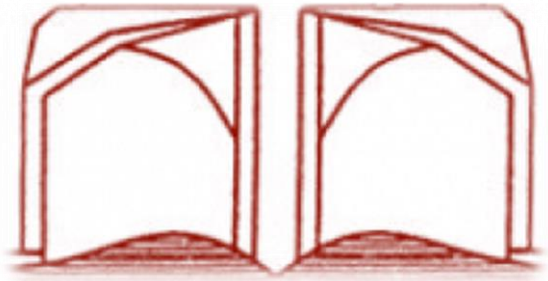


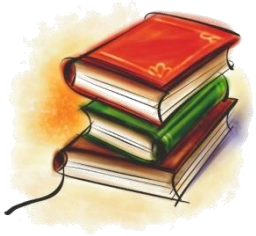
مبانی رایانش نرم

محاسبات زیستی: پردازش تکاملی

هادی ویسی

h.veisi@ut.ac.ir

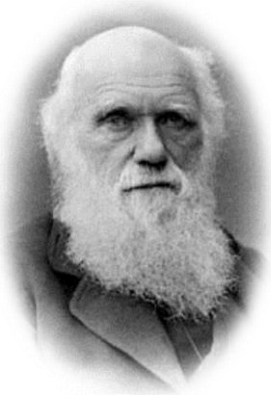
دانشگاه تهران – دانشکده علوم و فنون نوین



فهرست

- الگوریتم‌های جستجو: معرفی و مفاهیم
- پردازش تکاملی
  - مفاهیم
  - نمایش کروموزوم
  - جمعیت اولیه
  - تابع برازش
  - انتخاب (Selection)
  - تولید مثل: بازترکیب (Recombination)/هم‌برش (Crossover)، جهش (Mutation)
  - جایگزینی
  - شرایط توقف
  - کنترل قابلیت پویش و انتفاع
  - نظریه اسکیما
- الگوریتم ژنتیک: مثال

محاسبات زیستی

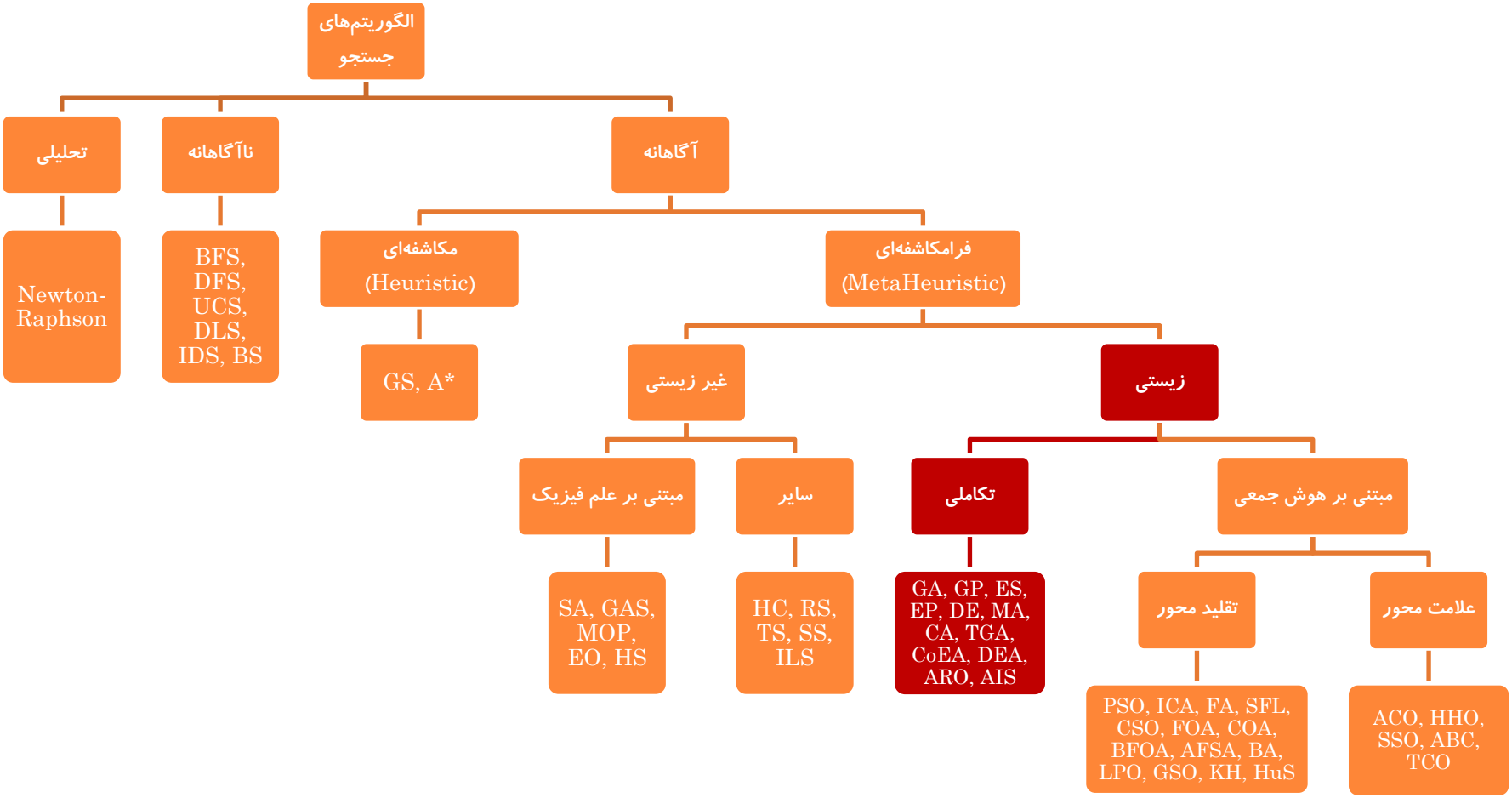


- **محاسبات زیستی (Evolutionary Computing)**
  - **حل مسائل بهینه‌سازی، جستجو و یادگیری ماشین با الهام از تکامل زیستی**

- **نظریه تکامل زیستی داروین (۱۸۵۹)**
  - حیوانات و گیاهان امروزی از نسل موجوات ماقبل تاریخ هستند-صدها میلیون سال از حیات می‌گذرد
  - حیات تنها با یک یا تعدادی ارگانیسم ساده شروع شده و بعدها تکامل یافته و تبدیل به میلیون‌ها گونه متفاوت امروزی شده است
  - تمامی فرآیند خلقت گونه‌های مختلف حیات، ناشی از یکی از نیروهای هدایت‌کننده در طبیعت با نام **انتخاب طبیعی** (Natural Selection) است

- **از بین رفتن نمونه‌های ضعیف و زنده ماندن نمونه‌های برتر = تکامل تدریجی**
- **انتخاب طبیعی راز بقای برترین‌ها در طبیعت و انتقال خصوصیات برتر به نسل بعد= قانون بقای اصلح** (Survival of the Fittest)

الگوریتم‌های جستجو . . .



الگوریتم‌های جستجو . . .

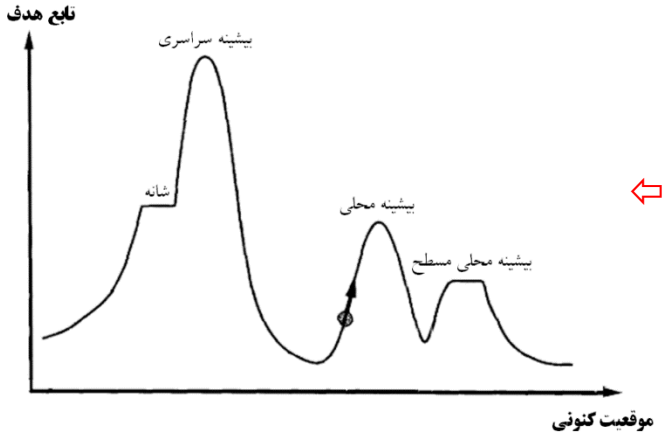
- **جستجوی تحلیلی (Analytical Search)**
  - **استفاده از توابع و روش‌های ریاضی برای یافتن حل بهینه: گرادیان، مشتق دوم**
    - نیوتون-رافسون (برای یافتن ریشه تابع)
- **جستجوی ناآگاهانه (Uninformed (Blind) Search)**
  - **عدم وجود از اطلاعات جانبی درباره نقاط فضای جستجو: تنها تشخیص هدف از غیرهدف**
  - **پیمایش فضای جستجو به صورت درختی**
  - **دو نوع**
    - کامل: پیمایش کامل فضای جستجو و تضمین یافتن یک راه حل (بهینه یا غیربهینه)
    - ناکامل: جستجو تا یافتن یک راه‌حل
  - **روش‌ها**
    - جستجوی اول سطح (Breadth-First Search)
    - جستجوی هزینه یکنواخت (Uniform-Cost Search)
    - جستجوی اول عمق (Depth-First Search)

الگوریتم‌های جستجو

- **جستجوی آگاهانه (Informed Search)**
  - **استفاده از یک تابع تخمین در مورد فضای جستجو (دانش مساله)**
  - **جستجوی اول-بهترین (Best-First Search) – مکاشفه‌ای**
    - جستجوی Tree-Search (Graph-Search)- انتخاب یک گره بر اساس تخمین هزینه گسترش آن
    - روش‌ها
      - جستجوی حریصانه (Greedy Search): جستجوی نزدیک‌ترین گره به هدف
      - جستجوی A* (A* Search): کمینه کردن کل هزینه برآورد شده- محاسبه زیاد و نیاز به حافظه زیاد
  - **جستجوی فرامکاشفه‌ای (Meta-Heuristic Search)**
    - روش‌های قدیمی: نیاز به نمایش فضای جستجو با درخت
      - عدم امکان نمایش فضای جستجو با درخت (به ویژه برای فضاهای بزرگ و نامنظم)
    - الهام از پدیده‌های طبیعی برای جستجو (زیستی و غیرزیستی)

جستجو: مفاهیم . . .

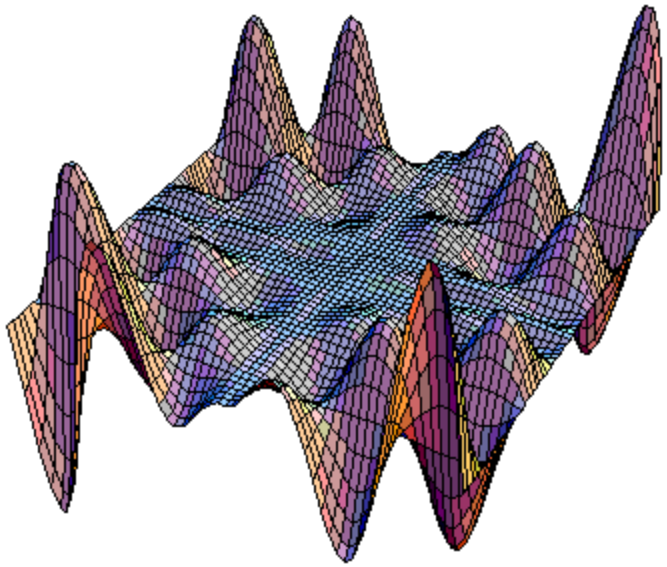
- **فضای جستجو و دورنمای برازش (Fitness Landscape) . . .**
  - **فضای جستجو: یک سرزمین حاوی کلیه مقادیر ممکن برای پاسخ مساله**
    - گاهی بسیار بزرگ و نامنظم
    - جستجوی هوشمندانه: پیمایش بخش مهم فضای پاسخ
  - **دورنمای برازش: پستی و بلندی‌های سرزمین جستجو**
    - تابع برازش (Fitness Function): تعیین پستی و بلندی‌های سرزمین جستجو
      - تعریف بر اساس اطلاعات مساله
      - در هر لحظه شامل نقطه پاسخ فعلی (موقعیت) و مقدار برازش (شایستگی) آن نقطه (ارتفاع)

