به نام پروردگار دانایی

هوش مصنوعی

الگوریتم‌های تکاملی

سید کاوه احمدی

Introduction

متافور اصلی پردازش تکاملی

حل مسئله		تکامل
مسئله	⟷	محیط
راه‌حل کاندید	⟷	نمونه
کیفیت	⟷	برآزندگی

برازندگی (Fitness) ← شانس برای بقاء و تکثیر

کیفیت (Quality) ← شانس برای ایجاد راه حل های جدید

تاریخچه

- 1948, Turing:
  - proposes “genetical or evolutionary search”
- 1962, Bremermann
  - optimization through evolution and recombination
- 1964, Rechenberg
  - introduces evolution strategies
- 1965, L. Fogel, Owens and Walsh
  - introduce evolutionary programming
- 1975, Holland
  - introduces genetic algorithms
- 1992, Koza
  - introduces genetic programming

تاریخچه

- 1985: first international conference (ICGA)
- 1990: first international conference in Europe (PPSN)
- 1993: first scientific EC journal (MIT Press)
- 1997: launch of European EC Research Network EvoNet

EC در قرن ۲۱

- سه کنفرانس اصلی شامل: CEC، GECCO، PPSN
- مجلات علمی EC
  - IEEE Transactions on Evolutionary Computation
    - Impact Factor: 4.81
  - Evolutionary Computing, MIT Press
    - Impact Factor: 2.109
  - Genetic Programming and Evolvable Machines, Springer
    - Impact Factor: 1.333

تکامل داروینی: بقای شایسته‌ترین (Survival of the fittest)

- همه‌ی محیط‌ها دارای منابع متناهی هستند.
  - یعنی می‌توانند تعداد محدودی از نمونه‌ها (individuals) را حمایت کند.
- نمونه‌ها در محیط دارای غریزه برای تکثیر / تولید مثل (reproduction) در چرخه حیات هستند.
- بنابراین چون افزایش جمعیت به شکل نمایی میسر نیست (به دلیل کمبود منابع) نوعی انتخاب اجتناب ناپذیر است.
- نمونه‌هایی که برای بدست آوردن منابع به شکل موثرتری عمل کرده‌اند، شانس بیشتری برای تکثیر دارند.
  - به عبارت دیگر آنهایی که بهتر از دیگران با محیط سازگار شده‌اند (قانون بقای شایسته‌ترین‌ها)
  - توجه شود که در تکامل طبیعی، برآزندگی یک معیار ثانویه است و انسان به افرادی که دارای نسل‌های (offspring) زیادی هستند، برآزندگی بالایی نسبت می‌دهد.

تکامل داروینی: تنوع باعث تغییر می‌شود

- صفات فنوتیپیک (Phenotypic traits)
  - ویژگی‌های رفتاری و تفاوت‌های فیزیکی نمونه‌ها که روی پاسخ به محیط (شامل سایر نمونه‌ها) تاثیر می‌گذارد (تعیین کننده برآزندگی).
  - بخشی از این صفات توسط وراثت و بخشی به وسیله عواملی در طول توسعه تعیین می‌شود.
  - هر فرد بیانگر یک ترکیب منحصر به فرد از صفات فنوتیپیکی می باشد که بوسیله محیط ارزیابی می‌شود.
    - اگر ارزیابی یک نمونه خوب باشد، آنگاه صفات فنوتیپیکی آن از طریق نسل‌هایش منتشر می‌شود.
    - در غیر اینصورت این صفات بدون داشتن نسلی از بین می‌روند.
  - برای هر نمونه منحصر به فرد است، بخشی از آن نتیجه تحولات تصادفی.

تکامل داروینی: تنوع باعث تغییر می‌شود

- اگر صفات فنوتیپیکی:
  - منجر به شانس بالاتری برای تولید مثل شوند
  - می‌بتواند به ارث برده شود
- و بنابراین در نسل‌های بعد تمایل به افزایش خواهند داشت،
- و منجر به ترکیبات جدیدی از صفات می‌شوند.

تکامل داروینی: خلاصه

- یک جمعیت متشکل از مجموعه متنوعی از نمونه‌ها است.
- ترکیب‌هایی از صفات که سازگارتری بهتری دارند، معمولا در جمعیت افزایش می‌یابند.

"واحد انتخاب" افراد هستند

- تغییرات رخ داده از طریق تغییرات تصادفی، منبع ثابتی برای ایجاد تنوع ژنتیکی هستند که به همراه انتخاب به این معنا است که:

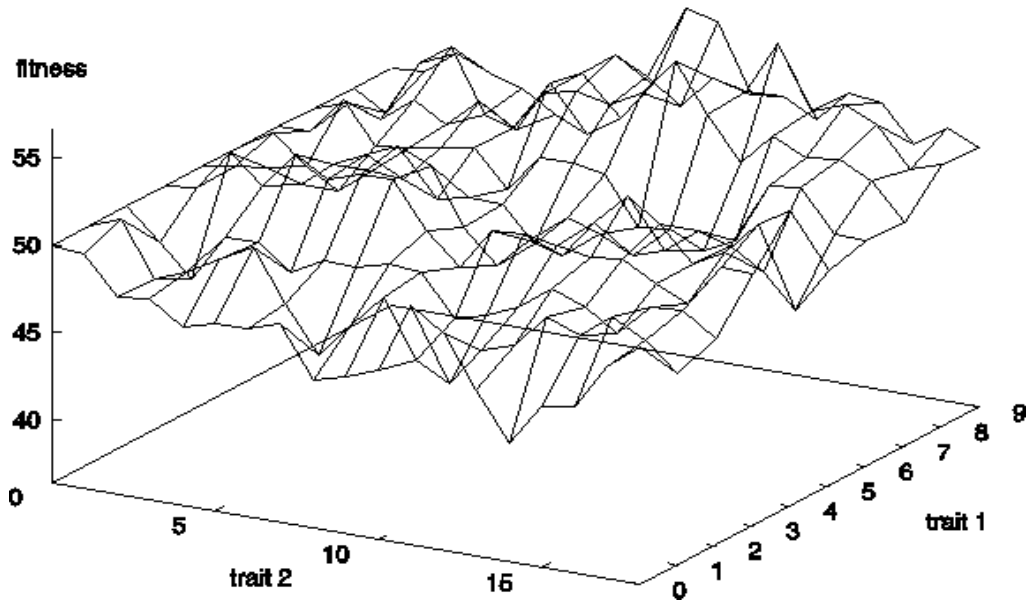
"واحد تکامل" جمعیت است

- به عدم وجود "نیروی هدایت" توجه کنید.

سطح تطبیق پذیر (Adaptive surface)

- می‌توان یک جمعیت با n صفت فنوتیپیکی را در فضایی n+1 بعدی (سطح /چشم انداز) تصور کرد که در آن ارتفاع متناظر با برآزندگی است.
- هر نمونه متفاوت (فنوتایپ) در جمعیت، بیانگر یک نقطه روی این سطح است.
- بنابراین جمعیت ابری از نقاط است که در طول زمان همین طور که تکامل می‌یابد، روی این سطح حرکت می‌کند – سازگاری (adaptation).

مثالی از جمعیتی با دو صفت



سطح تطبیق پذیر

- انتخاب جمعیت را به سمت بالای سطح فشار می‌دهد.
- تغییرات تصادفی در توزیع ویژگی‌ها می تواند باعث شود که جمعیت به سمت پایین حرکت کند و بنابراین از بهینه‌های محلی فرار کند.

