THOUGHTS ON EVOLUTIONARY CREATION

Tobi

December 18, 2024

20241123

> I wrote a Jupyter notebook that creates an initial pop of NNs and then mutates them
>
>> `https://github.com/tob-asco/gp-for-image-classification-cnn.git`
>>
>> mutations don’t have a tendency towards improvement
>>
>>> because of slow NN-teaching, I could only test on small pop sizes
>>
>> there’s a lot of parameters the EC-engineer needs to decide in mutations
>>
>>> e.g. an NN convolutional layer comes with a lot of choices (e.g. kernel sizes)
>>>
>>> in a sense this shifts the NN-hyperparameter optimization to an EC-hyperparameter optimization
>>>
>>> because the EC-space is either tightly bound or even discrete each EC-hyperparameter needs separate care
>>>
>>>> e.g. different built-in PyTorch loss functions need different target value encoding
>>>>
>>>> e.g. kernel sizes, paddings and dilations of convolutional blocks depend on each other and input size
>>>>
>>>> e.g. built-in PyTorch optimizers come with sub-parameters
>>>>
>>>>> the ”separate care” here is that one needs to understand those quite deeply
>>>>>
>>>>> the hierarchy of EC-hyperparameters makes muatation more intricate
>
> I want to start my first EC-run
>
>> Why?
>>
>>> Because I now want experience in the actual field (NNs were a dummy project)
>>
>> What?
>>
>>> Should be catchy so the topic itself creates attention
>>>
>>> Must be possible on my machine alone
>>>
>>> Probably needs to be a simulational run
>>>
>>>> So the CoDs’s values (for an ind.) are determined by simulation (of the environment)

20240905

> I’ll stick with Python all the way (GP and NNs)
>
> But I’m not fit with it, so I’ll need an IDE −¿ VS Code

20240904

> I went through the first few courses of learnpytorch.io
>
>> got a sufficient overview of PyTorch
>>
>>> it is a flexible enough non-code-intensive python library that I’ll use
>>>
>>> also I was able to move heavy computation to my GPU
>
> Now I want to create my first GP-run!

> > TODOs:
> >
> > > - find a way to represent an NN individual
> > >
> > > > the representation should be focused on easy-access hyperparameters
> > > > maybe a class where the constructor takes all hyperparameters as variable inputs
> > >
> > > - define the fitness measure
> > >
> > > > + high accuracy on test data
> > > > + small amount of network parameters (= number of weights)
> > > > + quick learning (i.e. a steep learning curve in the beginning)
> > >
> > > - define the selection
> > >
> > > > the 2019 Zhang paper uses a derivative of binary tournament selection
> > >
> > > - define crossover
> > > - mutation?
> > > - create an initial population
> > > - define weight initialization
> > > - choose which PyTensor layers individuals may use (= GP function set)
> > > - CoDs?
>
> Also, Laura recommended a nice book to simulate nature
>
> > Daniel Shiffman's "Nature of Code"
> >
> > > "Simulating Natural Systems with JavaScript"
> >
> > Possibly this could be used to create some environments for my EC-runs!
> >
> > > recall that EC-runs need complex environments to obtain a high dimensional space of CoDs

20240812

> Ok I finished a first read of Nielsen19 "Neural Nets & Deep Learning"
>
> > a well-written, quite basic, little hands-on introduction to (deep & convoluted) NNs
>
> Now I want to find the most efficient source code that offers maximal flexibility
>
> > flexibility: hyperparameters should all be choosable upon NN definition
> >
> > > net dimensions and kinds of layers
> > > activation functions
> > > cost functions
> > > parameter optimization techniques
> > >
> > > > SGD, Hessian / momentum-based
> > >
> > > learning parameters
> > >
> > > > η, epoch-count (resp. early stoppings)
> >
> > Nielsen's code is superseded because the core library "Theano" is unmaintained and integrated into "PyTensor"
> >
> > > I tried to update the code to use PyTensor instead but the functions have changed too much to understand
> >
> > I asked Silvano, he recommends learning PyTorch

20240810

> A little interlude thought on why sex is crucial in Nature's creations
>
> > otherwise, why would we find babies so cute
> > otherwise, why would sex be such a satisfying activity
> > otherwise, why would love be such a dominant feeling
> > otherwise, why would the peacock have such good-looking feathers