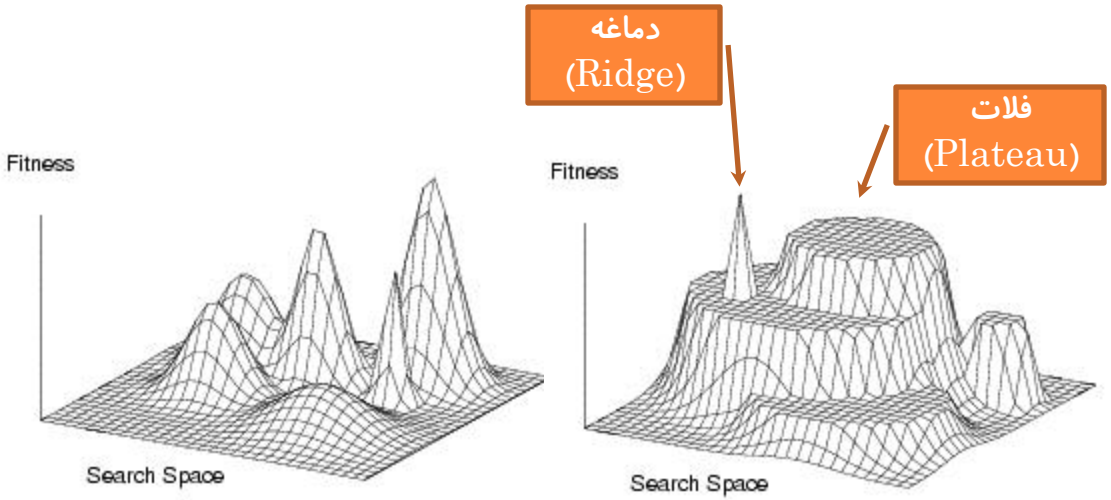


- ارتفاع متناظر با شایستگی: نقطه بهینه = بیشینه مقدار (بلندترین قله) ⇦
- ارتفاع متناظر با هزینه: نقطه بهینه = کمینه مقدار (عمیق‌ترین دره)

جستجو: مفاهیم ...

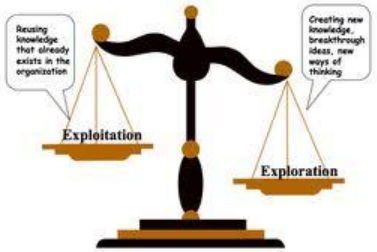
- فضای جستجو و دورنمای برازش (Fitness Landscape)
  - در مسائل واقعی، دورنمای برازش پیچیده است
    - فضای چندقله‌ای یا خارپشتی (multimodal)





جستجو: مفاهیم . . .

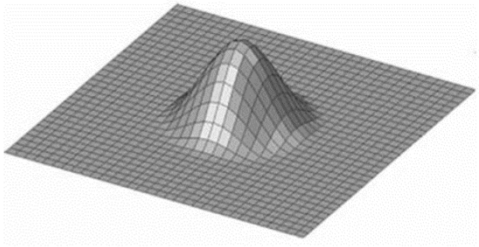
- **قابلیت پویش (Exploration)**
  - جستجوی آزادانه کل فضا بدون توجه به دستاوردهای آن در طول جستجو
  - رفتار تصادفی‌تر الگوریتم
- **قابلیت انتفاع (Exploitation)**
  - توجه به دستاوردهای الگوریتم در طول جستجو
  - رفتار حساب شده و محتاطانه
- **نیاز به تنظیم دو قابلیت بر اساس شرایط مساله**
  - ایجاد مصالحه (trade-off) بین این دو قابلیت با پارامترهای روش جستجو



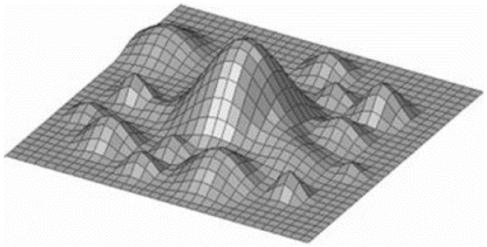
جستجو: مفاهیم . . .

قابلیت پویش و انتفاع

- برای مسائلی با دورنمای برازش منظم، باید انتفاع را تقویت و پویش را کم کنیم



- برای مسائلی با دورنمای برازش نامنظم (خارپشتی)، باید قابلیت پویش را زیاد کنیم



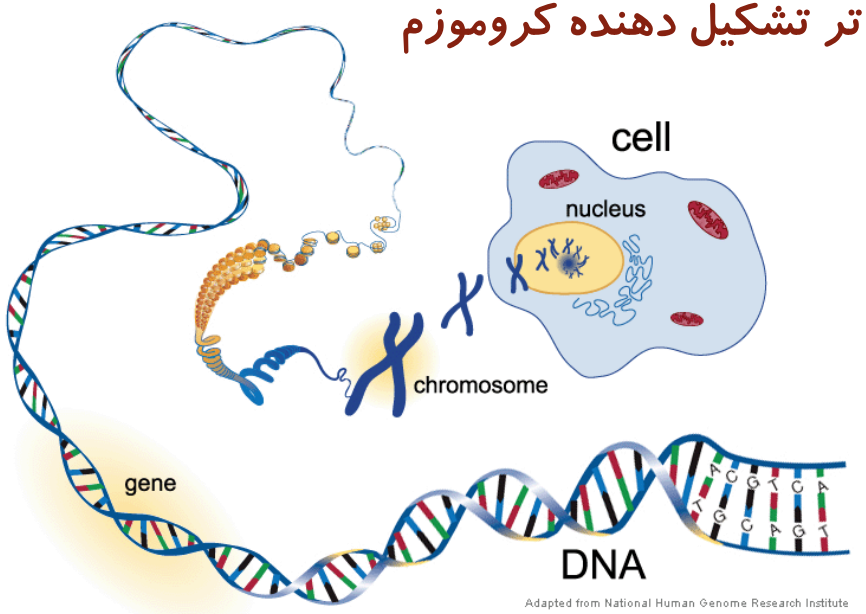
- جستجوی با بیشترین پویش= جستجوی تصادفی (Random Search)
- جستجوی با بیشترین انتفاع= جستجوی تپه نوردی (Hill-Climbing Search)

جستجو: مفاهیم

- مصالحه قابلیت‌های پویش و انتفاع با توجه به مساله
  - وجود یک الگوریتم مناسب برای هر مساله با توجه به شرایط آن
  - عدم وجود یک روش جستجوی بهینه برای کار در همه شرایط
    - No Free Lunch Theorem (از ناهار مجانی خبری نیست!)

پردازش تکاملی . . .

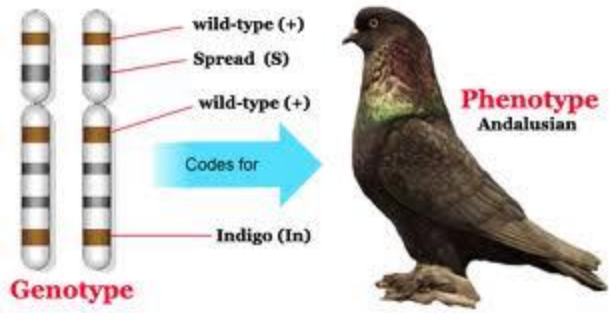
- مفاهیم . . .
  - کروزموزم (Chromosome)– محل ذخیره‌سازی اطلاعات ژنی یک موجود
    - تشکیل شده از DNA
    - انسان تقریباً ۲۲ هزار ژن در DNA خود دارد
    - رشته، گراف، درخت= پاسخ مساله
  - ژن (Gene)–واحدهای کوچک‌تر تشکیل دهنده کروموزم
    - ویژگی (مشخصه) داده‌ها

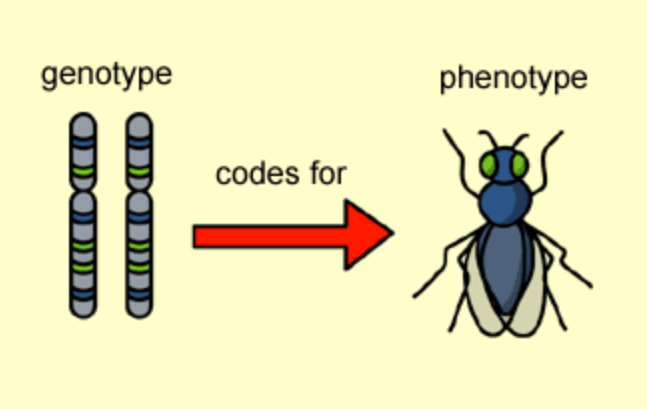


Adapted from National Human Genome Research Institute

پردازش تکاملی . . .

- مفاهیم . . .
  - ژنوتایپ (Genotype) – ترکیب تمام ژن ها برای یک فرد مشخص
  - فنوتایپ (Phenotype)- خصوصیات ظاهری یک فرد، حاصل شده از رمزگشایی یک ژنوتایپ
  - آلل (Allele)- مقادیر مجاز برای هر ژن
    - مقادیر مجاز برای مشخصه‌های هر پاسخ
  - برازش (Fitness)- میزان شایستگی یک موجود در جمعیت





پردازش تکاملی . . .

- مراحل یک الگوریتم تکاملی . . .

۱- تولید جمعیت اولیه (پاسخ‌های اولیه مساله)

۲- محاسبه برازش جمعیت ورودی

۳- انتخاب (Selection) برای تولید مثل (Reproduction): انتخاب والدین شایسته‌تر

- قانون بقای اصلح داروین

۴- بازترکیب (Recombination) والدین انتخاب شده: تولید یک یا چند فرزند با ترکیب ژن‌های دو یا چند والد با هم‌برش (Crossover)

- مفهوم جفت‌گیری

۵- جهش (Mutation) فرزندان تولید شده: تغییر تصادفی ژن‌ها در یک کروموزوم

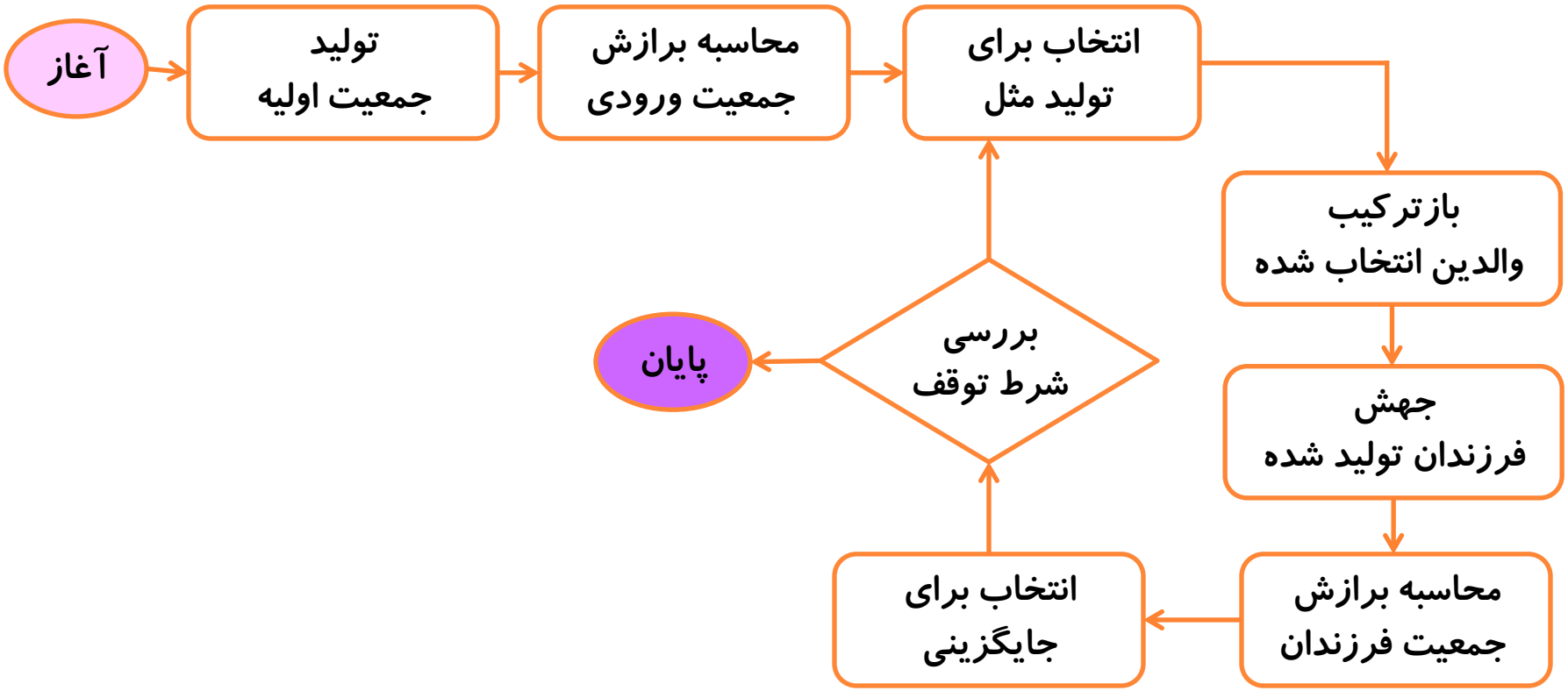
- یافتن مقادیر جدید برای ژن فرزندان (ایجاد تفاوت با والدین)

۶- محاسبه برازش جمعیت فرزندان: محاسبه شایستگی فرزندان جدید

۷- انتخاب برای جایگزینی (Replacement): تولید یک جمعیت به عنوان نسل جدید (از والدین قبلی و فرزندان جدید)

پردازش تکاملی . . .