ژنتیک طبیعی

- تمام اطلاعات لازم برای ایجاد یک ارگانیزم زنده در DNA مربوط به آن ارگانیزم کد شده است.
- ژنوتایپ (Genotype) (درون DNA) تعیین کننده فنوتایپ است.
- تبدیل ژن‌ها به صفات فنوتیپیکی (Genes → phenotypic traits) نگاشت پیچیده‌ای است:
  - یک ژن ممکن است روی صفات بسیاری تأثیر بگذارد (pleiotropy)
  - ممکن است یک صفت از ژن‌های بسیاری تأثیر بپذیرد (polygeny)
- تغییرات کوچک در ژنوتایپ سبب ایجاد تغییرات کوچکی در ارگانیزم زنده می‌شود – مثلا بلندی قد، رنگ مو و ...

ژن‌ها و ژنوم (Genes and the Genome)

- ژن‌ها در رشته‌هایی از DNA به نام کروموزوم کد شده‌اند.
- دیپلوئید (diploidy): در اکثر سلول‌ها از هر کروموزوم دو نسخه وجود دارد.
- به تمامی مواد ژنتیکی در یک فرد، ژنوم گفته می‌شود.
- در یک گونه، اکثر مواد ژنتیکی یکسان است.

مثال: انسان

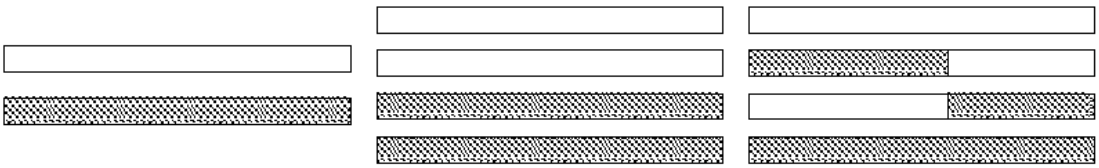
- در انسان، DNA در کروموزوم‌ها ساماندهی می‌شوند.
- سلول های بدن انسان شامل ۲۳ جفت کروموزوم است که با هم صفات فیزیکی شخص را تعریف می‌کنند.

سلول‌های تکثیر کننده

- گامت‌ها (Gametes) (اسپرم و سلول های تخمی) به جای ۲۳ جفت کروموزوم، شامل ۲۳ کروموزوم به صورت انفرادی می‌باشند.
- سلول‌هایی که از هر کروموزوم تنها یک کپی دارند هاپلوئید نام دارند (Haploid)
- گامت ها توسط نوع خاصی از تقسیم سلولی بنام میوز شکل می‌گیرند (meiosis)
- در حین میوز بر روی کروموزوم های جفتی، عملی به نام crossing-over اتفاق می‌افتد.

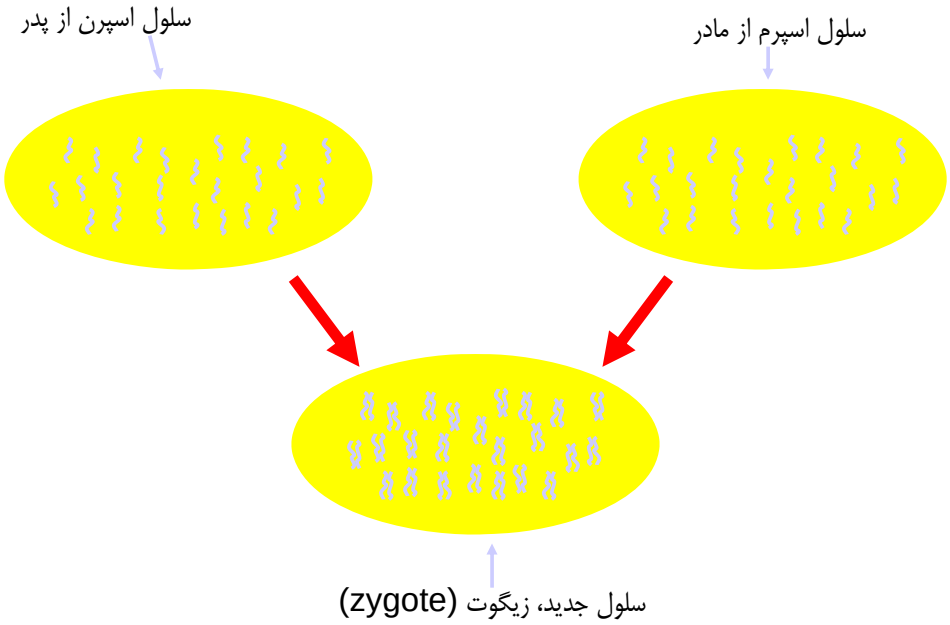
Crossing-Over در حین میوز

- کروموزوم‌های جفتی تراز می‌شوند و سپس هر کدام به دو تا تبدیل می‌شوند.
- در مرحله بعدی جفت کروموزوم داخلی در یک نقطه میانی به نام سانترومر (centromere) به یکدیگر متصل می‌شوند و بخش‌هایی از یکدیگر را با هم جابجا می‌کنند.



- خروجی یک کپی از کروموزوم‌های پدری و مادری به علاوه دو ترکیب کاملا جدید خواهد بود.

لقاح (Fertilisation)



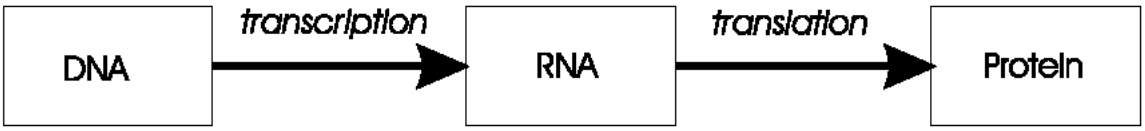
بعد از لقاح

- زیگوت جدید به سرعت تقسیم می‌شود و سلول‌های بسیاری را با محتویات ژنتیکی یکسان تولید می‌کند.
- اگرچه تمام سلول‌های ایجاد شده دارای ژن‌های یکسانی هستند اما بسته به اینکه مثلا در کجای ارگانیزم قرار می‌گیرند، به صورت‌های متفاوتی عمل می‌کنند.
- این فرآیند رفتارهای متفاوت در طول توسعه ontogenesis نامیده می‌شود.

Genetic code

- All proteins in life on earth are composed of sequences built from 20 different amino acids
- DNA is built from four nucleotides in a double helix spiral: purines A,G; pyrimidines T,C
- Triplets of these from codons, each of which codes for a specific amino acid
- Much redundancy:
  - purines complement pyrimidines
  - the DNA contains much rubbish
  - 43=64 codons code for 20 amino acids
  - genetic code = the mapping from codons to amino acids
- For all natural life on earth, the genetic code is the same !

Transcription, translation



A central claim in molecular genetics: only one way flow

Genotype → Phenotype

Genotype ↛ Phenotype

Lamarckism (saying that acquired features can be inherited) is thus wrong!

