

COSC-452

Evolutionary Computation

Lee Spector

Registration

- You must sign sign-in sheet to remain registered
- If not registered but want to be, email me

- Introductions
- Course information
- Evolutionary computation

- **Introductions**
- Course information
- Evolutionary computation

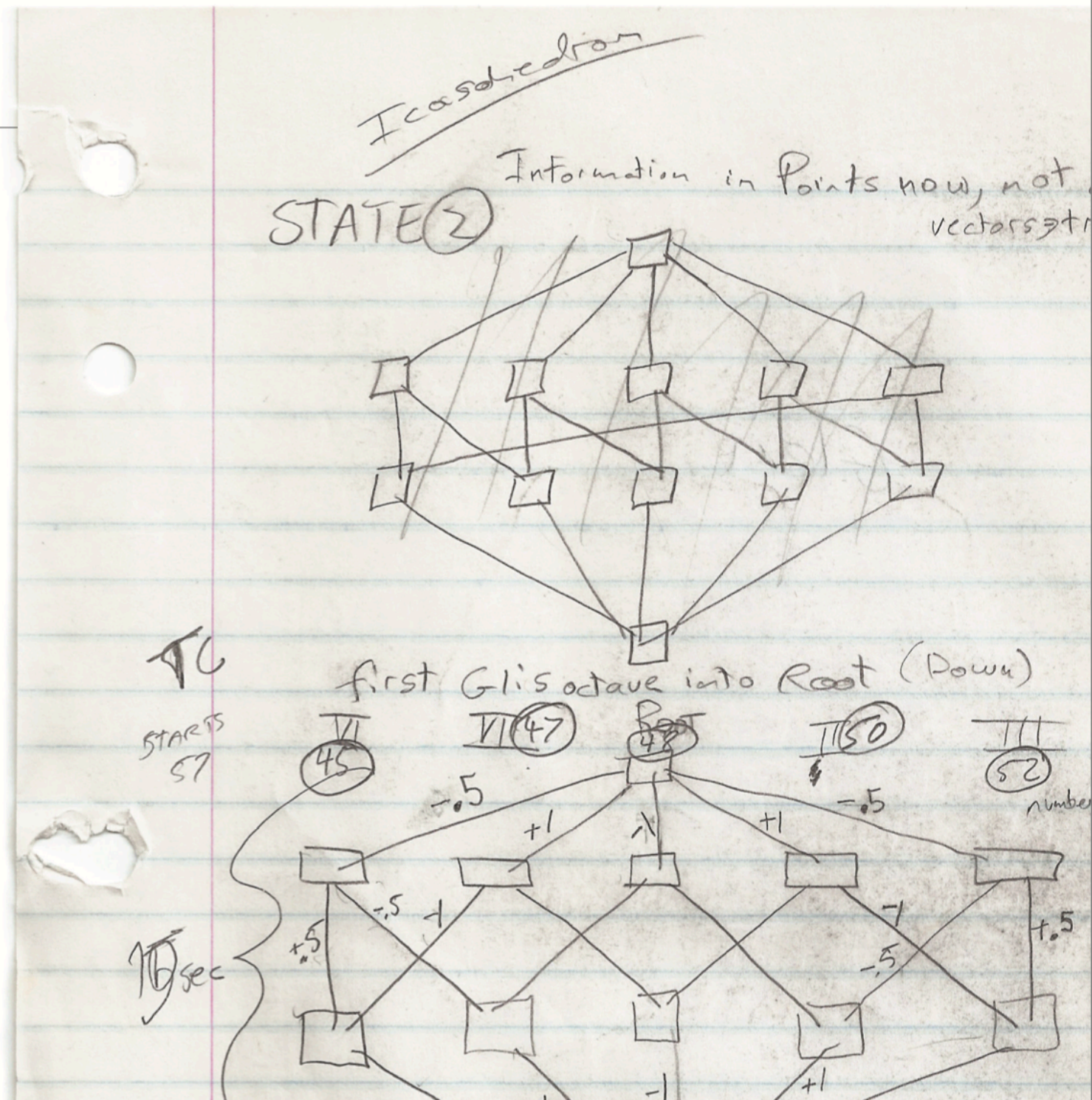
You

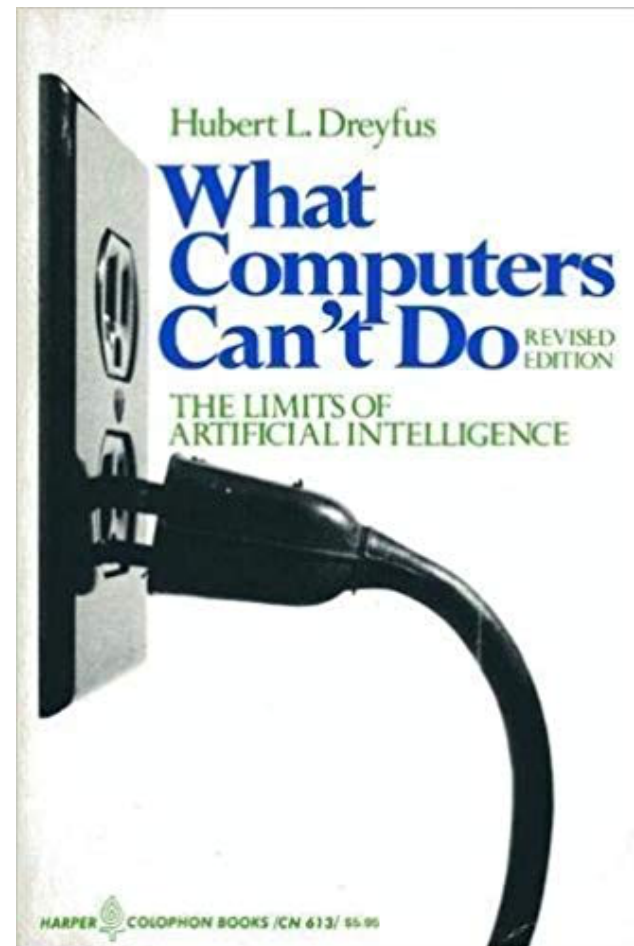
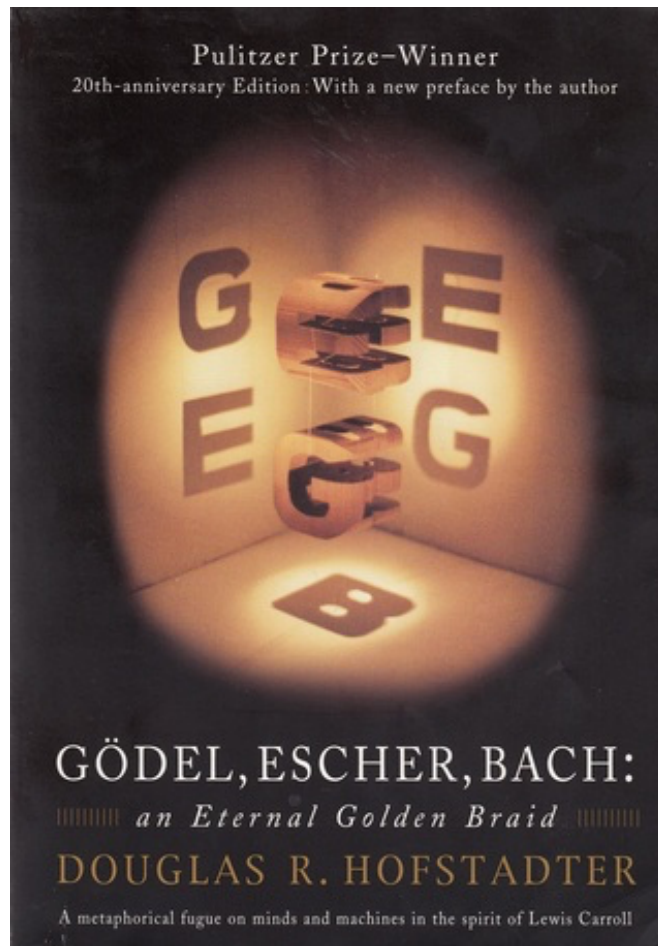
- Name
- Pronouns
- Year
- College if not Amherst
- Major(s) or possible major(s)
- Specific and/or non-computer-science interest(s)

MUSIC

The Spector Sound

"I happened to be a big Buckminster Fuller fan at the time," Spector explains, "and I decided to make a piece that translated the geometric objects he described in his





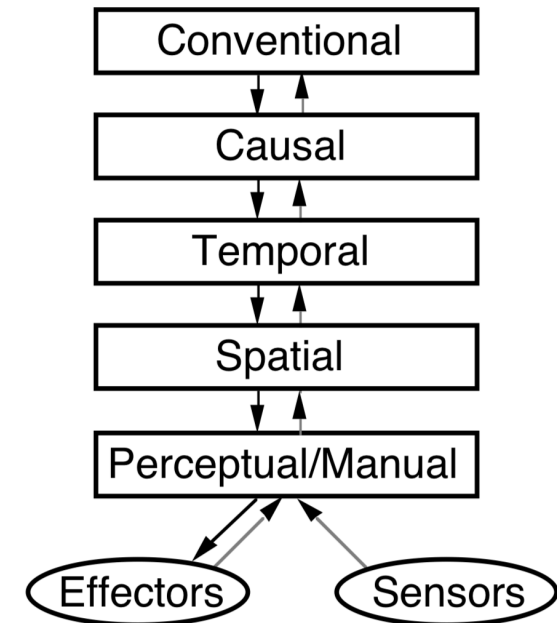
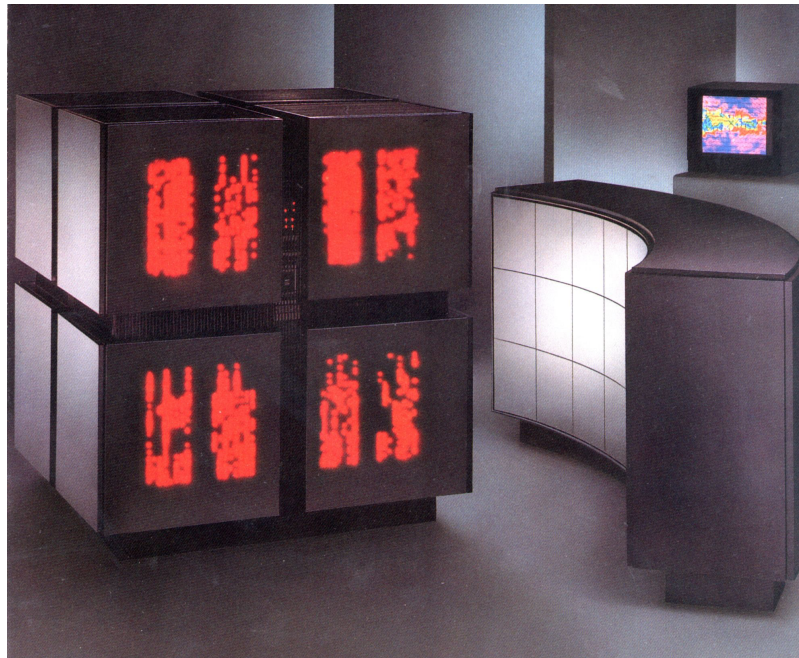


Figure 7. The specific levels of APE.