- مراحل یک الگوریتم تکاملی



پردازش تکاملی: نمایش کروموزوم

- از مهم‌ترین مراحل
  - فرموله کردن مساله برای الگوریتم تکاملی
  - تاثیر زیاد بر کارایی و پیچیدگی الگوریتم

- روش‌ها
  - بردار دودویی با طول ثابت (هر ژن یک عدد دودویی است): متغیرهای گسسته و اسمی

0	1	1	0	1	0	0	0	1

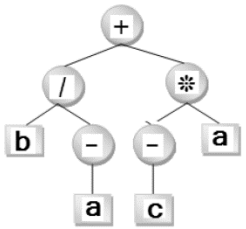
  - نمایش مبتنی بر اعداد حقیقی (هر ژن یک عدد حقیقی): برای متغیرهای پیوسته

1.12	32.6	15.1	19.4	6.19	0.12	10.5	12.3	65.1

  - جایگشت عناصر: مثلاً برای فروشنده دوره‌گرد

4	3	9	6	1	7	2	8	5

  - نمایش درختی: برای برنامه‌ها و روابط ریاضی



پردازش تکاملی: جمعیت اولیه

- جمعیت = تعدادی از راه‌حل‌های کاندید برای مساله
  - جمعیت اولیه = پاسخ اولیه مساله

  - روش تصادفی: مقدار تصادفی در بازه مجاز برای هر ژن (پوشش یکنواخت فضا)
  - روش هوشمندانه: تولید کروموزم‌های اولیه با برازندگی بالا (پوشش بخش‌ها مهم فضا)

  - اندازه جمعیت: معمولاً ثابت
    - افزایش اندازه جمعیت = تقویت قابلیت پویش: پوشش دادن فضای جستجوی بزرگ‌تر
    - افزایش اندازه جمعیت = تقویت قابلیت انتفاع: افزایش شانس عملگرهای تولیدمثل

    - افزایش اندازه جمعیت = افزایش محاسبات

پردازش تکاملی: تابع برازش

- تابع برازش: محاسبه میزان شایستگی پاسخ‌ها (نسل‌ها)
  - نگاشت شایستگی هر کروموزوم به یک مقدار عددی
  - کاربرد در عملگر انتخاب برای تعیین اعضای برازنده
  - وابسته به کاربرد
    - تعیین تابع برازش در برخی کاربردها کار مشکلی است

پردازش تکاملی: انتخاب . . .

- **انتخاب= یکی از عملگرهای اصلی پردازش تکاملی**
  - **بیانگر مفهوم بقای اصلح نظریه داروین**
  - **هدف: یافتن پاسخ‌های برتر مساله در جمعیت جاری برای اعمال عملگر تولید مثل**

  - **فشار انتخاب (Selective Pressure): میزان فشار عملگر انتخاب برای بردن جمعیت به راه‌حل‌های خوب**
    - فشار انتخاب زیاد = توجه بیش از اندازه به اعضای برازنده = کاهش تنوع جمعیت = افزایش قابلیت انتفاع = کاهش قابلیت پویش = همگرایی سریع (محلی)

  - **روش‌های مختلفی برای انتخاب**
    - انتخاب تصادفی
    - انتخاب نسبی (Proportional)
    - انتخاب رتبه‌ای (Rank-based)
    - انتخاب مسابقه‌ای (Tournament)
    - انتخاب برشی (Truncation)

پردازش تکاملی: انتخاب . . .

- **انتخاب تصادفی**
  - انتخاب هرکدام از اعضای جمعیت با احتمال برابر = عدم استفاده از برازش
  - کمترین فشار انتخاب

- **انتخاب نسبی (Proportional)**
  - شانس بیشتر برای موجوات برتر
  - محاسبه احتمال انتخاب عضو iام جمعیت:

برازش عضو iام
(محاسبه با تابع برازش)

$$p_i = \frac{f_i}{\sum_{j=1}^{N} f_j}$$

    - N= تعداد اعضای جمعیت

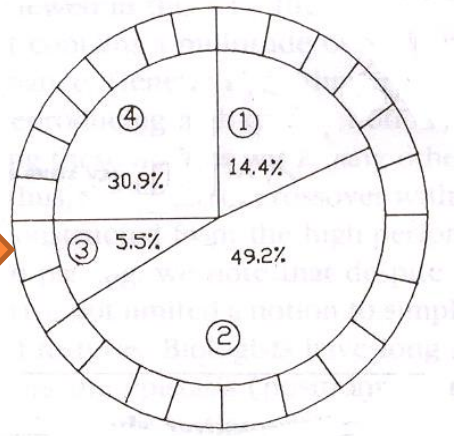
- **پیاده‌سازی انتخاب نسبی با چرخ روالت (Roulette Wheel Selection)**
  - در نظر گرفتن یک دایره (چرخ دوار) که به تعداد N بخش (قطعه) تقسیم شده است
  - هر قطعه مرتبط با هر عضو بوده که اندازه آن متناسب با احتمال آن عضو (برازندگی) است
  - چرخاندن چرخ به تعداد N بار و انتخاب عضوی که چرخ روی آن توقف می‌کند

پردازش تکاملی: انتخاب . . .

- انتخاب نسبی (Proportional)
  - فشار انتخاب زیاد چرخ رولت = همگرایی سریع در بهینه محلی
  - بهبود: چرخ رولت با بیش از یک اشاره‌گر
    - تعداد N اشاره‌گر
    - یک بار چرخاندن
    - تعدیل فشار انتخاب

No	Chromosome	F_i (Fitness)	p_i
1	01101	169	14.4
2	11000	576	49.2
3	01000	64	5.5
4	10011	361	30.9

Selection Point

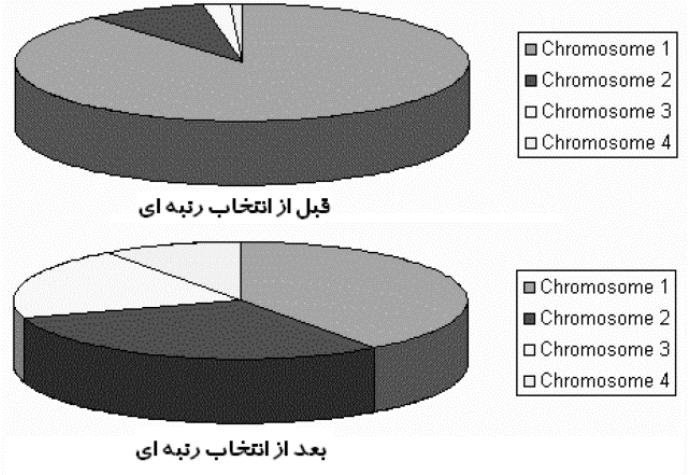


پردازش تکاملی: انتخاب . . .

- انتخاب رتبه‌ای (Rank-based)
  - استفاده از رتبه برازندگی اعضا به جای استفاده از مقدار مطلق برازندگی
    - برازش برترین عضو = N
    - برازش دومین عضو برتر = N-1
    - برازش سومین عضو برتر = N-2
    - ...
    - برازش آخرین عضو برتر = 1
    - محاسبه احتمال بر اساس برازش‌های جدید

$$p_i = \frac{f_i}{\sum_{j=1}^{N} f_j}$$

برازش عضو iام (محاسبه با روش فوق)



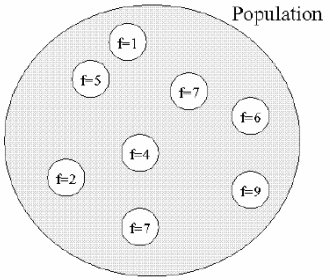
- کاهش فشار انتخاب در مقایسه با چرخ رولت
- عدم همگرایی زودرس

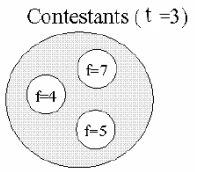
پردازش تکاملی: انتخاب . . .

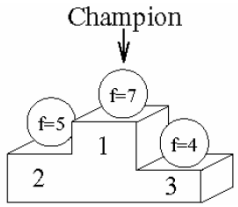
- انتخاب مسابقه‌ای (Tournament)
  - انتخاب یک گروه (t<N عضو) از جمعیت به صورت تصادفی
  - مقایسه برازش اعضای انتخاب شده و انتخاب بهترین عضو

- اگر t خیلی بزرگ نباشد، از انتخاب برترین افراد جلوگیری می‌شود (فشار انتخاب کم)
  - برای t=N، همیشه بهترین فرد انتخاب می‌شود = بیشترین فشار انتخاب
- اگر t خیلی کوچک باشد، شانس انتخاب ضعیف‌ترین افراد افزایش می‌یابد
  - برای t=1، الگوریتم انتخاب تصادفی

- مقدار معمول برای t: مقدار ۲ یا ۳







پردازش تکاملی: انتخاب . . .

- انتخاب برشی (Truncation)
  - مرتب کردن اعضای جمعیت بر اساس شایستگی آنها
  - انتخاب T درصد از اعضای برتر
  - انتخاب N عضو به صورت تصادفی (از میان T درصد از برترین اعضا)

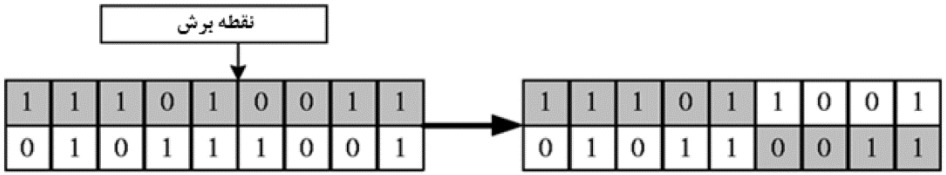
  - مقدار T بزرگ‌تر= فشار انتخاب کمتر
    - مقدار T=100 معادل انتخاب تصادفی

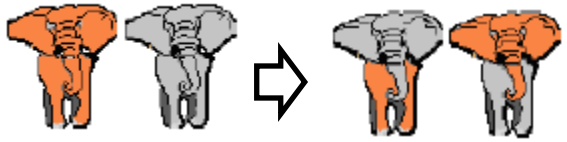
پردازش تکاملی: تولید مثل . . .