جهش

- برخی مواقع مواد ژنتیکی در طول این فرآیند به میزان بسیار کمی تغییر می‌کنند (خطای کپی)
- یعنی یک فرزند ممکن است حاوی اطلاعات ژنتیکی باشد که از هیچ کدام از والدینش به ارث نبرده باشد.
- این امر می‌تواند:
  - فاجعه آمیز باشد: offspring معتبر نباشد (بسیاری از مواقع)
  - خنثی باشد: ویژگی‌های جدید برازندگی را تغییر ندهند
  - مفید باشد: ویژگی‌های جدید قوی ایجاد شوند

انگیزه برای EC

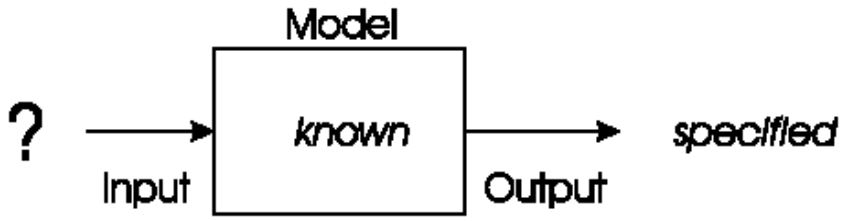
- همواره طبیعت به عنوان منبع الهام برای مهندسین و دانشمندان بکار رفته است.
- بهترین حل کننده‌های مسائل شناخته شده در طبیعت:
  - مغز (انسان): که "چرخ، شهر نیویورک، جنگ و ... " را ایجاد کرده است
  - مکانیزم تکامل: که مغز انسان را ایجاد نموده است
- پاسخ ۱: neurocomputing
- پاسخ ۲: پردازش تکاملی

انگیزه برای EC

- توسعه، تحلیل و اعمال روش‌های حل مسأله (الگوریتم‌ها) یک موضوع مرکزی در ریاضیات و علوم کامپیوتری می‌باشد.
- زمان برای تحلیل دقیق و کامل مسأله کاهش می‌یابد.
- پیچیدگی مسائلی که باید حل شوند افزایش می‌یابد.
- نتیجه:
  - نیاز به یک تکنولوژی قوی برای حل مسأله

انواع مسائل: بهینه‌سازی (Optimisation)

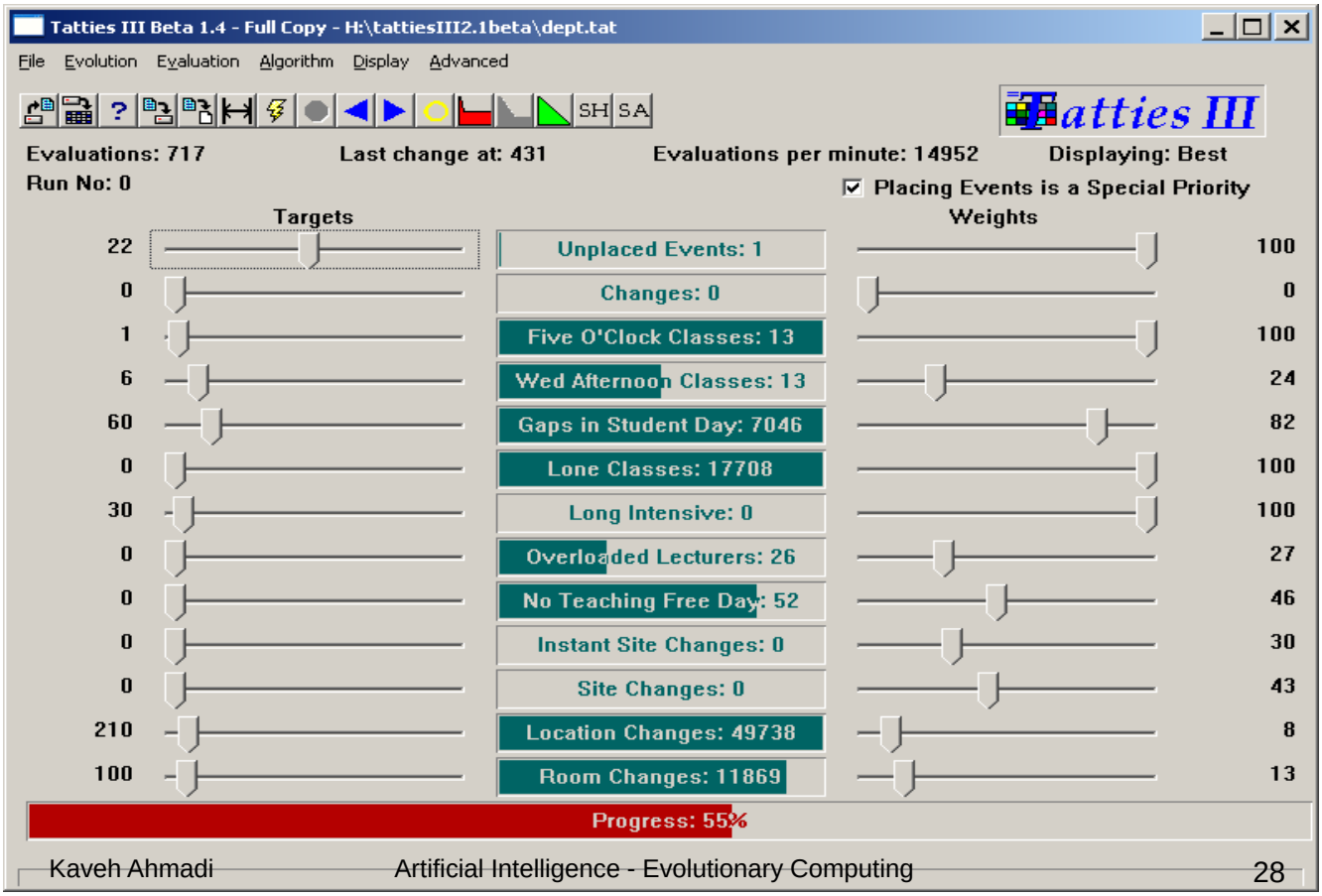
- مدلی از سیستم خود داریم و به دنبال ورودی‌هایی هستیم که به ما یک پاسخ مشخص شده می‌دهند.



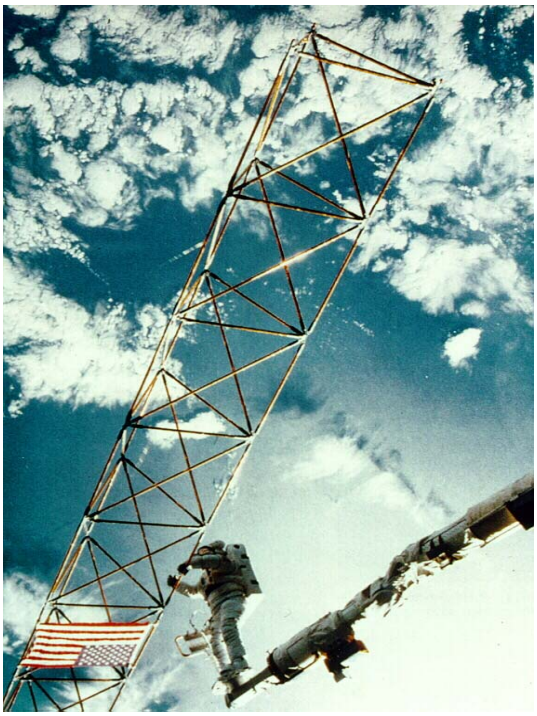
- مثال: جداول زمانی برای دانشگاه یا بیمارستان

مثال بهینه سازی: جدول زمانی دانشگاه

- فضای جستجوی بسیار بزرگ است.
- جداول زمانی باید خوب باشند.
- "خوب" بوسیله تعداد معیارهای رقابت کننده تعریف می‌شود:
  - دانشجویان ترجیح می‌دهند که در یک روز بیش از ۲ کلاس نداشته باشند.
  - اساتید میل دارند که برای انجام تحقیقات بیشتر روزهایشان آزاد باشد.
  - مدیریت دانشگاه می‌خواهد از فضای موجود حداکثر استفاده را کند و یا جابجایی‌ها میان ساختمان های مختلف را کاهش دهد.
- جداول زمانی باید امکان پذیر باشند
  - یک دانشجو نمی‌تواند در یک زمان در دو کلاس حضور داشته باشد.
  - در یک کلاس نمی‌توان به طور همزمان دو درس را ارائه کرد.
- قسمت بسیار بزرگی از فضای جستجو امکان پذیر نمی‌باشد.