20240722

> I'd like to get started with programming
> What are examples of CoD-problems?
>
> > control engineering - Regelungstechnik
> > to find methods against global warming
> > building human prostheses or body add-ons
> >
> > > create novel ways to communicate with the human body
> > > "body add-ons" refers to gadgets that enhance the human functions
> > >
> > > > like glasses, but not only to mitigate shortcomings but also for cool extras
> > >
> > > this requires a realistic simulation of physics and the body
> >
> > create actual animals that serve additionally choosable purposes
> >
> > > "Any stupid boy can crush a bug but all the profs in the world cannot make one"
> > > this is like extending Nature's Evolution with human-choosable CoDs
> > > I'm hoping to apply this against Global Warming, again
> >
> > composing songs
> >
> > > possible CoDs: not catchy, no surprise, unknown rhythm, too repetitive
> >
> > find energy-storage options
> >
> > > one could reduce to "chemical energy-storage"
> >
> > find energy-harvest options
> >
> > > i.e. find sources of energy that can be converted to electrical energy
> > > i.e. solutions to convert some free and existing energy into electrical energy
> > > I mean on the Earth, so we would have to simulate Earth for that...
> >
> > build an inverse microwave
> >
> > > a gadget that cools stuff within seconds
> >
> > find vaccines, medication, therapy methods
> >
> > > here CoDs would partly correspond to causes of human death
> >
> > define new techniques of human/goods transportation
> >
> > > like cars, plains, hyperloop, public transport, ...
>
> Ok, now examples that I could actually start with
>
> > minesweeper agent
> > navigate through a labyrinth
>
> Maybe first start with Neural Nets

> > because they are probably quicker to understand than all those simulators
> >
> > > because the given examples need a lot of simulation
> > >
> > > > because CoD-Evolution needs to simulate its ”environment”

20240721

> Some philosophical aspects AGAINST fitness measures as the driving force of Evolution
>
> > people love to quantify stuff/others, i.e. define a total order
> >
> > > either through money (jobs), or through citations (science), vote-count (politics), GPA (students), ...
> > > and this attitude has been a pain of mine for a long time - I dislike it almost categorically
> > > a fitness function is just another instance, so I’m naturally opposed
> >
> > fitness functions need uncountably much information which creates room for errors
> >
> > > to provide a continuous function, you would have to specify too much to consider each case separately
> > > so ceratin high-level paradigms are used to create this vast ”landscape of fitness”
> > > however, usually, edge cases with low-effort-high-fitness develop, creating ”gaps”
> > >
> > > > examples will be quite political, e.g. tax loopholes, planned obsolescence, child-labour
> >
> > on the other hand, a CoD/heaviside is defined by one point
> >
> > > we can see it as a fitness function, but highly discontinuous and almost everywhere constant
> > > in a sense, it is a collection of multiple binary fitness measures
> > >
> > > > recall from the ”Intro” book that a binary fitness measure is fatal
> > > >
> > > > > again, multiple CoDs are not binary, only one CoD is binary
> >
> > as in 20240717: my Evolution reads ”survival of the immune” instead of Darwin’s SOTF
> >
> > > don’t try to maximise 1 perfect measure
> > >
> > > > even if it were a (weighted) some of all susceptibilities to the CoDs
> > > >
> > > > > because then even the global maximum might die from some irrelevantly-weighted CoD
> > >
> > > instead, try to balance yourself between the various CoDs
> > >
> > > > ”well-rounded”
> > >
> > > rephrased, my Evolution has a non-scalar, non-continuous measure
> > >
> > > > and it is not high fitness, it is aiming for, but low non-fitness
> > > >
> > > > > which is not equivalent in non-scalar functions
> > > > >
> > > > > > as you would need to choose a norm thingy to translate
> > >
> > > it is similar to high-school GPA and university admission
> > >
> > > > you might only care about the grades in the subjects that you want to keep studying
> > > > but if you fail a single class, you’ll simply not be admitted to ANY university for ANY subject

20240719

> Why CoD-prescribed problems are efficiently solved by Stacked Random Search (SRS)

> > SRS is what I praised in the notes 20240705
> >
> > > where N random candidates are separated into G ”generations”/stacks
> > > first only N/G random candidates are produced - the first gen(eration)
> > > then new N/G candidates are created from the former generation
> > >
> > > > here, we may apply selection - a feedback loop between each generation
> > > > this is the crucial difference between Random Search and Evolutionary Search
> >
> > the question now: what kind of problems are best suited for this kind of solution creation?
> >
> > > the problem should have multiple independent subproblems
> > >
> > > > because of the separation/fragmentation into generations
> > > > I call this the fragmentation of the problem into subproblems
> > > > and I view each CoD as 1 fragment of the problem defined through the bundle of all CoDs
> > >
> > > solutions might then also split up into substructures - each concerned only with subproblems
> > > hence, such substructures can be detected in a generation and carried over to the next gen
> >
> > is Natural Evolution such a problem?
> >
> > > I think so, yes
> > > e.g., solutions have to avoid various CoDs
> > >
> > > > starving, being killed by others, dying from illnesses, suicide, dying from overpopulation, ...
> >
> > do we see such similar substructures across Nature’s species?
> >
> > > yes, again
> > > e.g. almost all species of plants are green because they use photosynthesis against the ”starving” CoD