- تولید مثل = تولید جمعیت جدید (فرزندان) از والدین انتخاب شده با
  - بازترکیب (Recombination) یا هم‌برش (Crossover)
    - معادل مفهوم جفت‌گیری
    - تولید یک یا چند فرزند با ترکیب ژن‌های تصادفی انتخاب شده از دو یا چند والد
    - اعمال روی اعضای (والدین) انتخاب شده در مرحله انتخاب (برازنده)
    - تولید فرزندان مشابه والدین: انتقال ژن‌های کروموزوم‌های والدین به فرزندان
      - به ارث بردن ژن‌های والدین و تولید پاسخ‌هایی با برازش بهتر
    - اعمال عملگر بازترکیب روی اعضای جمعیت با احتمال p_c
    - استفاده بیشتر از عملگر بازترکیب = افزایش قابلیت انتفاع
  - جهش (Mutation)
    - تغییر تصادفی ژن‌ها در کروموزوم
    - هدف: یافتن مقادیر جدید ژن برای فرزندان که در والدین نبوده است
      - افزایش تنوع ژنوتایپی
    - اعمال عملگر جهش روی فرزندان با احتمال p_m
    - افزایش بیشتر از عملگر جهش = افزایش قابلیت پویش

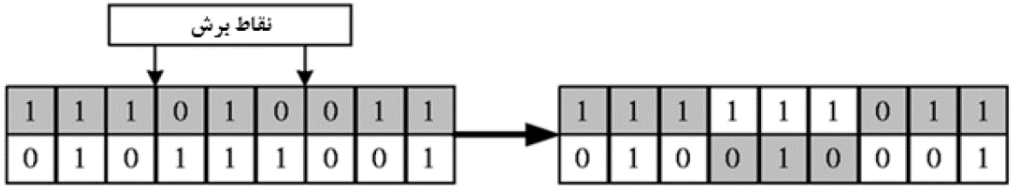
پردازش تکاملی: تولید مثل (بازترکیب) . . .

- حالت دودویی . . .
  - **هم‌برش (بازترکیب) تک نقطه‌ای** (One-point crossover)
    - انتخاب یک نقطه تصادفی و برش کروموزوم‌ها از این نقطه
    - بخش اول والد اول و بخش دوم والد دوم = فرزند اول
    - بخش دوم والد اول و بخش اول والد دوم = فرزند دوم



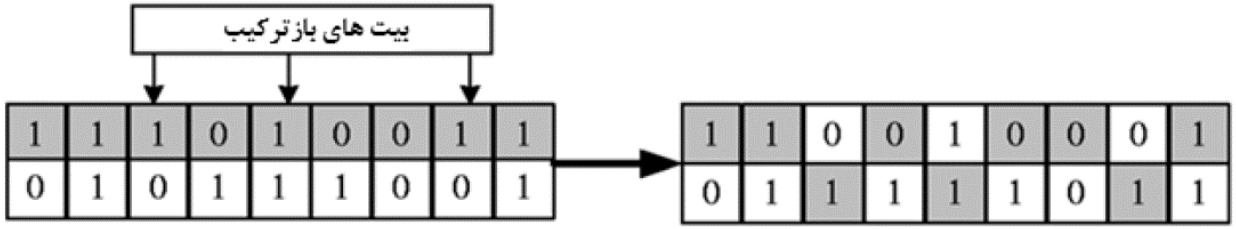


  - **هم‌برش (بازترکیب) دونقطه‌ای**
    - انتخاب دو نقطه تصادفی و برش کروموزوم‌ها از این نقاط
    - بخش اول و سوم والد اول و بخش دوم والد دوم = فرزند اول
    - بخش دوم والد اول و بخش اول و سوم والد دوم = فرزند دوم



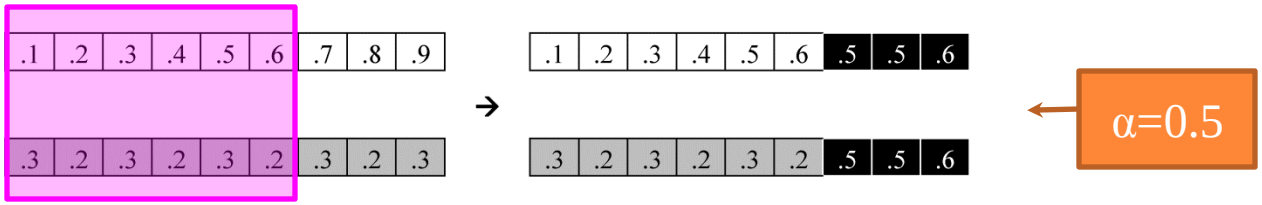
پردازش تکاملی: تولید مثل (بازترکیب) . . .

- حالت دودویی
  - هم‌برش (بازترکیب) یکنواخت (Uniform crossover)
    - انتخاب هر ژن فرزند از ژن متناسب یکی از دو والد
    - استفاده از یک توزیع تصادفی برای انتخاب ژن فرزند
    - شانس مشابه هر دو والد برای حضور در ژن فرزند
      - برای ضریب ترکیب 50% ، شانس هر دو والد برابر خواهد بود



پردازش تکاملی: تولید مثل (بازترکیب) . . .

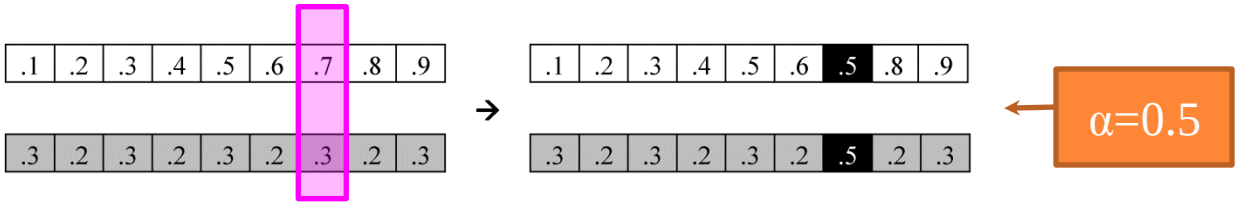
- حالت حقیقی . . .
  - **بازترکیب ساده** (Simple)
    - انتخاب یک بخش از کروموزوم‌ها
    - انتقال بخش اول از والد اول به فرزند اول
    - انتقال بخش اول از والد دوم به فرزند دوم
    - محاسبه ژن‌های بخش انتخاب نشده فرزند اول: جمع کردن مقدار ژن‌های دو کروموزوم و ضرب حاصل در α (بین ۰ و ۱)
    - محاسبه ژن‌های بخش انتخاب نشده فرزند دوم: جمع کردن مقدار ژن‌های دو کروموزوم و ضرب حاصل در $1-\alpha$



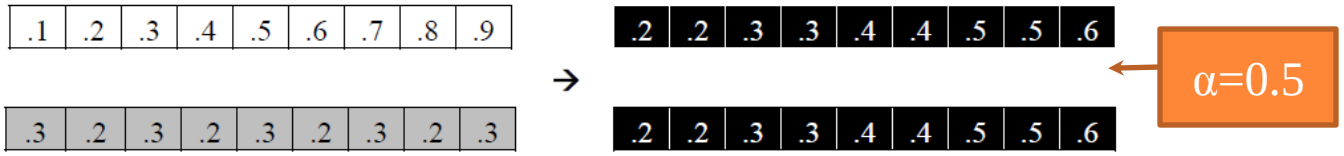
- توسعه: استفاده از عملگرهای دیگر، غیر از جمع

پردازش تکاملی: تولید مثل (بازترکیب) . . .

- حالت حقیقی
  - **بازترکیب حسابی ساده** (Simple Arithmetic)
    - مشابه بازترکیب ساده اما فقط یک ژن تغییر می‌کند



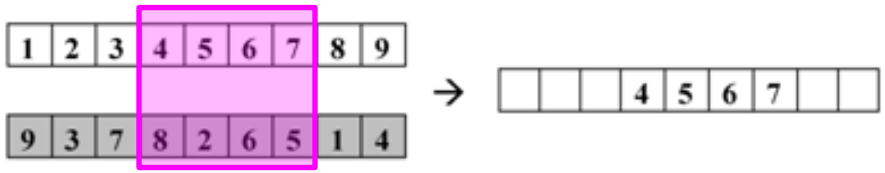
  - **بازترکیب حسابی کامل** (Whole Arithmetic)
    - مشابه بازترکیب ساده اما تمام ژن‌ها تغییر می‌کند
    - فرزندان شباهتی به والدین ندارند: افزایش قابلیت پویش



پردازش تکاملی: تولید مثل (بازترکیب) . . .

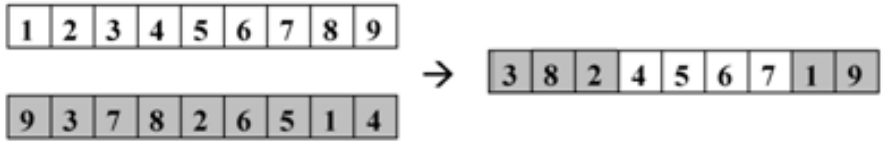
- حالت جایگشت . . .
  - بازترکیب ترتیبی (Order Recombination)
    - انتخاب دو نقطه تصادفی
    - گام اول: استفاده از والد اول = انتقال بخش میانی والد اول به فزند اول
    - گام دوم: استفاده از والد دوم = شروع از نقطه اول بخش پایانی برای استفاده از ژن‌های والد دوم در بخش پایانی فرزند اول
      - اگر مقدار ژنی قبلاً در فرزند وجود داشته باشد از آن صرف‌نظر می‌شود
    - ادامه این فرایند برای بخش آغازی (ابتدایی) فرزند اول

گام اول–استفاده از والد اول



- برای فرزند دوم
  - فرآیند قبل با عوض کردن جای دو والد

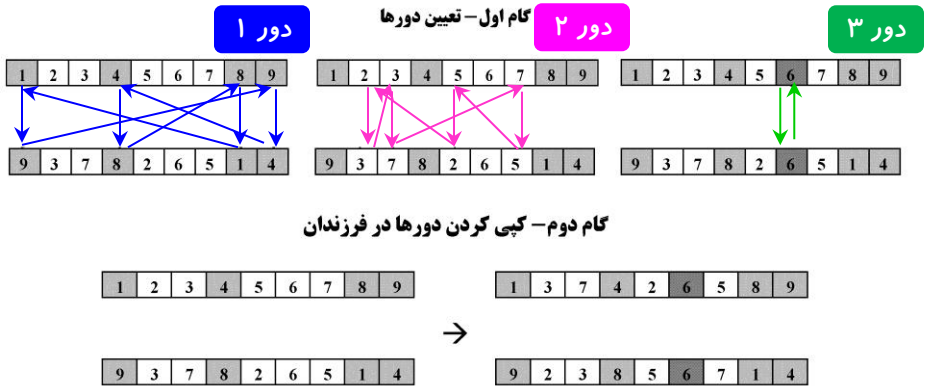
گام دوم– استفاده از والد دوم



پردازش تکاملی: تولید مثل (بازترکیب) . . .

حالت جایگشت

- **بازترکیب چرخشی** (Cycle Recombination) . . .
  - گام اول: تعیین دورها
    - شروع از ژن اول والد اول (در اینجا 1) و به همان موقعیت از ژن دوم بروید (ژن اول با مقدار 9)
    - مقدار ژن والد دوم (مقدار 9) را در والد اول جستجو کنید (موقعیت 9ام) و به همان موقعیت (9ام) از ژن دوم بروید
    - مقدار ژن والد دوم (مقدار 4) را در والد اول جستجو کنید (موقعیت 4ام) و به همان موقعیت (4ام) از ژن دوم بروید
    - تکرار گام فوق برای تا رسیدن به ژن اول والد اول (**دور**: در صورت رسیدن به نقطه‌ای که شروع کرده‌ایم)
    - تکرار الگوریتم فوق در صورت وجود ژن پیمایش نشده با شروع از اولین ژن پیمایش نشده در والد اول
  - گام دوم: تعیین ژن‌ها
    - انتقال ژن‌های دور اول از والد اول به فرزند اول
    - انتقال ژن‌های دور دوم از والد دوم به فرزند اول
    - انتقال ژن‌های دور سوم از والد اول به فرزند اول
    - انتقال ژن‌های دور چهارم از والد دوم به فرزند اول
    - ...
  - برای فرزند دوم: عوض کردن جای دو والد



نقاط شکست در تشکیل فرزندان = دورها

Hadi Veisi (h.veisi@ut.ac.ir)

پردازش تکاملی: تولید مثل (بازترکیب) . . .