ELSEVIER

Cognitive Science 25 (2001) 941–975

<http://www.elsevier.com/locate/cogsci>

COGNITIVE
SCIENCE

Partial and total-order planning: evidence from normal and prefrontally damaged populations

Mary Jo Rattermann^{a,*}, Lee Spector^b, Jordan Grafman^c, Harvey Levin^d,
Harriet Harward^e

^aDepartment of Psychology, Franklin & Marshall College, Lancaster, PA 17604, USA

^bSchool of Cognitive Science, Hampshire College, Amherst, MA 01002, USA

^cNational Institute of Neurological Disorders and Stroke, Building 10; Room 5C205; 10 Center Drive; MSC 1440, Bethesda, MD 20892-1440, USA

^dDepartment of Physical Medicine and Rehabilitation, Baylor University College of Medicine, 1333 Moursound Avenue, A 205, Houston, TX 77030, USA

^eCallier Center for Communication Disorders, University of Texas at Dallas, 1966 Inwood Road, Dallas TX 75235, USA

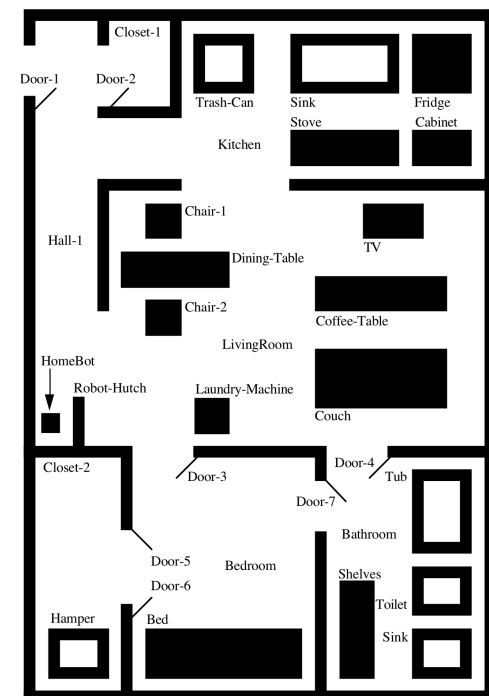
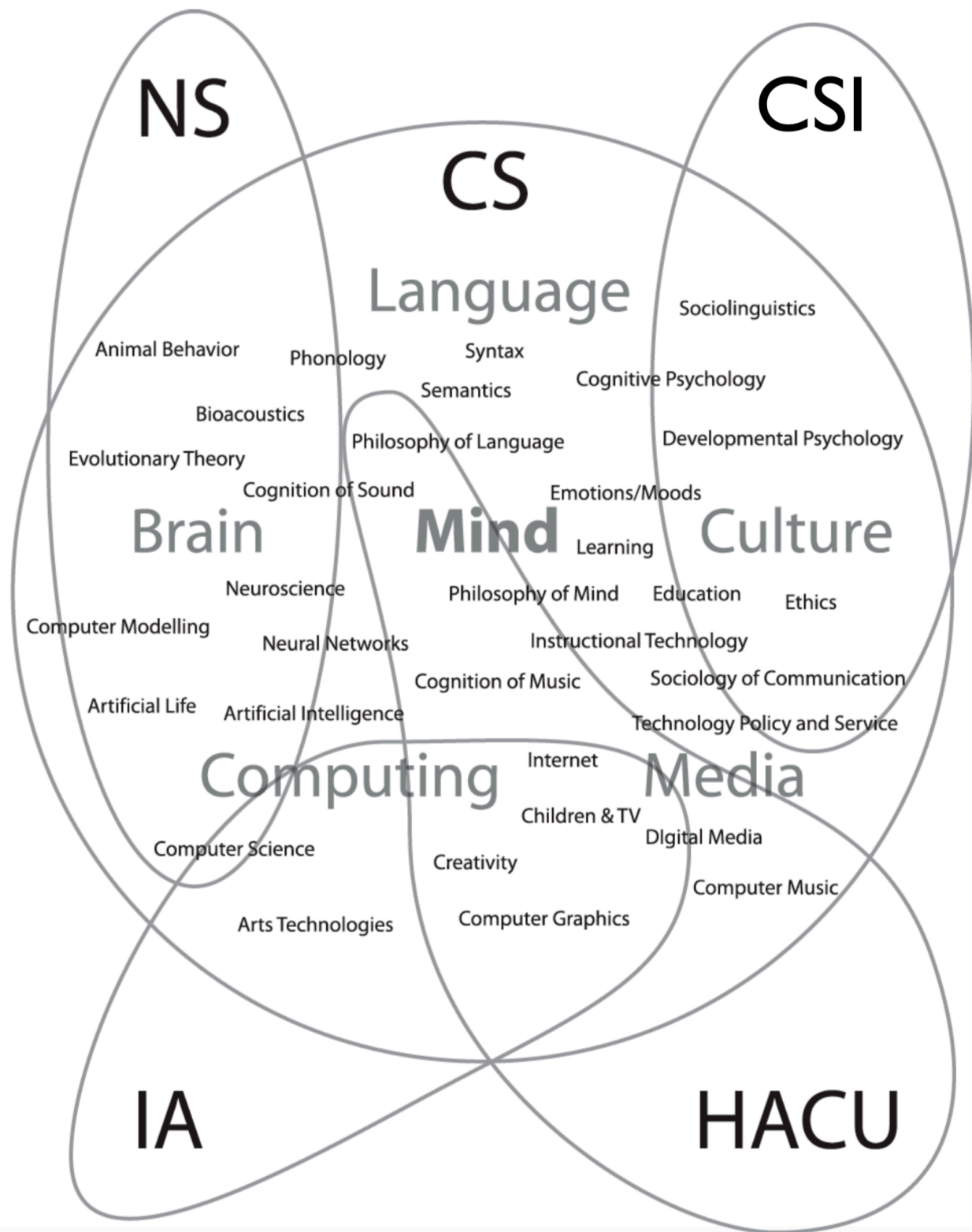


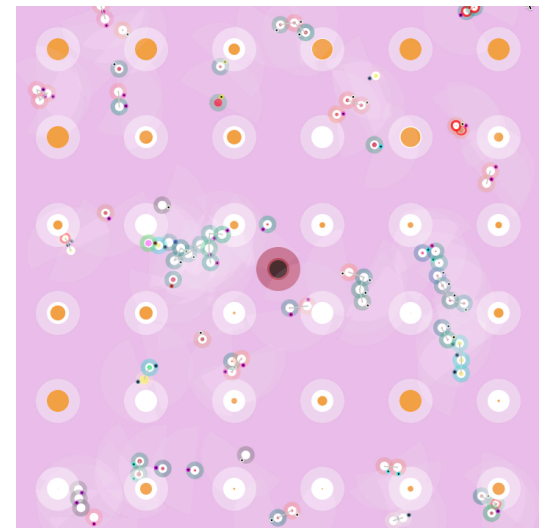
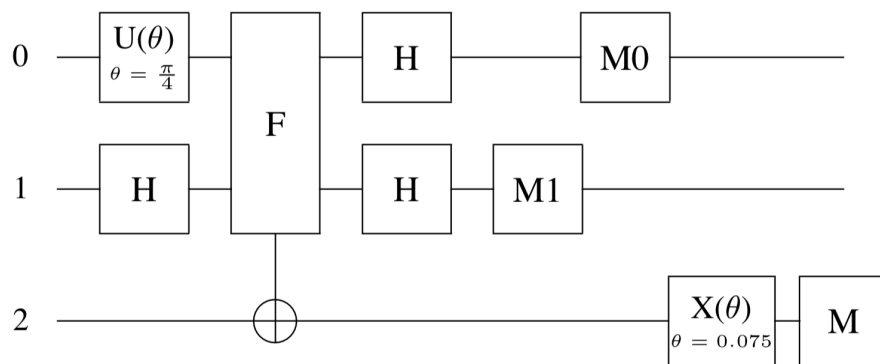
Figure 19. HomeBot's domain.



Advanced Topics in Artificial Intelligence
Algorithmic Arts
Animals and Animats: Natural and Artificial Intelligence and Behavior Artificial Intelligence
Artificial Intelligence in 3D Virtual Worlds
Beginning Coding for Science
Biocomputational Developmental Ecology
Code Immersion
Cognitive Science Fiction
Computational Models of Biological Systems
Computer Science Projects
Computing Concepts: Creative Machines?
Creative Programming Workshop
Current Issues in Cognitive Science
Evolutionary Computation
Genetic Programming
Hypertext
Introduction to Artificial Intelligence
Introduction to Cognitive Science
Introduction to Computer Science
Programming Creativity
Programming for Science
Programming Game Theory
Programming Language Paradigms
Quantum Computing with No Prerequisites of Any Kind
Radical Innovation in Digital Arts
Reasoning About Action
Research in Artificial Intelligence
Unconventional Computing
What Computers Can't Do (limits of computing)
When Machines Talk (natural language processing)

Integrated Teaching & Research

- Undergrad/grad/faculty collaboration
- Wide range of project areas
- Five College research group focusing on evolutionary computing



- Introductions
- **Course information**
- Evolutionary computation



Computer Science 452: Seminar in Computer Science: Evolutionary Computation

[Dashboard](#) ▶ [My courses](#) ▶ [1920S](#) ▶ [Computer Science](#) ▶ [Seminar in Computer Science: Evolutionary Computation \(COSC-452-1920S\)](#)

Turn editing on

NAVIGATION



[Dashboard](#)

[Site home](#)

▶ [Site pages](#)

▼ [My courses](#)

▼ [1920S](#)

▼ [Computer Science](#)

▶ [Data Structures 01 \(COSC-211-01-1920S\)](#)

▶ [Data Structures \(COSC-211-1920S\)](#)

▶ [Seminar in Computer Science: Evolutionary Computation \(COSC-452-1920S\)](#)

▶ [Participants](#)

[Grades](#)

▶ [General](#)

▶ [January 26 - February 1](#)

▶ [February 2 - February 8](#)

▶ [February 9 - February 15](#)

▶ [February 16 - February 22](#)

▶ [February 23 - February 29](#)

▶ [March 1 - March 7](#)



[Syllabus](#)

General course information and policies. Be sure to read all of this carefully.



[Schedule](#)

This is where you'll find detailed information about what we're doing each day, what you should read and turn in, etc. It will be populated and adjusted incrementally as the course proceeds, so check back frequently.



[Clojure Resources](#)

Links to Clojure texts, tools, tutorials, etc.



[Announcements](#)



[Anonymous Forum](#)

January 26 - February 1

February 2 - February 8



[Week 1 Portfolio](#)



[Automatic Quantum Computer Programming: A Genetic Programming Approach](#)

For this week, you should just read Chapter 4: "Genetic and Evolutionary Programming"

February 9 - February 15

Syllabus

Amherst College, Spring 2020

COSC-452: Evolutionary Computation

Instructor: Professor [Lee Spector](#) (he/him), SCCE C211, lspector@amherst.edu

Class Meetings: Tuesdays and Thursdays, 2:30-3:50, in SCCE A331

Office hours:

Sign up for regular slots (M 1:30-3:30 & 4:30-5:30, Tu 1:30-2:15, Th 12:30-2:15) [here](#).

Other times can be arranged by [email](#).

Description: Evolutionary computation techniques harness mechanisms of biological evolution, including mutation, recombination, and selection, to build software systems that solve difficult problems or shed light on the nature of evolutionary processes. In this course students will explore several evolutionary computation techniques and apply them to problems of their choosing. The technique of genetic programming, in which populations of executable programs evolve through natural selection, will be emphasized.

Objectives:

- To understand and apply evolutionary computation methods.
- To develop skills in "functional"-style programming.
- To conduct independent and collaborative programming-based project work.

Overview: In the first part of the course we will survey the field of evolutionary computation, develop functional programming skills in the Clojure programming language, and use Clojure to begin implementing and experimenting with evolutionary algorithms. Near the middle of the term we will pitch project ideas, vote on pitches, and form project groups. For the remainder of the semester we will engage in project work while also exploring advanced topics in evolutionary computation research. The last week of the semester will be dedicated to project presentations.

Texts: All readings and other materials will be distributed on the class Moodle site.

Software: Java and **IntelliJ IDEA** with the **Cursive** plugin, or an acceptable alternative Clojure IDE (as discussed in class).

Hardware: IntelliJ/Cursive is installed on Science Center computers, but it would be best also to install it on a computer of your own if you are able to do so.

Expectations and Grading: You are expected to attend class, read the assigned readings, participate in all class activities (except when you really can't or shouldn't, for example because of illness), submit all assigned work on time, and respond appropriately to feedback. You are also expected to demonstrate, through your submitted work and participation, that you have engaged seriously with the course material and mastered it to the extent that you are able. You will get an **A** if you meet these expectations, an **A+** if you meet them and produce exceptionally strong work, and an **A-** if you fall short but only in minor ways. You will get a **B** if you fall short in significant ways but nonetheless demonstrate substantial engagement and learning, and you will get a lower grade for weaker performance. I will provide feedback on your performance periodically. If you have questions about your grade as the course progresses, then please check in with me.

Portfolios: Most of your work will be submitted as updates to a cumulative portfolio of text and code that you will be building throughout the semester. You will submit an update to your portfolio each week, each of which should have a name like "Week 1", "Week 2", etc., and each of which should contain:

- Overview: A text file containing a brief description of the week's updates (max one page). For collaborative work, each collaborator's contribution must be explicitly noted.
- RICE Report: A text file containing a very brief (one or two sentence) report on the week's RICE activity (see below).
- Text: A folder containing new notes on readings, ideas for project work, and other reflections on the course material.
- Code: A folder containing new code developed for work in the course.