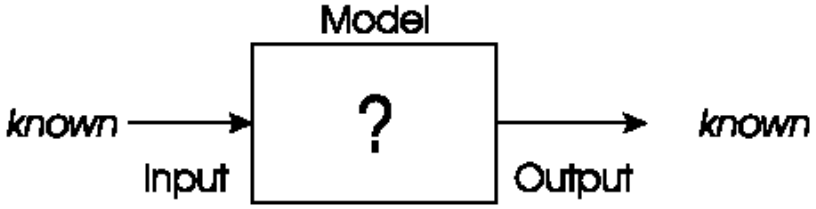
مثال بهینه‌سازی: ساختار ماهواره



- طراحی‌های بهینه ماهواره توسط NASA به منظور حدکثر سازی مقاومت در برابر ارتعاشات
- تکامل: ساختارهای طراحی
- برازندگی : مقاومت در برابر لرزه
- "خلاقیت" تکاملی

انواع مسأله ۲: مدلسازی (Modelling)

- مجموعه مرتبطی از ورددی‌ها و خروجی‌ها داریم و به دنبال مدلی هستیم که بازای هر ورودی خروجی درستی تولید کند.
- یادگیری ماشین به صورت تکاملی



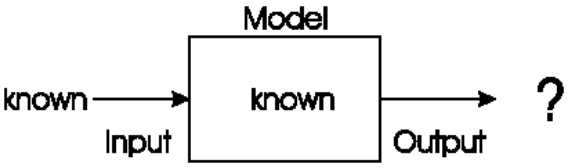
مثال مدلسازی: اعتبار سنجی متقاضیان وام

- بانک انگلیس یک مدل اعتبار سنجی برای پیش بینی رفتار بازپرداخت وام برای متقاضیان جدید توسعه داده است.

- تکامل: مدل های پیش بینی کننده
- برازندگی: دقت مدل بر روی داده های قبلی

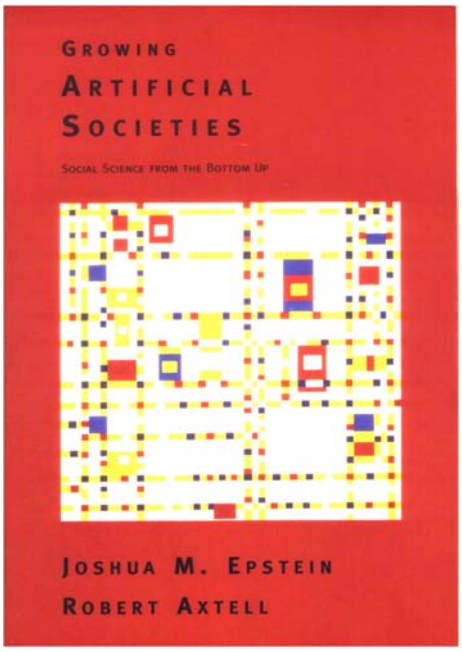
انواع مسائل: شبیه سازی (Simulation)

- یک مدل به ما داده شده. می‌خواهیم خروجی متناظر با ورودی‌های مختلف را بدانیم.
- مثال: حیات مصنوعی، اقتصاد تکاملی



Simulation example: evolving artificial societies

- Simulating trade, economic competition, etc. to calibrate models
- Use models to optimise strategies and policies
- Evolutionary economy
- Survival of the fittest is universal (big/small fish)



Demonstration: magic square

- یک فضای 10x10 با یک مربع 3x3 داخل آن داریم.
- مسئله: اعداد ۱ تا ۱۰۰ را در فضا فرار بده به شکلی که
  - جمع‌های افقی، عمودی، قطری برابرند (۵۰۵).
  - مربع 3x3 یک راه‌حل برای ۱ تا ۹ ارائه می‌کند.

Demonstration: magic square

- رویکرد تکاملی به حل این معما:
  - ایجاد چیدمان شروع تصادفی
  - ساخت N جهش در چیدمان ایجاد شده
  - نگهداری فرزند ایجاد شده با کمترین خطا
  - ایست زمانی که خطا صفر شد

Demonstration: magic square

- Software by M. Herdy, TU Berlin
- Interesting parameters:
  - Step1: small mutation, slow & hits the optimum
  - Step10: large mutation, fast & misses (“jumps over” optimum)
  - Mstep: mutation step size modified on-line, fast & hits optimum
- Start: double-click on icon below
- Exit: click on TUBerlin logo (top-right)

Application

What is an Evolutionary Algorithm?

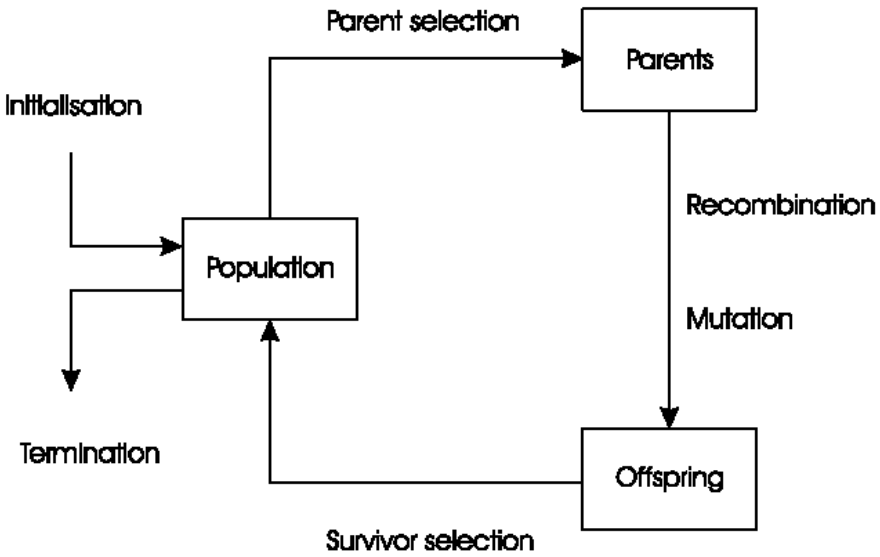
یک الگوریتم تکاملی چیست؟ ایده اصلی

- یک محیط از جمعیتی از افراد با منابع محدود تشکیل می‌شود.
- رقابت برای این منابع باعث انتخاب افرادی می‌شود که بهتر از بقیه با محیط تطبیق یافته‌اند.
- نمونه‌ها به عنوان والد برای ایجاد نمونه‌های جدید از طریق آمیزش (recombination) و جهش (mutation) عمل می‌کنند.
- برازندگی نمونه‌های جدید ارزیابی می‌شود و نمونه‌های جدید به منظور بقا به رقابت می‌پردازند (احتمالا با والدین)
- در طول زمان انتخاب طبیعی باعث افزایش برازندگی جمعیت می‌شود.

یک الگوریتم تکاملی چیست؟

- الگوریتم‌های تکاملی در دسته الگوریتم‌های «تولید و تست» قرار می‌گیرند.
- ویژگی‌های الگوریتم‌های تکاملی:
  - الگوریتم‌های تکاملی مبتنی بر جمعیت می‌باشند: مجموعه‌ای از راه حل‌های کاندید را به طور همزمان پردازش می‌کنند.
  - الگوریتم‌های تکاملی اغلب از عملگر آمیزش به منظور ترکیب اطلاعات موجود در چندین راه‌حل در یک راه حل جدید استفاده می‌کنند.
  - الگوریتم‌های تکاملی اتفاقی (غیرقطعی) می باشند.
  - انتخاب، تنوع را کاهش می‌دهد و به عنوان به شکلی عمل می‌کند که کیفیت را مجبور به بهبود کند.