حالت جایگشت

- **بازترکیب چرخشی (Cycle Recombination) – مثال**

Parent1: 8 4 7 3 6 2 5 1 9 0
Parent2: 0 1 2 3 4 5 6 7 8 9

- دور ۱
  - شروع از اولین ژن والد اول (مقدار 8) و رفتن به همان موقعیت از والد دوم (مقدار 0)
  - یافتن مقدار 0 در والد اول (موقعیت 10) و رفتن به موقعیت معادل در والد دوم (مقدار 9)
  - یافتن مقدار 9 در والد اول (موقعیت 9) و رفتن به همان موقعیت در والد دوم (مقدار 8)

Parent1: 8 4 7 3 6 2 5 1 9 0
Parent2: 0 1 2 3 4 5 6 7 8 9

- دور ۲
  - شروع از اولین ژن والد اول (مقدار 4) و رفتن به همان موقعیت از والد دوم (مقدار 1)
  - یافتن مقدار 1 در والد اول (موقعیت 8) و رفتن به موقعیت معادل در والد دوم (مقدار 7)
  - یافتن مقدار 7 در والد اول (موقعیت 3) و رفتن به همان موقعیت در والد دوم (مقدار 2)
  - یافتن مقدار 2 در والد اول (موقعیت 6) و رفتن به همان موقعیت در والد دوم (مقدار 5)
  - یافتن مقدار 5 در والد اول (موقعیت 7) و رفتن به همان موقعیت در والد دوم (مقدار 6)
  - یافتن مقدار 6 در والد اول (موقعیت 5) و رفتن به همان موقعیت در والد دوم (مقدار 4)

Parent1: 8 4 7 3 6 2 5 1 9 0
Parent2: 0 1 2 3 4 5 6 7 8 9

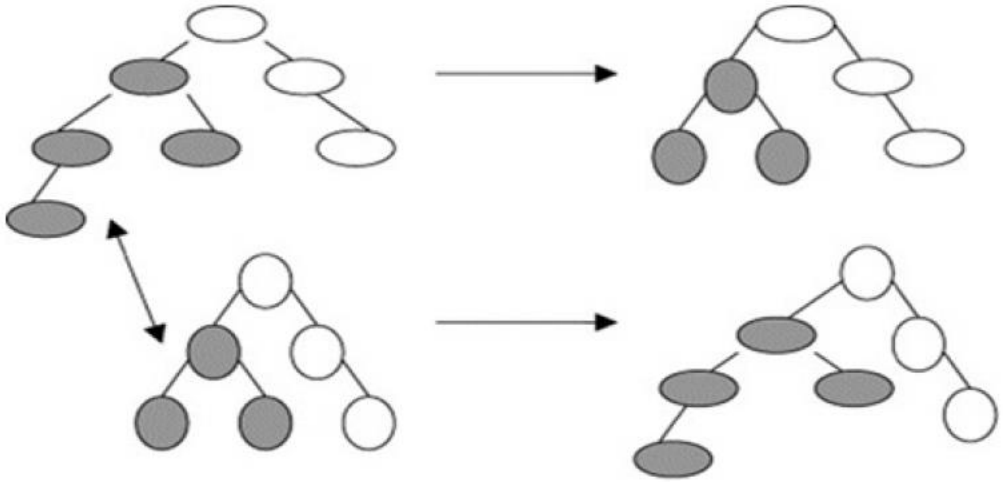
- دور ۳
  - شروع از اولین ژن والد اول (مقدار 3) و رفتن به همان موقعیت از والد دوم (مقدار 3)

Parent1: 8 4 7 3 6 2 5 1 9 0
Parent2: 0 1 2 3 4 5 6 7 8 9

Child1: 8 1 2 3 4 5 6 7 9 0
Child2: 0 4 7 3 6 2 5 1 8 9

پردازش تکاملی: تولید مثل (بازترکیب)

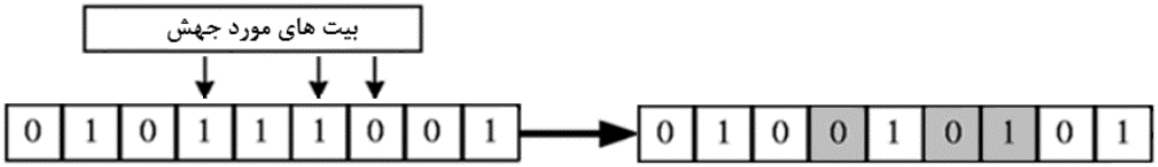
- حالت درخت
  - در نظر گرفتن نقاط شکست در دو والد
  - جابجایی زیردرخت‌های انتخاب شده



پردازش تکاملی: تولید مثل (جهش) . . .

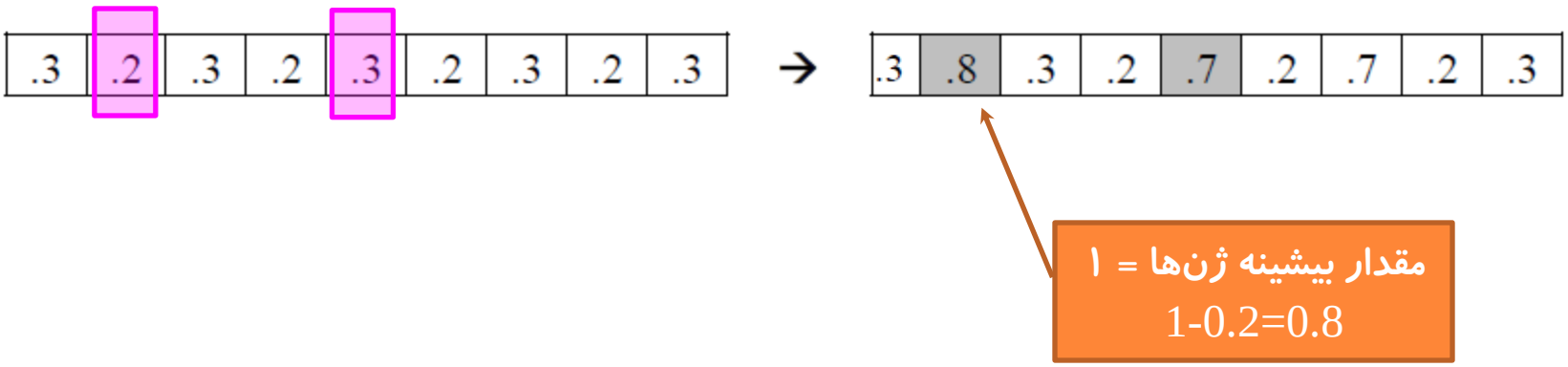
- حالت دودویی: معکوس‌سازی بیت (Bit-flipping mutation)
  - انتخاب یک یا چند بیت به صورت تصادفی
  - تغییر مقدار آن بیت (۰ به ۱ و برعکس)

- عدم استفاده از اطلاعات موجود



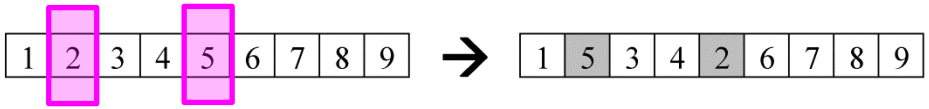
پردازش تکاملی: تولید مثل (جهش) . . .

- حالت حقیقی: جهش مکمل (Complement Mutation)
  - انتخاب یک یا چند ژن برای ژن
  - کم کردن مقدار بیشینه ممکن برای آن ژن از مقدار جاری آن ژن
    - جمع مقادیر قبلی و جدید ژن = مقدار بیشینه ممکن برای آن ژن

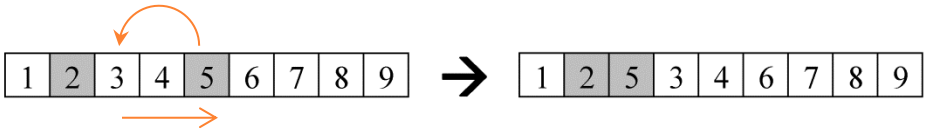


پردازش تکاملی: تولید مثل (جهش) . . .

- حالت جایگشت . . .
  - جهش جابجایی (Swap Mutation)
    - انتخاب دو ژن به صورت تصادفی و جابجا کردن مقادیر آنها



  - جهش درجی (Insert Mutation)
    - انتخاب دو ژن به صورت تصادفی
    - کپی کردن ژن دوم در ژن بعد از ژن اول
    - شیفت دادن ژن‌های دیگر به راست



پردازش تکاملی: تولید مثل (جهش) . . .

- حالت جایگشت
  - جهش درهم‌سازی (Scramble Mutation)
    - انتخاب دو ژن به صورت تصادفی
    - جابجایی تصادفی مقادیر ژن‌های بین دو نقطه

- افزایش فاصله بین دو نتقطه انتخابی= قابلیت پویش بیشتر

  - جهش معکوس (Inversion Mutation)
    - انتخاب دو ژن به صورت تصادفی
    - جابجا کردن مقادیر ژن‌های بین دو نقطه به صورتی که نسبت به وسط آنها وارونه باشند
      - گذاشتن آینه در نقطه وسط بین دو نقطه

پردازش تکاملی: تولید مثل (جهش)

- حالت درخت
  - تغییر مقدار مربوط به یک گره
  - مقدار جدید = تصادفی

پردازش تکاملی: جایگزینی . . .

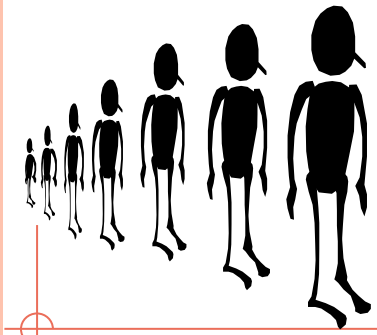
- گزینش جمعیت جدید (پاسخ‌های جدید)
  - از روی جمعیت والدین (پاسخ‌های فعلی) و جمعیت فرزندان (پاسخ‌های جدید)
  - دو نوع کلی
    - جایگزینی حالت پایدار (پایا) (Steady State Replacement)
    - جایگزینی نسلی (Generational Replacement)

- جایگزینی حالت پایدار (پایا) (Steady State Replacement) . . .
  - حفظ کردن بخش بزرگی از جمعیت والدین
  - جایگزینی بخشی از والدین با بهترین فرزندان تولید شده
  - حفظ بافت قبلی جمعیت و گوناگونی جمعیت: جلوگیری از همگرایی به بهینه محلی

  - پارامتر کنترلی: p_{rep} = درصدی از والدین که تغییر می‌کنند
    - افزایش این پارامتر = کاهش تنوع = همگرایی سریع
    - شکاف نسلی (generation gap): میزان همپوشانی نسل کنونی و نسل بعد

پردازش تکاملی: جایگزینی . . .