In general, you should be submitting only new materials in each update, since all of your submissions taken together, over the course of the semester, will constitute your final portfolio. In some cases, however, it may make sense to submit an updated version of an item from a previous week. If you do this, then you should be sure to note that you are doing so, and clearly describe what is new in your Overview.

RICE: Required Immersive Collaborative Experience: A Required Immersive Collaborative Experience (RICE) session is an out-of-class activity in which you meet with a randomly assigned partner and spend at least 30 minutes discussing and/or collaboratively engaging in work for the course, with at most one screen between the two of you. You will be assigned a RICE partner each week, and each week you should include a report in your portfolio, with the following format:

Who: (partner name)

When: (when you met)

Where: (where you met)

What: (what you did, in just a sentence or maybe two)

If you are unable to meet with your RICE partner in a particular week, then you should submit a description of the circumstances. If there is someone in the class with whom you would prefer not to be paired, please send me an email about this, which I will treat as confidential.

Demonic Coding: In some class sessions we will engage in Demonic Coding sessions:

- The class is split into "coders" and "demons," and each demon is paired with a coder.
- Coders begin coding on whatever they are working on for the course.
- Demons observe, ask questions, and make suggestions. Demons with fewer skills than their coders can ask more questions, while those with more skills can make more suggestions. Roughly 50% of a coder's time should be devoted to demonic interactions, with the rest devoted to making progress on the code.
- From time to time, demons rotate to other coders.
- Halfway through the session, all coders become demons and all demons become coders.

Demonic coding sessions may be started sometimes without warning, so you should always have access to your code.

Moodle forum: You are strongly encouraged to submit questions and answers to the class Moodle forum. Anonymity is the default, and you are welcome to remain anonymous or to identify yourself depending on the situation. I will not look at identities of anonymous posters unless it is necessary to deal with a problem.

Support: My goal is for you to succeed in this course, to learn a lot and to earn a good grade. If you are having trouble, then I want to know about it and I want to help. Please make proactive use of the support systems built into the course (the Moodle forum and my office hours), and contact me if you may need additional support.

Phones and other devices: Feel free to use whatever devices you want if they help you to engage with the class by taking notes, doing searches, executing code, etc., but you should indeed be 100% engaged in the collective work of the class throughout every class meeting.

Accommodations: If you have a documented disability that requires accommodations, you will need to register with Accessibility Services for coordination of your academic accommodations. You can reach them via email at accessibility@amherst.edu, or via phone at 413-542-2337. Once you have your accommodations in place, I will be glad to meet with you privately during my office hours or at another time to discuss the best implementation of your accommodations.

Honor code: The [Amherst College Honor Code](#) applies to this course, as it does to all other Amherst College courses.

Schedule

This schedule will be augmented and adjusted as the course proceeds.

Class 1 (Tuesday, January 28)

Before:

- Nothing

In class:

- Introductions
 - What we will do in this course and why
-

Class 2 (Thursday, January 30)

Before:

- Read "[And now, digital evolution](#)," by Lee Spector
- Read "[Evolution of artificial intelligence](#)," by Lee Spector
- Read the "Getting Started" section of the [Cursive User Guide](#)
- Begin reading [Chapter 3: "Do things: a Clojure Crash Course"](#) of *Clojure for the Brave and True*

In class:

- Installfest
- [Clojinc](#)

Class 3 (Tuesday, February 4)

Before:

- Submit Week 1 portfolio
- Read Chapter 4: "Genetic and Evolutionary Programming" of *Automatic Quantum Computer Programming: A Genetic Programming Approach*, by Lee Spector
- Read "[Beating the Averages](#)," by Paul Graham
- Finish reading [Chapter 3: "Do things: a Clojure Crash Course"](#) of *Clojure for the Brave and True*

In class:

- [Clojestions](#)
 - EvolveSum
-

Class 4 (Thursday, February 6)

Before:

- Read "[Analyzing a Decade of Human-Competitive \("HUMIE"\) Winners: What Can We Learn?](#)," by Kannappan et al.
- Try [4Clojure](#)

In class:

- Humies
- Demonic coding

- Introductions
- Course information
- **Evolutionary computation**

Evolutionary Computation

Lee Spector

Amherst College, Hampshire College, UMass Amherst

This material is based upon work supported by the National Science Foundation under Grant No. 1617087. Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of the National Science Foundation.



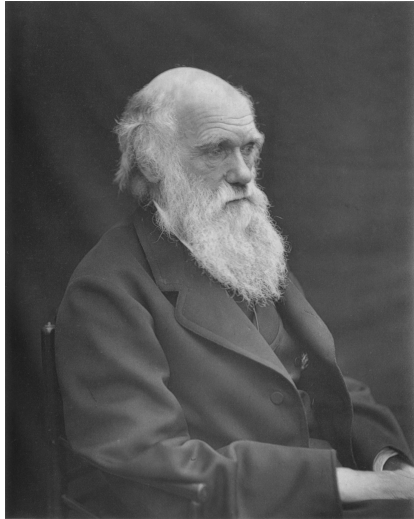
Outline

- What is evolutionary computation?
- What can it do?
- Improving it
- Connections
- COSC-452

Outline

* **What is evolutionary computation?**

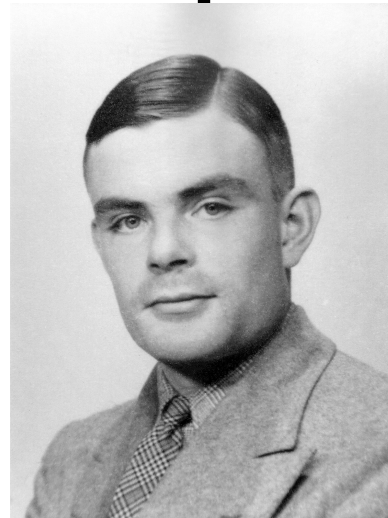
- What can it do?
- Improving it
- Connections
- COSC-452



Charles
Darwin



Ada
Lovelace



Alan
Turing

https://en.wikipedia.org/wiki/Charles_Darwin

<https://www.cnn.com/ampstories/tech/meet-ada-lovelace-the-first-computer-programmer>

<https://www.britannica.com/biography/Alan-Turing>

EvoBio & CompSci

- Bioinformatics
- Modeling & simulation
- Artificial life
- Evolutionary algorithms

EvoBio & CompSci

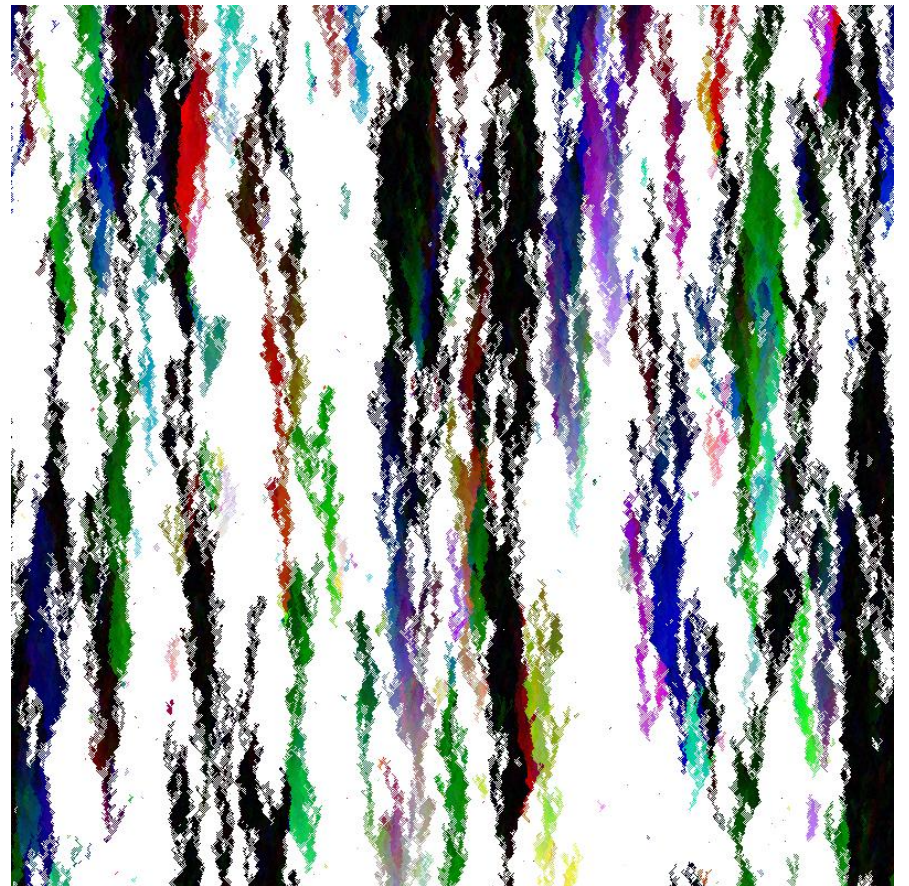
* **Bioinformatics**

- Modeling & simulation
- Artificial life
- Evolutionary algorithms



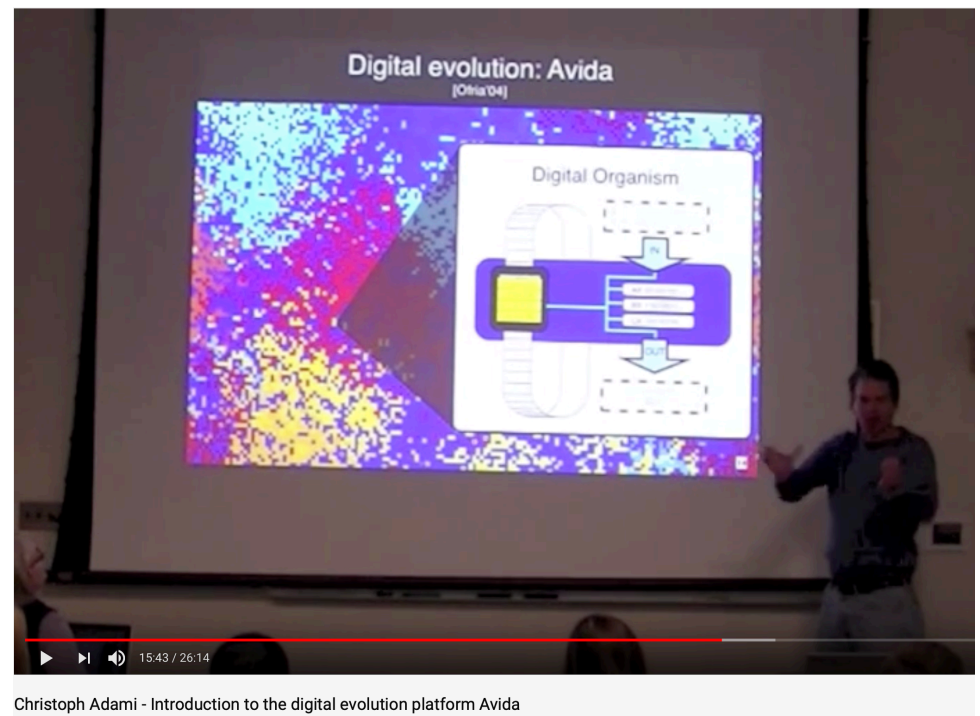
EvoBio & CompSci

- Bioinformatics
- * **Modeling & simulation**
- Artificial life
- Evolutionary algorithms