شمای کلی الگوریتم های تکاملی



شبه کد یک الگوریتم تکاملی

```
BEGIN
  INITIALISE population with random candidate solutions;
  EVALUATE each candidate;
  REPEAT UNTIL ( TERMINATION CONDITION is satisfied ) DO
    1 SELECT parents;
    2 RECOMBINE pairs of parents;
    3 MUTATE the resulting offspring;
    4 EVALUATE new candidates;
    5 SELECT individuals for the next generation;
  OD
END
```


انواع مختلف الگوریتم های تکاملی

- از نظر تاریخی، انواع مختلف الگوریتم‌های تکاملی در نحوه بازنمایی با یکدیگر تفاوت دارند:
  - رشته‌های دودویی: الگوریتم‌های ژنتیک (Genetic Algorithms)
  - بردارهایی از اعداد حقیقی: استراتژی‌های تکامل (Evolution Strategies)
  - ماشین‌های متناهی الحالت: برنامه نویسی تکاملی (Evolutionary Programming)
  - درخت‌ها: برنامه‌نویسی ژنتیک (Genetic Programming)
- بهترین استراتژی:
  - انتخاب بازنمایی مناسب (طبیعی یا ساده) برای مسأله.
- انتخاب عملگرهای مناسب برای نحوه بازنمایی.
  - عملگرهای انتخاب تنها از مقدار برازندگی استفاده می‌کنند و بنابراین به بازنمایی بستگی ندارند.

مولفه های یک الگوریتم تکاملی

- به منظور پیاده‌سازی یک الگوریتم تکاملی باید مؤلفه‌های زیر مشخص شوند:
  - بازنمایی (نحوه تعریف افراد)
  - تابع ارزیابی (یا تابع برازندگی)
  - جمعیت
  - مکانیزم انتخاب والد
  - عملگرهای ژنتیکی (آمیزش و جهش)
  - مکانیزم انتخاب بازمانده (استراتژی جایگزینی)
- علاوه بر این موارد، باید نحوه تولید جمعیت اولیه و شرط (شرایط) توقف الگوریتم نیز مشخص شوند.

روش‌های بازنمایی (تعریف نمونه‌ها) – Representations

- راه حل های کاندیدا (نمونه‌ها) در فضای فنوتایپ قرار دارند.
- این راه حل‌ها به صورت کروموزوم‌ها کد می‌شوند که در فضای ژنوتایپ (genotype) قرار دارند.
  - کد کردن: فنوتایپ ← ژنوتایپ (لزوماً یک به یک نیست)
  - دیکد کردن: ژنوتایپ ← فنوتایپ (باید یک به یک باشد)
- کروموزم‌ها حاوی ژن‌ها می‌باشند که در موقعیت‌هایی (معمولا ثابت) به نام locus قرار دارند و دارای مقادیری (allele) هستند.
- برای یافتن بهینه سراسری، هر راه حل امکان پذیر باید بتواند در فضای ژنوتایپ بازنمایی شود.

تابع ارزیابی (برازندگی)

- بیانگر نیازمندی‌هایی است که جمعیت باید با آن ها تطبیق یابد
  - نوعی تابع کیفیت (quality function) یا تابع هدف (objective function)
- به هر فنوتایپ یک مقدار برازندگی (عدد حقیقی) نسبت می‌دهد و اساس انتخاب را تشکیل می‌دهد.
  - بنابراین هر قدر مقادیر متفاوت‌تری را نسبت دهد، بهتر است
- نوعا برازندگی کمیتی است که باید بیشینه شود
  - تبدیل یک مسأله کمینه سازی به یک مسأله بیشینه سازی (و برعکس) کار ساده‌ای است.

جمعیت

- راه‌حل‌های ممکن برای مسأله را نگهداری می‌کند (بازنمایی آنها را).
- معمولا دارای اندازه ثابتی است و یک چندمجموعه (multiset) از ژنوتایپ‌ها است.
- معمولا عملگرهای انتخاب کل جمعیت را در نظر می‌گیرند، یعنی احتمال تکثیر هر کروموزوم نسبت به نسل فعلی محاسبه می‌شود.
- برخی از Eaهای پیشرفته ساختاری خاص را روی جمعیت در نظر می‌گیرد. مثلا گرید.
  - پیشرفته‌تر: ساختارهای پیتسبرگ در برابر میشیگان
- تعداد برازندگی‌ها / فنوتایپ‌ها / ژنوتایپ‌های متفاوت در جمعیت معرف تنوع (Diversity) آن جمعیت است.

مکانیزم انتخاب

- افراد را بر اساس مقدار برازندگی آنها به عنوان والد برای تولید نسل انتخاب می‌کند.
- معمولا احتمالاتی است:
  - راه حل‌هایی که دارای کیفیت بالاتری هستند نسبت به راه حل‌هایی که دارای کیفیت پایین‌تری هستند، شانس بیشتری برای انتخاب شدن دارند.
  - اما چنین چیزی تضمین شده نیست.
  - حتی بدترین فرد حاضر در جمعیت نیز شانس انتخاب شدن دارد.
- همین طبیعت غیر قطعی به فرار از بهینه های محلی کمک می‌کند.

عملگرهای ژنتیکی (Variation Operators)

- نقش آنها تولید نمونه‌های کاندیدای جدید است.
- این عملگرها معمولا بر اساس تعداد عملوندها (تعداد نمونه‌های ورودی) در دو دسته قرار می‌گیرند:
  - چندی = ۱: عملگرهای جهش (mutation)
  - چندی > ۱: عملگرهای آمیزش (recombination)
  - چندی = ۲: عملگر Crossover
- بحث های بسیاری در مورد اهمیت نسبی عملگرهای ژنتیکی (آمیزش و
- جهش) وجود دارد:
  - امروزه اغلب الگوریتم‌های تکاملی از هر دو استفاده می‌کنند.
  - انتخاب عملگرهای ژنتیکی خاص، وابسته به نحوه بازنمایی است.

عملگر جهش (Mutation)

- بر روی یک ژنوتایپ عمل می‌کند و یک ژنوتایپ جدید ایجاد می‌کند.
- در جهش، تصادفی بودن یک عنصر ضروری است و باعث تمایز آن از دیگر عملگرهای هیوریستیک می‌شود.
- اهمیت آن بستگی به روش بازنمایی و نوع الگوریتم تکاملی دارد:
  - در GA دودویی- یک عملگر پس زمینه و مسوول حفظ و ایجاد تنوع می‌باشد.
  - در EP برای ماشین‌های متناهی الحالت و متغیرهای پیوسته – تنها عملگر جستجو
  - در GP به ندرت استفاده می‌شود.
- عملگر جهش می‌تواند تضمین کننده پیوستگی فضای جستجو باشد (اثبات همگرایی)

عملگر آمیزش (Recombination)

- اطلاعات والدین را در فرزندان ادغام می‌کند.
- انتخاب اینکه چه اطلاعاتی باید ادغام شوند، اتفاقی است.
- اغلب فرزندان ممکن است بدتر و یا مانند والدینشان باشند.
- این عملگر با این امید انجام می‌شود که برخی از فرزندان با ترکیب عناصر ژنوتایپ‌ها که منجر به ایجاد ترکیب‌های بهتری از ویژگی‌ها می‌شوند، از والدینشان بهتر باشند.
- این اصل هزاران سال توسط پروش دهندگان گیاهان و حیوانات استفاده شده است.