20240718

> my midnight tai chi walk sparked some thoughts:

> > EVOLUTIONARY CREATION (EC) is the name of my goal and it is different from GP
> >
> > > EC runs should only affect the environment, not HOW evolution evolves its candidates
> > >
> > > > namely through the definition of CoDs
> > > > not through genetic operators (GO)
> > > >
> > > > > ideally, the GOs should be defined canonically
> > > > >
> > > > > > so, no subjectivity involved
> > > > > > "crossover" shall evolve from self-cloning
> > > > > > mutation might be constructable from first principles
> > > > > >
> > > > > > > like the 2nd law of thermodynamics
> > >
> > > EC doesn't search for a global extremum, rather it searches for immunity
> > >
> > > > you can view CoDs as a negative fitness measure, but it is highly discontinuous
> > > >
> > > > > because each CoD has a heaviside function associated to it
> > > > > either you pass a CoD (survival = good fitness) or not (death = worst fitness)
> > >
> > > EC tries to create a trend towards creativity
> > >
> > > > creativity is encouraged if 2 requirements of development are met:
> > > >
> > > > > R1. the rules have to be crystal clear
> > > > >
> > > > > > I feel like CoDs are more clear than "maximise this smooth function"
> > > > >
> > > > > R2. the solution space should be left as unconstrained as possible, in the best case not even definable
> > > > >
> > > > > > this is why we do not want to pre-impose any concrete GOs
> > > > > > this is why a representation of variable-length bit-strings is my preferred choice
> >
> > if, in a GP run, GOs are highly specific, it is more an optimization than creation

20240717

> Where is evolution taking us?
> towards better?
>
> > is the human design very good?
> >
> > > I would love to have wings, e.g.
> > > also, I would prefer not to have been born with an appendix
> > > breathing under water would also be cool
> > > how about running as fast as a jaguar
> > > or having some sort of wheel construction instead of two feet
> > > what about seeing UV to know better when to protect from sunlight
> > > how about having some sensors to feel infections without measuring
> > > I would love to be a little less lazy
> > > why do I get addicted to stuff that doesn't do me good in the first place?
> > > why aren't people always happy?
> > > why aren't people more peaceful?
> > >
> > > > actually, there might be a trend towards this
> > > > I mean, compare to (movies of) the Middle Age
> >
> > are the primitive bacteria alive today as good as 4B years evolution would give if there were a trend towards better?

> towards more or less co-dependent?
>
> > co-dependence is referring to a species' dependence on another species
> > I feel like complex species almost always depend on others
> >
> > > e.g. lions depends on antelopes
> > > e.g. don't some simple bacteria have zero co-dependence?
> > > however, e.g. viruses cannot survive without host (I think)
> > >
> > > > so the inverse doesn't really hold
> > >
> > > as soon as energy from the sunlight is not sufficient, a co-dependence evolves, I think
> > > what about trees, they are pretty independent
> > >
> > > > yet, they are pretty complex, no?
> >
> > the very first species ever, by definition, had no co-dependence
> >
> > > so, indeed, relative to this example, co-dependence cannot decrease
>
> towards more complex?
>
> > this actually sounds the most promising
> > pro: in GP there's a lot of bloat evolving in later generations
> > pro: everyone would agree that the human design is extremely complex
> > pro: entropy increases with higher complexity
> >
> > > because the amount of candidates (= microstates) increases with higher complexity (= macrovariable)
> >
> > pro: all non-complex alive species may just have had a later starting point
>
> Death as the central ingredient in GP:

> > because it is the central ingredient in Nature's evolution, in my opinion
> >
> > First, define a set of causes of death (CoDs)
> >
> > > a CoD is one test that a given individual may either pass (survive) or fail (die)
> > >
> > > CoDs may be completely different from one another
> > >
> > > > this is the entry point for the programmer to guide his GP run
> > > >
> > > > the programmer may create a CoD for each property that shall NOT be found in the "fittest" individuals
> > > >
> > > > > AHAAA: a shift of perspective from "who is the fittest" to "who is not fit enough"
> > > > >
> > > > > > this essentially remedies my complaint 6. from 20240714
> > >
> > > CoDs should come with a number, the CoD's risk
> > >
> > > > the risk should directly correspond to the frequency with which the risk's CoD is used as a test on candidates
> >
> > Second, test the candidates on CoDs over time
> >
> > > within a generation, the GP run randomly - weighed by the risks - tests candidates on some CoDs
> > >
> > > this will produce a diverse new gen that is immune to a diverse selection of CoDs
> > >
> > > the goal is to find a generation that is immune to (almost) all CoDs
> > >
> > > > is this the ultimate goal of Evolution: immunity?
> >
> > a CoD can be used to integrate two different categories of tendencies into a GP run
> >
> > > 1. environment
> > >
> > > > the difficulties posed by the environment can be phrased as CoDs
> > > >
> > > > in fact, that is the more obvious category
> > >
> > > 2. goal
> > >
> > > > what you want to achieve can be phrased through a CoD
> > > >
> > > > e.g. you may define that blockage of low frequencies in an analog circuit is a CoD
> > > >
> > > > e.g. you may define that letting high frequencies pass is also a CoD
>
> Self-replication as the source of all genetic operators
>
> > in a perfect GP run, the programmer may not choose genetic operators, yet all known genetic operators may evolve, and more!
> >
> > > I want to mimic Nature's evolution here, but I don't know how replication actually started!
> > >
> > > > I want to say cloning, i.e. haploidy
> > > >
> > > > > it seems so easy, almost canonical
> > > >
> > > > but I don't really know
> > > >
> > > > I'll go with it, because I don't have the necessary background/idea to replace it with
> > >
> > > the hope is that cloning shall be superseded as a replication mechanism in favour of a better, self-developed mechanism
> > >
> > > > as in Nature - because cloning couldn't repair fatally mutated DNA as well as diploid reproduction
> > > >
> > > > > e.g. with humans, the father-mother-reproduction is extremely safe
> > > > >
> > > > > > as opposed to incest (which is as close to cloning as I have examples of)
> > > > >
> > > > > cf. the nice chapter on homologous DNA in the non-Koza GP introduction book

20240714

> I had some time near Harlaching in order to meditate over Darwin’s Evolution and ”survival of the fittest”
> 1. ”fittest” sounds like there’s exactly ONE survivor/species and all the others are dead
>
> > plain wrong, of course
> > right now there is a myriad of living (hence ”fittest”) species on Earth
> > there is also insane cross-dependence between them
> > so try to stop thinking of a global maximum in evolution
>
> 2. Darwin’s SOTF doesn’t capture the essence about how humans in the last 100 years have evolved
>
> > here, the evolution of mind, intelligence, society, etc. is independent of survival
> > I want a similar driving force for humans, their creativity and esprit
>
> 3. there is not a single living entity that has survived more than 1
>
> > so SOTF cannot be about individuals, right?
> > so it should be about something like a species
> >
> > > remember, a species is the set of organisms that mates with each other
>
> 4. there is no reference to breeding
>
> > would you say that a mosquito is ”the fittest”?
> >
> > > I mean, we’re killing those beasts almost routinely
> >
> > no, but this is an example of a species that excels at their breeding rate
>
> 5. I feel like Darwin’s Evolution says that there is a long-time tendency towards BETTER
>
> > I think this would only be true if the environment would have a long-time tendency towards harder
>
> 6. there’s too much about fitness when it should be more about DEATH
>
> > because it’s less about being super or the best, but more about avoiding death
> > ”survival of the death-avoiding” $\sim$ ”survival of the surviving”

20240711

> crossover and replication are examples of genetic operators that have evolved
>
> > e.g. cloning
> >
> > > bacteria that couldn’t clone itself just died
> > > bacteria that could would be dominant in the next generation
> >
> > e.g. male-female crossover
> >
> > > higher organisms that would just clone itself were too susceptible to DNA mutation
> > > higher organisms that used male-female crossover mitigated this susceptibility and survived
> >
> > ChatGPT gives a lot of crossover examples that are different to human crossover
> >
> > > Parthenogenesis, Polyembryony, Hermaphroditism, Assisted Reproductive Strategies (e.g. bee hives), Brood Parasitism
>
> so why should this be implemented by the genetic programmer, huh?
> I need to find a general, canonical framework that may create such genetic operators through program evolution