- جایگزینی حالت پایدار (پایا) (Steady State Replacement)
  - تصادفی: جایگزینی فرزند با یک والد که به صورت تصافی انتخاب می‌شود
  - بدترین: جایگزینی بدترین والد با فرزند
  - رقابتی: انتخاب تصادفی مجموعه‌ای از والدین و جایگزینی بدترین آنها با فرزند
  - قدیمی‌ترین: والدی که زودتر وارد شده، زودتر خارج می‌شود، امکان حذف بهترین پاسخ
  - محافظه کار: انتخاب دو والد که یکی از آنها پیرترین والد است و جایگزینی بدترین آنها با فرزند؛ حفظ بهترین والد پیر
  - نخبه‌گرایی: حفظ بهترین پاسخ‌ها (والدها)



پردازش تکاملی: جایگزینی

- جایگزینی نسلی (Generational Replacement)
  - جایگزینی کل والدین (نسل قبل) با کل فرزندان (نسل جدید)
  - جایگزین شدن بهترین عضو در جمعیت والدین با ضعیف‌ترین عضو در جمعیت فرزندان
    - نخبه سالاری (Elitism): جلوگیری از نابودی بهترین پاسخ
  - همگرایی سریع الگوریتم
  - در استراتژی تکاملی: روش انتخاب $(\mu+\lambda)$ و (μ,λ)
    - روش $(\mu+\lambda)$: تعداد μ عضو برتر والدین و تعداد λ فرزند انتخاب می‌شوند
    - روش (μ,λ): تعداد μ عضو برتر از میان λ فرزند انتخاب شده و به نسل بعد منتقل می‌شوند

پردازش تکاملی: شرایط توقف

- رسیدن به بهترین پاسخ
  - برای حالتی که مقدار برازش بهترین پاسخ را داریم
  - همیشه ممکن نیست
- محدود کردن تعداد نسل‌ها (تعداد تکرار الگوریتم)
- راکد شدن (Stagnant) جمعیت
  - عدم تغییر جمعیت در نسل‌های متوالی
  - شمارش تعداد نسل‌هایی که بهترین پاسخ تغییر نکرده است

پردازش تکاملی: کنترل قابلیت پویش و انتفاع . . .

- کنترل پارامترها
  - احتمال بازترکیب (p_c): افزایش (کاهش) باعث افزایش (کاهش) قابلیت انتفاع
  - احتمال جهش (p_m): افزایش (کاهش) باعث افزایش (کاهش) قابلیت پویش
  - درصد جایگزینی (p_{rep}): افزایش (کاهش) باعث افزایش (کاهش) قابلیت انتفاع
  - تعداد اعضای مورد گزینش در انتخاب مسابقه‌ای (p_{tourn}): افزایش (کاهش) باعث افزایش (کاهش) قابلیت انتفاع
    - انتخاب یک گروه از جمعیت به صورت تصادفی و انتخاب بهترین عضو
  - درصد اعضای مورد بررسی در انتخاب برشی (p_{trunc}): افزایش (کاهش) باعث افزایش (کاهش) قابلیت پویش
    - مرتب کردن اعضای جمعیت بر اساس شایستگی آنها، انتخاب T درصد از اعضای برتر و انتخاب N عضو به صورت تصادفی (از میان T درصد از برترین اعضا)

پردازش تکاملی: کنترل قابلیت پویش و انتفاع . . .

- به کارگیری روش مناسب برای انتخاب، بازترکیب، جهش و جایگزینی
  - انتخاب چرخ رولت ساده: تقویت قابلیت انتفاع
  - انتخاب چرخ رولت با چنداشاره‌گر: تقویت قابلیت پویش
  - بازترکیب تک نقطه‌ای در مقایسه با چند نقطه‌ای: قابلیت انتفاع بیشتر
  - بازترکیب یکنواخت: بهترین انتخاب برای تقویت قابلیت پویش
  - جهش: تخریب بیشتر در کروموزوم‌ها = قابلیت پویش بیشتر
  - جایگزینی حالت پایدار در مقایسه با جایگزینی نسلی قابلیت پویش بیشتری دارد

پردازش تکاملی: کنترل قابلیت پویش و انتفاع . . .

- **حفظ تنوع جمعیت: کرانه‌سازی (Niching)**
  - **جمعیت به سمت کرانه‌ها (نقاط بهینه) می‌روند**

  - **اشتراک برازش (Fitness Sharing)**
    - ایده به اشتراک‌گذاری منابع محدود برای موجوداتی که در یک منطقه زندگی می‌کنند
    - تشویق الگوریتم برای پویش بیشتر با در نظر گرفتن برازش کاذب برای اعضا
      - تغییر مقدار برازش واقعی
  - **انبوه‌سازی (Crowding)**
    - جایگزینی اعضای جدید با اعضای مشابه خود در جمعیت

    - روش اول: انتخاب بخشی از اعضا در جمعیت (حدود ۱۰٪) برای اعمال عملگر انتخاب و جهش، سپس، برای هر فرزند جدید تعداد CF (Crowding Factor) (عددی بین ۲ تا ۵) نفر از جمعیت انتخاب شده و با آن فرزند مقایسه می‌شوند تا فرزند جایگزین شبیه‌ترین عضو شود

    - روش دوم (انتخاب مسابقه‌ای محدود شده): جلوگیری از رقابت یک پاسخ با پاسخ‌های خیلی متفاوت از آن

Hadi Veisi (h.veisi@ut.ac.ir)

پردازش تکاملی: کنترل قابلیت پویش و انتفاع

- **حفظ تنوع جمعیت: گونه‌سازی (Speciation)**
  - **ایده: شیرها با شیرها و فیل‌ها با فیل‌ها جفت‌گیری می‌کنند**
  - **گونه: دسته‌ای از اعضا که شباهت بیشتری به همدیگر دارند**
  - **ایده گونه‌سازی: تنها اعضای مشابه اجازه تولیدمثل دارند**
    - کاهش پاسخ‌های مهلک (Lethal): برازش خیلی ضعیف، جلوگیری از بازترکیب اعضای کرانه‌های مختلف
  - **روش اول: در بازترکیب، یک عضو با نام m را انتخاب کرده، سپس از میان جمعیت یک عضو تصادفی انتخاب شده، اگر فاصله m با این عضو کمتر از یک آستانه باشد، بازترکیب انجام می‌شود**
  - **روش دوم (روش Flag bits): هر کروموزوم با یک (دو گونه) یا چند بیت نشانه برچسپ زده می‌شود که نشانگر گونه کروموزوم است.**
    - هر کروموزوم فقط به یک گونه تعلق دارد.
    - برازش نسبی عضو iام در گونه jام برابر با $|s_j|/|f_i|$ است ($|s_j|$ اندازه گونه jام).
    - تقویت برازش نسبی اعضای حاضر در مجموعه‌های کوچک
    - عمل بازترکیب تنها روی اعضای یک گونه

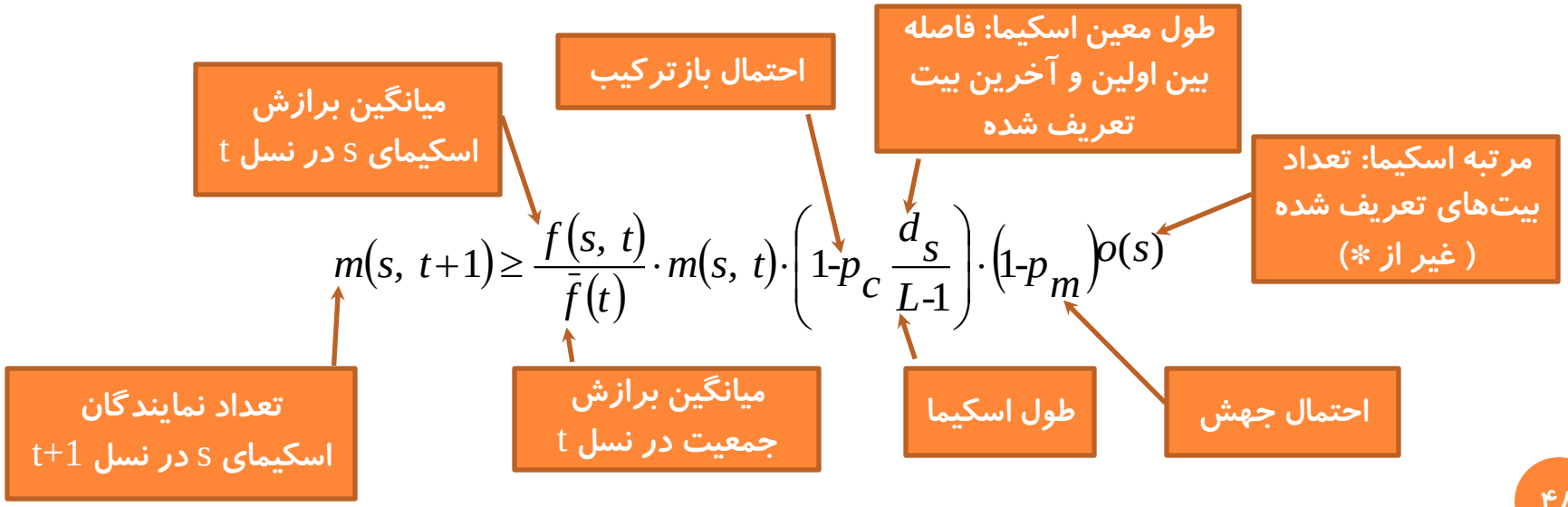
پردازش تکاملی: بهینه‌سازی چند هدفه

- مثال: می‌خواهید به یک تور دیدنی، ارزان قیمت، ۷ روزه بروید.

- هدف: بهینه‌سازی با چند هدف (چند تابع برازش)
  - استفاده از توابع تجمیعی (Aggregation Functions): ترکیب توابع برازش مختلف با وزن‌های متفاوت (مثلاً جمع وزن‌دار)
  - رویکرد جمعیتی (Population-based): تفکیک جمعیت به زیرجمعیت‌هایی برابر با تعداد اهداف و بهینه‌سازی مستقل آنها
    - کاربرد عمده در مسائل با تعداد اهداف زیاد
    - ضعف: عدم توان در حفظ پاسخ‌های غیرمغلوب (Pareto) (پاسخ بهینه با توجه به همه اهداف)
      - حذف پاسخ‌هایی که در همه هداف مساوی هستند (و ممکن است بهینه سراسری باشند)
  - رویکرد مبتنی بر پرتو (Pareto-based): شناسایی پاسخ‌های غیرمغلوب و سعی در حفظ آنها
    - کاربرد در مسائلی که ارزش همه اهداف مساله یکسان است

پردازش تکاملی: اسکیما (Schema) . . .

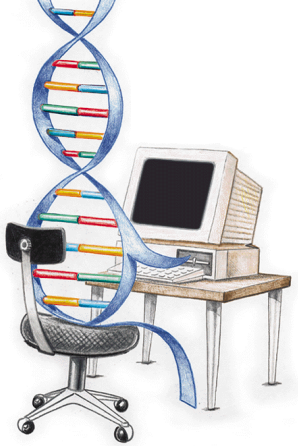
- نظریه‌ای برای بیان درستی عملکرد الگوریتم تکاملی
  - توسط هالند (Holland)
  - فرموله کردن تکامل در جمعیت در طی زمان
  - تعداد اسکیماها در جمعیت به سمت برازش نسبی خود میل می‌کند
    - اسکیما: مجموعه‌ای از رشته بیت‌های تشکیل شده از 0، 1و * (که *= صفر یا یک = don't care)
  - عدم ارائه تصویر درست از رفتار الگوریتم!



پردازش تکاملی: اسکیما (Schema)

- بلوک سازنده (Building Block)
  - اسکیمایی که دارای سه خاصیت زیر است:
    - دارای برازندگی بالاتر از میانگین برازندگی جمعیت است
    - دارای طول کوتاه باشد
    - دارای مرتبه پایین باشد

- تعداد بلوک‌های سازنده در طول نسل‌ها به صورت نمایی افزایش می‌یابد.
- بلوک‌های سازنده با عملگرهای ژنتیکی با یکدیگر ترکیب شده و بلوک‌های با مرتبه و طول بزرگ‌تر می‌سازند.
- تکرار این فرآیند منجر به یافتن پاسخ بهینه می‌شود.



پردازش تکاملی: انواع الگوریتم‌ها

- الگوریتم ژنتیک (Genetic Algorithm)
- برنامه‌نویسی ژنتیک (Genetic Programming)
- استراتژی تکامل (Evolutionary Strategy)
- برنامه‌نویسی تکاملی (Evolutionary Programming)
- تکامل تفاضلی (Differential Evolution)
- الگوریتم فرهنگی (Cultural Algorithm)
- الگوریتم هم‌تکاملی (Co-Evolutionay Algorithm)
- الگوریتم ممتیک (Memtic Algorithm)
- بهینه‌سازی تولیدمثل غیرجنسی (Asexual Reproduction Optimization)
- ...