جایگزینی (Survivor Selection)

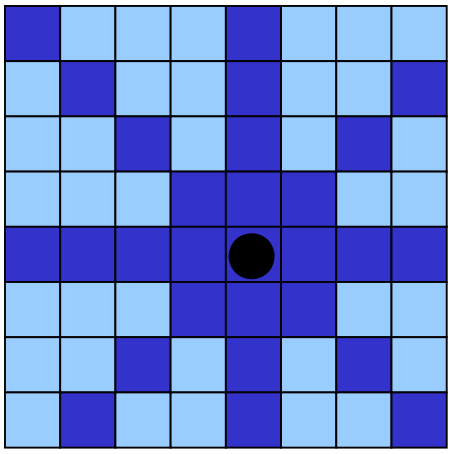
- اغلب الگوریتم‌های تکاملی از یک اندازه ثابت برای جمعیت استفاده می‌کنند. بنابراین به روشی برای رفتن به نسل بعدی (با انتخاب افراد از میان والدین و فرزندان) نیاز دارند.
- اغلب قطعی می‌باشد:
  - مبتنی بر برازندگی: مثلا رتبه‌بندی والدین + فرزندان بر اساس برازندگی‌ها و انتخاب بهترین‌ها
  - مبتنی بر سن: به تعداد والدین فرزند ایجاد می‌شود و سپس تمامی والدین حذف می‌شوند
  - برخی مواقع به صورت ترکیبی (elitism)

مقدار دهی جمعیت اولیه و توقف (Initialisation / Termination)

- مقدار دهی جمعیت اولیه معمولا به صورت تصادفی صورت می‌گیرد.
- می‌تواند شامل راه‌حل‌های موجود باشد و یا از هیوریستیک‌های خاص مسأله برای ایجاد جمعیت اولیه استفاده کند
- شرایط توقف در هر نسل بررسی می‌شود:
  - رسیدن به یک مقدار خاص از برازندگی
  - رسیدن به یک حداکثر تعداد مجاز از نسل‌ها
  - رسیدن به یک سطح حداقل از نظر تنوع
  - رسیدن به یک تعداد مشخص از نسل‌های متوالی بدون بهبود برازندگی

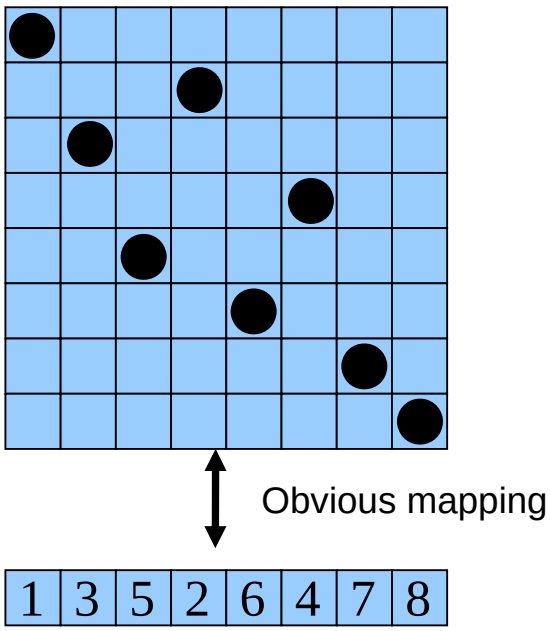
مثال: مسأله ۸–وزیر

- هشت وزیر را بر روی یک صفحه شطرنج هشت در هشت قرار دهید به طوری که هیچ دو وزیری نتوانند یکدیگر را تهدید کنند



مسأله ۸–وزیر: بازنمایی

- فنوتایپ:
  - یک پیکره‌بندی از صفحه
- ژنوتایپ:
  - یک جایگشت از اعداد ۱ تا ۸

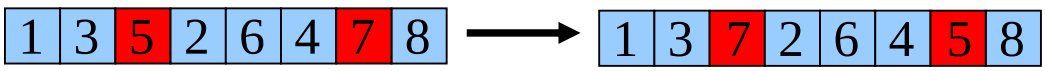


مسأله هشت وزیر: ارزیابی برازندگی

- محاسبه خطا برای یک وزیر:
  - تعداد وزیرهای تحت حمله توسط آن وزیر
- محاسبه خطا برای یک پیکره‌بندی
  - مجموع خطای تمامی وزیرها
- توجه: خطا باید کمینه شود
- محاسبه برازندگی یک پیکره بندی
  - بیشینه سازی عکس خط

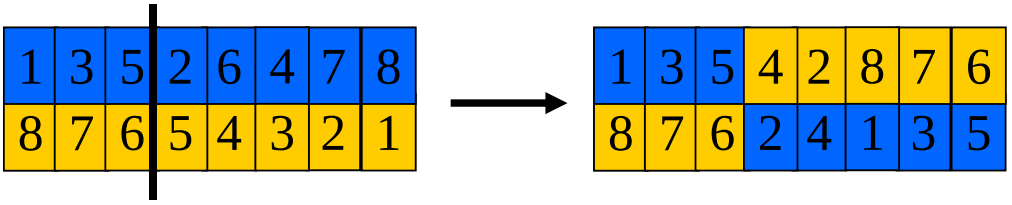
مسأله ۸–وزیر: جهش

- تغییر کوچک در یک جایگشت
  - مثلاً: جابجا نمودن مقادیر دو مکان به صورتصادفی



مسأله ۸-وزیر: آمیزش

- ترکیب نمودن دو جایگشت در دو جایگشت جدید
- یک نقطه crossover به صورت تصادفی انتخاب کن.
- بخش‌های اول را در فرزندان کپی کن
- بخش دوم هر فرزند را با درج مقادیر از والد دیگر ایجاد کن
  - به ترتیب وقوع مقادیر
  - با شروع از مکان بعد از نقطه crossover
  - با پرش از روی مقادیر موجود در فرزند



مسأله ۸-وزیر: انتخاب

- انتخاب والد:
  - ۵ والد را انتخاب کن و ۲تای بهتر را برای انجام crossover برگزین.
- انتخاب بازمانده (جایگزینی):
  - هنگام درج یک فرزند جدید در جمعیت، به روش زیر یک عضو موجود را برای جایگزینی انتخاب کن:
    - کل جمعیت را بر اساس مقادیر برازندگی به صورت نزولی مرتب کن
  - لیست را به ترتیب از بالا به پایین پیمایش کن
  - فرزند جدید را با اولین عنصری که برازندگی آن کمتر از برازندگی فرزند داده شده می‌باشد جایگزین کن.

مسأله ۸–وزیر: خلاصه

- توجه کنید که جدول زیر تنها یک مجموعه از انتخاب‌های ممکن را برای عملگرها و پارامترها نشان می‌دهد

Representation	Permutations
Recombination	"Cut-and-crossfill" crossover
Recombination probability	100%
Mutation	Swap
Mutation probability	80%
Parent selection	Best 2 out of random 5
Survival selection	Replace worst
Population size	100
Number of Offspring	2
Initialisation	Random
Termination condition	Solution or 10,000 fitness evaluation

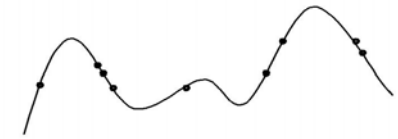
تمرین

- فرموله سازی مناسب برای الگوریتم کوله پشتی صفر و یک ارائه دهید.