> > maybe programs should ”die” after some time
> >
> > > DEATH - it is central in my opinion
> > > maybe there’s exactly two ingredients: DEATH + MUTATION
> > >
> > > > so, not so much about absolute and relative fitness?
> > > > more about whether or not a certain fitness level is not passed - DEATH
> > > > this goes towards the thought that evolution doesn’t create better, but more complex/robust
> >
> > just like my reasoning from the ”cloning” example above, this may create cloning
> > cloning, in turn, may evolve into some sort of crossover
> > ah, the beauty - it’s time to start, damn’ it..
> >
> > > I’ll wait and read a little more, this seems to be sparking lots of ideas
>
> ”Mimicking evolution” is not well-specified because it can be done on many scales
>
> > some examples that go from small scale to high scale:
> >
> > > small: clone parts of a living creation
> > > larger: understand functionality of parts of living creations and replicate it elsewhere
> > >
> > > > e.g. the bug’s dirt repulsion system (from iENA) used in modern gadgets
> > > > e.g. neural nets as inspired from brain’s neurons
> > >
> > > larger: mimic the way that humans brain evolved from apes to today
> > >
> > > > by copying homologous DNA recombination techniques in crossover
> > >
> > > larger: mimic the way that humans evolved from bacteria to today
> > >
> > > > by copying mutation and fitness evaluation
> > >
> > > larger: mimic the way that entropy and death creates stuff
> > >
> > > > death here involves a model of an environment together with natural laws
> > > > at this scale we have no fitness measure, just mutation (=entropy) and death
> > >
> > > (largest?: mimic the way that entropy creates stuff)
> > >
> > > > maybe death can be seen as entropy itself
> >
> > I feel a resemblance with the energy scale in fundamental physics
> >
> > > the higher the energy the more fundamental the set of laws
> > > the higher the energy the more canonical and pure
> > > the higher the energy the more abstract and further away from experiment

20240710

> been reading 1998’s ”Genetic Programming An Introduction” by Banzhaf, Nordin, Keller, Francone
> I’ve put a handwritten list of interesting paragraphs inside the book
> the book is ok, but it is too theoretical
>
> > this field is not about formality, it is about trial and error!
>
> there’s too little about creation

> > they say the boundaries of optimization and creation are ”fuzzy”
> >
> > > maybe, but still the distinction is central in my opinion
> >
> > a perfect GP setup should be as restrictive as a sheet of paper together with a pencil
> >
> > > you should be able to write down Maxwell’s equations
> > > you should be able to write down lyrics for your new song
> > > you should be able to sketch a scene from your last dream on it
>
> however they have taught me important aspects of biology
>
> > e.g. a species is defined to be those individuals that mate with each other
> >
> > > because two DNAs that are non-homologous cannot be recombined
> > >
> > > > homologous DNAs means that their coarse structures agrees
> > > > homology is needed to put upper bounds on the impact of recombination
> >
> > e.g. that sex’s prime feature is to mitigate the impact of heavy mutation rather than being a source of diversity
> >
> > > because heavy mutation in the father’s DNA cannot be passed on to the kid if homology is demanded
>
> isn’t creative/artistic work partly also like an internal ”evolution of thought”?
>
> > e.g. when writing lyrics:
> >
> > > 1. the composing artist has a topic in mind
> > >
> > > > this topic greatly defines the fitness measure for the upcoming verses
> > >
> > > 2. he begins by writing something
> > > 3. he sings the verse in his head and evaluates how well this fits his intentions
> > > 4. if ok, he goes to the next verse; if not sufficient, he might change parts of the verse
> > >
> > > > this sounds like selection in the first case; mutation in the second case
> >
> > e.g. when sketching a picture from memory
> >
> > > 1. the artist draws a new part
> > > 2. she looks at the newly emerged picture as a whole
> > > 3. she evaluates the new picture against her memory of what it was before the new part
> > > 4. she selects: either deletes the new part, or keeps going with it

20240706

> started reading Holland’s ”Adaption in Natural And artificial Systems”

> > the new preface includes ”I still find the 1975 preface surprisingly relevant. About the only change I would make would be to put more emphasis on improvement and less on optimization”
> >
> > > I interpret as him slightly agreeing with me that evolution is not an optimization problem
> > >
> > > > e.g. I think it’s flawed to consider any of the currently living species a global maximum and the others not
> > > >
> > > > > there is cross-dependence for example and this is just not reflected in a single number as fitness measure
> > > > > still, Koza’s examples have totally convinced me already that a single fitness measure can greatly drive algo creation
> > > > >
> > > > > > the point being that actual evolution in Nature is not as simple
> >
> > ok this book seems to be super dull, sorry!
> >
> > > there’s way to much formality, the field is not about formality, just like Koza says too