الگوریتم ژنتیک . . .

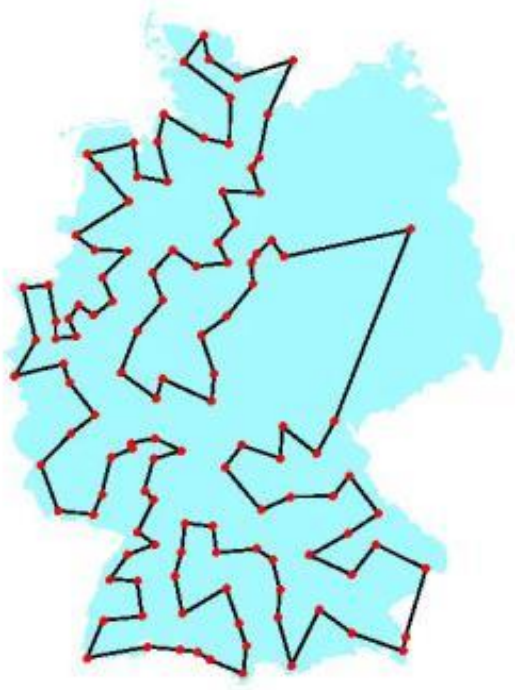
- تاریخچه
  - اولین بار توسط Fraser در ۱۹۵۷
  - ادامه توسط Bremermann در ۱۹۶۲ و Reed در ۱۹۶۷
  - تکمیل و توسعه توسط Holland در ۱۹۷۵
    - هالند = پدر الگوریتم ژنتیک
- کاربرد
  - جستجو، بهینه‌سازی، یادگیری ماشین، کنترل، زمان‌بندی کارها، رباتیک
- عملگرها
  - انتخاب (مدل‌سازی قانون بقای اصلح)
  - تولید مثل: بازترکیب و جهش

الگوریتم ژنتیک . . .

- خصوصیات الگوریتم ژنتیک استاندارد
  - استفاده از نمایش رشته بیتی
  - طول ثابت و یکسان برای کروموزوم‌ها
  - تعداد اعضای جمعیت ثابت
  - استفاده از عملگر انتخاب نسبی برای انتخاب والدین
  - استفاده از بازترکیب تک نقطه
  - استفاده از جهش معکوس‌سازی بیت
  - احتمال بازترکیب بالا (حدود ۰.۹۵ و بیشتر)
  - احتمال جهش پایین: برابر با $1/L$ که L طول کروموزوم است

الگوریتم ژنتیک: مثال . . .

- فروشنده دوره‌گرد (Travelling Salesman Problem)
  - یافتن کوتاه‌ترین مسیر برای یک فروشنده با عبور از n شهر
  - از تمامی شهرها دقیقاً یک بار بگذرد و به شهر اول برگردد
  - جزو مسائل NP-Hard
  - تعداد کل راه‌حل‌ها برای n شهر = $0.5(n-1)!$
    - برای ۵ شهر = 12
    - برای ۱۰ شهر = 1.814.400
    - برای ۳۰ شهر = $1.3*10^{32}$
  - حل با الگوریتم ژنتیک



الگوریتم ژنتیک: مثال (فروشنده دوره‌گرد) . . .

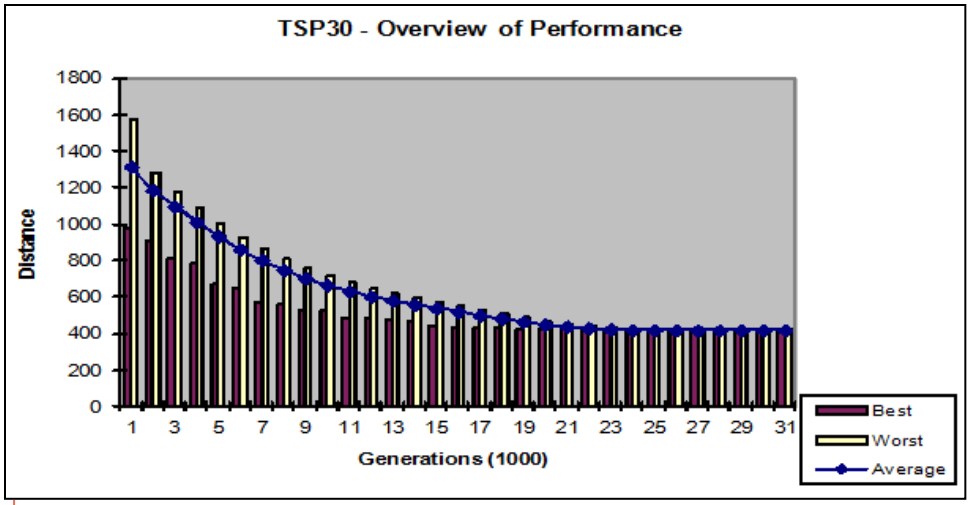
- نمایش کروموزوم: جایگشت
  - برای شهرهای تهران (۱)، اصفهان (۲)، مشهد (۳)، شیراز (۴)، تبریز(۵)، زاهدان (۶)
    - [136425]
- برازش: جمع کل فاصله (هزینه) تور
  - فاصله کم‌تر = برازش بهتر

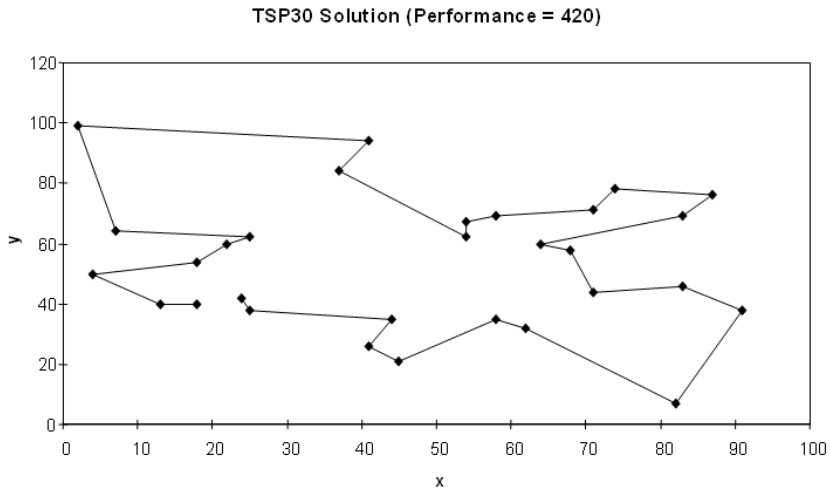
	تهران	اصفهان	مشهد	شیراز	تبریز	زاهدان
تهران	۰	۴۳۹	۸۹۴	۹۲۴	۱۵۶۷	۱۵۶۷
اصفهان	۴۳۹	۰	۱۲۲۲	۴۸۵	۱۱۹۰	۱۱۹۰
مشهد	۸۹۴	۱۲۲۲	۰	۱۳۷۴	۹۵۱	۹۵۱
شیراز	۹۲۴	۴۸۵	۱۳۷۴	۰	۱۵۲۳	۱۱۰۰
تبریز	۱۵۶۷	۱۱۹۰	۹۵۱	۱۵۲۳	۰	۲۱۶۶
زاهدان	۱۵۶۷	۱۱۹۰	۹۵۱	۱۱۰۰	۲۱۶۶	۰

- انتخاب: مسابقه‌ای، رتبه‌ای، ...
- بازترکیب: ترتیبی یا چرخشی
- جهش: جابجایی، درجی، جهش، معکوس

الگوریتم ژنتیک: مثال (فروشنده دوره‌گرد)

- پارامترها: برای ۳۰ شهر
  - جمعیت: ۵۰
    - جمعیت اولیه: تصادفی (می‌تواند هوشمندانه هم باشد)
  - تعداد نسل‌ها (تکرار): ۱۰۰۰۰
  - نرخ جهش (p_m)= ۵٪ (می‌تواند در طول الگوریتم افزایش یابد)





Hadi Veisi (h.veisi@ut.ac.ir)

الگوریتم ژنتیک: مثال . . .

- چینش بهینه حروف فارسی بر روی صفحه کلید
  - اهمیت چینش بهینه حروف فارسی بر روی صفحه‌کلید برای تایپ (به ویژه تایپ زیاد)
    - راحت‌تر و سریع‌تر کردن تایپ: جابجایی کمتر انگشتان در حین تایپ
  - بهینه‌سازی با الگوریتم ژنتیک: جستجو در فضای چینش‌های مختلف حروف

[سجاد مرادی، سعید شیری قیداری، بهینه‌سازی چینش حروف فارسی بر روی صفحه کلید با استفاده از الگوریتم‌های ژنتیکی، یازدهمین کنفرانس سالانه انجمن کامپیوتر ایران، ۱۳۸۴]

الگوریتم ژنتیک: مثال (صفحه کلید) . . .

- نمایش کروموزوم
  - هندسه صفحه‌کلید ثابت است
    - تعداد ۳۳ نشانه (۳۲ حرف الفبای فارسی و حرف همزه "ء")
    - سه ردیف صفحه‌کلید، دارای ۱۲، ۱۱ و ۱۰ کلید

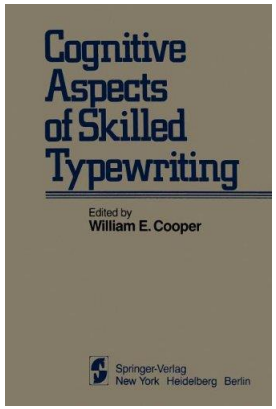
  - نمایش جایگشت
    - هر ژن = یک حرف
    - کروموزوم = برداری از حروف فارسی (بیانگر ترکیب‌های مختلف حروف فارسی روی صفحه‌کلید)
    - هر بردار شامل ۳۳ عنصر

ض	ص	ث	ق	ف	. . .	ژ
۱	۲	۳	۴	۵	. . .	۳۳

- تعداد چینش‌های مختلف: $33! = 8.7*10^{36}$

الگوریتم ژنتیک: مثال (صفحه کلید) . . .

- تابع برازش . . .
  - بیانگر میزان راحت یا سخت بودن تایپ با چینش جاری حروف بر روی صفحه‌کلید
  - یک مساله پیچیده ارگونومیک



- چهار هدف را برای طراحی یک صفحه‌کلید کارا
  - برابر بودن کاری که دو دست انجام می‌دهند
  - بیشترین تایپ حروف به صورت متناوب با دو دست
  - کمترین تکرار تایپ دو حرف متوالی با یک انگشت
  - بیشترین تایپ حروف بر روی کلیدهای پایه‌ای (کلیدهای ردیف وسط)

- تابع برازش = کمی کردن چهار هدف فوق

الگوریتم ژنتیک: مثال (صفحه کلید) . . .

- **تابع برازش . . .**
  - **اندازه‌گیری مقدار دو هدف اول: C_{hand}= هزینه استفاده از یک دست برای تایپ دو حرف پشت سر هم**
    - برابری بودن کاری که دو دست انجام می‌دهند
    - بیشترین تایپ حروف به صورت متناوب با دو دست
  - **اندازه‌گیری هدف سوم: C_{finger}= هزینه استفاده از یک انگشت برای تایپ دو حرف پشت سر هم**
    - کمترین تکرار تایپ دو حرف متوالی با یک انگشت
  - **اندازه‌گیری هدف چهارم و برخی عوامل دیگر: $C_{ergonomic}$= هزینه تایپ یک حرف با توجه به موقعیت آن حرف بر روی صفحه‌کلید.**
    - بیشترین تایپ حروف بر روی کلیدهای پایه‌ای (کلیدهای ردیف وسط)
    - استفاده از انگشتان مختلف دست
    - میزان جابجایی انگشتان روی صفحه کلید

الگوریتم ژنتیک: مثال (صفحه کلید) . . .