رفتار یک الگوریتم تکاملی

- فازهای مختلف در بهینه سازی یک تابع برازندگی یک بعدی

- فاز اولیه
  - توزیع تصادفی افراد در کل فضای جستجو



- فازمیانی
  - جمعیت اطراف قله‌ها پراکنده است

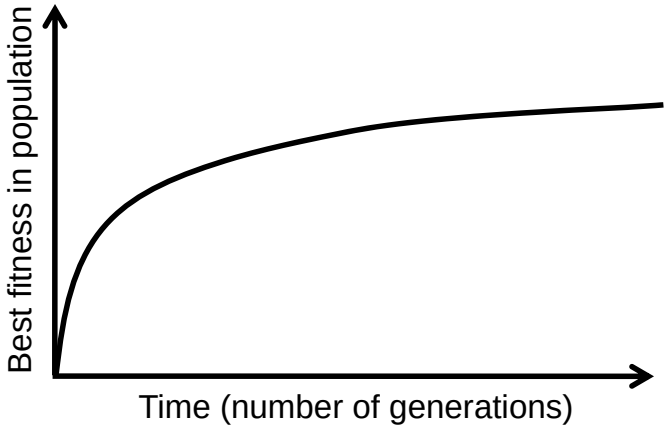
- فاز نهایی
  - جمعیت در اطراف قله‌های مرتفع پراکنده است

Exploration vs. Exploitation

- فازهای مختلف جستجو اغلب برحسب موارد زیر دسته بندی می‌شوند:
  - اکتشاف (Exploration): تولید افراد جدید در نواحی آزمایش نشده از فضای جستجو
  - بهره برداری (Exploitation): تمرکز جستجو در نزدیکی راه حل‌های خوب
- در فرآیند جستجوی تکاملی باید بین این دو توازن برقرار شود:
  - Exploration زیاد: عدم کارآیی جستجو، پاسخ بهینه
  - Exploitation زیاد: افزایش میزان حریصانه بودن جستجو، سریع، پاسخ غیر بهینه
- همگرایی زودرس (premature convergence): از دست دادن سریع تنوع جمعیت و افتادن در دام بهینه‌های محلی

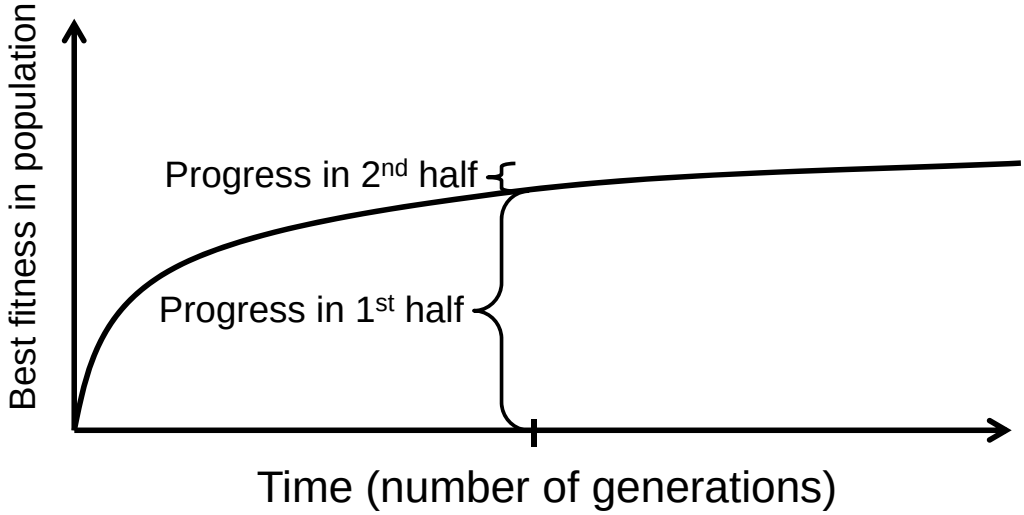
یک نمودار اجرای نوعی

- نشانگر «anytime behavior»
  - معنی anytime: جستجو را می‌توان در هر زمان دلخواه متوقف نمود و الگوریتم یک راه حل می‌یابد (اگرچه زیر بهینه)



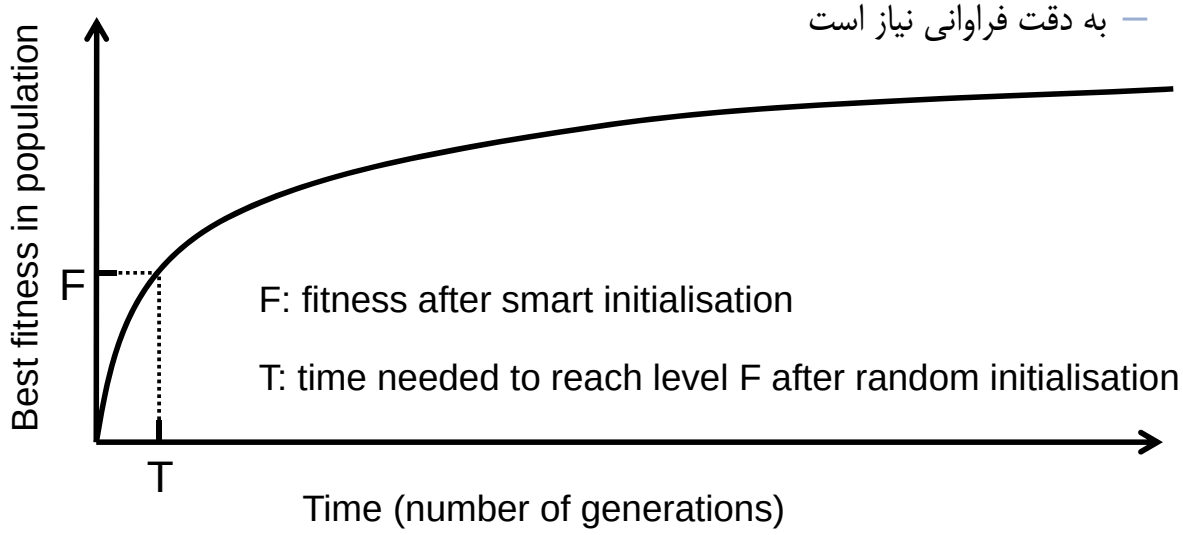
آیا اجراهای طولالانی مفیدند؟

- پاسخ: بستگی به مقدار مورد نیاز برازندگی دارد
  - بهتر است اجراهای کوتاه‌تر بیشتری انجام شود