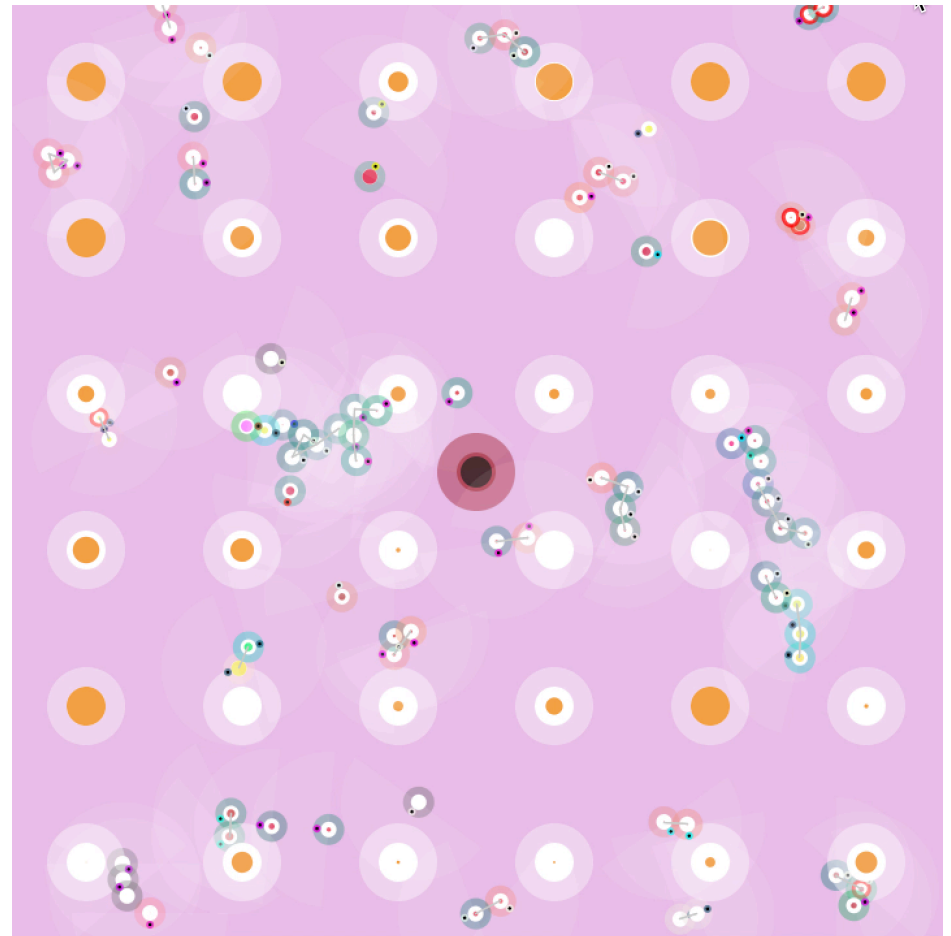
EvoBio & CompSci

- Bioinformatics
- Modeling & simulation
- * **Artificial life**
- Evolutionary algorithms



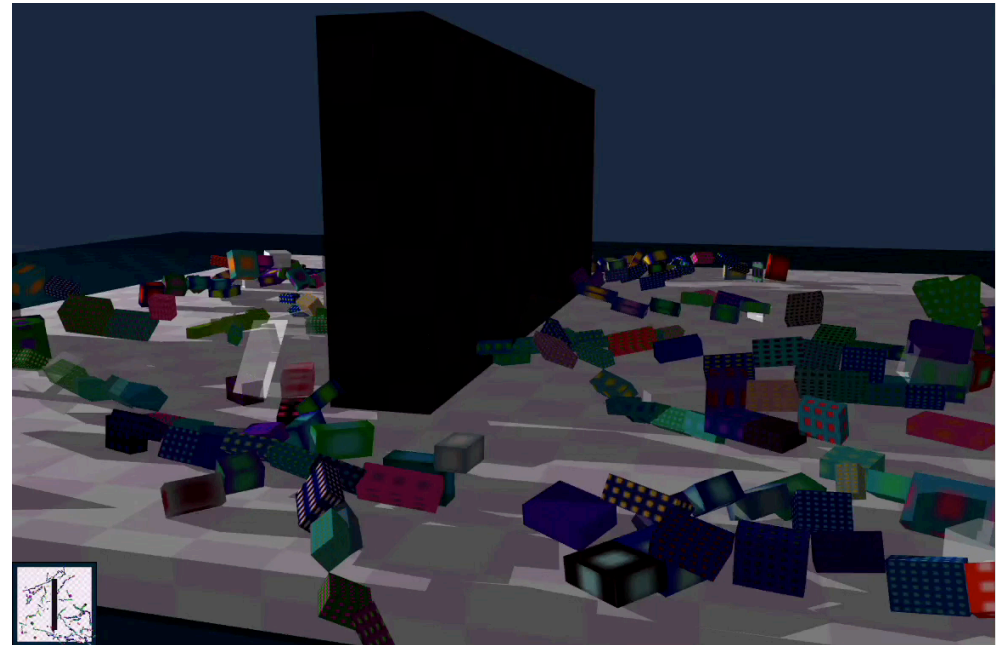
EvoBio & CompSci

- Bioinformatics
- Modeling & simulation
- * **Artificial life**
- Evolutionary algorithms



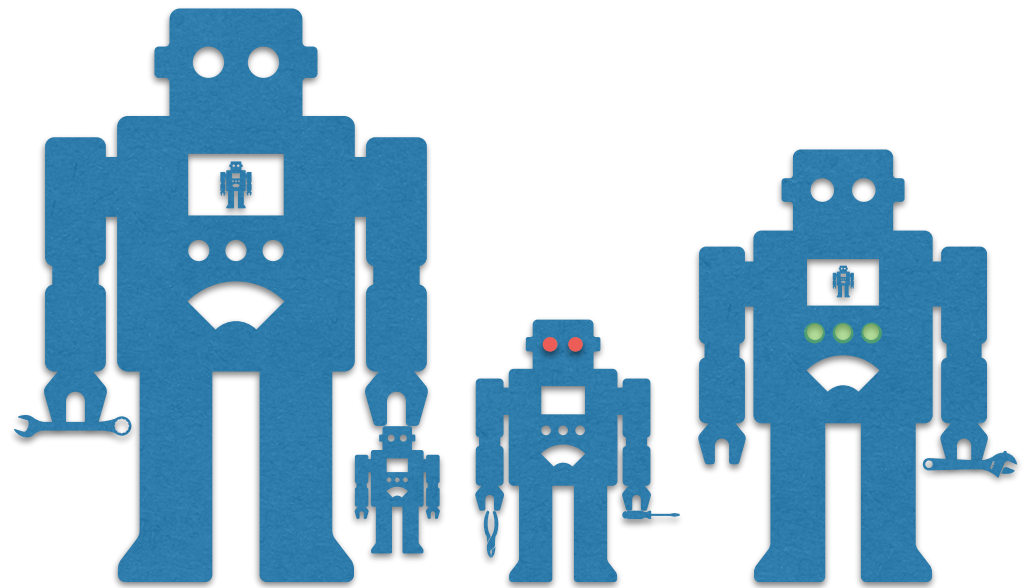
EvoBio & CompSci

- Bioinformatics
- Modeling & simulation
- * **Artificial life**
- Evolutionary algorithms



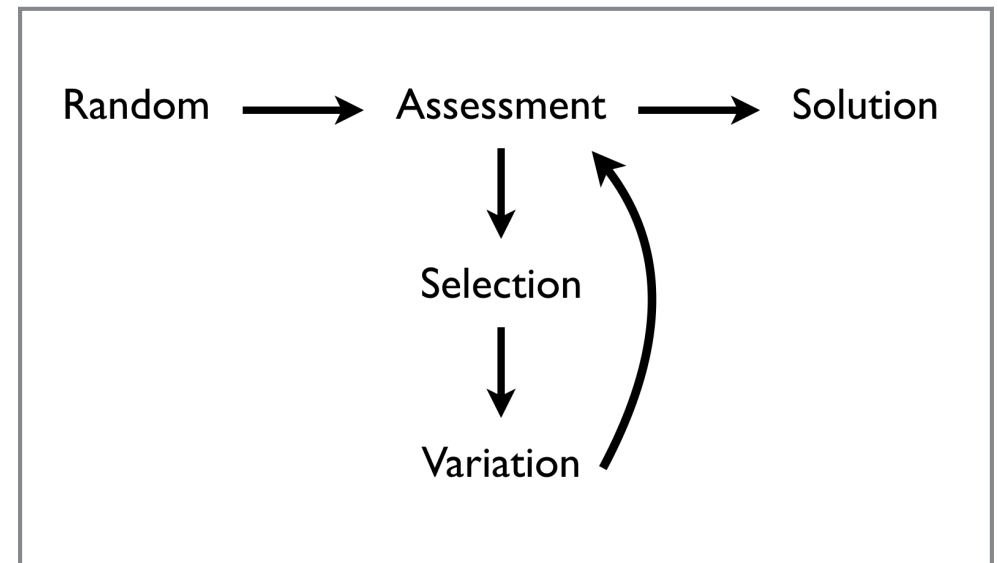
EvoBio & CompSci

- Bioinformatics
- Modeling & simulation
- * **Artificial life**
- Evolutionary algorithms