20240705

> started reading Koza’s GP1
> he says ”Nature creates structure over time by applying natural selection”
> and he says that nature uses none of the following principles that humans tend to do science with:
>
> > correctness, consistency, justifiability, certainty, orderliness, parsimony, decisiveness
>
> so maybe Nature itself has been constructed also avoiding these principles
> so maybe Nature itself has been ”evolved”
>
> > the fitness measure may be whether or not the candidate universe lives
> >
> > > e.g. has a cosmological constant such that the candidate universe’s expansion is stable
> >
> > the different candidates may be encoded through different sets of laws
> >
> > > so the ”evolution of universes” in effect evolves its laws to be ever more stable, i.e. ”living”
> >
> > this of course implies that multiple universes are ”simultaneously” ”present”
> > furthermore it implies that these candidate universes have to have ”sex”
> >
> > > wuahh, weird.. let’s say ”candidate universe crossover”
>
> Randomness vs. Evolution
>
> > Randomness is like Evolution with only 1 Gen
> >
> > > Allowed size and complexity of the candidates dictates how big a (Gaussian) random distribution has to be to find a suitable candidate.
> >
> > Evolution is like ”stacked”-Randomness, where each layer is 1 Gen
> >
> > > Use the same size of population and divide it by after how many generations you want to terminate.
> > > Again - Gaussian or otherwise - randomly produce the first Gen, pick the best ones and build a new Gen ”based on the best”.
> > > Repeat for all Gens.
> >
> > Evolution sounds sooo much more efficient/better/clever
> >
> > > Note that the two examples have the same amount of candidates to be generated
> > > and hence they share the amount of fitness evaluations as each candidate needs only be evaluated once (as it’s fitness is constant in time)

20240704

> other possible GP applications
>
>> battery design
>>
>>> mainly a choice of material
>>
>> superconductor design
>>
>>> mainly a choice of material

20240703

> today I'm watching Koza's video tapes that accompany his books: GP1
>
>> I watched it yesterday
>> the tree representation of a program (nodes = functions; leaves = input) is seminal for crossover
>>
>>> you simply choose random nodes from the parents and exchange them to produce two children
>>
>> a great amount of examples, some of them are really amazing!
>>
>>> I loved the inverse kinematic robot
>>
>> my criticism:
>>
>>> you have to provide the functions, the evolution cannot create new atomic functions, i.e. building blocks
>
> GP2
>
>> Automatically Defined Functions (ADFs) are the main topic
>>
>>> they provide the framework for the algorithms to create their own subroutines
>>> the human has to specify the precise framework's structure
>>>
>>>> e.g. the amount of ADFs
>>>> e.g. the amount of input parameters per ADFs
>>>> e.g. the hierarchy between multiple ADFs
>>>>
>>>>> i.e. who is the outer function allowed to call the inner ones
>>>>
>>>> but in the very end he talks about evolving also this structure through evolution
>>>>
>>>>> not sure what the take-away was here, though - was it better or worse?
>>>
>>> FRACTALITY i.e. SCALABILITY !! (I think)
>
> GP3

> > he lists 16 (!) demands on the evolution and the evolved final algo
> >
> > > 1. Start only with "What needs to be done" (i.e. a high-lvl instruction)
> > > 2. Tells us how to do it (huh?)
> > > 3. Produces a computer program
> > > 4. Automatically determines program size (huh?)
> > > 5. Code reuse
> > > 6. Parameterized reuse (why not in 5.?)
> > > 7. Internal storage
> > > 8. Iterations, loops, recursions
> > > 9. Self-organization of hierarchies
> > > 10. Automatically determines architecture (includes at least 9., right?)
> > > 11. Wide range of constructs (huh?)
> > > 12. Well-defined
> > > 13. Problem-independent (big huh?)
> > >
> > > > this might mean that e.g. both low-pass and band-pass filters can be evolved
> > > > i.e. using the same evolution, population size, generation count
> > >
> > > ——————- up to here he feels like he checked them ——————
> > > 14. Wide applicability (vague..)
> > > 15. Scalable (huh?)
> > > 16. Human-competitive ("most important")
> >
> > he quotes Turing's seminal 1950 paper
> >
> > > TODO: read it
> >
> > he explains what mutation and crossover are and how frequently they're employed
> >
> > > mutation is a random growth of a deleted node; improbable (1crossover is copying an existing node; very probable (85
> >
> > architecture-altering operations add or delete:
> >
> > > ADFs, their args, iterations, loops, recursions, memory
> >
> > automatic synthesis of analog electrical circuits
> >
> > > I LOVE THIS IDEA
> > > there's an amazing animation on how trees dictate circuits in the video
> > > lots of examples of true AI
> > >
> > > > it's amazing !!!
> >
> > "GP delivers a very high ratio of A-to-I" (artificial-to-input-intelligence)
>
> GP4