آیا تلاش برای ایجاد جمعیت اولیه به صورت هوشمند ارزش دارد؟

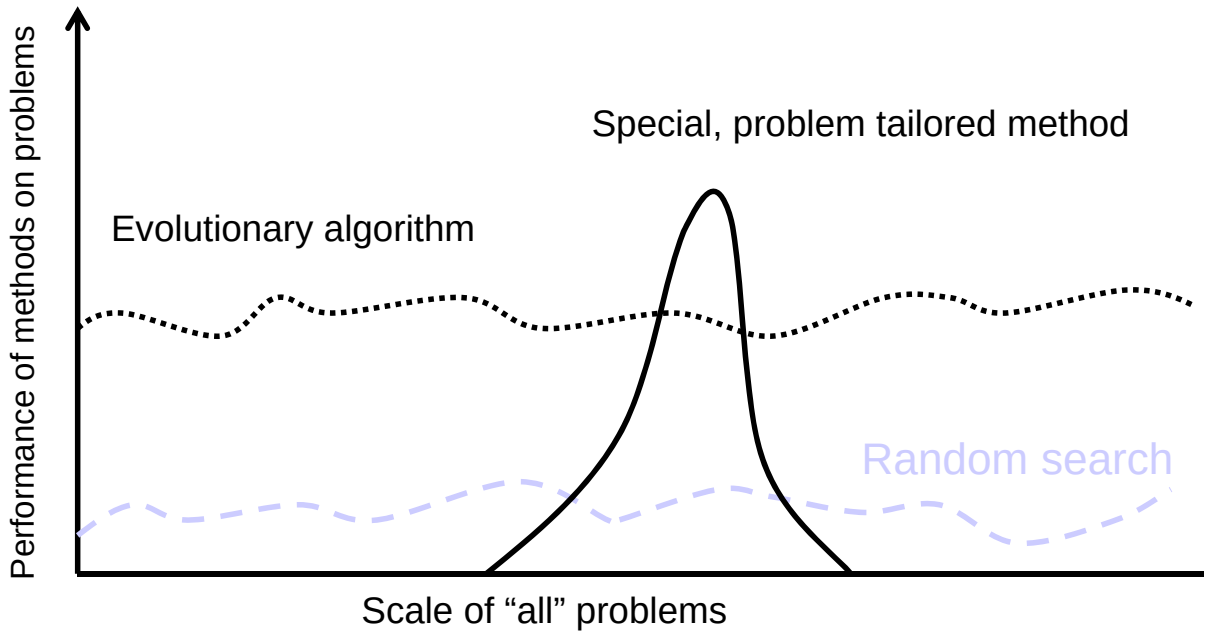
- پاسخ: بستگی دارد:
  - احتمالاً، اگر راه‌حل ها یا روش‌های خوبی وجود دارد
  - به دقت فراوانی نیاز است



الگوریتم های تکاملی در مفهوم

- دیدگاه‌های زیادی در مورد استفاده از الگوریتم‌های تکاملی به عنوان یک ابزار قوی در حل مسأله وجود دارد:
- برای اغلب مسائل یک ابزار خاص مسأله ممکن است:
  - از یک الگوریتم جستجوی عمومی بر روی اغلب نمونه ها بهتر عمل کند
  - کاربرد محدودی داشته باشد
  - بر روی تمام نمونه‌ها به خوبی عمل نکند
- هدف فراهم کردن یک ابزار قوی با عملکرد خوب بر روی گستره وسیعی از مسائل و نمونه ها است.

الگوریتم های تکاملی به عنوان حل کننده مسأله دیدگاه گلدبرگ ۱۹۸۹



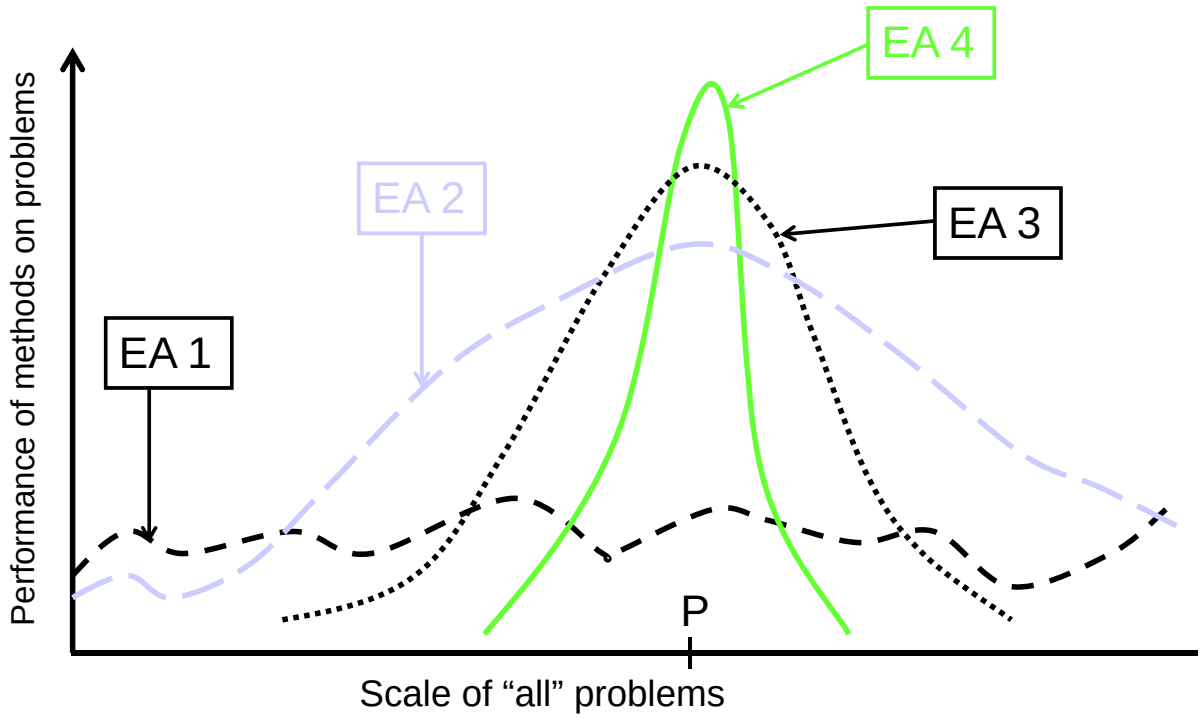
تئوری No Free Lunch

- عملکرد هیچ الگوریتمی به طور متوسط روی همه مسائل، نمی‌تواند بهتر از عملکرد جستجوی تصادفی باشد.
- یعنی نمایش منحنی عملکرد EA همواره در بالای منحنی عملکرد جستجوی تصادفی اساساً نادرست است.

الگوریتم‌های تکاملی و دانش دامنه

- دهه ۹۰
  - اضافه نمودن دانش خاص مسأله به الگوریتم‌های تکاملی (عملگرهای ویژه
- ژنتیکی، عملگرهای ویژه ترمیم و ...)
- نتیجه: «تغییر شکل» منحنی عملکرد EA
  - بهتر در مورد مسائل از نوع داده شده
  - بدتر در مورد مسائل از نوع دیگر
  - مقدار دانش اضافه شده متغیر است
- تئوری اخیر پیشنهاد می‌کند که جستجو به دنبال یک الگوریتم «همه منظوره» بی فایده است.

دیدگاه Michalewicz (1996)



پردازش تکاملی و بهینه سازی تکاملی

- بهینه سازی سراسری: جستجو برای یافتن بهترین راه حل X^* از یک مجموعه ثابت S
- روش‌های قطعی
  - مانند شاخه و حد (branch and bound, …)
  - یافتن X^* را تضمین می‌کند اما ممکن است در یک زمان بیش از چند جمله‌ای اجرا شود.
- روش‌های هیوریستیک (تولید و تست)
  - قوانینی برای تصمیم گیری در مورد $x \in S$ بعدی که باید تولید شود.
  - تضمینی برای اینکه بهترین راه حل یافته شده راه حل بهینه سراسری باشد وجود ندارد.

پردازش تکاملی و جستجوی همسایگی

- بسیاری از هیوریستیک‌ها یک ساختار همسایگی را بر روی S تحمیل می‌کنند (روش‌های جستجوی محلی)
- چنین هیوریستیک‌هایی تضمین می‌کنند که بهترین نقطه یافته شده بهینه محلی باشد (مانند تپه نوردی)
  - اما در اغلب مسائل، بهینه‌های محلی فراوانی وجود دارد.
  - اغلب در یافتن یک راه حل خوب (نزدیک به بهینه سراسری) بسیار سریع است.
- ویژگی‌های الگوریتم‌های تکاملی
  - مبتنی بر جمعیت ← فرار از بهینه‌های محلی – قابل استفاده در فضاهای حالت بزرگ
  - استفاده از چندین عملگر جستجوی اتفاقی
  - عملگرهای ژنتیکی ویژه با چندی بزرگتر از ۱
  - انتخاب اتفاقی