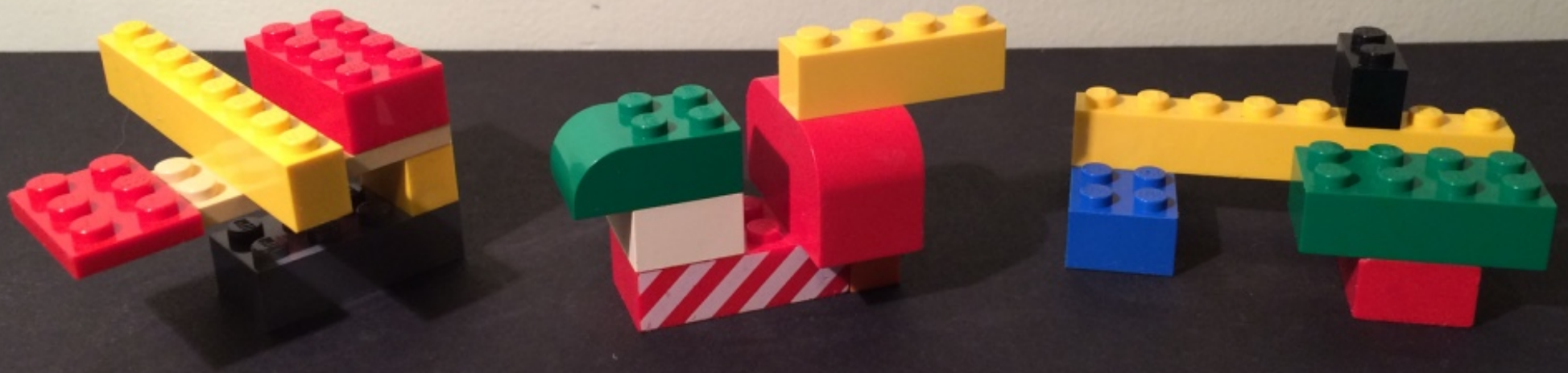
EvoBio & CompSci

- Bioinformatics
- Modeling & simulation
- Artificial life

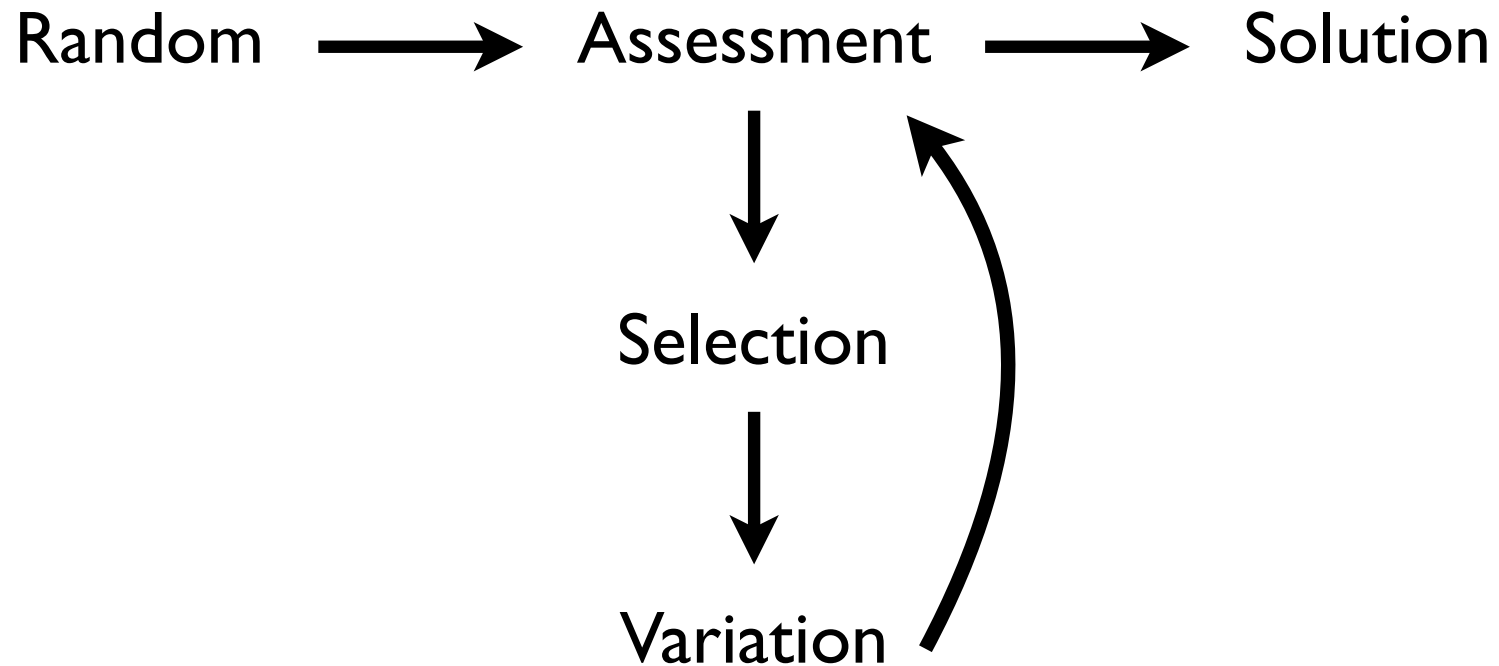


* **Evolutionary algorithms**

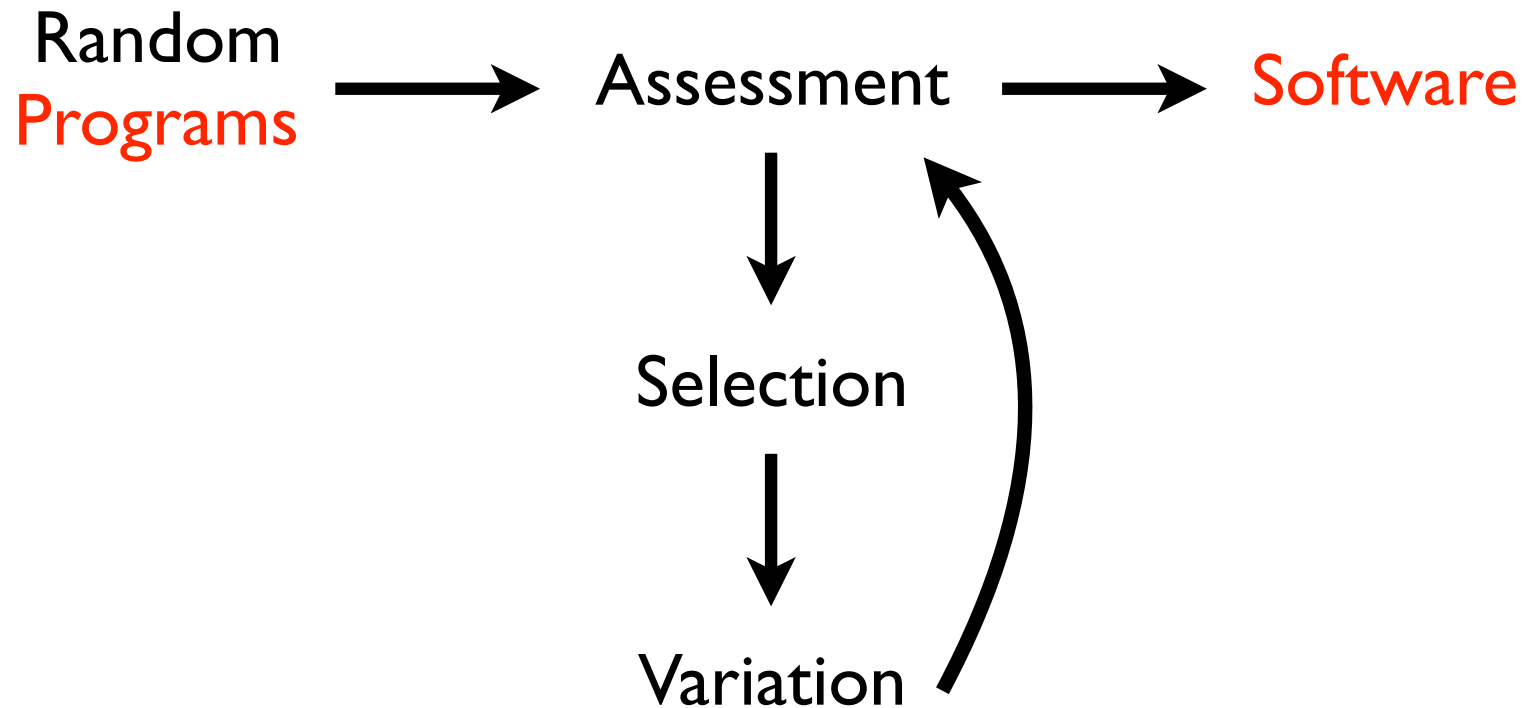
Evolving LEGO bridges



Evolutionary Algorithms



Genetic Programming



Outline

- What is evolutionary computation?

- ✱ **What can it do?**

- Improving it
- Connections
- COSC-452



Annual "Humies" Awards For Human-Competitive Results

Produced By Genetic And Evolutionary Computation

The result was ***patented as an invention*** in the past is an improvement over a patented invention or would qualify today as a patentable new invention.

The result is equal to or better than a result that was accepted as a ***new scientific result*** at the time when it was published in a peer-reviewed scientific journal.

The result is equal to or better than a result that was placed into a database or archive of results maintained by an ***internationally recognized panel of scientific experts***.

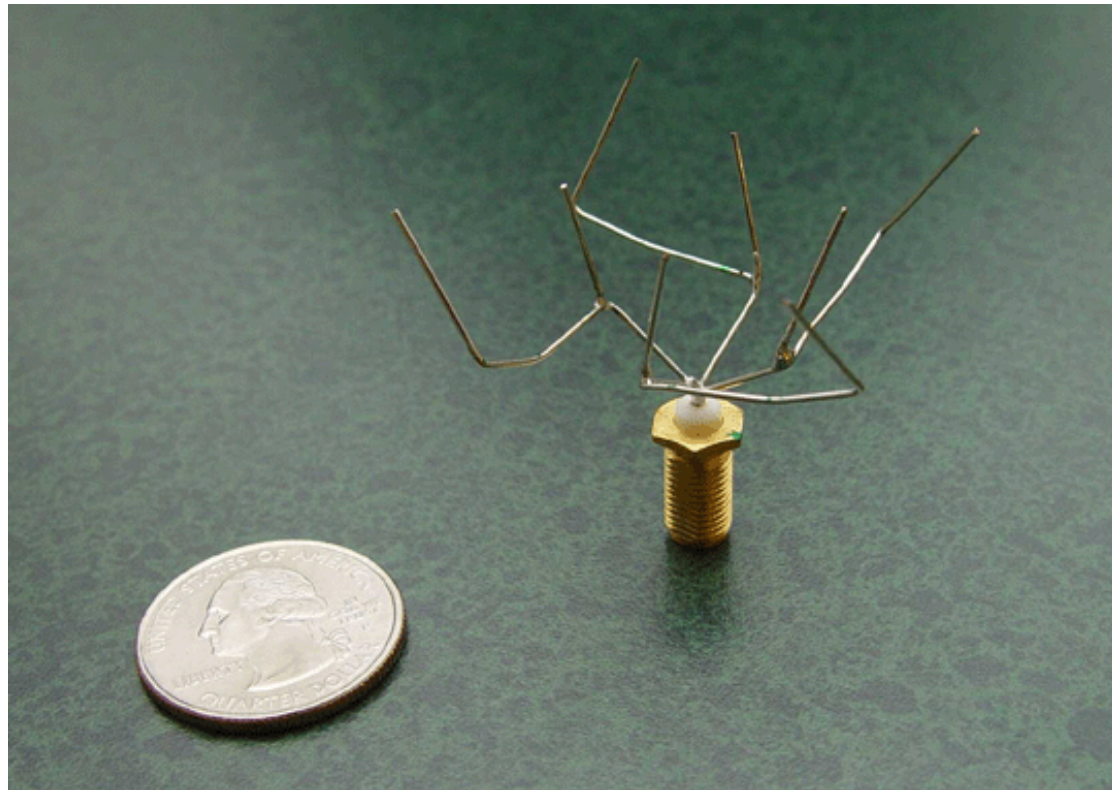
The result is ***publishable in its own right*** as a new scientific result independent of the fact that the result was mechanically created.

The result is ***equal to or better than the most recent human-created solution*** to a long-standing problem for which there has been a succession of increasingly better human-created solutions.

The result is equal to or better than a result that was considered an ***achievement in its field*** at the time it was first discovered.

The result ***solves a problem of indisputable difficulty*** in its field.

The result holds its own or ***wins a regulated competition involving human contestants*** (in the form of either live human players or human-written computer programs).



An Evolved Antenna for Deployment on NASA's Space Technology 5 Mission

Jason D. Lohn, Gregory S. Hornby, Derek S. Linden
NASA Ames Research Center

Humies Gold Medal, 2004

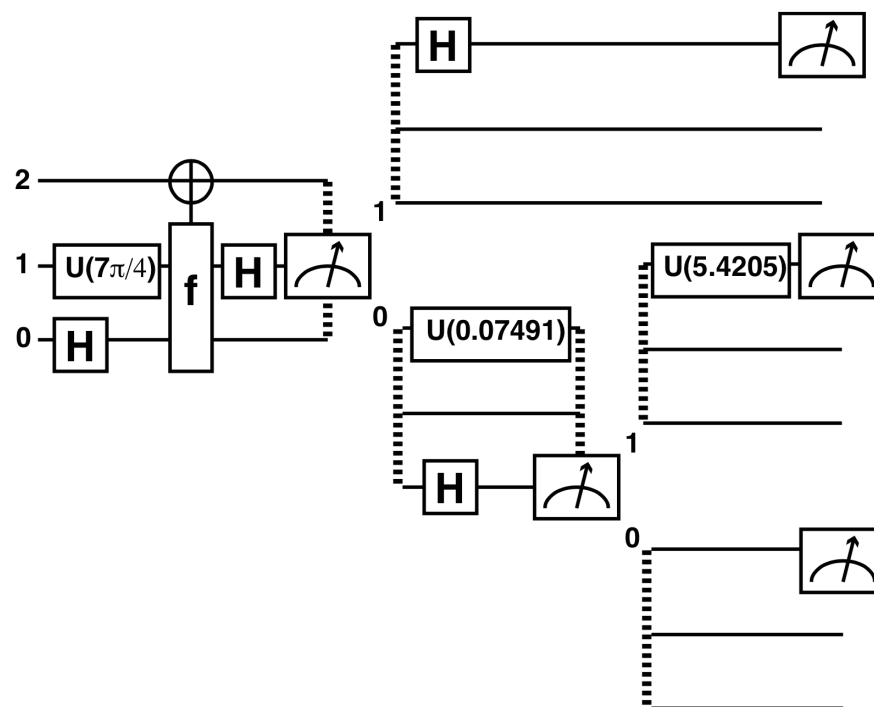
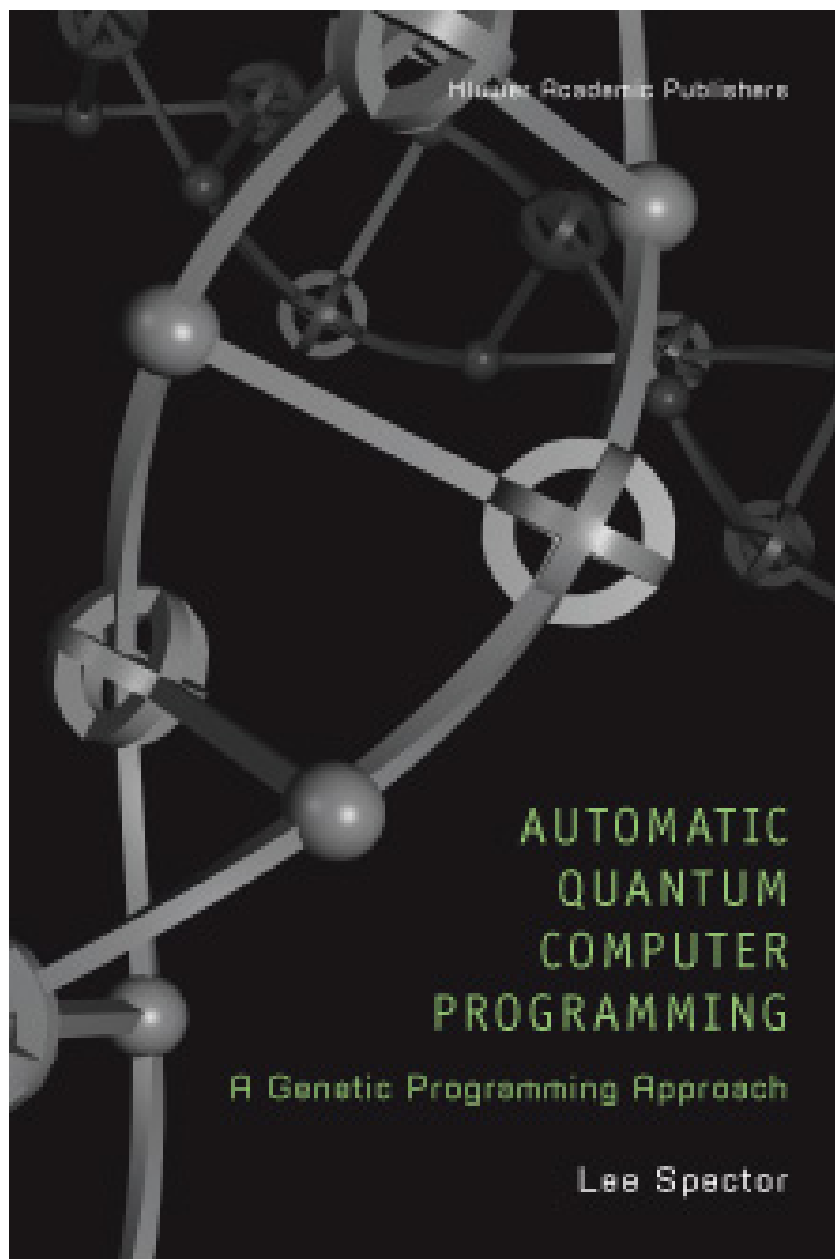