> > it's about how broad GP solutions can be (cf. point 13. in GP3)
> >
> > > 1. it's easy and quick to change an evolution from one problem to another
> > >
> > > > if the problems are in the same field, usually only change the fitness measure
> > > > if the problems are from non-overlapping fields, additionally change nodes and leaves
> > >
> > > 2. one can evolve algos with extra free variables to make it a multi-problem solution
> > >
> > > > here, the fitness measure includes an integration over the free variables
> > > > using the evolved algo w/ different free variable inputs yields solutions to different problems
> > > > e.g. "general purpose controller" which they actually tried to patent
> >
> > there's lots of overlap between the other movies...
> > "it is a system that works, based on a system that works"
> > GP is the unique strategy among AI and machine learning methods - at 2003 - that:
> >
> > > has duplicated numerous already patented results
> > > has produced patentable new results
> > > has produced "parameterized topology" (point 2. above)
>
> using Linux CLPs, one might be able to evolve pipe commands that solve something
> possible hot applications that jump to my head:

> > aesthetics
> >
> > > maybe there is something deterministic going on in our aesthetic brain parts
> > >
> > > > like a neural net, hehe
> > > >
> > > > > then bring back the GP-NN (genetic programming - neural net) loop !!
> > > > >
> > > > > > i.e. train a NN to be able to judge aesthetics (i.e. make it the fitness measure)
> > > > > > and thus evolve some masterpieces by GP
> >
> > NNs (neural nets) themselves
> >
> > > you know how complicated ChatGPT's language model is? (something about 96-fold loop)
> > > in general, NNs have a predefined structure
> > >
> > > > amount of layers, amount of neurons, amount of inputs, amount of outputs, etc.
> > >
> > > let GP create this structure for you
> > > i.e. have a training set ready for each candidate-NN, train it and measure its fitness by how well it predicts
> >
> > GP itself - meta!!!
> >
> > > there's some (albeit little) structure predefined for a GP-evolution
> > >
> > > > e.g. nodes and leaves (of the program tree)
> > > > e.g. the whole idea behind it (but this might get out-of-hand)
> > >
> > > use GP to predefine this structure
> > >
> > > > but this GP needs also nodes and leaves... which one?
> > > > well, a candidate can choose to ignore so the more the better (maybe; hopefully)
> >
> > math and physics
> >
> > > of course, I don't yet know what the nodes and leaves should be !!
> > > but the two are areas with some characteristics that are shared across the field of GP-applications
> > >
> > > > e.g. we seek a solution to a problem
> > > >
> > > > > this sounds pretty tautological, but I don't think e.g. writing poetry can be phrased this way
> > > >
> > > > e.g. we might be able to tell automatically if a candidate is a (perfect) solution
> > >
> > > it might be impossible to have a non-discrete/non-binary fitness measure, idk
> > >
> > > > not even sure if that breaks the GP-apparatus, though
> > >
> > > so there's a big ? on these examples
> > >
> > > > but it would be freaking AMAZING

20240702

> there is criticism on metaphor methodology where new optimization methods try to be sold b.m.o. good metaphors and not high efficiency
>
> > `https://en.wikipedia.org/wiki/List_of_metaphor-based_metaheuristics#Criticism`
>
> Coevolution provides Free Lunch algorithms
>
> > Coevolution is, I think, when two species mutually depend
> > https://de.wikipedia.org/wiki/Koevolution
>
> Phenotype vs. Genotype = Problem space vs. Solution space `https://en.wikipedia.org/wiki/Genetic_`

`representation`

Genotype = the set of all genetic code of a candidate
Phenotype = the set of all observable characteristics of a candidate

e.g. physical form and structure, its developmental processes, its biochemical and physiological properties, its behavior, and the products of behavior
it results from two basic factors

the expression of an organism's genetic code (its genotype) and the influence of environmental factors

problem space contains concrete solutions
search space contains the encoded solutions
mapping from search space to problem space is called genotype-phenotype mapping

genetic operators apply to the search space
for evaluation, elements of the search space are mapped to elements of the problem space via genotype-phenotype mapping

4 main paradigms in EAs: Genetic Algorithm (GA), Evolution Strategies (ES), Evolutionary Programming (EP), Genetic Programming (GP) GA

focuses on recombination, i.e. crossover, i.e. sex
2 (or more) algorithm-parents are chosen and a new one is created from them

