - PEEL2 system, 2003
    - Bottom-up, specific to general
      - GMPE (Grammar Model-Based Program Evolution), 2004
    - Clearly many other possibilities are possible
      - But may require identifying large repeated sub-trees
  - De Jong & Bosman 2004
    - Similar ideas
    - Sampled expression trees, rather than derivation trees
      - Cost of parsing all individuals

Grammar Learning EDA-GP Example: GMPE

GMPE Structure

- Initial probability model: context-free grammar describing the solution space, uniform RHS probabilities for each LHS
- Sample from grammar
- Truncation selection
- Generate preliminary grammar covering exactly the selected instances
- Generalise the grammar ‘appropriately’
  - Stochastic grammar
  - Required to be a specialisation of the initial grammar


Truncation
Selection
Elite
Grammar
Formation
S
Specific
Grammar
Generalisation
S → B or B 0.05
B → C or D 0.90
B → not B 0.05
Generalised
Grammar
Sampling
Population (derivation trees)
GMPE
Process

GMPE Grammar Generalisation

- Based on Stolcke's merge operator
- Example

$X1 \rightarrow \lambda 1$

$X2 \rightarrow \lambda 2$

after merge:

$Y \rightarrow \lambda 1$

$Y \rightarrow \lambda 2$

How to Choose Merges to try?

- Problem:
  - Can’t try all possible merges
    - Quadratic in size of specific tree
- Solution:
  - Randomly try merges
  - Repeat until rate of successful merges falls too low

Acceptance Criterion for Merges?

- We follow Stolcke in using Minimum Encoding Inference
  - Fitness = Cost of encoding model + Cost of encoding errors
  - MIE minimises the fitness measure
  - In our case, accept a merge iff it reduces the model cost
- However we differ from Stolcke in how to compute the model cost
  - Stolcke uses a fairly simple cost metric of the kind often used in Minimimum Description Length (MDL) inference
  - Ignores prior knowledge about the likely form of solutions

MDL vs MML for Grammar Learning

- MDL works well in one-shot grammar learning
  - Easy and fast to compute
  - Values quite close to those given by MML
- Unfortunately, errors accumulate in multi-shot learning
  - Such as in EDA-GP
  - We were unable to get Stolcke's metric to work
    - Led to severe oscillatory behaviour between under- and over-generalisation
- Very careful analysis, leading to a more complex MML metric
  - Especially, Dirichlet prior for stochastic grammar probabilities
  - Led to much more stable behaviour
- Details in Yin Shan's thesis

GMPE Evaluation

- Works well on some difficult test problems
  - Can find solutions at much lower evaluation cost than other GP and EDA-GP algorithms
  - Can indeed find a useful description of the space of good solutions
- Difficult to scale because of
  - High sampling bias
    - Exponentially increasing with model depth
  - Very high learning cost
    - No model revision
    - Waste of information from truncation selection

Current and Future Directions

Sampling Bias in Grammar-Based EDA-GP

- Three primary sources of bias:
  - Amplification of sampling bias due to strict dependence structures
  - Discarding of infinite trees
  - Discarding of too-deep trees

Sampling Bias: Current Work

- Analysis of the amplification of sampling bias due to strict dependence
  - Experimental analysis
    - In press
  - Theoretical analysis
    - Largely complete, but not yet properly written up
- Proposals for Remediation
  - Proposed mechanism similar to likelihood weighting for evidence
  - Practically, seems to overcome the problem
  - No theoretical analysis yet

Sampling Bias: Future Work

- Analysis of the effect of discarding infinite trees
  - Should be feasible to compute the effect for each production
  - May be feasible to re-weight evidence from instances to compensate
    - Almost certainly requires an assumption that in the limit, fitness is independent of tree size
      - I.e. selection wouldn't change the sampling bias
    - Some justification for this from GP

Sampling Bias: Future Work

- Analysis of the effect of discarding too-large trees
  - Much more complex to compute the effect
  - May be reasonable to assume that the effect is the similar to discarding infinite trees, and appropriately increase the compensation weighting

Model Learning → Model Revision

- Incorporate Stolcke-style generalisation into a grammar revision framework
  - Complex, because the stochastic grammar has discarded information that may later become essential
  - Requires retaining additional population information beyond what is incorporated into the grammar
- We had made substantial progress on this
  - Recently, have been distracted from this work by the analysis of sampling bias, but hope to return to it soon

Greater Use of Fitness Information

- Stolcke-style learning requires weighted comparisons anyway
  - The number of instances contributing to this merge decision
- Easy to incorporate fitness levels into the weights
  - Results in non-integer weights, but algorithms are essentially unchanged

Alternative Grammar-Learning Mechanisms?

- Not clear that generalising learning is really the best choice for EDA-GP
  - Generalising learning often suffers from local optima
    - Difficult to foresee the benefit from a sequence of merge operations
  - Specialising learning might offer greater advantages
    - Especially, more readily incorporated into a grammar-revision framework
- Suggestions anyone?

Alternative Grammar Models

- Previously mentioned Ant-TAG
  - Fairly trivial version
    - Independent probability model, no grammar learning
  - TAGs have many advantages for this kind of work
    - But not aware of suitable work in TAG learning
- Suggestions anyone?

Summary

Summary

- GP is worth doing
  - Can often solve problems for which there is no other obvious method of attack
- GP needs improvement
  - Often fails to solve problems that don't seem unreasonably difficult
  - Doesn't scale well
  - Only describes a single solution, not a solution set
- EDA-GP has a lot of potential
  - Might scale better
  - Might describe a solution set well

Summary

- Two main kinds of EDA-GP
  - PPT-based
  - Grammar-based
- Grammar-based EDA-GP has a number of variants
  - No learnt structure
  - Learning grammar annotations
  - Learning grammar structure
- Some issues
  - Very high sampling bias
  - How to manage model revision?
  - How to incorporate maximum information from population?
  - How to learn suitable grammars?

Muchas Gracias