Figure 8.11. A gate array diagram for an evolved solution to the AND/OR oracle problem. The gate marked “f” is the oracle. The sub-diagrams on the right represent the possible execution paths following the intermediate measurements.

Lee Spector
Hampshire College

Humies Gold Medal, 2004

Genetic Programming for Finite Algebras

Lee Spector
Cognitive Science
Hampshire College
Amherst, MA 01002
lspector@hampshire.edu

David M. Clark
Mathematics
SUNY New Paltz
New Paltz, NY 12561
clarkd@newpaltz.edu

Ian Lindsay
Hampshire College
Amherst, MA 01002
iml04@hampshire.edu

Bradford Barr
Hampshire College
Amherst, MA 01002
bradford.barr@gmail.com

Jon Klein
Hampshire College
Amherst, MA 01002
jk@artificial.com

$$\begin{aligned} &(((((((X * (Y * X)) * X) * Z) * (Z * X)) * ((X * (Z * (X * (Z * Y)))) * Z)) * \\ &Z) * Z) * (Z * (((X * ((Z * Z) * X) * (Z * X))) * X) * Y) * (((Y * (Z * (Z * \\ &Y)))) * (((Y * Y) * X) * Z)) * (X * (((Z * Z) * X) * (Z * (X * (Z * Y)))))) \end{aligned}$$

Humies Gold Medal, 2008

International Journal of Algebra and Computation | Vol. 28, No. 05, pp. 759-790 (2018)

Evolution of algebraic terms 3: Term continuity and beam algorithms

David M. Clark ✉ and Lee Spector

Fixing software bugs in 10 minutes or less using evolutionary computation

University of New Mexico

Stephanie Forrest

ThanhVu Nguyen

University of Virginia

Claire Le Goues

Westley Weimer



Humies Gold Medal, 2009

Yavalath

Yavalath is an abstract board game for two or three players, invented by a computer program called LUDI. It has an easy rule set that any player can pick up immediately, but which produces surprisingly tricky emergent play.

Yavalath is available from [nestorgames](http://nestorgames.com), making it the first — and still only — computer-generated game to be commercially published, together with its sister game [Pentalath](http://pentath.com).

In October 2011, Yavalath was ranked in the top #100 abstract board games ever invented on the [BoardGameGeek](http://boardgamegeek.com) database. This helped it win the GECCO "Humies" gold medal for human-competitive results in evolutionary computation for 2012.

Here is a Yavalath [article](#) in the November 2013 issue of Bitcoin magazine.

Rules

The board starts empty.

Two players take turns adding a piece of their colour to an empty cell.

Win by making a line-of-4 (or more) pieces of your colour.

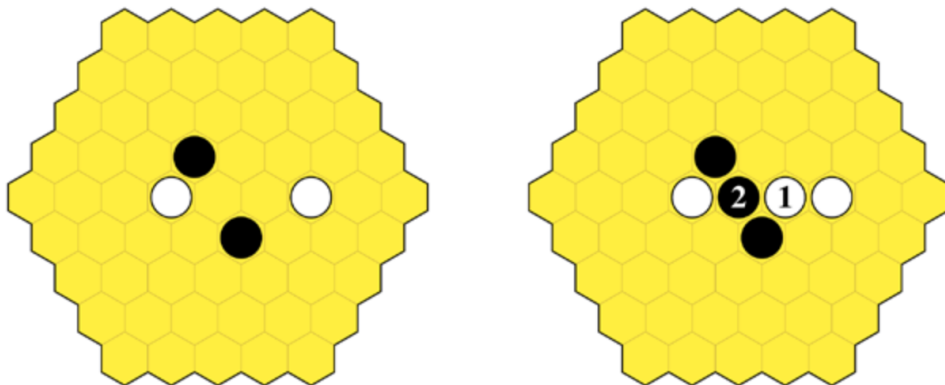
Lose by making a line-of-3 pieces of your colour beforehand.

Draw if the board otherwise fills up.

No, players are not allowed to pass.

Tactics and Strategy

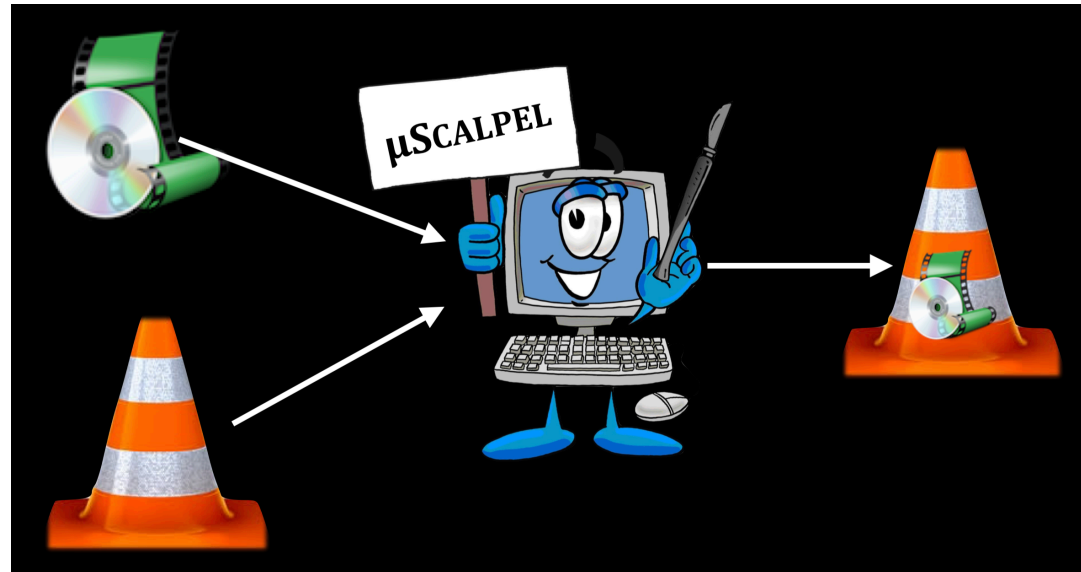
The key tactical play in Yavalath is the *forcing move*, as shown below. White move 1 forces Black to lose with the blocking move 2.



Cameron Browne
Imperial College London

Humies Gold Medal, 2012

Automated Software Transplantation



Earl T. Barr, Mark Harman, Yue Jia, Alexandru Marginean, Justyna Petke
University College London

Humies Gold Medal, 2016

Darwinian Data Structure Selection

Michail Basios
University College London, UK
michail.basios@cs.ucl.ac.uk

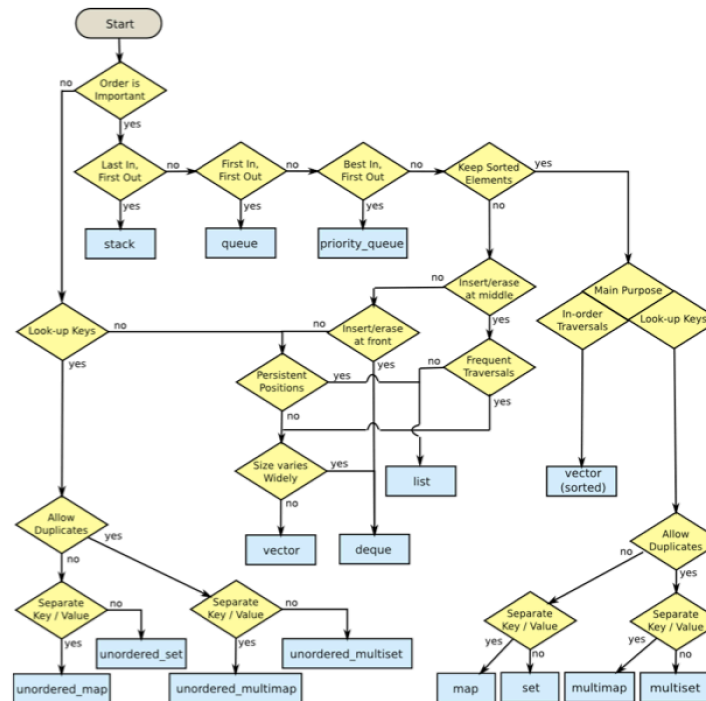
Lingbo Li
University College London, UK
lingbo.li@cs.ucl.ac.uk

Fan Wu
University College London, UK
fan.wu@cs.ucl.ac.uk

Leslie Kanthan
University College London, UK
l.kanthan@cs.ucl.ac.uk

Earl T. Barr
University College London, UK
e.barr@cs.ucl.ac.uk

Data Structure Selection/Optimisation Process



Hmmm, maybe just use defaults that work in most cases.