ES `https://en.wikipedia.org/wiki/Evolution_strategy`

problem space = solution space
dynamic mutation

it is fixed that each mutation follows a normal distribution
the σ of these normal distributions is what is "dynamic"

it is calculated from the 2 parent's recombined σs that is mutated (i.e. changed) via an exponentiated normally distributed random variable

the new σ is then the std. deviation that mutates the actual decision variables of the candidate

this means that a normally distributed (w/ the new σ) random variable is added to the old decision variable to give the new decision variable

upside

we search with increasing precision as the σ may get narrower and narrower

–¿ sounds like fractality

downsides

danger of getting stuck in larger invalid areas of the search space

–¿ fractality may help

EP

there's also recombination, but the primary operator is mutation
what is really special is that solutions/candidates are Finite State Machines (FSM)

they are like programs that have multiple (but finitely many) states between which the transition

how to transition (for fixed input) is called the FSM's "behaviour"

this behaviour is what is optimized through mutation (and recombination) in EP

> GP https://en.wikipedia.org/wiki/Genetic_programming
>
>> there's also mutation, but the primary operator is crossover
>> it is concerned only with decision trees i.e. actual software programs
>>
>>> trees: nodes are functions/operations; leaves are variables/constants
>>
>> it evolves programs to find the best output to a given input
>> it recombines subtrees of the parents during crossover
>> one goal is "algorithm discovery"!
>>
>>> this sounds pretty much like what I had in mind, but let's keep digging..
>>
>> One big guy in this field was John Koza [http://www.genetic-programming.com/johnkoza.html]
>>
>>> read https://www.popsci.com/scitech/article/2006-04/john-koza-has-built-invention-machine/,
>>> I also saved it as pdf by Keats
>>> his books' videos are at https://human-competitive.org/
>>
>> GECCO: conference on genetic and evolutionary computation https://gecco-2024.sigevo.org/HomePage
>>
>>> yearly, starts in 12 days, haha...
>>
>> A still-active big guy is Mengjie Zhang at Wellington https://people.wgtn.ac.nz/Mengjie.Zhang
>>
>>> he is "available" for PhD supervisions
>>
>> The "Humies" are an competition for the best GP-created results that are "better or equal" to human results
>>
>>> cf. https://human-competitive.org/awards
>>> there's 8 possible ways how a result can classify as "human-competitive"

20240701

> Ok this is a broad field: Evolutionary Algorithm (EA)
>
>> this is even a Wiki SERIES, cf. https://en.wikipedia.org/wiki/Category:Evolutionary_algorithms
>>
>>> "[EA] is a heuristic optimization algorithm using techniques inspired by mechanisms from organic evolution such as mutation, recombination, and natural selection to find an optimal configuration for a specific system within specific constraints."
>>
>> but I think this is only about optimization
>>
>>> I'm still hoping that Evolution is not actually an optimization problem
>>>
>>>> e.g. no absolute fitness function in Evolution
>>>> e.g. time-dependence in the fitness metric in Evolution
>>>
>>> creation, I think, is qualitatively different from optimization
>>
>> Natural Evolution Strategies (NES) seem to treat the fitness function as a black box
>>
>>> read "High Dimensions and Heavy Tails for Natural Evolution Strategies"
>>>
>>>> the heavy tails ought to incorporate also Black Swans, i.e. non-Gaussians
>>>>
>>>>> although I don't know which distribution was the one that was Gaussian in the first place
>>
>> No fractality at first sight
>>
>>> I want to treat different scales on equal footing
>>>
>>>> a leave solution + a branch solution + a stem solution = tree solution

20240627

> Bremermann I think wanted to do what I want to do:
>
> > creation of solutions/optimization to given problems/cost-functions by mimicking evolution
>
> What I don't want to follow is the strong demand to mimic the biology of evolution
>
> > it's more the guiding principle of evolution that I want to see
> > abstractly and concisely
> >
> > > e.g. "survival of the fittest"
> > >
> > > > but that's probably not it
> > > > sex is missing in it
> >
> > maybe I don't even want to mimic evolution but Nature
> >
> > > fractal-like self-similarity across scales
> > >
> > > > read "the Black Swan" by Taleb
> > > > look at the Wiki page on the "Mandelbrot set"
> >
> > or maybe I want a combination of the two:
> >
> > > evolutionary randomness
> > > + fractally searching across scales
> > >
> > > > for the randomness, i.e. the possible algorithms/creations
>
> Recursion and fractality are similar
>
> > read `https://www.spiceworks.com/tech/devops/articles/what-is-dynamic-programming/`