- تابع برازش . . .
  - تابع برازش = مجموع سه فاکتور
  - محاسبه برای یک مجموعه متن فارسی

$$\text{Fitness}(\text{layout}) = \sum_{w_i \in W} \sum_{l_j \in w_i} [C_{\text{hand}}(l_j, l_{j-1}) + C_{finger}(l_j, l_{j-1}) + C_{ergonomic}(l_j)]$$

- W مجموعه تمامی کلمات موجود در متن مورد استفاده
- w_i کلمه iام از مجموعه W
- l_j حرف jام از کلمه w_i

- هزینه فشردن کلیدها: مصاحبه با تایپیست‌های حرفه‌ای (راست دست)

Jeffrey S. Goettl, Alexander W. Brugh, Bryant A. Julstrom: Call me e-mail: arranging the keyboard with a permutation-coded genetic algorithm. SAC 2005: 947-951

الگوریتم ژنتیک: مثال (صفحه کلید) . . .

- **تابع برازش**

$$\text{Fitness}(\text{layout}) = \sum_{w_i \in W} \sum_{l_j \in w_i} [C_{\text{hand}}(l_j, l_{j-1}) + C_{finger}(l_j, l_{j-1}) + C_{ergonomic}(l_j)]$$

- **$C_{\text{hand}}(l_j, l_{j-1})$= مقداری ثابت برای حالتی که دو حرف jام و j-1ام با یک دست تایپ شود**
  - در غیراین‌صورت مقدار آن صفر است
  - مقدار ثابت = یک‌چهارم $C_{\text{finger}}(l_j, l_{j-1})$
- **$C_{\text{finger}}(l_j, l_{j-1})$= مقداری ثابت برای حالتی که دو حرف jام و j-1ام با یک انگشت تایپ شود،**

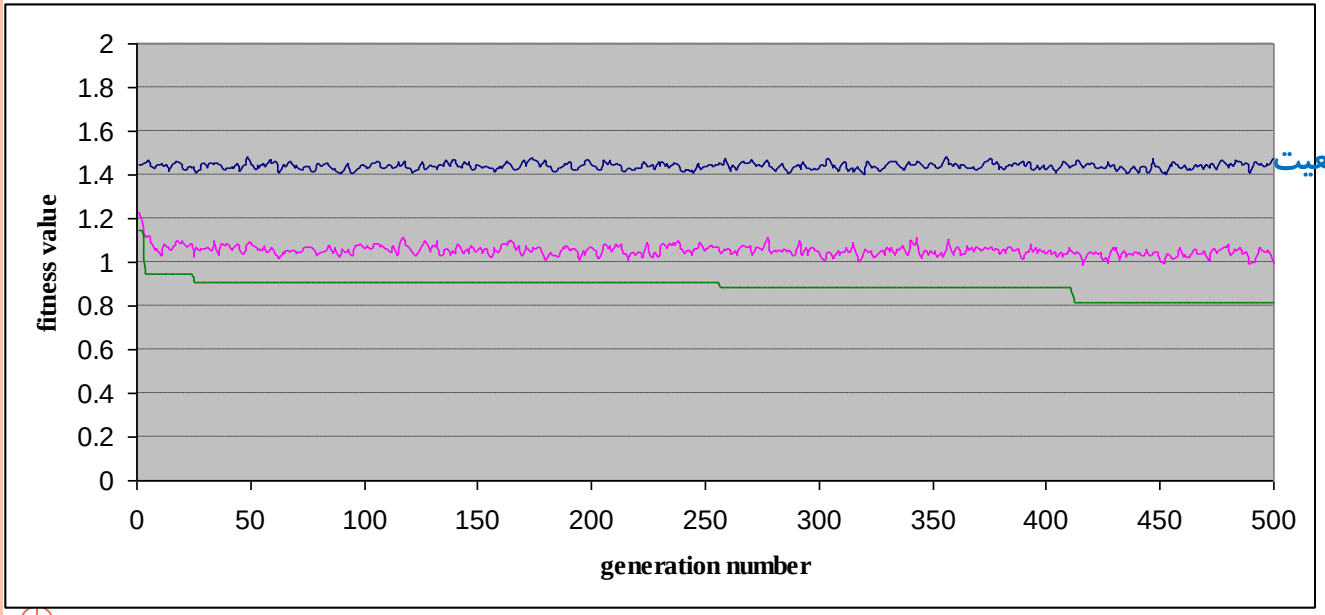
  - در غیراین‌صورت مقدار آن صفر است
  - مقدار ثابت = متوسط اعداد هزینه فشردن کلیدها
- **$C_{\text{ergonomic}}(l_j)$= مقدار معادل هزینه فشردن کلیدها**

الگوریتم ژنتیک: مثال (صفحه کلید) . . .

- عملگرها
  - در اینجا تنها از عملگر جهش استفاده شده است
    - دلیل عدم استفاده از عملگر بازترکیب: هزینه زمانی بالا
  - عملگر جهش
    - سعی در حفظ پاسخ‌های برگزیده
    - در نظر گرفتن درصدی از پاسخ‌ها به عنوان جامعه نخبگان (بیشترین مقدار برای تابع برازش): جابجایی چهار ژن به صورت تصادفی
    - برای سایر پاسخ‌ها (افراد عادی جمعیت): جابجایی ۱۲ ژن

الگوریتم ژنتیک: مثال (صفحه کلید) . . .

- ارزیابی: پارامترها
  - تعداد اعضای جمعیت: ۱۰۰ کروموزوم
  - نسل اول: به صورت تصادفی
  - درصد تشکیل‌دهنده جامعه نخبگان برای عملگر جهش ۱۰٪ کل جمعیت
  - تعداد کل نسل‌ها: ۵۰۰ نسل



- ارزیابی: پاسخ

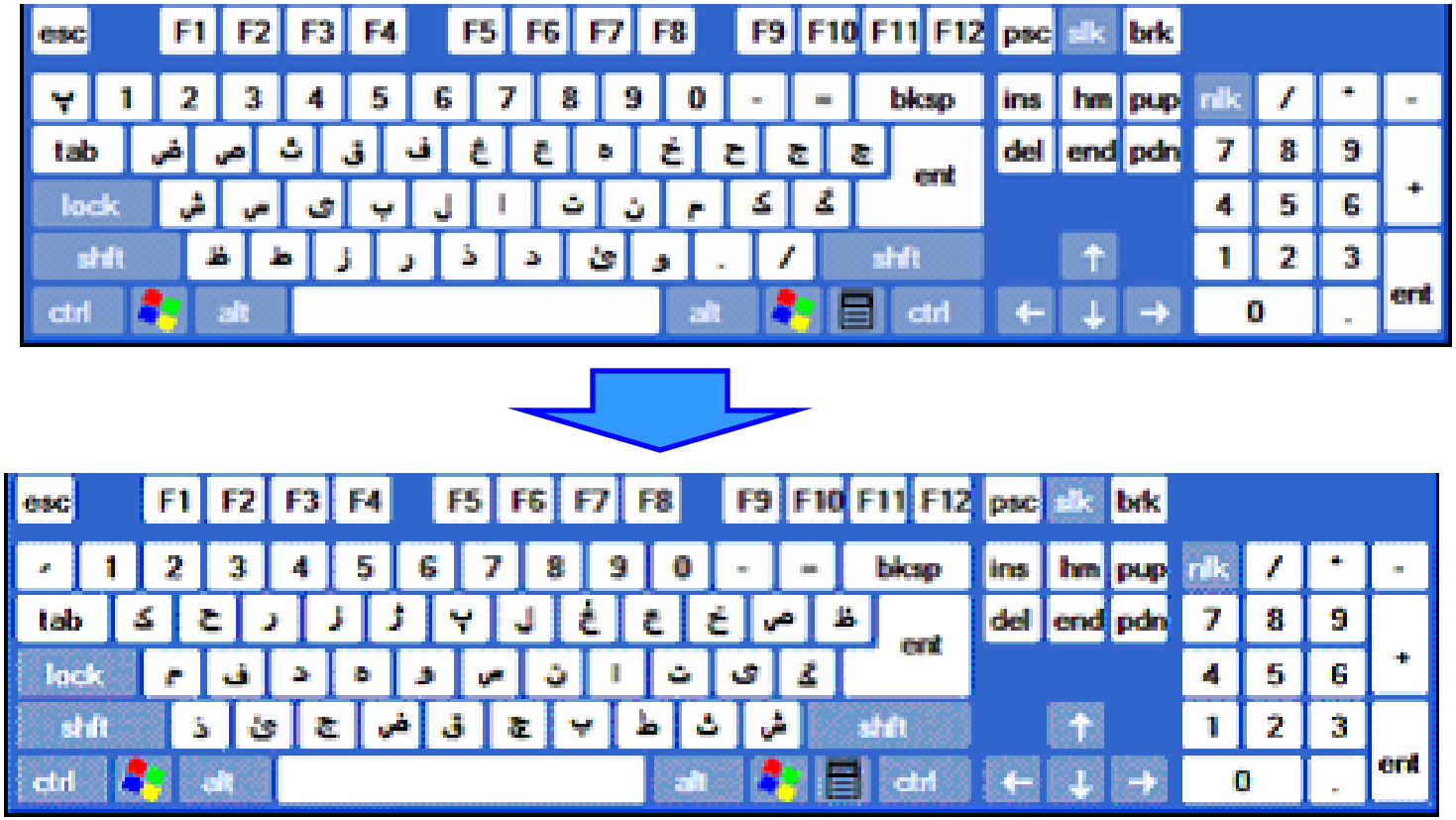
متوسط مقادیر برازش همه اعضای جمعیت

متوسط مقادیر برازش جامعه نخبگان

بهترین برازش

الگوریتم ژنتیک: مثال (صفحه کلید)

- ارزیابی
  - هزینه بهترین چینش با الگوریتم ژنتیک = ۰.۸۱۵ هزینه چینش کنونی



برنامه‌نویسی ژنتیک

- برنامه‌نویسی ژنتیک (Genetic Programming)
  - ارائه شده توسط Koza برای تکامل برنامه‌های کامپیوتری
    - هدف: پیدا کردن برنامه بهینه
  - کاربردها: برنامه‌نویسی خودکار، برنامه‌ریزی، درخت تصمیم، طراحی شبکه عصبی، ...
  - می‌توان آن را یکی از الگوریتم‌های ژنتیک دانست!
  - تفاوت اصلی با الگوریتم ژنتیک: استفاده از نمایش درختی
  - ویژگی‌ها
    - طول متغیر برای کروموزوم (ویژگی انحصاری)
    - تعداد اعضای جمعیت ثابت است
    - استفاده از عملگرهای بازترکیب (با احتمال بیشتر از ۹۰٪) و جهش (کم‌تر از ۱٪)
    - استفاده از عملگر تغییر معماری (Architecture Alternation)
      - ویرایش برخی قوانین (مثلاً جایگزینی x and x با x)
      - شناسایی بلوک‌های سازنده و جلوگیری از تغییر بلوک‌های مفید با عملگرهای تولید مثل
  - تابع برازش: کارایی موجود (برنامه) بر روی یک مجموعه آزمون

استراتژی تکامل

- استراتژی تکامل (Evolutionary Strategy)
  - ارائه شده توسط ریچنبرگ در سال ۱۹۶۰
  - هدف: بهینه‌سازی فرآیند تکامل است
  - علاوه بر ویژگی‌های ژنی برای هر موجود، پارامترهای استراتژی هم وجود دارد
    - مدل‌سازی رفتار موجود در محیط
    - تکامل هم‌زمان ویژگی‌های ژنی و پارامترهای استراتژی
  - کاربردها: بهینه‌سازی، طراحی کنترل‌گر، سیستم‌های قدرت
  - ویژگی‌ها
    - نمایش اعداد حقیقی
    - تعداد ثابت برای اعضای جمعیت
    - استفاده از عملگر جهش با نرخ تطبیقی (بازترکیبی در موارد محدودی استفاده می‌شود)
      - پاسخ‌های بهتر دارای نرخ جهش کم و برعکس
    - هر موجود دارای پارامترهای استراتژی مربوط به خود است
    - استفاده از جایگزینی‌های $(\mu+\lambda)$ و (μ,λ) [مراجعه به چند اسلاید قبل]