<https://www.human-competitive.org/sites/default/files/basiosslides.pptx>

Humies Bronze Medal, 2019

<https://www.human-competitive.org/awards>

Physics

Evaluation of bi-objective treatment planning for high-dose-rate prostate brachytherapy—A retrospective observer study

Stefanus C. Maree^{1,*}, Ngoc Hoang Luong², Ernst S. Kooreman¹, Niek van Wieringen¹, Arjan Bel¹, Karel A. Hinnen¹, Henrike Westerveld¹, Bradley R. Pieters¹, Peter A.N. Bosman^{2,3}, Tanja Alderliesten¹

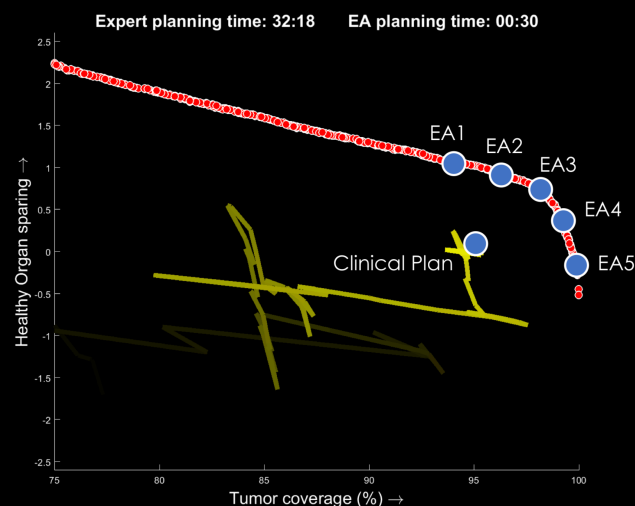
¹Department of Radiation Oncology, Amsterdam UMC, University of Amsterdam, Amsterdam, The Netherlands

²Life Sciences and Health Research Group, Centrum Wiskunde & Informatica, Amsterdam, The Netherlands

³Algorithmics group, Department of Software Technology, Faculty of Electrical Engineering, Mathematics and Computer Science, Delft University of Technology, Delft, The Netherlands

Human competition

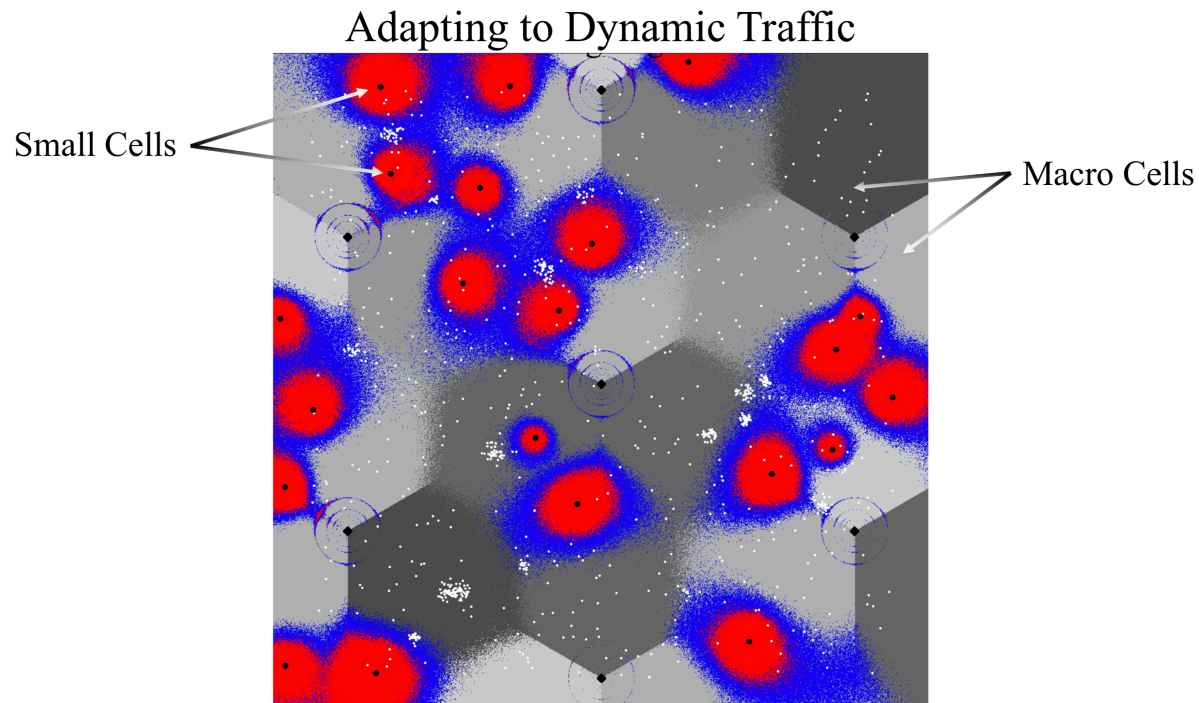
- 18 prostate cancer patients
- Blinded comparison between 6 plans
- 3 experienced radiation oncologists



Humies Silver Medal, 2019

Automated Self-Optimization in Heterogeneous Wireless Communications Networks

David Lynch^{ID}, Michael Fenton, David Fagan, Stepan Kucera, *Senior Member, IEEE*,
Holger Claussen, *Senior Member, IEEE*, and Michael O'Neill



Humies Gold Medal, 2019

Automatic identification of wind turbine models using evolutionary multiobjective optimization

William La Cava ^a  , Kourosh Danai ^a, Lee Spector ^b, Paul Fleming ^c, Alan Wright ^c, Matthew Lackner ^a

 [Show more](#)


<https://doi.org/10.1016/j.renene.2015.09.068>

[Get rights and content](#)

Highlights

- Accurate, succinct models of wind turbine dynamics are identified from normal operating data.
- A novel evolutionary multi-objective optimization system is described.
- The proposed method produces physically meaningful models without prior knowledge of the system.
- The method is bench-marked against other modeling techniques.

Multidimensional genetic programming for multiclass classification

William La Cava^a  , Sara Silva^{b, c, d}, Kourosh Danai^e, Lee Spector^f, Leonardo Vanneschi^c, Jason H. Moore^a

 **Show more**

<https://doi.org/10.1016/j.swevo.2018.03.015>

[Get rights and content](#)

Abstract

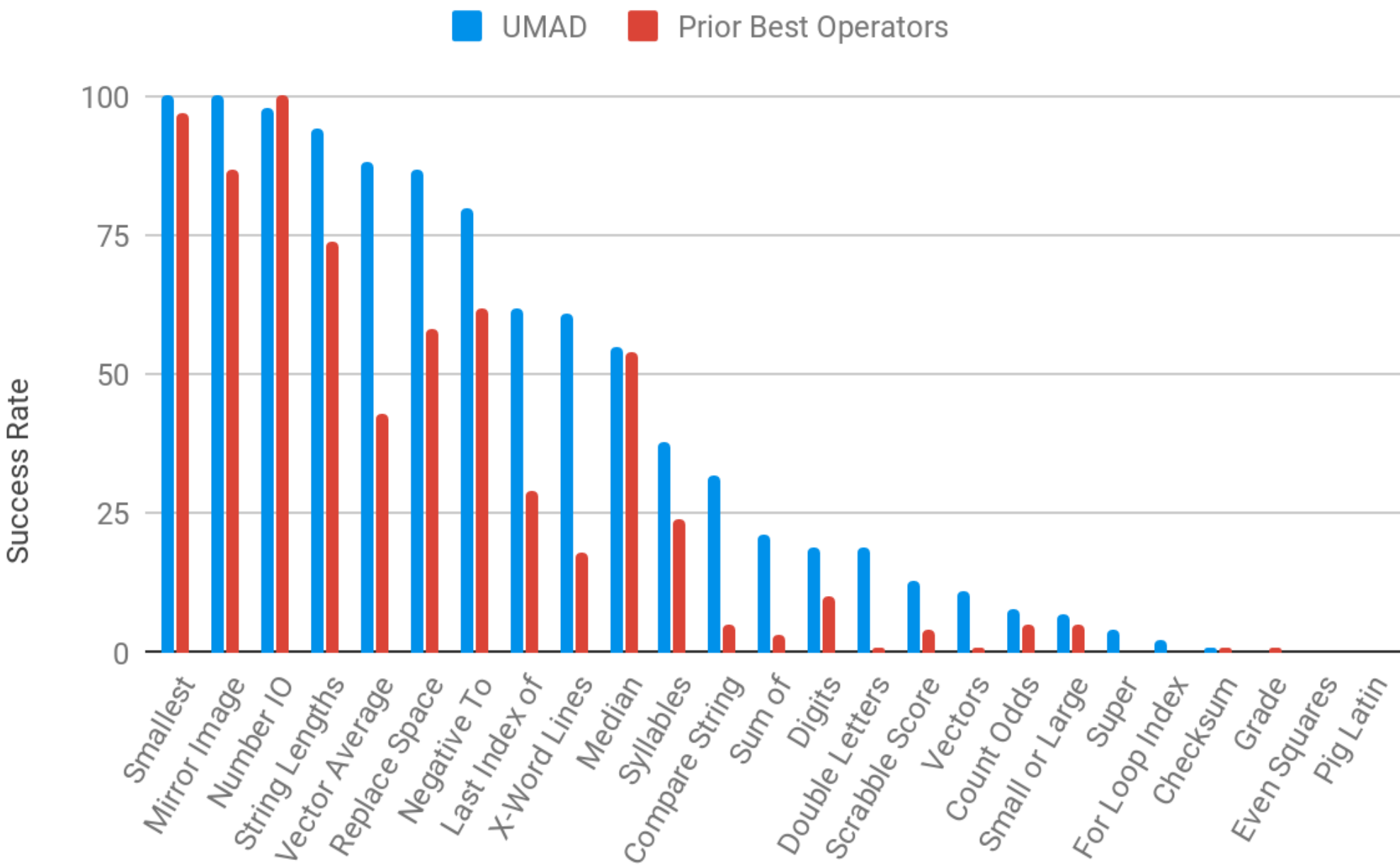
We describe a new **multiclass classification** method that learns multidimensional feature transformations using **genetic programming**. This method optimizes models by first performing a transformation of the feature space into a new space of potentially different **dimensionality**, and then performing classification using a **distance function** in the transformed space. We analyze a novel program representation for using genetic programming to represent multidimensional features and compare it to other approaches. Similarly, we analyze the use of a **distance metric** for classification in comparison to simpler techniques more commonly used when applying genetic programming to multiclass classification. Finally, we compare this method to several state-of-the-art **classification techniques** across a broad set of problems and show that this technique achieves competitive test accuracies while also producing concise models. We also quantify the **scalability** of the method on problems of varying dimensionality, sample size, and difficulty. The results suggest the proposed method scales well to large feature spaces.

Software Synthesis

- 29 benchmark problems taken from intro CS textbooks
- Require multiple data types and control structures
- Driven by software tests, input/output pairs
- Used for studies of program synthesis, by us and by others

7. **Replace Space with Newline (P 4.3)** Given a string input, print the string, replacing spaces with newlines. Also, return the integer count of the non-whitespace characters. The input string will not have tabs or newlines.
8. **String Differences (P 4.4)** Given 2 strings (without whitespace) as input, find the indices at which the strings have different characters, stopping at the end of the shorter one. For each such index, print a line containing the index as well as the character in each string. For example, if the strings are “dealer” and “dollars”, the program should print:

```
1 e o
2 a l
4 e a
```



Application	Count	Application Category
Antennas	1	Engineering (19)
Biology	2	Science (7)
Chemistry	1	Science (7)
Computer vision	2	Computer science (7)
Electrical engineering	1	Engineering (19)
Electronics	5	Engineering (19)
Games	6	Games (6)
Image processing	3	Computer science (7)
Mathematics	2	Mathematics (3)
Mechanical engineering	4	Engineering (19)
Medicine	2	Medicine (2)
Operations research	1	Engineering (19)
Optics	2	Engineering (19)
Optimization	1	Mathematics (3)
Photonics	1	Engineering (19)
Physics	1	Science (7)
Planning	1	Computer science (7)
Polymers	1	Engineering (19)
Quantum	3	Science (7)
Security	1	Computer science (7)
Software engineering	3	Engineering (19)

Problem Type Count

Classification 5

Clustering 1

Design 20

Optimization 8

Planning 1

Programming 4

Regression 3

Kannappan, K., L. Spector, M. Sipper, T. Helmuth, W. La Cava, J. Wisdom, and O. Bernstein. 2015. Analyzing a decade of Human-competitive ("HUMIE") winners -- what can we learn? In *Genetic Programming Theory and Practice XII*. New York: Springer.

Evolution, the Designer

WHAT WOULD DARWIN SAY? | LEE SPECTOR

And now, digital evolution

The Boston Globe

By Lee Spector | August 29, 2005

RECENT developments in computer science provide new perspective on "intelligent design," the view that life's complexity could only have arisen through the hand of an intelligent designer. These developments show that complex and useful designs can indeed emerge from random Darwinian processes.

“Darwinian evolution is itself a designer worthy of significant respect, if not religious devotion.”

What it is **Not** Good For

- Quite a lot
- It is solving "problems beyond the reach of other forms of AI"
- But also **not** solving problems **within** the reach of other forms of AI

Outline

- What is evolutionary computation?
- What can it do?
- * **Improving it**
 - Connections
 - COSC-452

Areas for Improvement

- Representation
- Variation
- Selection

Areas for Improvement

- Representation
- Variation
- * **Selection**

Parent Selection

- Traditionally based on overall scores
- Roulette wheels or tournaments
- Unbalanced, qualitatively diverse test sets



Biological Selection

- Survive challenges that you happen to face
- Until you can reproduce
- Each challenge may be competitive

Lexicase Selection

- Don't use overall scores
- To select single parent:
 1. Shuffle test cases
 2. First test case – keep best* individuals
 3. Repeat with next test case, etc.Until one individual remains
- Selected parent may be specialist, not great on average, but lead to generalists later

Problem name	Lexicase	Tournament
Replace Space With Newline	57	13
Syllables	24	1
String Lengths Backwards	75	18
Negative To Zero	72	15
Double Letters	5	0
Scrabble Score	0	0
Checksum	0	0
Count Odds	4	0

Diversity

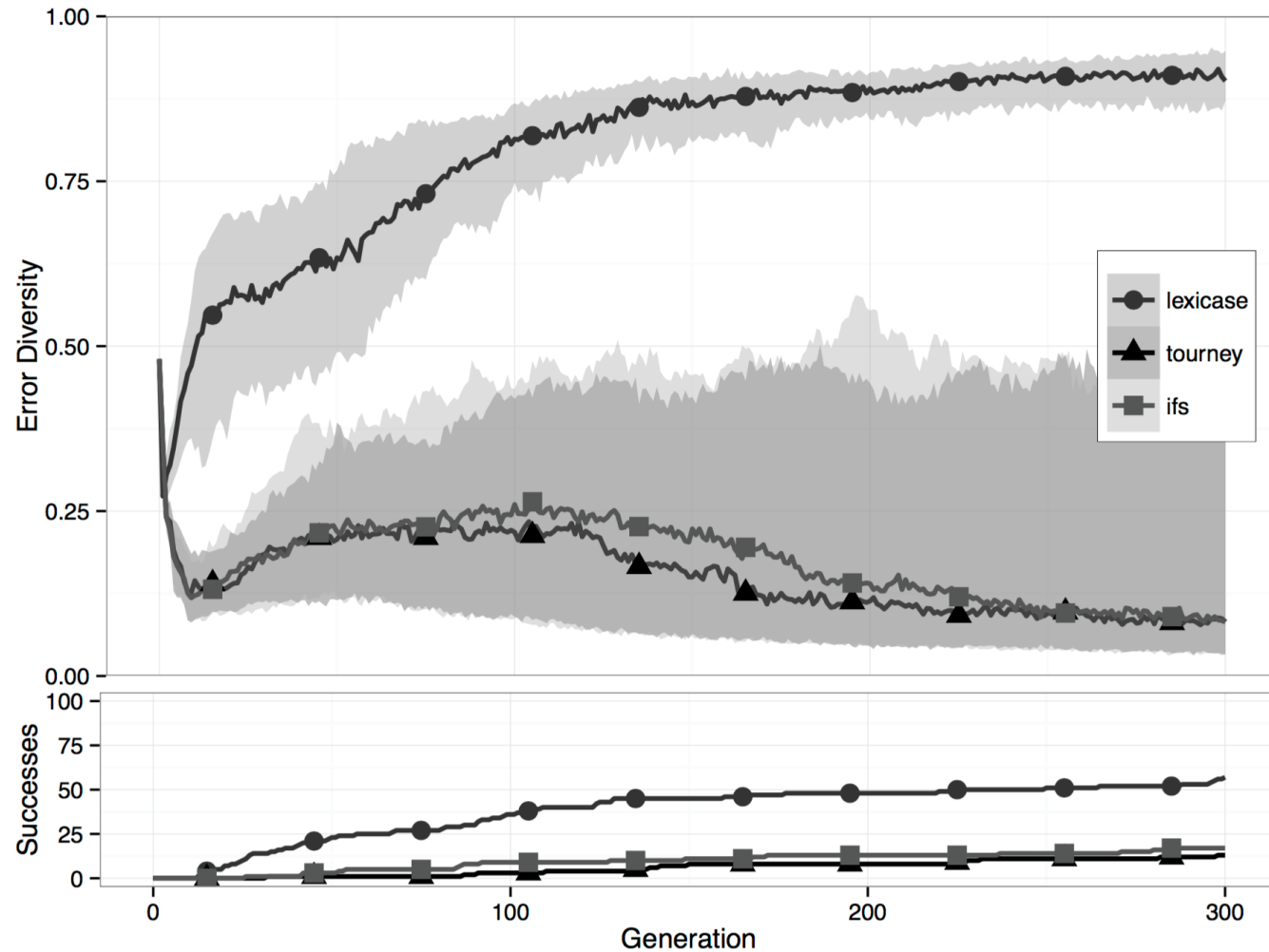


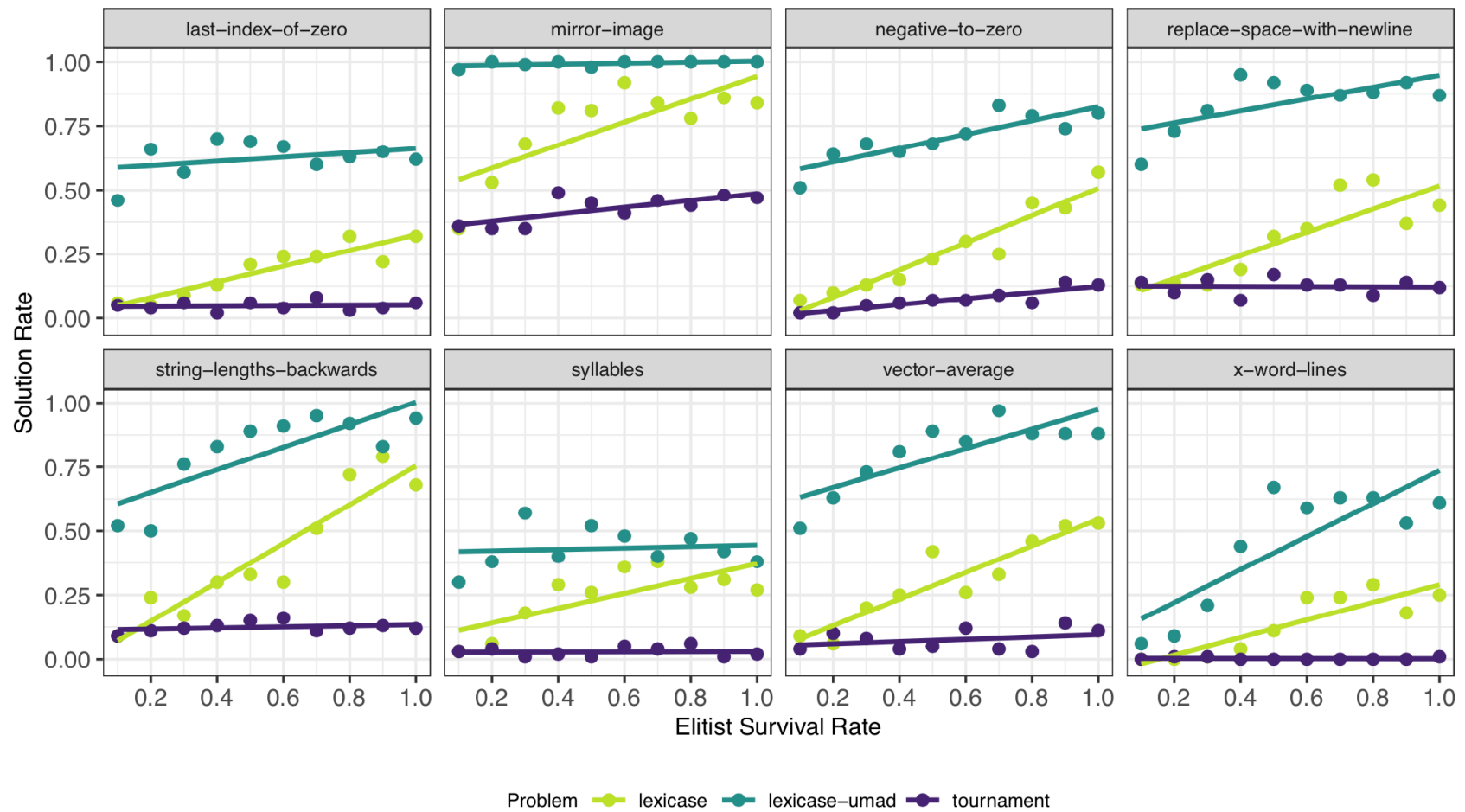
Fig. 1 Replace Space With Newline – error diversity

GPTP-2015

Why Does it Work?

- Prior results: Diversity alone is not the answer
- Hypothesis: Selecting specialists is important
- Test by only allowing programs with good overall scores (good non-specialists) to be selected
- Degraded performance would suggest that specialists are important

Specialists Help



General Lessons?

In what ways might specialists be important in:

- Other forms of machine learning?
- Biological evolution?
- Engineering teams?
- Educational communities?

Outline

- What is evolutionary computation?
- What can it do?
- Improving it

* **Connections**

- COSC-452

Connections

- Machine learning
- Software engineering
- Programming languages
- Theory
- Evolutionary biology
- Applications

Outline

- What is evolutionary computation?
- What can it do?
- Improving it
- Connections

* **COSC-452**

[Home](#) » [Academics](#) » [Majors](#) » [Computer Science](#) » [Sem: Evolutionary Comp](#)



Note: this is preliminary information about this course. Final course information will be published shortly before the start of the semester.

Sem: Evolutionary Comp

• SEM: EVOLUTIONARY COMP

SPRING 2020

Seminar in Computer Science: Evolutionary Computation

Listed in: [Computer Science](#), as COSC-452

Formerly listed as: COSC-40

Faculty

[Lee Spector](#) (Section 01)

Description

Evolutionary computation techniques harness the mechanisms of biological evolution, including mutation, recombination, and selection, to build software systems that solve difficult problems or shed light on the nature of evolutionary processes. In this course students will explore several evolutionary computation techniques and apply them to problems of their choosing. The technique of genetic programming, in which populations of executable programs evolve through natural selection, will be emphasized.

Requisite: COSC 112. Limited to 20 students. Preference given to Computer Science majors. Spring semester. Professor Spector.

COSC-452

- Projects applying and/or improving EC
- First half of course: preparing to do this
- First **weeks** of course: learning **Clojure**, a language in which progress will be faster:
 - Child of Java and Lisp
 - Functional, high level, concise
 - Full GP system in a few hundred lines
- Prerequisite: COSC-112

<https://insights.stackoverflow.com/survey/2017>



Top Paying Technologies

Top Paying Technologies by Region

Worldwide

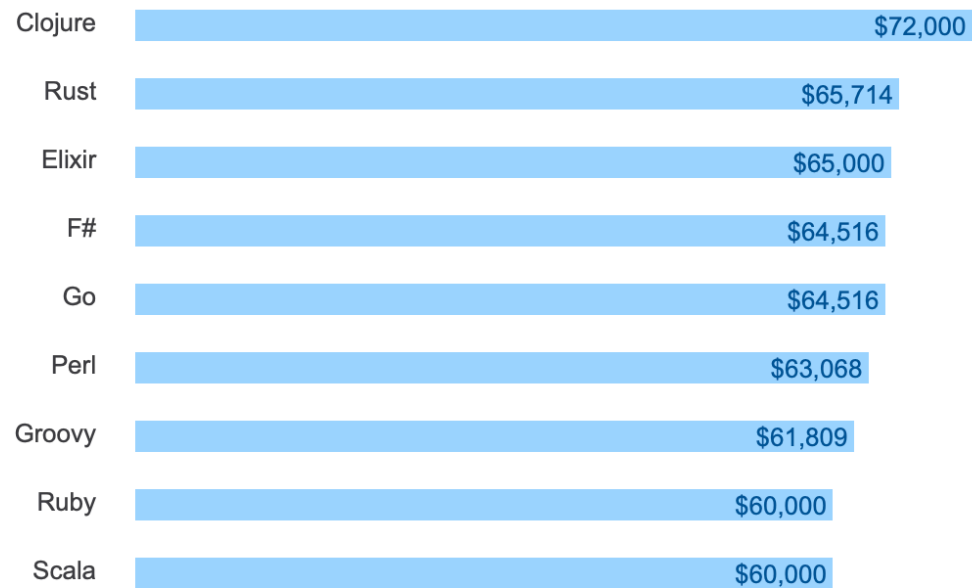
US

UK

Germany

India

France



Takeaways

- Many EvoBio/CompSci intersections
- Evolutionary algorithms use variation and selection to solve hard, interesting, and important problems
- Ample opportunities for improvement and application
- Specialists appear to be important
- CS-452 will be fun

Questions?

Reminders

- Read "[And now, digital evolution](#)," by Lee Spector
- Read "[Evolution of artificial intelligence](#)," by Lee Spector
- Read the "Getting Started" section of the [Cursive User Guide](#)
- Begin reading [Chapter 3: "Do things: a Clojure Crash Course"](#) of *Clojure for the Brave and True*